



# EUROPLEXUS

A Computer Program for the Finite Element Simulation of Fluid-Structure  
Systems under Transient Dynamic Loading

## USER'S MANUAL



JRC89891

Commissariat à l'énergie atomique  
Direction de l'énergie nucléaire  
Département de Modélisation des Systèmes et  
Structures  
Service des Etudes Mécaniques et Thermiques  
Laboratoire d'Etudes de Dynamique

European Commission  
Joint Research Centre  
Directorate for Space, Security and Migration  
Safety and Security of Buildings

EUROPLEXUS manual generated on:

November 11, 2020 at 11:28 A.M..

# Contents

<b>1</b>	<b>GETTING STARTED</b>	<b>19</b>
1.1	ABOUT EUROPLEXUS	19
1.2	Installation	20
1.2.1	License types	20
1.2.2	Installation under Windows	20
1.2.3	Installation under Linux	20
1.2.4	Manual	20
1.2.5	Benchmarks	20
1.3	Data flow	21
1.3.1	Files in EPX	21
1.4	Mesh generation	23
1.4.1	k-file (LS-DYNA)	23
1.4.2	med (SALOME)	24
1.4.3	CAST3M	24
1.5	EUROPLEXUS Inputs	25
1.5.1	First input sample	25
1.5.2	Elements	26
1.5.3	Sandwich Elements	27
1.5.4	Materials	27
1.5.5	Element erosion	28
1.5.6	Fluid calculations	28
1.5.7	INTRODUCTION TO FLUID-STRUCTURE INTERACTION	28
1.5.8	Restart	29
1.5.9	Index of important commands	30
1.6	Outputs/Post processing	32
<b>2</b>	<b>PREPARATION OF THE INPUT DATA</b>	<b>33</b>
2.1	WRITING CONVENTIONS	34
2.2	USE OF LITERAL VARIABLES	36
2.3	PROCEDURE /LECTURE/	37
2.4	PROCEDURE /PROGRESSION/	42
2.5	PROCEDURE /CTIME/	43
2.6	PROCEDURE /LCHP/	45
2.7	PROCEDURE /LECDDL/	46
2.8	MPI PARALLEL CALCULATIONS	48
2.9	READING DATA FROM EXTERNAL FILES	49
<b>3</b>	<b>SPATIAL DISCRETIZATION (ELEMENT TYPES)</b>	<b>50</b>
3.1	ELEMENT TYPES	50
3.1.1	1-D ELEMENTS	51
3.1.2	2-D ELEMENTS	56
3.1.3	3-D ELEMENTS	65
3.2	SANDWICH (MULTI-LAYER) ELEMENTS	79
3.2.1	LOCATION AND NUMBER OF THE INTEGRATION POINTS	79
<b>4</b>	<b>GROUP A—PROBLEM TYPE AND DIMENSIONING</b>	<b>81</b>
4.1	TITLE AND PRELIMINARY INFORMATION	82
4.1.1	TITLE	82
4.1.2	INPUT DATA ECHO AND INPUT CHECK UP	82
4.2	INTERACTIVE (FOREGROUND) EXECUTION	83

4.3	FILE MANAGEMENT	84
4.3.1	DEFAULT FILE NAMES	84
4.3.2	EXPLICIT FILE OPENING	87
4.3.3	EXPLICIT OUTPUT DIRECTORY DEFINITION	90
4.4	TYPE OF MESH, PROBLEM AND LISTING	91
4.4.1	MODELING OF ADVECTION-DIFFUSION PHENOMENA	102
4.4.2	TYPE OF OUTPUT LISTING	103
4.5	MPI GLOBAL OPTIONS	105
4.6	DIMENSIONING	107
4.6.1	DIMENSIONS RELATIVE TO GROUP B (GEOMETRY)	108
	NODES	109
	NUMBER OF ELEMENTS	110
	ADAPTIVITY (Adaptive Mesh Refinement)	111
	DECOHESION (Automatic Separation of the Elements)	113
	GRID MOTION (A.L.E.)	114
	SPACE INTEGRATION FOR SHELL AND BEAM ELEMENTS	115
	MEMORY FOR ED1D CALCULATIONS	116
4.6.2	DIMENSIONS RELATIVE TO GROUP C (MATERIALS)	117
4.6.3	DIMENSIONS RELATIVE TO GROUP D (CONNECTIONS)	118
4.6.4	DIMENSIONS RELATIVE TO GROUP G (PRINTOUTS)	120
4.6.5	DIMENSIONS RELATIVE TO GROUP I (CALCULATION)	121
4.6.6	DIMENSIONS RELATIVE TO ADVECTION-DIFFUSION	122
4.6.7	END OF DIMENSIONING	123
<b>5</b>	<b>GROUP B—MESH AND GRID MOTION</b>	<b>124</b>
5.1	OPTIONAL MESH MANIPULATION COMMANDS	125
5.2	MESH IN COCO-LIKE OR IN FREE FORMAT	127
5.2.1	GEOMETRY	127
5.2.2	ELEMENT ZONES	129
5.2.3	EXAMPLE: MESH FROM FORMATTED EXTERNAL FILE	131
5.3	MESH IN CASTEM FORMAT	134
5.3.1	Superposed elements in a Cast3m mesh (color problem)	136
5.4	MESH IN I-DEAS FORMAT	137
5.5	MESH IN LS-DYNA FORMAT (K-FILE)	138
5.6	GRID MOTION IN AN A.L.E. COMPUTATION	139
5.6.1	AUXILIARY FILE	142
5.6.2	“SUIVRE”	143
5.6.3	“LIGNE”	144
5.6.4	“PLAN”	145
5.6.5	“TETR”	146
5.6.6	“HEXA”	147
5.6.7	“PRIS”	148
5.6.8	“PYRA”	149
5.6.9	“CONTOUR”	150
5.6.10	“SLIP”	152
5.6.11	“AUTO”	153
5.6.12	“MEAN”	155
5.6.13	“DIRE”	156
5.6.14	“QUAD”	157
5.6.15	“SPEC”	158
5.6.16	“MECA”	159

5.6.17	“ELAS”	160
5.6.18	“GLOB”	161
5.6.19	USER’S ROUTINE “COOGRI”	162
5.7	MESH REFINEMENT FOR WAVEFRONT TRACKING	164
5.8	ADAPTIVITY	166
<b>6</b>	<b>GROUP C—GEOMETRIC COMPLEMENTS</b>	<b>174</b>
6.1	AUXILIARY FILE	175
6.2	FS-Core GEOMETRIC COMPLEMENTS	176
6.3	ADDED MASSES	177
6.4	Automatic mass distribution : MAPM	178
6.5	THICKNESS OR SECTION	179
6.6	GEOMETRICAL PARAMETERS FOR SHELL ELEMENTS	181
6.7	EXCENTRICITY FOR SHELL ELEMENTS	183
6.8	SANDWICHES AND LAYERS	184
6.9	GEOMETRY OF BEAMS	186
6.10	DIAMETERS	190
6.11	NAMED ELEMENT GROUPS	192
6.12	NAMED NODE GROUPS	197
6.13	ELEMENT COLORS	204
6.14	RTM COMPOSITE MATERIALS	205
6.15	PFEM METHOD	206
6.16	FLYING DEBRIS MODEL	207
6.17	DISPLACEMENT EROSION	216
6.18	STRUCTURED FLUID GRID MODEL	217
6.19	AUTOMATIC GENERATION OF SPECTRAL MICRO MESH	220
6.20	ELEMENT-SPECIFIC EROSION	222
6.21	MESH ORIENTATION	223
6.22	AUTOMATICALLY GENERATED SPH PARTICLES	225
6.23	WATER TABLES	228
6.24	HELIUM TABLES	231
6.25	PIPE JUNCTIONS	233
6.26	TUBM (3D-1D JUNCTION)	235
6.27	TUYM (3D-1D JUNCTION)	237
6.28	CORRESPONDENCE BETWEEN NODES	239
6.29	SPH SHELL ELEMENT (SPHC)	241
6.30	DISCRETE ELEMENT MODEL (ELDI)	243
6.31	MULTILAYER ELEMENT CMC3	247
6.32	ORTHOTROPY	248
6.33	ORTHOTROPY FOR 3D SHELLS	250
6.34	PARTICLE ELEMENT (BILLE)	252
6.35	RIGID BODIES (JRC Implementation)	254
<b>7</b>	<b>GROUP C1—MATERIALS</b>	<b>256</b>
7.1	LIST OF MATERIALS	257
7.2	AVAILABLE MATERIALS FOR ELEMENT TYPES	259
7.3	AUXILIARY FILE	269
7.4	LOCALISED DAMPING	270
7.5	NON-LINEAR SUPPORTS : ”APPU”	271
7.6	NON-LINEAR SUPPORTS : ”SUPP”	274
7.7	SOLID MATERIALS	276

7.7.1	LINEAR ELASTICITY . . . . .	276
7.7.2	RESL: NONLINEAR SPRING IN THE LOCAL REFERENCE FRAME . . . . .	278
7.7.3	GENERIC LINEAR ELASTICITY . . . . .	280
7.7.4	GENERIC PLASTICITY . . . . .	281
7.7.5	RESG: NONLINEAR SPRING IN THE GLOBAL REFERENCE FRAME . . . . .	282
7.7.6	DRUCKER-PRAGER . . . . .	284
7.7.7	VON MISES MATERIAL . . . . .	286
	PERFECTLY PLASTIC VON MISES . . . . .	287
	ISOTROPIC VON MISES . . . . .	289
	DYNAMIC VON MISES . . . . .	291
	TEMPERATURE-DEPENDENT VON MISES . . . . .	296
7.7.8	STEINBERG-GUINAN . . . . .	298
7.7.9	VON MISES ORTHOTROPIC GRID MODEL . . . . .	300
7.7.10	LEM1 . . . . .	302
7.7.11	ZALM . . . . .	304
7.7.12	LMC2 . . . . .	307
7.7.13	CONCRETE: Old version . . . . .	312
7.7.14	CONCRETE: DYNAR LMT (BLMT) . . . . .	319
7.7.15	BPEL: MODEL FOR PRESTRESSING CABLE-CONCRETE FRICTION . . . . .	322
7.7.16	CONCRETE: MAZARS-LINEAR ELASTIC LAW WITH DAMAGE . . . . .	323
7.7.17	DADC: Dynamic Anisotropic Damage Concrete . . . . .	325
7.7.18	DPDC: Dynamic Plastic Damage Concrete . . . . .	328
7.7.19	DAMAGE . . . . .	334
7.7.20	EOBT: ANISOTROPIC DAMAGE OF CONCRETE (EDF) . . . . .	336
7.7.21	ENGR: ELASTIC GRADIENT DAMAGE MATERIAL . . . . .	338
7.7.22	LINEAR MULTI-LAYER . . . . .	341
7.7.23	CHANG-CHANG MULTI-LAYER MODEL . . . . .	345
7.7.24	LINEAR ORTHOTROPY . . . . .	348
7.7.25	ORTS : LINEAR ORTHOTROPY (Local basis) . . . . .	350
7.7.26	ORTE : ELASTIC DAMAGE ORTHOTROPY (only in 3D) . . . . .	353
7.7.27	ODMS : ONERA DAMAGE MODEL (only in 3D) . . . . .	358
7.7.28	WOOD . . . . .	371
7.7.29	ORSR : RATE DEPENDENT LINEAR ORTHOTROPY (Local basis) . . . . .	377
7.7.30	MASS . . . . .	384
7.7.31	PHANTOM . . . . .	386
7.7.32	FREE (USER'S MATERIAL) . . . . .	387
7.7.33	FREE MATERIAL OF TYPE STRUCTURE . . . . .	389
7.7.34	FREE MATERIAL OF TYPE FLUID . . . . .	392
7.7.35	FREE MATERIAL OF TYPE MATERIAL POINT . . . . .	396
7.7.36	FREE MATERIAL OF TYPE MECHANISM . . . . .	399
7.7.37	FREE MATERIAL OF TYPE BOUNDARY CONDITIONS . . . . .	402
7.7.38	FREE PARTICLE MATERIAL . . . . .	406
7.7.39	VON MISES (ISPRA IMPLEMENTATION) . . . . .	410
7.7.40	VM1D MATERIAL . . . . .	412
7.7.41	DONE MATERIAL . . . . .	413
7.7.42	VON MISES WITH VISCOPLASTIC REGULARIZATION . . . . .	416
7.7.43	DRUCKER PRAGER WITH VISCOPLASTIC REGULARIZATION . . . . .	418
7.7.44	COMPOSITE MATERIAL (LINEAR ORTHOTROPIC) ISPRA IMPLEMENTATION . . . . .	421
7.7.45	MODIFIED CAM-CLAY MATERIAL . . . . .	423

7.7.46	MODIFIED CAM-CLAY MATERIAL WITH VISCOPLASTIC REGULARIZATION	427
7.7.47	FUNE (SPECIALIZED CABLE MATERIAL)	431
7.7.48	JOHNSON-COOK MODEL	433
7.7.49	LUDWIG-PRANDTL MODEL	437
7.7.50	LUDWIK MODEL	439
7.7.51	ZERILLI-ARMSTRONG MODEL	441
7.7.52	DRUCKER-PRAGER WITH HYDROSTATIC POST-FAILURE (JRC)	443
7.7.53	ALUMINIUM FOAM	445
7.7.54	GLRC: REINFORCED CONCRETE FOR SHELLS	448
7.7.55	HYPERELASTIC MATERIAL	460
7.7.56	MINT: MATERIAL FOR INTERFACE ELEMENT	466
7.7.57	THE SL-ZA MODEL	468
7.7.58	RTM composite material	470
7.7.59	TVMC (LOI ELASTOPLASTIQUE POUR COMPOSITES)	473
7.7.60	HILL MATERIAL MODEL	476
7.7.61	GLASS MATERIAL	478
7.7.62	BL3S: REINFORCED CONCRETE LAW FOR DEM	480
7.7.63	LAMINATED SECURITY GLASS MATERIAL	485
7.7.64	SMAZ: MAZARS-LINEAR ELASTIC LAW WITH DAMAGE FOR SPHC ELEMENTS	487
7.7.65	SLIN: LINEAR ELASTIC LAW WITH DAMAGE FOR SPHC ELEMENTS	489
7.7.66	JCLM	490
7.7.67	VPJC	492
7.7.68	RIGI (Rigid Material)	500
7.7.69	DCMS (Damage in Coarsely Meshed Shells)	501
7.7.70	MOONEY-RIVLIN MATERIAL	503
7.7.71	OGDEN MATERIAL	506
7.7.72	BLATZ-KO MATERIAL	509
7.8	FLUID MATERIALS	512
7.8.1	GENERIC IDEAL GAS	514
7.8.2	FLUID	515
7.8.3	TAIT EoS	519
7.8.4	PERFECT GAS	521
7.8.5	STIFFENED GAS	523
7.8.6	SOURCE	525
7.8.7	SODIUM-WATER REACTION (“NAH2”)	526
7.8.8	SODIUM-WATER REACTION (“RSEA”)	529
7.8.9	WATER	533
7.8.10	HELIUM	538
7.8.11	1D WALL	540
7.8.12	PIPE BREAK PARAMETERS	542
7.8.13	MULTIPLE MATERIALS	544
7.8.14	LIQUID	546
7.8.15	TUBE BUNDLES	548
7.8.16	HOMOGENEISATION OF TUBE BUNDLES	552
7.8.17	PUFF	554
7.8.18	MIEG (Mie-Grüneisen)	556
7.8.19	ADCR	558
7.8.20	EXVL—VAN LEER HYDROGEN DETONATION	563
7.8.21	JWL—JONES-WILKINS-LEE LAW	567

7.8.22	CHOC—RANKINE-HUGONIOT SHOCK	571
7.8.23	GPDI—DIFFUSIVE VAN LEER PERFECT GAS	572
7.8.24	VAN LEER PERFECT GAS	575
7.8.25	ADCJ	577
7.8.26	FLUID PARTICLE	581
7.8.27	PRGL—POROUS JELLY	583
7.8.28	VAN DER WAALS GAS	585
7.8.29	JWLS	587
7.8.30	USER-DEFINED FLUID (FLUT)	592
7.8.31	MATERIAL FOR MINERAL OIL PYROLISIS	603
7.8.32	ADVECTION-DIFFUSION FLUID (ADFM)	606
7.8.33	MULTICOMPONENT FLUID MATERIAL (MCGP)	607
7.8.34	MULTICOMPONENT FAR-FIELD FLUID MATERIAL (MCFF)	610
7.8.35	MULTIPHASE MULTICOMPONENT FLUID MATERIAL (FLMP)	612
7.8.36	SG2P—Multicomponent Stiffened Gases - Fully Conservative Formulatoin	614
7.8.37	SGMP—Multicomponent Stiffened Gases models	620
7.8.38	BUBBLE MODEL	626
7.8.39	CDEM—Discrete Equation Method for Combustion	627
7.8.40	DEMS—Discrete Equation Method for Two Phase Stiffened Gases	641
7.8.41	GAZD—Detonation in gas Mixture	648
7.8.42	HMEM—Homogeneous two-phase multi-miscible-component fluid	655
7.9	IMPEDANCES	660
7.9.1	ABSORBING MATERIAL	665
7.9.2	TOTAL ABSORBING MATERIAL	667
7.9.3	HEAD LOSS	669
7.9.4	GRID	670
7.9.5	IMPOSED PRESSURE	672
7.9.6	DIAPHRAGM	674
7.9.7	MEMBRANE	676
7.9.8	CRITICAL MASS FLOW RATE	678
7.9.9	CLOSED BOTTOM	680
7.9.10	PUMP	681
7.9.11	FOLLOWING FORCE	683
7.9.12	CRITICAL MASS FLOW RATE COUPLING (NAH2)	684
7.9.13	CRITICAL MASS FLOW RATE COUPLING (RSEA)	686
7.9.14	“VANNE” - SAFETY AND REGULATING VALVES	688
7.9.15	SWING CHECK VALVE WITH FLUID-STRUCTURE COUPLING	690
7.9.16	FLUIDE-STRUCTURE GRID	692
7.9.17	PERFORATED PLATE (JRC)	694
7.9.18	RUPTURE DISK (JRC)	696
7.9.19	STACEY’S 1ST ORDER ABSORBING BOUNDARY (JRC)	698
7.9.20	RUPTURE DISK FOR MC FORMULATION (JRC)	700
7.9.21	ABSORBING MATERIAL VAN LEER	702
7.9.22	CONDITION AT INFINITY VAN LEER	703
7.9.23	IMPOSED PRESSURE VAN LEER	704
7.9.24	IMPOSED PERFECT GAS MASS FLOW RATE VAN LEER	705
7.9.25	RIGID OBSTACLE VAN LEER	706
7.9.26	SAFETY VALVE (JRC)	707
7.9.27	RUPTURE DISK (JRC NEW)	715
7.9.28	ABSORBING MATERIAL (JRC implementation)	717
7.9.29	ABSORBING MATERIAL (Zienkiewicz for geotechnical materials)	719



7.9.30	AIR BLAST WAVE . . . . .	722
7.9.31	NTNU's SIMPLIFIED FSI MODEL . . . . .	726
7.9.32	FRAGILE PLATE . . . . .	729
7.9.33	DATA RECORDING FOR VISUALIZATION . . . . .	731
7.9.34	CLVF ABSORBING MATERIAL (SUPERSONIC OUTLET) . . . . .	733
7.9.35	CLVF CONDITIONS AT INFINITY (INFINITELY HUGE RESERVOIR) . . . . .	734
7.9.36	CLVF IMPOSED PRESSURE (SUBSONIC OUTLET) . . . . .	736
7.9.37	CLVF IMPOSED MASS FLOW RATE (SUPERSONIC INLET) . . . . .	738
7.9.38	CLVF SUBSONIC INLET . . . . .	740
7.9.39	CLVF LODI QUASI 1-D CONDITION . . . . .	742
7.9.40	CLVF FOURIER MODES IN 2D . . . . .	744
7.9.41	CLVF RIEMANN 3-D CONDITION . . . . .	746
7.9.42	CLVF TIME-DEPENDENT PRESSURE . . . . .	748
7.9.43	SAFETY VALVE FOR VFCC (JRC) . . . . .	749
7.10	MECHANISMS . . . . .	758
7.10.1	IMPOSED FORCE . . . . .	760
7.10.2	MOTOR COUPLE . . . . .	761
7.10.3	IMPOSED FORCE AND COUPLE . . . . .	762
7.10.4	DIRECT CURRENT MOTOR . . . . .	763
7.10.5	SPRING . . . . .	764
7.10.6	LIGR . . . . .	766
7.10.7	FUNCTIONS RELATED TO MECHANISMS . . . . .	768
7.10.8	MECHANISM INERTIA . . . . .	769
7.10.9	SERVOMECHANISM . . . . .	770
7.10.10	ELECTRIC PARAMETRES . . . . .	771
7.10.11	REDUCER . . . . .	772
7.10.12	TACHYMETRIC GENERATOR . . . . .	773
7.11	ASSIGNING MATERIALS TO MULTILAYER SHELL ELEMENTS . . . . .	774
7.12	JOINT PROPERTIES . . . . .	776
7.12.1	BUSHING ELEMENT . . . . .	776
<b>8</b>	<b>GROUP D—LINKS</b> . . . . .	<b>781</b>
8.1	LIST OF LINKS . . . . .	781
8.2	LINK CATEGORY . . . . .	785
8.3	OPTIONS FOR COUPLED LINKS . . . . .	786
8.4	OPTIONS FOR "LIAISON" LINKS . . . . .	789
8.5	AUXILIARY FILE . . . . .	793
8.6	BLOCKAGES . . . . .	794
8.7	TIME-LIMITED BLOCKAGES . . . . .	795
8.8	GUIDE (SLIDE CHANNEL) . . . . .	796
8.9	GEOMETRIC BILATERAL RESTRAINTS (CONTACTS) . . . . .	797
8.9.1	PLANE/LINEAR RESTRAINT (CONTACT PLAN) . . . . .	798
8.9.2	SPHERICAL/CIRCULAR RESTRAINT . . . . .	800
8.9.3	CYLINDRICAL RESTRAINT . . . . .	801
8.9.4	CONICAL RESTRAINT . . . . .	802
8.9.5	TOROIDAL RESTRAINT . . . . .	803
8.9.6	PLANE/LINE OF SYMMETRY RESTRAINT . . . . .	805
8.10	IMPOSED CIRCULAR SHAPE . . . . .	807
8.11	RELATIONS . . . . .	808
8.12	ARMATURES . . . . .	811
8.13	CROSSING . . . . .	814

8.14 ACBE: REBAR(FEM)-CONCRETE(DEM) LINK . . . . .	815
8.15 LCAB: LINK BETWEEN PRESTRESSING CABLES AND CONCRETE . . . .	817
8.16 ARLQ: SHELL-3D MESH COUPLING WITHIN ARLEQUIN FRAMEWORK .	818
8.17 IMPOSED MOTIONS . . . . .	819
8.18 CONNECTIONS BETWEEN SHELLS AND SOLID ELEMENTS . . . . .	820
8.19 INTERFACES . . . . .	822
8.20 FLUID-STRUCTURE COUPLING (FLST) . . . . .	824
8.21 FLUID-STRUCTURE COUPLING (FLSR) . . . . .	825
8.22 FLUID/IMMERSED STRUCTURE INTERACTION (FLSX) . . . . .	831
8.23 FLUID-STRUCTURE INTERACTIONS . . . . .	833
8.23.1 FLUID-STRUCTURE CONNECTION (FS) . . . . .	834
8.24 UNILATERAL RESTRAINT [OBSOLETE] . . . . .	835
8.25 IMPACT . . . . .	836
8.26 GAPS ("JEUX") . . . . .	839
8.27 BUTEE: LIMITED DISPLACEMENT . . . . .	841
8.28 SLIDING LINES AND SLIDING SURFACES . . . . .	842
8.29 METHOD OF PARTICLES AND FORCES . . . . .	848
8.30 SMOOTHED PARTICLE HYDRODYNAMICS METHOD (SPH) . . . . .	850
8.31 CONNECTING FINITE AND DISCRETE ELEMENT MODELS . . . . .	852
8.32 BIFURCATION CONNECTION . . . . .	853
8.33 ADHESION CONNECTION . . . . .	854
8.34 TUBM CONNECTION (3D-1D JUNCTION) . . . . .	855
8.35 TUYM CONNECTION (3D-1D JUNCTION) . . . . .	856
8.36 TUYA CONNECTION (3D-1D JUNCTION) . . . . .	857
8.37 RIGID BODY (SOLIDE INDEFORMABLE) CEA Implementation . . . .	858
8.37.1 INERTIA . . . . .	860
8.38 ARTICULATION . . . . .	862
8.38.1 RIGID ARTICULATION (VERR) . . . . .	863
8.38.2 PIVOT . . . . .	864
8.38.3 PIN JOINT (ROTU) . . . . .	866
8.38.4 SLIDER (GLISSIERE) . . . . .	867
8.38.5 SLIDING PIVOT . . . . .	869
8.38.6 IMPOSED RELATIVE DISPLACEMENT (DRIT) . . . . .	871
8.38.7 CONNECTION BETWEEN SHELL AND BEAM (TGGR) . . . . .	873
8.38.8 CONNECTION BETWEEN SHELL AND BEAM (CRGR) . . . . .	874
8.39 ROTATION . . . . .	876
8.40 IMPOSED TIME-DEPENDENT ROTATIONAL MOTION . . . . .	877
8.41 TIME-LIMITED IMPOSED ROTATIONAL MOTION . . . . .	878
8.42 CONSTANT DISTANCE CONNECTION ("DIST") . . . . .	879
8.43 BARYCENTRIC JUNCTION . . . . .	880
8.44 RIGID JUNCTION . . . . .	882
8.45 GLUE - Glueing together two meshes . . . . .	885
8.46 CONTACTS DEFINED BY SPLINE FUNCTIONS . . . . .	887
8.47 COLLISIONS . . . . .	889
8.48 FLUID-STRUCTURE SLIDING OF ALE TYPE (FSA) . . . . .	892
8.49 RIGID-BOUNDARY/FLUID SLIDING OF ALE TYPE . . . . .	894
8.50 IMPACT/CONTACT BY PINBALL MODEL (PINB) . . . . .	895
8.51 CONTACT/IMPACT BY GENERALIZED PINBALL MODEL (GPIN) . . . .	902
8.52 FLUID-STRUCTURE SLIDING BY "FSS" . . . . .	907
8.53 NODE TO SHELL CONNECTOR . . . . .	914
8.54 WEAK FLUID-STRUCTURE COUPLING 2 (FLSW) . . . . .	915

8.55	NODE ON FACET ELEMENT . . . . .	921
8.56	FINITE-ELEMENT/SPECTRAL-ELEMENT INTERFACE . . . . .	923
8.57	NAVIER-STOKES (INCOMPRESSIBILITY) . . . . .	924
8.58	PIPELINE RUPTURE CONNECTION . . . . .	925
8.59	SURFACE PRESSURE MEASURED IN AN ELEMENT (PELM) . . . . .	926
8.60	UNCOUPLED HANGING LINKS . . . . .	928
8.61	PRESCRIBED DAMAGE FOR GRADIENT DAMAGE MATERIALS . . . . .	929
8.62	DRAG FORCES ON 3D BEAMS/BARS EMBEDDED IN A FLUID . . . . .	930
<b>9</b>	<b>GROUP E—FUNCTIONS AND INITIAL CONDITIONS</b>	<b>933</b>
9.1	FUNCTIONS . . . . .	934
9.1.1	TABLE FUNCTION . . . . .	936
9.1.2	SUBROUTINE <i>TABANA</i> . . . . .	937
9.1.3	HARMONIC FUNCTION . . . . .	940
9.1.4	SWEEP SINE FUNCTION . . . . .	942
9.1.5	ABAUQUE : PARAMETRISED TABLE FUNCTION . . . . .	943
9.2	ENERGY INJECTION HISTORY (JRC) . . . . .	944
9.3	INITIAL CONDITIONS . . . . .	946
9.3.1	AUXILIARY FILE . . . . .	947
9.3.2	INITIAL DISPLACEMENTS . . . . .	948
9.3.3	INITIAL VELOCITIES . . . . .	949
9.3.4	INITIAL VELOCITIES FOR RIGID BODIES . . . . .	951
9.3.5	INITIAL VELOCITIES FOR CELL-CENTRED FINITE VOLUMES . . . . .	952
9.3.6	INITIAL CONDITIONS FOR CELL-CENTRED FINITE VOLUMES . . . . .	953
9.3.7	INITIAL STRESSES . . . . .	957
9.3.8	INITIAL TEMPERATURES . . . . .	958
9.3.9	NODAL TEMPERATURES FOR ADVECTION-DIFFUSION (JRC) . . . . .	959
9.3.10	INITIAL ENERGY SUPPLY . . . . .	960
9.3.11	INITIAL BENDING STRESSES DUE TO TEMPERATURE GRADIENT . . . . .	961
9.3.12	GRAVITY LOADING . . . . .	962
9.3.13	PRESCRIBED CONSTANT FLOW RATE . . . . .	965
9.3.14	INITIAL ROTO-TRANSLATIONAL VELOCITY . . . . .	969
9.3.15	INITIALISATIONS FROM A PREVIOUS ALICE FILE . . . . .	972
9.3.16	INITIAL STATUS OF MULTICOMPONENT FLOW (JRC) . . . . .	974
9.3.17	INITIAL TENSOR OF STRESS . . . . .	976
9.3.18	INITIAL TENSOR OF STRAIN . . . . .	978
9.3.19	INITIALISATIONS FROM A MED FILE . . . . .	980
9.3.20	INITIALISATIONS FROM A STATIC ANALYSIS . . . . .	981
9.3.21	PRESCRIBED MASS FLOW RATE . . . . .	983
9.3.22	PRESCRIBED INITIAL EQUILIBRIUM . . . . .	984
9.3.23	PRESCRIBED INITIAL CRACK IN SPHC MODEL . . . . .	986
9.3.24	INITIAL CONDITIONS FOR ADAPTIVITY . . . . .	987
9.3.25	MATERIAL RE-INITIALIZATION WITHOUT ADAPTIVITY . . . . .	992
9.3.26	SKIPPING ELEMENTS . . . . .	993
9.3.27	INITIAL CONDITIONS BY READING MAP FILE (BLAST LOADING) . . . . .	994
9.3.28	INITIAL DAMAGE FOR GRADIENT DAMAGE MATERIALS . . . . .	995
9.3.29	INITIALIZATION FROM A MAP FILE . . . . .	995

<b>10 GROUP F—LOADS</b>	<b>997</b>
10.1 AUXILIARY FILE . . . . .	1001
10.2 CONSTANT LOADS . . . . .	1002
10.3 FACTORIZED LOADS . . . . .	1004
10.3.1 DISPLACEMENT . . . . .	1005
10.3.2 FORCE . . . . .	1006
10.3.3 PRESSURE . . . . .	1007
SEGMENT PRESSURE . . . . .	1008
SHELL PRESSURE . . . . .	1011
FACE PRESSURE . . . . .	1014
NODE PRESSURE . . . . .	1016
PRESSURE ON DISCRETE ELEMENTS . . . . .	1018
10.3.4 ADDITIONAL ACCELERATION . . . . .	1020
10.3.5 TABLE . . . . .	1021
10.4 PROGRAMMED LOADS . . . . .	1022
10.4.1 FORCE . . . . .	1024
10.4.2 CARDS . . . . .	1027
10.4.3 PRESSURE . . . . .	1028
10.4.4 ROUTINE . . . . .	1031
EXAMPLE 1 . . . . .	1038
EXAMPLE 2 . . . . .	1040
10.5 ADVECTION-DIFFUSION "LOADS" (JRC) . . . . .	1042
10.5.1 PRESCRIBED TEMPERATURE . . . . .	1043
10.5.2 PRESCRIBED HEAT FLUX . . . . .	1044
10.5.3 PRESCRIBED HEAT GENERATION . . . . .	1045
10.5.4 PRESCRIBED CONVECTIVE HEAT TRANSFER . . . . .	1046
10.5.5 PRESCRIBED RADIATION HEAT TRANSFER . . . . .	1047
10.5.6 PRESCRIBED TIME-DEPENDENT PRESSURE . . . . .	1048
10.5.7 PRESCRIBED VELOCITIES . . . . .	1049
10.5.8 PRESCRIBED PARALLEL VELOCITIES . . . . .	1050
10.6 SEISMIC-LIKE LOADS FOR USE WITH SPECTRAL ELEMENTS . . . . .	1051
10.6.1 PUNCTUAL SEISMIC LOAD SOURCES . . . . .	1052
10.6.2 PLANE WAVE SEISMIC LOAD SOURCES . . . . .	1055
10.6.3 SEISMIC MOMENT LOADS . . . . .	1057
10.7 NEW CONSTANT LOADS . . . . .	1059
10.8 IMPOSED TIME-DEPENDENT LOADS . . . . .	1061
10.9 DYNALPY LOADS . . . . .	1063
10.10 AIR BLAST (AIRB) LOADING . . . . .	1064
<b>11 GROUP G—PRINTOUT AND STORAGE OF RESULTS</b>	<b>1068</b>
11.1 SELECTIVE PRINTOUTS ("ECRITURE") . . . . .	1069
11.2 PRINTABLE QUANTITIES . . . . .	1072
11.2.1 NODE-RELATED QUANTITIES . . . . .	1072
Coordinates . . . . .	1072
Displacements, velocities, accelerations and forces . . . . .	1072
11.2.2 ELEMENT-RELATED QUANTITIES . . . . .	1072
Stresses and material parameters . . . . .	1072
11.3 STRESSES AND DEFORMATIONS . . . . .	1073
11.3.1 TOTAL DEFORMATIONS . . . . .	1076
11.4 MATERIAL PARAMETERS ("ECROU") . . . . .	1077
11.5 TIME CHOICE (PROCEDURE /CTIME/) FOR THE PRINTOUTS . . . . .	1078

11.6	NODES OR ELEMENTS TO BE PRINTED . . . . .	1079
11.7	RESULT FILES . . . . .	1080
11.8	POST-PROCESSING BY I-DEAS MASTER SERIES . . . . .	1094
11.9	OUTPUT REGIONS . . . . .	1097
11.10	MEASUREMENTS AND MESH QUALITY . . . . .	1101
11.10.1	Mesh Quality assessment . . . . .	1103
11.11	SAVING FILE FOR SUCCESSIVE RESTART . . . . .	1106
<b>12</b>	<b>GROUP H—OPTIONS</b>	<b>1108</b>
12.1	OPTIONS RELATED TO THE TIME-STEP . . . . .	1110
12.2	OPTIONS RELATED TO THE DAMPINGS . . . . .	1115
12.3	OPTIONS FOR FINITE ELEMENTS AND GEOMETRIC ISSUES . . . . .	1118
12.4	OPTIONS FOR FLYING DEBRIS . . . . .	1122
12.5	OUTPUT OPTIONS . . . . .	1123
12.6	RETURNING TO DEFAULT OPTIONS . . . . .	1130
12.7	OPTIONS FOR AN ADVECTION-DIFFUSION COMPUTATION . . . . .	1131
12.8	OPTIONS FOR ALE CALCULATIONS IN STRUCTURES . . . . .	1133
12.9	OPTIONS FOR DEBUGGING . . . . .	1134
12.10	PHANTOM OPTION (Element erosion by time) . . . . .	1137
12.11	CLASS (For a post-treatment with the directive REGION) . . . . .	1138
12.12	SHOCK AND IMPACT OPTIONS . . . . .	1139
12.13	OPTIONS FOR FSA/FSR . . . . .	1140
12.14	OPTIONS FOR NODE-CENTERED FINITE VOLUMES . . . . .	1142
12.15	OPTIONS FOR MULTIPHASE MULTICOMPONENT FLUIDS . . . . .	1143
12.16	OPTIONS FOR AUTOMATIC REZONING IN ALE COMPUTATIONS . . . . .	1145
12.17	OPTIONS FOR CELL-CENTRED FINITE VOLUMES . . . . .	1148
12.18	OPTIONS FOR CONNECTIONS ("LIAISONS"/LINKS) . . . . .	1153
12.19	OPTIONS FOR GRAPHICAL RENDERING . . . . .	1166
12.20	OPTIONS FOR MESH-ADAPTIVE COMPUTATIONS . . . . .	1168
12.21	STRAIN RATE FILTERING OPTION . . . . .	1171
12.22	OPTIONS FOR PARALLEL COMPUTING . . . . .	1172
12.23	OPTIONS FOR GRADIENT DAMAGE MODELS . . . . .	1173
<b>13</b>	<b>GROUP I—TRANSIENT CALCULATION DEFINITION</b>	<b>1175</b>
13.1	STRUCTURES . . . . .	1176
13.2	INTERFACES . . . . .	1182
13.3	XFEM . . . . .	1186
13.4	"CALCUL" DIRECTIVE . . . . .	1190
13.5	ED1D INPUT DECK . . . . .	1194
13.6	PLAY (interactive commands) . . . . .	1195
13.7	QUALIFICATIONS . . . . .	1197
13.8	"SUITE" OR "FIN" . . . . .	1201
<b>14</b>	<b>GROUP ED—POST-TREATMENT BY EUROPLEXUS</b>	<b>1202</b>
14.1	TITLE AND CHOICE OF RESULTS FILE . . . . .	1204
14.2	DIMENSIONING . . . . .	1206
14.3	OUTPUTS . . . . .	1207
14.4	CREATING A REDUCED RESULTS FILE . . . . .	1209
14.5	PRINTOUTS ON THE LISTING . . . . .	1211
14.6	GRAPHIC OUTPUTS . . . . .	1214
14.6.1	Post-processing in adaptivity . . . . .	1217
14.6.2	Curve (Nodal Variables) . . . . .	1218

14.6.3 Curve (Element Variables) . . . . .	1223
14.6.4 Curve (Combinations) . . . . .	1228
14.6.5 Curve (Regional Balances) . . . . .	1234
14.6.6 Curve (Global Quantities) . . . . .	1238
14.6.7 Curve (Quantities from LOG file) . . . . .	1242
14.6.8 Curve in space (Nodal Variables) . . . . .	1244
14.6.9 Curve in space (Element Variables) . . . . .	1248
14.6.10 Curve Read In from a File . . . . .	1253
14.6.11 Curve Defined By The User . . . . .	1256
14.6.12 Set of Pochhammer-Chree curves . . . . .	1259
14.6.13 Drawings (TRACE) . . . . .	1262
14.6.14 Output on file (XMGR) . . . . .	1267
14.6.15 Output on file (K2000) . . . . .	1269
14.6.16 Output on file (LIST) . . . . .	1271
14.6.17 Find value on a curve (FVAL) . . . . .	1273
14.7 VISUALIZATIONS . . . . .	1274
<b>15 GROUP O—INTERACTIVE COMMANDS</b>	<b>1276</b>
15.1 Primary interactive commands . . . . .	1277
15.2 Keystrokes and mouse events in the OpenGL graphical visualizer . . . . .	1285
15.3 CALCUL options . . . . .	1287
15.4 TRACE options . . . . .	1288
15.5 AVI file generation from a sequence of bitmaps (MAVI) . . . . .	1303
15.6 GOTRAC: a simple looping mechanism . . . . .	1305
15.7 CAMERA parameters and options . . . . .	1306
15.8 SLERP parameters and options . . . . .	1308
15.9 SCENE options . . . . .	1310
15.9.1 Objects Menu Parameters . . . . .	1313
15.9.2 Geometry Menu Parameters . . . . .	1315
15.9.3 Vectors Menu Parameters . . . . .	1324
15.9.4 Choice of a vector field . . . . .	1326
15.9.5 Iso Menu Parameters . . . . .	1327
15.9.6 Choice of an iso field . . . . .	1330
15.9.7 Text Menu Parameters . . . . .	1335
15.9.8 Colors Menu Parameters . . . . .	1336
15.9.9 Lights and Materials Menu Parameters . . . . .	1338
15.9.10 POV-Ray Menu Parameters . . . . .	1340
Defining POV-Ray lights . . . . .	1342
Defining POV-Ray textures . . . . .	1343
15.10 TITLES options . . . . .	1346
<b>16 GROUP V—The built-in OpenGL Graphical Visualizer</b>	<b>1347</b>
16.1 Preparing to use the built-in graphical visualizer . . . . .	1348
16.2 Interactive code execution . . . . .	1349
16.3 Keyboard commands . . . . .	1350
16.4 Mouse-driven motions . . . . .	1351
16.5 The Main menu . . . . .	1352
16.6 The Objects menu . . . . .	1354
16.6.1 The Draw hidden as menu . . . . .	1355
16.7 The Geometry menu . . . . .	1357
16.7.1 The Navigation menu . . . . .	1358

16.7.2	The Near plane tolerance menu . . . . .	1360
16.7.3	The Projection menu . . . . .	1360
16.7.4	The References menu . . . . .	1360
16.7.5	The Faces menu . . . . .	1361
16.7.6	The Lines menu . . . . .	1361
16.7.7	The Points menu . . . . .	1362
16.7.8	The Shrinkage menu . . . . .	1363
16.7.9	The Pinballs menu . . . . .	1364
16.7.10	The Initial geometry menu . . . . .	1365
16.7.11	The Flying debris menu . . . . .	1366
16.7.12	The FLSR domains menu . . . . .	1366
16.7.13	The FLSW domains menu . . . . .	1367
16.7.14	The Gpinballs menu . . . . .	1368
16.7.15	The Links (coupled) menu . . . . .	1370
16.8	The Vectors menu . . . . .	1372
16.8.1	The Scale menu . . . . .	1373
16.8.2	The Length menu . . . . .	1373
16.8.3	The Color scheme menu . . . . .	1374
16.8.4	The Options menu . . . . .	1374
16.9	The Isovalues menu . . . . .	1375
16.9.1	The Fill elements representations menu . . . . .	1376
16.9.2	The Scale menu . . . . .	1377
16.9.3	The Color scheme menu . . . . .	1378
16.10	The Text menu . . . . .	1379
16.11	The Colors menu . . . . .	1380
16.11.1	The 1) Select color menu . . . . .	1380
16.11.2	The 2) Apply it to menu . . . . .	1381
16.12	The Lights/Mats menu . . . . .	1383
16.12.1	The Light X menu . . . . .	1384
16.12.2	The Light Y menu . . . . .	1384
16.12.3	The Light Z menu . . . . .	1384
16.12.4	The Light ambient menu . . . . .	1385
16.12.5	The Light diffuse menu . . . . .	1385
16.12.6	The Light specular menu . . . . .	1385
16.12.7	The Light shininess menu . . . . .	1385
16.12.8	The Light model ambient menu . . . . .	1386
16.12.9	The 1) Select material menu . . . . .	1386
16.12.10	The 2) Apply it to menu . . . . .	1386
16.13	The Win/Copy menu . . . . .	1388
16.13.1	The Bitmap format menu . . . . .	1389
<b>17</b>	<b>GROUP SR—SAVING AND RESTART</b>	<b>1390</b>
17.1	SAVING . . . . .	1390
17.1.1	DEFINING SAVING IN THE INPUT CODE . . . . .	1390
17.1.2	SAVING VIA COMMAND FILE . . . . .	1390
17.2	RESTART . . . . .	1391
17.3	DIRECTIVE "SAUVE" (OBSOLETE FORM) . . . . .	1393
17.4	DIRECTIVE "REPRISE" . . . . .	1396
17.5	DATA NECESSARY FOR RESTARTS . . . . .	1398

<b>18 GROUP RM—CHANGE OF TOPOLOGY (“REMAILAGE”)</b>	<b>1400</b>
18.1 AUTOMATIC REMESHING . . . . .	1401
18.2 MANUAL REMESHING . . . . .	1403
18.3 CHECKING THE REMESHING . . . . .	1405
<b>19 GROUP EX—EXAMPLES</b>	<b>1406</b>
19.1 BENDING OF A BEAM . . . . .	1407
19.2 IMPACT ON A CIRCULAR PLATE . . . . .	1410
19.3 EXPLOSION IN A TANK . . . . .	1413
19.4 MODELLING OF PERFORATED PLATES . . . . .	1417
<b>20 DEVELOPMENT NOTES</b>	<b>1420</b>
20.1 PROGRAMMING GUIDELINES . . . . .	1420
20.1.1 Programming Language . . . . .	1420
20.1.2 F77 Programming . . . . .	1420
20.1.3 F90 Programming . . . . .	1420
20.1.4 Code Profiling . . . . .	1420
20.2 PRECOMPILER KEYWORDS . . . . .	1423
20.3 MANUAL CREATION . . . . .	1424
20.3.1 PAGE NUMBERING CONVENTIONS . . . . .	1424
20.4 ACKNOWLEDGEMENTS, LICENSES . . . . .	1425
20.4.1 lib_vtk_io . . . . .	1425
20.4.2 Blas, Lapack . . . . .	1425
20.4.3 OpenMPI . . . . .	1426
20.4.4 Glut . . . . .	1427
20.4.5 f90gl, f90glu, f90glut . . . . .	1427
<b>21 BIBLIOGRAPHY</b>	<b>1428</b>
Essential EURDYN Bibliography (before 1986) . . . . .	1429
Plexis-3C/EUROPLEXUS Bibliography . . . . .	1432
1986 . . . . .	1432
1987 . . . . .	1432
1988 . . . . .	1433
1989 . . . . .	1433
1990–1991 . . . . .	1433
1992 . . . . .	1434
1993 . . . . .	1434
1994 . . . . .	1435
1995 . . . . .	1436
1996 . . . . .	1437
1997 . . . . .	1437
1998 . . . . .	1439
1999 . . . . .	1440
2000 . . . . .	1441
2001 . . . . .	1441
2002 . . . . .	1442
2003 . . . . .	1442
2004 . . . . .	1444
2005 . . . . .	1444
2006 . . . . .	1445
2007 . . . . .	1445
2008 . . . . .	1446



2009	1447
2010	1449
2011	1450
2012	1452
2013	1453
2014	1453
2015	1454
2016	1455
2017	1457
2018	1459
2019	1459
2020	1461
Castem-Plexus/EUROPLEXUS Bibliography	1462
1978	1462
1979	1462
1980	1462
1981	1462
1982	1463
1983	1464
1984	1465
1985	1466
1986	1468
1987	1469
1988	1470
1989	1471
1990	1472
1991	1473
1992	1474
1993	1475
1994	1476
1995	1477
1996	1478
1997	1479
1998	1481
1999	1482
2000	1483
2001	1486
2002	1488
2003	1489
2004	1489
2005	1490
2006	1490
2007	1490
2008	1491
2009	1492
2010	1492
2011	1493
2012	1494
2013	1495
2014	1496
2015	1497

2016 . . . . .	1497
2018 . . . . .	1498
2019 . . . . .	1498
Samtech Plexus/EUROPLEXUS Bibliography . . . . .	1499
1999 . . . . .	1499
2000 . . . . .	1499
2001 . . . . .	1499
2003 . . . . .	1499
2009 . . . . .	1499
EDF CASTEM-PLEXUS/EUROPLEXUS Bibliography . . . . .	1500
1982 . . . . .	1500
1983 . . . . .	1500
1984 . . . . .	1500
2004 . . . . .	1500
2005 . . . . .	1500
2006 . . . . .	1500
2007 . . . . .	1501
2008 . . . . .	1501
2009 . . . . .	1501
2010 . . . . .	1502
2011 . . . . .	1502
2012 . . . . .	1502
2013 . . . . .	1502
2015 . . . . .	1503
2016 . . . . .	1503
ONERA EUROPLEXUS Bibliography . . . . .	1505
1997 . . . . .	1505
2014 . . . . .	1505
2015 . . . . .	1505
2016 . . . . .	1505
2017 . . . . .	1505
2018 . . . . .	1505
<b>22 Keywords Index</b>	<b>1506</b>

# 1 GETTING STARTED

## 1.1 ABOUT EUROPLEXUS

EUROPLEXUS is a computer code being jointly developed since 1999 by CEA (CEN Saclay, DMT) and EC (JRC Ispra, SS&M) under a collaboration contract. It stems from CEA's CASTEM-PLEXUS (a program belonging to the CASTEM system) and the previous CEA-EC joint product PLEXIS-3C.

The code analyses 1-D, 2-D or 3-D domains composed of solids (continua, shells or beams) and fluids. Fluid-structure interaction is also taken into account.

The program uses an explicit algorithm (central-difference) for the discretization in time and therefore it is best adapted to rapid dynamic phenomena (fast transient dynamics) such as explosions, impacts, crashes etc. Geometric non linearity (large displacements, large rotations, large strains), and the non-linearity of materials (plasticity, viscoplasticity, etc) are fully taken into account.

The spatial discretization is mainly based on the Finite Element or Finite Volume method. Other formulations such as SPH (Smoothed Particle Hydrodynamics), Spectral ELements, Diffuse Elements etc. are also available or under development. Numerous element types and a comprehensive library of material types for solids, fluid and special media (e.g. impedances) are available.

Three main descriptions are available in the code: the Lagrangian description which is well suited for the structural domain, the Eulerian description useful for purely fluid problems, and the Arbitrary Lagrangian Eulerian (ALE) description which is typically used in fluid-structure interaction problems.

EUROPLEXUS is interfaced to various pre- and post-processing programs that enable the meshing of the studied domain (e.g. CEA's Cast3m) and the visualization of the results (e.g. Cast3m, ParaView or EUROPLEXUS itself).

Different types of licenses are available of EUROPLEXUS. A limited version of the code can be downloaded. For research and education these licenses are mainly for free. Details can be found on the web page of EUROPLEXUS (<http://www-epx.cea.fr/>). This User's manual is updated daily and can be downloaded from <http://europlexus.jrc.ec.europa.eu/>.

A large bibliography concerning EUROPLEXUS as well as its ancestors is provided at the end of the present manual (see Section BIB). Many of the cited documents are available to EUROPLEXUS developers in electronic form on the EUROPLEXUS Consortium web site (<https://europlexus.jrc.ec.europa.eu/>).

## 1.2 Installation

### 1.2.1 License types

Different types of licenses are available of EUROPLEXUS. A limited version of the code can be downloaded for free. For research and education a full version can be obtained for free. Details can be found on the web page of EUROPLEXUS (<http://www-epx.cea.fr/>). This User's manual is updated daily (development version) and can be downloaded from <http://europlexus.jrc.ec.europa.eu/>.

### 1.2.2 Installation under Windows

to be written

### 1.2.3 Installation under Linux

to be written

### 1.2.4 Manual

The getting started manual is part of the EPX manual containing all information of the code. It is divided in main sections (GROUPS). In general an EPX input file can be created by following all groups and taking the needed commands.

A large bibliography concerning EUROPLEXUS as well as its ancestors is provided at the end of the present manual (see **GBIBB**). Many of the cited documents are available to EPX developers in electronic form on the EPX Consortium web site (<https://europlexus.jrc.ec.europa.eu/>).

### 1.2.5 Benchmarks

to be written

### 1.3 Data flow

The first step in learning an FE tool is to understand its particular data flow. Figure 1 presents a general description of the procedure how to perform an EPX calculation.



Figure 1: General data flow in EPX

The mesh creation concerns the creation of nodes and elements. The meshes were mainly stored in external files. The next step is to define materials, loadings and calculation parameters in an EPX input file. The calculation of the inputs is done normally via the command line tool. The result files can be then assessed by several post-processing tools. The detailed data flow is shown in 2.

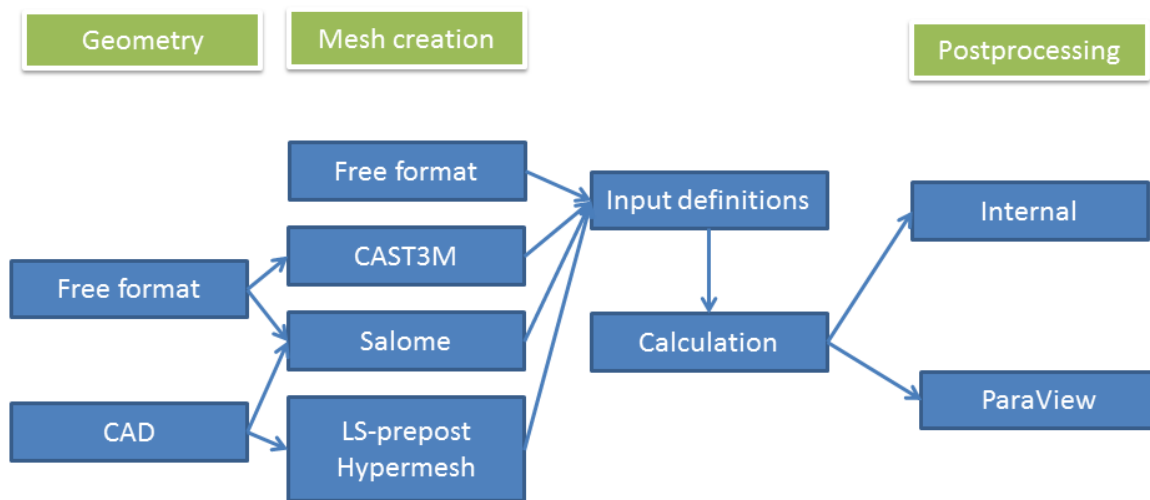


Figure 2: Detailed data flow in EPX

#### 1.3.1 Files in EPX

extension	description	ref
epx	input file	<a href="#">GBINT_0018</a>
k	Mesh file: k file format from ls.dyna	<a href="#">GBINT_0016</a>
listing	Listing output	
log	Log file	<a href="#">GBH_0020</a>

med	Mesh file: salome	GBINT_0016
msh	Mesh file: Cast3m	GBINT_0016
pvd	ParaView time step file	GBG_0070
std	Standard outputs (error messages)	
vtu	ParaView result file	GBG_0070

---

## 1.4 Mesh generation

One of the fundamental steps to perform a FE/FV calculation is to create the meshes. There are several possibilities in EUROPLEXUS to perform that task.

- **Free format:** nodes and elements are written in the EPX input file in a specific format. This is very effective for very small models but not feasible for bigger models. The advantage is that the mesh is included in the input file and not separated. It is therefore used for many benchmarks. There is no known FE mesher that can produce this mesh format. Further information about the format are given in [GBB\\_0020](#).
- **k-file (LS-DYNA):** This is a text file input where nodes, elements and sets of them are defined based on a given format. The advantage of that format is that it can be produced from different graphical mesh generators like LS-Prepost (freeware) or Hypermesh.
- **med (SALOME):** Med files are files produced by the open source tool SALOME. With SALOME also full EPX inputs can be created.
- **CAST3M:** CAST3M is a FEM software from CEA that is freely available. It can produce msh files that can be used very efficiently in EUROPLEXUS. The meshes were created by defining points, lines, surfaces etc. in a script language. That is very powerful but also difficult to learn.

### 1.4.1 k-file (LS-DYNA)

The format of the input is described in the LS-DYNA manuals. It is very simple and can therefore also be written by scripts.

The following not exhaustive list of tools can create k-files:

- **LS-prepost:** Free graphical tool from LS-DYNA (<http://www.lstc.com/lsp/>). Some support to create FE meshes from CAD files.

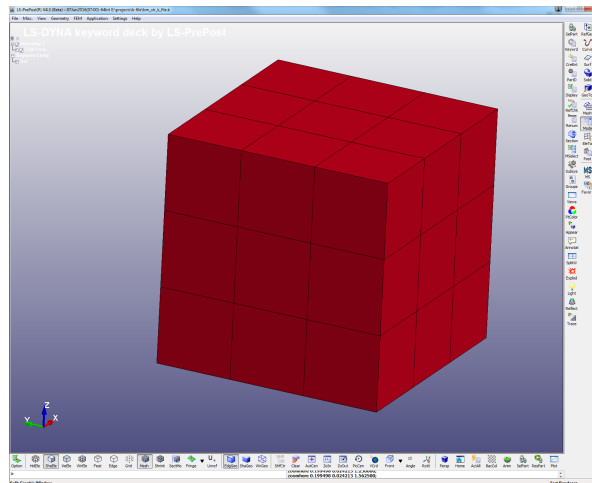


Figure 3: LS-Prepost mesh generation

- **HyperMesh:** Very big graphical tool from Altair (<http://www.altairhyperworks.it/product/HyperMesh>). Can efficiently be used for the conversion of CAD geometries to FE meshes.
- **ANSYS**

### 1.4.2 med (SALOME)

SALOME is an open-source software that provides a generic platform for Pre- and Post-Processing for numerical simulation. It is based on an open and flexible architecture made of reusable components. The software can be downloaded on the webpage: <http://www.salome-platform.org/>.

### 1.4.3 CAST3M

The CAST3M .msh-file format is the mesh file format with the widest support in EUROPLEXUS. Nevertheless, the mesh creation can only be done with the FE software tool CAST3M. This tool is quite powerful since the mesh generation can easily be parametrised and automatized. But it needs additional effort to be learned. For an introduction in the CAST3m methodology it is referred to the CST3M webpage (<http://www-cast3m.cea.fr/>)



## 1.5 EUROPLEXUS Inputs

### 1.5.1 First input sample

Let's start with an very easy EUROPLEXUS example. It is recommend to start with such a very easy calculation in order to test the installation.

```

1      impact0                                !title of the problem
2      ECHO                                    !Output on the console
3      KFIL                                    !mesh file definition
4      TRID LAGR                               !3d structural calculation
5      GEOM Q4GS PART 1 TERM                   !element definition
6      COMP EPAI 2 LECT PART 1 TERM            !Thickness
8      MATE LINE RO 7800 YOUN 2.E11 NU 0.3     !Material definition: linear
9              LECT PART 1 TERM               !for part 1
10     LINK COUP                               !Links (coupled)
11         BLOQ 123 LECT NSET 1 NSET 2 TERM    !Boundaries
12     INIT VITE 3 -110 LECT PART 1 TERM        !Initial conditions
13     ECRI FICH PVTK TFRE 1.0E-3              !Output as ParaView
14             VARI DEPL                       !Output variable displacement
15     OPTI NOTE LOG 1                         !Log file written each step
16         CSTA 0.5                           !Stability step
17     CALC TINI 0.0 TFIN 100.E-3              !Start and end time of calculation
18     FIN
```

1. The first line contains the title of the calculation. It is important to give that title. Otherwise, the first input line will be taken as the title.
2. ECHO indicated that the output will be written on the command line and not only to the listing.
3. KFIL identifies that a k-file in LS-DYNA format will be read. The name of the file can be added after the command included in ' '. If the name is not given, the default will be chosen as the name of the epx input file with the extension .k.
4. TRID identifies a three-dimensional calculation and LAGR a purely structural one.
5. The elements that were read via the mesh file must be attached to element types. That is done with the command GEOM. A list of all element types is given in [GBINT\\_0080](#). Some general elements are listed in xxx. The structure of the element type allocation is that first, the element type is given and second the elements are chosen. Here, the PART 1 from the k-file is taken. The keywords depend on the mesh file type used. Here a shell element of type Q4GS is chosen. The command must be closed with TERM as soon as all elements are defined.
6. Depending on the element type several additional definitions can be given with COMP. All available commands in this section are described in [GBC\\_0010](#). Here, the thickness of 2 is set to all elements of PART 1 with the command EPAI. The classical structure of reading elements or nodes is to use LECT xxx TERM. This procedure is described more in detail in [GBINT\\_0050](#).
7. This line contains the material definition. The complete list of all materials is collected on page [GBC\\_0100](#). A linear material is defined.
8. Selection of the elements for the given material

9. The links (e.g. boundary conditions) are defined here with LINK ([GBD\\_0010](#)). A coupled approach (Lagrange Multiplier) is chosen (COUP).
10. Boundary conditions are given here with BLOQ (blockages). The number afterwards identifies that all three directions are blocked. The nodes concerned are again taken with LECT. Here two NSETs were chosen.
11. Initial conditions were given in that line as an initial velocity for PART 1. Initial conditions were described more in detail on page [GBE\\_0040](#).
12. The outputs were defined with ECRI (see [GBG\\_0010](#)). FICH indicates that the outputs were written in a separate file and not in the listing. PVTk means the ParaView output files. As default these files were binary. ASCII files can be written by adding FORM. With TFRE, the frequency of the output steps is given (see [GBINT\\_0057](#)).
13. With VARI (in case of PVTk) the output fields can be defined (here displacement - DEPL).
14. Some options can be given with OPTI ([GBH\\_0010](#)). LOG 1 indicates that the log file is written per step. This should not be done in case of MPI calculations. NOTE DFEINS the output of the energy check.
15. CSTA the stability step. A value of 0.5 indicates that the calculated stability step will be multiplied by 0.5 for safety.
16. The calculation is started with CALC ([GBI\\_0020](#)). TINI (initial time) and TFIN (end time) must be given.
17. The input file must be closed with FIN.

### 1.5.2 Elements

Several elements are available for EUROPLEXUS. Its full list is given on [GBINT\\_0080](#). It is very important to know the history behind the elements. These were created in the past either by JRC or CEA and its formulation in the back can be totally different. This means also that not all elements are available for all materials. A table with all possible material-element combinations is given in ([GBC\\_0100](#)) In this general introduction, the main elements for structural and fluid calculations were presented. Details about further elements (like SPH or diffuse elements) can be taken from the list.

The following structural elements are recommended. They vary depending their mechanical assumptions. Further details are given in the description of the elements.

1. Solid elements: [CUBE/CUB8](#), [PRIS](#), [TETR](#). CUBE is a cubic element with reduced integration while
2. Beams: POUT for beams. The cross section information can be given with EPAI.
3. Triangular shells: [T3GS](#), [DST3](#), [DKT3](#)
4. Quadrilateral shells: [Q4G4](#), [Q4GS](#), [Q4GR](#)
5. Loading surface elements: [CL3T](#), [CL3D](#)
6. Material points: [PMAT](#), [DEBR](#)

Fluid calculations can be done by using finite elements or finite volumes. The accuracy of finite elements is not very high. Therefore, only finite volumes are recommended to use for fluid or fluid structure interaction calculations. The following finite volumes can be taken: **CUVF**, **PRVF**, **TEVF**, **PYVF**. Further information about fluid calculations by using finite volumes are given here:

### 1.5.3 Sandwich Elements

The code also allows to use layered elements for beams and in particular for shell elements. This means that the materials of the integration points through the thickness could be different. More information are given in **GBINT\_0110**. As an example, the benchmark **bm\_str\_lsgl01** could be used.

### 1.5.4 Materials

Several material laws are implemented in EUROPLEXUS. The full list of materials can be found on **GBC\_0100**. Not all elements accept all material types. **GBC\_0100** shows also the possible material-element combinations.

The table below presents some materials that have a general use.

number	name	ref	law of behaviour
74	ABSE		
21	CLVF	<b>7.9.34</b>	Boundary conditions for finite volumes
109	DADC	<b>7.7.17</b>	Dynamic Anisotropic Damage Concrete
111	DPDC	<b>7.7.18</b>	dynamic plastic damage concrete
87	DPSF	<b>7.7.43</b>	Drucker Prager with softening and viscoplastic regularization
83	DRPR	<b>7.7.52</b>	Drucker Prager Ispra model
12	DRUC	<b>7.7.6</b>	Drucker-Prager
19	DYNA	<b>7.7.7</b>	dynamic Von Mises isotropic rate-dependent
17	FANT	<b>7.7.31</b>	phantom: ignore the associated elements
9	GAZP	<b>7.8.4</b>	perfect gas
118	GGAS	<b>7.8.1</b>	generic ideal gas material
116	GLIN	<b>7.7.3</b>	generic linear material
117	GPLA	<b>7.7.4</b>	generic plastic material
48	GVDW	<b>7.8.28</b>	Van Der Waals gas
40	GZPV	<b>7.8.24</b>	perfect gas for Van Leer
95	HYPE	<b>7.7.55</b>	hyperelastic material (Model of Mooney-Rivlin, Hart-Smith and Ogden)
4	ISOT	<b>7.7.7</b>	isotropic Von Mises
108	JCLM	<b>7.7.66</b>	Johnson-Cook with Damage Lemaitre-Chaboche for SPHC
50	JWL	<b>7.8.21</b>	explosion (Jones-Wilkins-Lee model)
66	JWLS	<b>7.8.29</b>	Explosion (Jones-Wilkins-Lee for solids)
72	LEM1	<b>7.7.10</b>	Von Mises isotropic coupled with damage (type Lemaitre)
1	LINE	<b>7.7.1</b>	linear elasticity
23	LIQU	<b>7.8.14</b>	incompressible (or quasi-) fluid
70	LMC2	<b>7.7.12</b>	Von Mises isotropic coupled with damage (Lemaitre) with strain-rate sensitivity
26	MASS	<b>7.7.30</b>	mass of a material point
85	MAZA	<b>7.7.16</b>	Mazars-linear elastic law with damage
2	PARF	<b>7.7.7</b>	perfectly plastic Von Mises
125	RIGI	<b>7.7.68</b>	Rigid material (for rigid bodies)

---

99	SLZA	7.7.57	Steinberg-Lund-Zerilli-Armstrong
35	VM23	7.7.39	Von Mises elasto-plastic radial return
2/4/5/19	VMIS	7.7.7	Von Mises materials
76	VMJC	7.7.48	Johnson-Cook
78	VMLP	7.7.49	Ludwig-Prandtl
79	VMLU	7.7.50	Ludwik
84	VMSF	7.7.42	Von Mises with softening and viscoplastic regularization
77	VMZA	7.7.51	Zerilli-Armstrong
120	VPJC	7.7.67	visco-plastic Johnson-Cook
67	ZALM	7.7.11	Zerilli-Armstrong with damage Lemaitre-Chaboche

---

### 1.5.5 Element erosion

Element erosion means that elements are deleted from the table of elements and were not treated any more. This is a particular procedure in explicit codes since the general energy balance is violated by eroding elements. There are several reasons why element erosion could be indicated:

1. The material has reached a failure mode (damage or other criteria)
2. The element became so distorted that it cannot be treated any more (CROI)
3. The time step size of the element is too small (CALC TFAI)
4. Parts of the model should be removed at a certain time (e.g. certain elements have to become “fantom”, i.e. have to be eroded, at a chosen time [GBH\\_0100](#)), or due to further criteria (e.g. displacement-driven erosion [GBC\\_0067](#)).
5. The user decides to erode some elements interactively at the current time reached by the calculation [GBO\\_0010](#).

The objective in most of the cases is that the calculation is not stopped due to critical behaviour of material or elements.

In all cases, the keyword EROS (see [GBA\\_0030](#)) must be added in the beginning (before DIME and GEOM). This keyword is followed by CROI as soon as element erosion for distorted elements is needed. Erosion due to too small time steps sizes can be activated in the CALC PART by adding TFAI (see [GBI\\_0020](#)).

In case of failure erosion, the ratio between failed and total gauss point in an element can be given. This parameter must be written immediately after EROS. The global value for the material failure element erosion can be overwritten for parts of the elements by COMP EROS (see [GBC\\_0069](#)).

### 1.5.6 Fluid calculations

### 1.5.7 INTRODUCTION TO FLUID-STRUCTURE INTERACTION

EUROPLEXUS offers a rich variety of models for Fluid-Structure Interactions (FSI). The following is a short introduction to FSI and a tentative classification of the models available in the code, in order to guide the user in the choice of the most appropriate FSI models for the applications of interest. For a more detailed overview of the available FSI models see e.g. [\[303\]](#).

Fluid-structure interaction (FSI) phenomena play an important role in many areas, ranging from aeronautical and space applications, to civil and marine/offshore engineering and to the transport industry, to name just a few. The EUROPLEXUS development team has been involved for many years in the development of numerical methods for FSI modeling applied

to safety studies—initially for the nuclear industry and more recently for conventional power plants (electrical machinery)—to civil engineering (vulnerability of buildings and other critical infrastructures to terrorist attacks) and to land mass transports (blast effects in railway stations, metro lines, rolling stock).

All these studies are characterized by the violent blast loading, resulting either from an accident or from an intentional attack, and by the very short time scale (fast transient dynamics). Strong pressure waves propagate in the fluid and load the surrounding structures, which typically undergo large deformations and in some cases reach complete failure and fragmentation.

For this class of problems, an explicit time marching algorithm is usually adopted, where the fluid is modelled as compressible and inviscid (Euler equations). An Arbitrary Lagrangian Eulerian (ALE) formulation is adopted for the fluid sub-domain, while the structure is Lagrangian.

Three different discretization approaches are available in the code for the fluid sub-domain: finite elements (FE), node-centred finite volumes (NCFV) and cell-centred finite volumes (CCFV):

- In the FE case, kinematic variables (such as the velocity  $v$ ) are discretized at the element nodes, while state variables (such as the fluid pressure  $p$ ) are discretized at Gauss points, typically located at the element centroid.
- In the NCFV case, a virtual FV (dual) mesh centred on fluid nodes is automatically built up starting from the FE-like (primal) mesh provided in input, and all variables are discretized at the nodes.
- In the CCFV case, the FV mesh looks similar to the FE case, but all variables are discretized at the volume centres. Note that in this case the ‘nodes’ carry no relevant information other than their position, used to compute the volume.

The coupling between the fluid (ALE) and the structure (Lagrangian) is realized by suitable FSI algorithms. Two broad classes of algorithms are available in the code. The first class uses a **strong** approach, based on constraints imposed on the (velocity of) fluid and structure nodes at the F-S interface. The second class uses a **weak** approach, based on direct application of fluid pressure forces to the structure. This terminology (strong/weak) is tentatively adopted here in an attempt to characterize the different nature of the two approaches, but it should not be confused with other uses of the same terms in the literature, in particular with weak (i.e. integral) forms in continuum mechanics. Traditionally, strong FSI algorithms are mainly used in FE, while weak FSI algorithms are mainly used in FV.

Yet another classification of FSI algorithms concerns the degree of deformation/damage that the structure can undergo (and thus the type of application). One class of **basic** algorithms is suitable for large motion and large deformation of structures, but only provided these do not fail. Another class of algorithms can go up to complete failure, and fragmentation, of the loaded structures. Finally, FSI algorithms can be classified in three types according to spatial discretization: (nodally) conforming, non-conforming, and embedded (or immersed). The first two types are mostly used in applications without structural failure (but there are exceptions), while embedded algorithms are the only ones capable of dealing with extreme loading cases where the structure fails and breaks up in pieces.

The following Table summarizes the architecture of a typical FSI model, consisting of a *detection* module and of an *enforcement* module. The various types of approaches (*basic* / *embedded* or *strong* / *weak*) are briefly summarized.

The following Table completes the classification of the available FSI models, by showing the type of spatial discretization (*conforming*, *non-conforming* or *embedded*), the name of the input directive (when applicable/needed), and the associated fluid discretization(s).

### 1.5.8 Restart

Table 3: A classification of FSI algorithms

FSI Algorithm	<b>FSI Detection</b>	<b>Basic</b>	No structural failure, moderate rotations.
		<b>Embedded</b>	Structure can fail, arbitrary rotations.
	<b>FSI Enforcement</b>	<b>Strong</b>	Constraints on $F$ and $S$ velocities are imposed, e.g. by Lagrange multipliers (implicit).
		<b>Weak</b>	Pressure forces are transmitted from the fluid to the structure; structure motion provides weak feedback on fluid ( $S = \text{master} / F = \text{slave}$ ).

Table 4: The available FSI algorithms

	<b>Detection Strategy</b>	<b>Spatial Discretization</b>	<b>Enforcement Strategy</b>	<b>Name / Command</b>	<b>Use with</b>
FSI Algorithm	<b>Basic</b> (no structural failure)	<b>Conforming</b> $F$ - $S$ meshes	<b>Strong</b>	FSA	FE, NCFV
			<b>Weak</b>	Merge $F$ - $S$ nodes	CCFV
		<b>Non- conforming</b> $F$ - $S$ meshes	<b>Strong</b>	FSA	FE, NCFV
			<b>Weak</b>	Declare non-matching $F$ -nodes	CCFV
	<b>Embedded</b> (structure can fail)	$S$ -mesh is <b>immersed</b> in the $F$ -mesh	<b>Strong</b>	FLSR	FE, NCFV
			<b>Weak</b>	FLSW	CCFV

### 1.5.9 Index of important commands

This is a non exhaustive list of important commands that may be useful to understand the basic working of EUROPLEXUS. Additional lists are given for the materials [GBC\\_0100](#) and elements [GBINT\\_0080](#).

command	main group	description	ref
BLOQ	LINK	Boundary conditions	<a href="#">GBD_0030</a>
CALC		Calculation definitions	<a href="#">GBI_0020</a>
COMP		Geometric complements	<a href="#">GBC_0010</a>
COUP	LINK	Treatment of the links as coupled (Lagrange multipliers)	<a href="#">GBD_0010</a>
CSTA	OPTI	Time step safety coefficient	<a href="#">GBH_0020</a>
DECO	LINK	Treatment of the links as decoupled	<a href="#">GBD_0010</a>
ECHO	—	Output on the console	<a href="#">GBA_0020</a>
ECRI		Output of the results	<a href="#">GBG_0010</a>
EPAI	COMP	Thickness (e.g. of shell elements)	<a href="#">GBC_0040</a>
GEOM		Mesh and grid motion	<a href="#">GBB_0010</a>
KFIL	—	mesh file definition (k-file)	<a href="#">GBA_0030</a>
INIT		Initial conditions	<a href="#">GBE_0040</a>
LAGR	—	Structural calculation	<a href="#">GBA_0030</a>
LINK		Links	<a href="#">GBD_0010</a>

---

LOG	OPTI	Log file .log is created	GBH_0020
MATE		Material definition	GBC_0100
NOTE	OPTI	No energy check printed per each step	GBH_0020
OPTI		Definition of options	GBH_0010
PVTK	ECRI	Output as ParaView	GBG_0070
TFIN	CALC	End time of calculation	GBI_0020
TINI	CALC	Start time of calculation	GBI_0020
TRID	—	3D calculation	GBA_0030

---

## 1. INT.70

## 1.6 Outputs/Post processing

1. Writing outputs in EPX in files
2. Internal postpro
3. Interactive work
4. ParaView



## 2 PREPARATION OF THE INPUT DATA

### Rule 1:

All the data may be coded in FREE format (unless a fixed format is explicitly chosen by the user for specific data, e.g. the geometry). One or more blanks and/or an end of line act as a separator between data items.

Only columns 1 to 72 of each 'data card' are analysed. A data card is a record, i.e. a sequence of up to 72 characters terminated by a new line character.

### Rule 2:

The data consist in a sequence of instructions or DIRECTIVES, (each one possibly including a number of OPTIONS and SUBOPTIONS) that can be specified using keywords.

Only the FIRST FOUR LETTERS ARE COMPULSORY for the coding of keywords, since these are unique in each situation. EUROPLEXUS ignores characters beyond the fourth one when decoding a keyword.

### Rule 3:

Numeric values are coded with or without a decimal point:

Example:

12 24. .3 1.3E-4 1E4 .5D+2 -.4E 00

### Rule 4:

Comment 'cards' (records, in the sense defined above) can be freely interspersed with the data, except that the first record of a data file cannot be a comment (it is the problem title, see below).

A comment card begins with a dollar (\$) or an asterisk (\*) in column 1. Alternatively, it is possible to use an exclamation point (!) which may be placed at any position in the 'card' and not only in column 1.

Any characters that follow one of these special characters on an input card are ignored by EUROPLEXUS. By using an exclamation point, it is therefore possible to add comments on the same line as the data.

Note that the program does include in the input data echo comments written at the beginning of the output listing.

A semicolon is considered as the end of a card.

Example:

"TRID" "ALE" ! 3D ALE computation

## 2.1 WRITING CONVENTIONS

### Object:

To ensure that the necessary keywords, optional key-words as well as the numerical values to be entered show up clearly in the syntax of the instructions, the following conventions have been adopted.

### Description of language conventions:

Keywords are enclosed in quotation marks: "TRIDIM".

Anything not enclosed in quotation marks represents numerical values or something else to be described. Names representing character strings are enclosed in apostrophes:

```
< "SAUVE"  nb  ifreq  < "PROT"  'mykey'  > >
```

When a sequence is optional, it is enclosed in angle brackets: (<...>).

If there is a choice between several sequences among which ONE ONLY is compulsory, the sequences are enclosed between ( |[...]| ) symbols, each of them separated by semicolons (;). The list of sequences can be written on several lines, for example:

```
|[ "DEPL" ; "VITE" ; "ACCE" ]|
```

or

```
|[
  "DEPL" ;
  "VITE" ;
  "ACCE"
]|
```

If there is a choice between several sequences among which ONE AT MOST is compulsory, the sequences are enclosed between ( \$[...]\$ ) symbols, each of them separated by semicolons (;). Again, the list of sequences can be written on several lines, for example:

```
$[ "ECHO" ; "NOEC" ]$
```

or

```
$[
  "ECHO" ;
  "NOEC"
]$
```

If a sequence can be repeated as many times as wanted, it is enclosed in parentheses: (...).

If a sequence must be repeated, for example *nf* times, it is framed by: *nf*\*(...). The reading of *nf* must precede immediately the first sequence:

```
( "VMIS" "ISOT" "RO" rho "YOUNG" e "NU" nu ...
  ... "TRAC" nf*( sig eps ) /LECTURE/ )
```

In this case, the input data should contain an integer (**nf**), immediately followed by **nf** couples of values. For example, if **nf**=3, the sequence **nf\*( sig eps)** could be written as:

```
3      0.  0.      1.E8 1.E-3      1.E9 1.E-1
```

In order to simplify the writing, a symbolic name is sometimes assigned to frequently used sequences, that are described only once and then referred to simply by their name enclosed in /, as in:

```
"MASSE" ( /LECDDL/      xm /LECTURE/  )
```

These named sequences are sometimes called procedures. It should be stressed that these names should not appear as such in an input data, but have to be replaced by the appropriate sequence of keywords and values. Named sequences should not be confused with keywords or directives. Their names therefore do not obey to the 4-character rule like keywords.

## 2.2 USE OF LITERAL VARIABLES

### Object :

It is possible to replace any data (a number, a keyword or a text) by a literal variable.

The name of this variable must start by % (per cent), followed by at most 16 alphanumeric characters.

Before being used for the first time, a literal variable must be assigned a value.

In order to assign a value to a literal variable, type its name followed by the equals sign (=) and then by the value, like in the Fortran programming language.

### Example :

```
. . .
%diameter_1 = 0.0548      %diameter_2 = 0.2027
%tube_1 = lig_ent        %tube_2 = lig_tot
. . .
COMPLEMENT
  DIAMETRE DROIT %diameter_1  LECT %tube_1 TERM
  DIAMETRE DROIT %diameter_2  LECT %tube_2 TERM
. . .
```

### Comments :

It is possible to re-define a literal variable (i.e., change its ‘contents’), at any place in the input data set.

The assignment sign ‘=’ must lie on the same input line as the variable name in the input data set.

A literal variable represents JUST ONE data: in the previous example, %tube\_1 and %tube\_2 represent each one a single word, that is an object name.

## 2.3 PROCEDURE /LECTURE/

### Object:

Procedure used to define a set of integers. Most of the time it is used to specify a list of nodes or of elements.

### Syntax:

Explicit definition (direct list of values):

```
| "LECT"  n1  n2  . . .  nk  "TERM"          |
```

Implicit definition (by using an increment `npas`):

```
| "LECT"  ndeb  "PAS"  npas  nfin  "TERM"      |
```

Definition using the objects created by the mesh generator "GIBI":

```
| "LECT"    ( 'nomobjet' )  "TERM"            |
```

Definition using the "permanent groups" created by the mesh generator "I-DEAS":

```
| "LECT"    ( 'nomgroup' )  "TERM"            |
```

Definition using the selections created by the LS-DYNA k-file format:

```
| "LECT"    ( 'kfileselections' )  "TERM"      |
```

All the elements or all the nodes are concerned (special keyword `TOUS`):

```
| "LECT"  "TOUS"  "TERM"          |
```

None of the elements or none of the nodes are concerned (special keyword `NONE`). This can be useful to avoid an error message in directives where the specification of elements or nodes is optional:

```
| "LECT"  "NONE"  "TERM"          |
```

Difference, intersection or symmetric difference between two sets (each set being defined by one of the above syntaxes):

```
| "LECT" <first_set> "DIFF" <second_set> "TERM" |
| "LECT" <first_set> "INTR" <second_set> "TERM" |
| "LECT" <first_set> "SDIF" <second_set> "TERM" |
```

### 'kfileselections':

The selection of entities from the LS-DYNA k-file can be done with the following commands

```
$[PART partnr; NSET nsetnr; SSHE sshell; SSOL ssolid; SBEA sbeam;  
  NODE nodenr; ELEM elemnr]$
```

#### **partnr**

Elements from PART partnr (also the part name can be used with a maximum length of 32 characters).

#### **nsetnr**

Nodes from NSET number nsetnr.

#### **ssolid**

Solid elements from SET\_SOLID number ssolid.

#### **sshell**

Shell elements from SET\_SHELL number sshell.

#### **sbeam**

Beam elements from SET\_BEAM number sbeam.

#### **elemnr**

Elements with the numbers elemnr. These are the numbers in the k-file and not the internal epX-numbers.

#### **nodenr**

Nodes with the numbers nodenr. These are the numbers in the k-file and not the internal epX-numbers.

### **Comments:**

The explicit and implicit syntaxes can be linked together. For example, to obtain the integers 3, 5, 2, 4, 6, 8, 10, 14, 15, 18, 21, 24, write:

```
LECT 3 5 2 PAS 2 10 14 15 PAS 3 24 TERM
```

For the implicit syntax, the step **npas** can be negative.

In the case of GIBI objects, the procedure extracts, if necessary, the indexes of the nodes or of the elements that constitute the objects/groups defined by the user. The directive where the procedure **/LECTURE/** appears determines if the indexes indicate nodes or elements.

If the **DIFF**, **INTR** or **SDIF** keywords are used, they must be at the end of the directive, as shown in the examples below. In other words, first the basic set must be defined, followed by one of these three keywords and then by the definition of the second set.

From now on this procedure will be called **/LECTURE/** or **/LECT/**.

**Remarks:**

EUROPLEXUS systematically checks the coherence of the indexes taking into account the expected type (nodes or elements).

Once they have been read, the indexes are classified in ascending order (if this does not harm the concerned directive).

If GIBI object names are mixed up with EUROPLEXUS numbers, separate the sequences by the keyword **SUIT**.

I-DEAS permanent group names can be freely mixed up with EUROPLEXUS indexes.

The keywords **DIFF**, **INTR** and **SDIF** may only be used in **/LECT/ures** that return lists of elements or nodes *ordered* in growing sequence and *without repeated items*. These are the vast majority in the code.

In any case, the finale result of the **/LECT/ure** may not be the empty set. An error is issued in this case.

To indicate **all** the elements or all the nodes in the model, the advised syntax is, as indicated above: **LECT TOUS TERM**. Other (obsolete) forms of the same directive are also accepted sometimes, for example the short syntax **TOUS** (i.e. without the keyword **LECT**). These alternative syntaxes are still accepted for compatibility with old input files, but might become unsupported in the short future. Note that the full syntax (**LECT TOUS TERM**) is the only supported syntax for use in conjunction with operations introduced by the keywords **DIFF**, **INTR** or **SDIF**.

**Examples:**

```
LECT 3 5 2 4 6 8 10 14 15 18 21 24 TERM
LECT 3 5 2 PAS 2 10 14 15 PAS 3 24 TERM
LECT 3 5 10 PAS -2 2 14 15 PAS 3 24 TERM
LECT objet1 objet2 SUIT 25 27 TERM
LECT objet1 SUIT 25 27 SUIT objet2 TERM
LECT toto tata DIFF titi tutu TERM
LECT TOUS DIFF titi tutu TERM
LECT 1 PAS 3 25 INTR titi tutu TERM
LECT toto tata SDIF 1 5 PAS 2 28 TERM
```

**Warning concerning the DIFF operator:**

Pay attention in the use of the DIFF operator in /LECT/ures where a set of **nodes** is required. For example, the expression:

```
LECT coco DIFF caca TERM
```

used in a context where a set of nodes is expected, and when **coco** is an object composed of elements, is evaluated as follows:

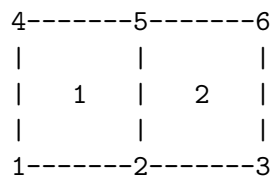
- First, the nodes of all the elements belonging to the **coco** object are evaluated.
- Next, the nodes belonging to the object **caca** are evaluated. If **caca** contains only nodes, these are directly available. If **caca** contains only elements, then all the nodes belonging to such elements are evaluated.
- Finally, the nodes of the second set belonging also to the first set are removed from the first set.

Note that this might not be the result you want or expect. In particular, in the case that both **coco** and **caca** contain elements, the resulting set of nodes is different from (smaller than) the set of nodes belonging to the difference between the two element sets. If the latter is what you want, proceed as follows: first, define a named group of elements (say **edif**) containing the difference of the elements, by using the COMP GROU directive; then, use directly the name **edif** in the /LECT/ expecting the nodes:

```
COMP GROU 1 'edif' LECT coco DIFF caca TERM ! element group
. . .
!LINK COUP BLOQ 1 LECT coco DIFF caca TERM ! wrong !!!
LINK COUP BLOQ 1 LECT edif TERM ! ok : nodes of the
! element group
```



The following example may also help clarify the matter. Assume the following simple mesh:



and assume that the Cast3m object `mesh` contains elements 1 and 2, while the object `right` contains element 2.

Then, in a `/LECT/` directive looking for nodes (e.g. the following blockage directive), the expression:

```
LINK COUP ... BLOQ 1 LECT mesh DIFF right TERM
```

will return nodes 1 and 4 (but *not* nodes 2 and 5). This is because: first the nodes of `mesh` are extracted (nodes 1 to 6); then the nodes of `right` are extracted (nodes 2, 3, 5 and 6); and finally these are subtracted from the first set, thus leaving only nodes 1 and 4.

If one wants instead to select all the nodes of the difference between the element sets (i.e. nodes 1, 2, 4 and 5), one can proceed as follows:

```
COMP GROU 1 'edif' LECT mesh DIFF right TERM
. . .
LINK COUP ... BLOQ 1 LECT edif TERM
```

The first directive builds up the element group containing the difference between the two element sets (difference between *elements*), which results into element 1. The second directive then extracts all the nodes of such element set, i.e. nodes 1, 2, 4 and 5.

## 2.4 PROCEDURE /PROGRESSION/

### Object:

Procedure used to prescribe a group of real numbers. For example, it can be used to specify the values of the physical times at which the solution has to be printed or stored.

### Syntax:

Explicit definition :

```
| "PROG" r1 r2 . . . rk "TERM" |
```

Implicit definition :

```
| "PROG" rdeb "PAS" rpas rfin "TERM" |
```

### Comments:

The explicit and implicit syntaxes can be linked together. For example, to obtain the values 3., 5., 2., 4., 6., 8., 10., 14., 15., 18., 21., 24. write:

```
"PROG" 3. 5. 2. "PAS" 2. 10. 14. 15. "PAS" 3. 24. "TERM"
```

For the implicit syntax, the step `rpas` can be negative.

From now on this procedure will be called `/PROG/`.

### Remark:

Once they have been read, the values are classified in ascending order (if this does not harm the instruction concerned).

## 2.5 PROCEDURE /CTIME/

### Object:

Procedure used to choose time values in the form of equidistant values (in time steps or in time values) or user-defined values. Typically, this can be used to specify the times at which output operations such as printouts, storage of data for restart or post-processing, etc., should take place during a computation.

### Syntax:

```
< "FREQ" ifreq > < "TFRE" tfreq >  
< "NUPA" /LECT/ > < "TIME" /PROG/ >
```

#### FREQ

A fixed frequency in time steps is chosen.

#### ifreq

Value of the frequency in time steps, starting from step 0.

#### TFRE

A fixed frequency in time is chosen.

#### tfreq

Value of the frequency in time units, starting from the initial time.

#### NUPA

A series of time steps is specified by the user.

#### /LECT/

Definition of the series of time step numbers.

#### TIME

A series of time values is specified by the user.

#### /PROG/

Definition of the series of time values.

### Comments:

The above optional forms of specifying time values can be freely combined. For example:

```
FREQ 10 NUPA LECT 10 15 35 TERM TFRE 1.E-3  
TIME PROG 0.5E-3 PAS 1.E-3 3.5E-3 TERM
```

is a valid specification and prescribes that the concerned event (e.g., a printout) will take place each 10 steps, and at steps 15 and 35, and each 1.E-3 time units, and at time values of 0.5E-3, 1.5E-3, 2.5E-3 and 3.5E-3.

If repeated values occur (such as in the above example for step number 10), they are automatically dealt with, i.e. only one event takes place for repeated values.

Note that if a time frequency is specified (keyword **TFRE**), the resulting time values take into account the **initial time** of the calculation declared in directive **CALC** by the **TINI** keyword. For example, if **TFRE 2.E-3** and **TINI 0.0**, then the corresponding event will take place at times 2.E-3, 4.E-3, 6.E-3 etc. But if **TFRE 2.E-3** and **TINI 1.0E-3**, then the corresponding event will take place at times 3.E-3, 5.E-3, 7.E-3 etc.

In a few instances, the keyword **"TEMPS"** can be used as an alias for **"NUPA"**. This keyword has been maintained for backwards compatibility, but should no longer be used in newly-written input data.

### Warning

Be aware that values given in time units with the above **"TFRE"** and **"TIME"** keywords are rounded to the closest number of 'time normalization units', as described below in the directive **"OPTION"**, see page H.20. The default value of time normalization unit is 1 picosecond (1.D-12 s). If necessary, this can be changed by the **OPTI TION** option described on page H.20.

## 2.6 PROCEDURE /LCHP/

### Object:

Procedure used to read a field of values ('champoint') generated by CASTEM2000.

### Syntax:

```
"LCHP" nomobjet "TERM"
```

Reads a CASTEM2000 object called `nomobjet`, of the type 'champoint', previously stored in CASTEM2000 with the directive "SAUV". The object is read and stored for successive use by other directives (see e.g. "EPAI").

## 2.7 PROCEDURE /LECDDL/

### Object:

It often occurs in a set of EUROPLEXUS input data that one has to define, for some nodes, the degrees of freedom according to which it is necessary to add a mass, prescribe the direction of a velocity, set a displacement to zero, and so on.

These degrees of freedom are identified by the numbers from 1 to 7 as described hereafter. See also the description of the available elements.

### Meaning of the numbers:

#### AXISYMMETRIC CASE (see AXIS):

Solid elements:

- 1 : d.o.f. along R;
- 2 : d.o.f. along Z.

Shell elements:

- 1 : d.o.f. along R;
- 2 : d.o.f. along Z;
- 3 : rotation d.o.f.

#### TWO DIMENSIONAL CASES (see CPLA and DPLA):

Same as for the axisymmetric case (but with X and Y instead of R and Z).

#### THREE DIMENSIONAL CASE (see TRID):

Solid elements:

- 1 : d.o.f. along X;
- 2 : d.o.f. along Y;
- 3 : d.o.f. along Z.

Shell, beam and pipe elements:

- 1 : d.o.f. along X;
- 2 : d.o.f. along Y;
- 3 : d.o.f. along Z;
- 4 : d.o.f. of rotation around X;

- 5 : d.o.f. of rotation around Y;
- 6 : d.o.f. of rotation around Z;
- 7 : fluid d.o.f. in coupled 1-D (for pipes only).

**Conventions:**

The degrees of freedom are specified by the corresponding numbers, one immediately after the other, i.e. without blanks.

Examples:

346    ==>   d.o.f.   3, 4, and 6;  
1426   ==>   d.o.f.   1, 4, 2, and 6.

The number of degrees of freedom described must correspond to the problem type (plane, axisymmetric, tridimensional) and to the element type.

In the following this procedure is called /LECDDL/.

## 2.8 MPI PARALLEL CALCULATIONS

Parallel calculations on distributed memory clusters yield the following requirements:

- Calculations must be launched through a working MPI library installed on the system (for questions about hardware or compatibility between EUROPLEXUS executable and MPI library, please contact CEA).
- Since data distribution is achieved through domain decomposition, subdomains must be used (see **STRUCTURE** and **INTERFACE** directives on page I.15). If these directives are ignored, subdomains and interfaces are automatically created using default parameters corresponding to the following input:

```
STRUCTURE AUTO ROB  
INTERFACE LINK NOMU
```

One subdomain is attached to each thread of the parallel simulation. Data between subdomains are exchanged through MPI messages.

For details about parallel algorithms implemented in EUROPLEXUS, please consult:

- Parallel development plan for EUROPLEXUS [\[828\]](#)
- Progress report [\[836\]](#)
- Description and evaluation of the first available parallel version [\[842\]](#)



## 2.9 READING DATA FROM EXTERNAL FILES

For convenience, it is sometimes useful to read some (bulky) input data for EPX not from the normal input file (.EPX) but from an external file.

For example, consider the case where a complex set of initial conditions or of external loads must be specified, with thousands of input lines. Sometimes, such complex files can be produced by some external software, which has no relation to (and is not interfaced with) EPX.

In order not to clutter the (main) input file .EPX, the use may proceed as follows:

- In the .EPX file, in place of the (entire) main directive concerned (INIT or CHAR in this example), use the INCLUDE command, followed by the name of the file in quotes, i.e. for example: INCLUDE 'init.txt' or INCLUDE 'char.txt'.
- Make available the chosen .txt file, which should contain the text of the complete directive, followed by a line containing the keyword RETURN.

When EPX encounters the INCLUDE directive, it continues reading the input from the specified external file. When it encounters the RETURN keyword in the external file, it goes back to reading from the main input file, at the line immediately following the INCLUDE statement.

### Comments:

At the moment, this mechanism can be used for all main directives of EPX, except the GEOM directive in a direct calculation and the SORT directive during post-processing of results with EPX (but work is ongoing for these directives).

The inclusion mechanism is not recursive, i.e. an included file cannot call another included file.

### 3 SPATIAL DISCRETIZATION (ELEMENT TYPES)

This Section gives a short description of all the “element” types available in EUROPLEXUS for the spatial discretization of the problem to be solved.

The code contains various formulations, including:

- Finite Elements, which may be used for both structural and fluid parts of the model;
- Finite Volumes, which are suited for the fluid parts;
- Spectral Elements, which can be used for the discretization of continuum-like solid parts which behave linearly (e.g. small-strain wave propagation);
- SPH particles, often used for high-speed impact problems;
- Diffuse Elements.

#### 3.1 ELEMENT TYPES

**Object:**

We describe now the different elements available for a one-dimensional, two-dimensional (plane or axisymmetric), or three-dimensional problem, by specifying, for each problem type:

- the number associated to the element type (num.),
- the element name for EUROPLEXUS (Name),
- the element name for the CASTEM meshing (Gibi),
- the developer (JRC or CEA),
- the number of nodes (npt),
- the number of degrees of freedom per node (dof),
- the number of integration points where the stresses will be computed (ngp).

Specific information relative to each particular element type follows these tables.

**Remark:**

Stresses are computed at the integration points.

In the output (on the listing or on the result file) all the stresses for a given element and for each of its integration points will be printed. See chapter [GBC.0010](#) as well.

### 3.1.1 1-D ELEMENTS

The various elements available in 1D are presented below.

num.	Name	Dev	Gibi	npt	dof	ngp	Remarks
22	<b>TUBE</b>	SEG2	CEA	2	1	1	fluid only (rigid tube)
23	<b>TUYA</b>	SEG2	CEA	2	7	2	tube coupled with FSI
24	<b>CL1D</b>	POI1	CEA	1	1	1	fluid boundary condition
25	<b>BIFU</b>	SUPE	CEA	1:9	7	1	bifurcation junction
26	<b>CAVI</b>	SUPE	CEA	1:9	1	1	cavity junction
31	<b>CLTU</b>	POI1	CEA	1	7	1	boundary condition with FSI
44	<b>ED1D</b>	SEG2	JRC	2	1	1	1D/2D-3D structural coupling
146	<b>BREC</b>	SEG2	CEA	2	7	1	pipeline rupture
147	<b>TUVF</b>	SEG2	CEA	2	1	1	rigid tube (1D vfcc)
148	<b>TYVF</b>	SEG2	CEA	2	7	1	tuvf coupled with FSI (1D vfcc)
149	<b>BIVF</b>	SEG2	CEA	2	1	1	bifurcation junction (1D vfcc)
150	<b>CAVF</b>	SEG2	CEA	1:9	1	1	cavity junction (1D vfcc)

For these elements (apart ED1D), the "EULER" option is mandatory (see page **GBA\_0030**). Furthermore, the mesh nodes must have three coordinates. Thus, the directives for the type of problem treated by these elements will always be of the form:

```
"TRIDIM"  "EULER"
```

Note that, for the ED1D elements the ED1D is used to perform coupled 1-D/multi-d calculations, therefore the problem must be declared either "AXIS" or "CPLA" or "DPLA" or "TRID" and the "EULE" is not needed.

These elements are specified hereafter:

#### TUBE

This element allows to model the fluid contained within a fixed pipeline. It is assumed that fluid properties are the same in all points of a give cross-section of the pipe (one-dimensional calculation).

#### CL1D

This element allows to introduce the different boundary conditions or localised singularities, in a pipeline meshed by elements of type "TUBE".

#### CAVI

To assemble the different branches of a pipeline. The junction is done by a finite volume with an attached constitutive law, unlike in the case of a simple bifurcation.

#### BIFU

This element allows to simply specify the relationships between the inputs and outputs of different branches which are joined together. The conservation of mass flow rate and pressure is ensured.

**TUYA**

This element allows to treat the motions of pipelines in the presence of an internal flow. It results from the superposition of elements “TUBE” and “POUTRE”.

**CLTU**

Similarly to “CL1D”, this element further allows to introduce coupled boundary conditions of the fluid-structure type (e.g. a closed pipe end).

**BREC**

This element allows to model a pipeline rupture. Before the optional rupture instant, this element behaves like a bifurcation.

**TUVF**

This 1D finite volume element allows to model the fluid contained within a fixed pipeline. It is assumed that fluid properties are the same in all points of a give cross-section of the pipe (one-dimensional calculation).

**TYVF**

This 1D finite volume element allows to treat the motions of pipelines in the presence of an internal flow. It results from the superposition of elements “TUVF” and “POUTRE”.

**BIVF**

This 1D finite volume element allows to simply specify the relationships between the inputs and outputs of 2 branches which are joined together. The conservation of mass flow rate is ensured.

**Warning:**

The BIVF element can be used only with 2 branches of fixed pipeline at the moment (tuvf element).

**CAVF**

To assemble the different branches of a pipeline composed of 1D finite volume element. The junction is done by a finite volume with an attached constitutive law, unlike in the case of a simple bifurcation.

**Warning:**

The 1D finite volume element are under development and testing at CEA and EDF and should therefore be used with great care.

## ED1D

A 2-node element used as an interface between a 1D structure and a multi-D structure.

This element can be used to couple a 2D or 3D model to a 1D model to be calculated by the EURDYN-1D code, developed at JRC Ispra (this “code” is now embedded within EUROPLEXUS, so that its usage is transparent to the EUROPLEXUS user). Of the two nodes, one is used to define the location of the interface (i.e. the point at which forces are transmitted between the two structures), while the other is only used to indicate the direction in space along which coupling is enforced.

This direction remains unchanged during deformation. The distance between the two nodes is therefore irrelevant. The 1D structure has to be separately modelled by EURDYN-1D, which is seen by EUROPLEXUS as a standard element module. A special set of input data for EURDYN-1D (ED1D “input deck”) has to be prepared. This must be included within the normal EUROPLEXUS input file, immediately after the CALCUL directive and before any additional EUROPLEXUS directives (see page I.23 and the EURDYN-1D manual listed in the References: ([33])).

The ED1D input deck must be immediately preceded by a line containing “ED1D START” (capitals, starting in column 1, followed only by blanks if any) and be immediately followed by a line containing “ED1D END” (capitals, starting in column 1, followed only by blanks if any).

The following (fixed) logical unit numbers are used by the EURDYN-1D module:

- 01 (formatted) used to copy the ED1D input deck, thus “extracting” it from the normal EUROPLEXUS input file. At run end it contains the list of 1D space plots storages;
- 04 (unformatted) used to store data for TPLOT (time plots);
- 33 (unformatted) used to store data for TPLOT (space plots);
- 34 (unformatted) used to store data for restart.

Note that units 33 and 34 are used by default to store data for TPLOT (space plots) and for a restart, respectively. They are both unformatted. These values, however, can be changed from the input.

Since EURDYN-1D is a specialised module, its usage can lead to important savings in the overall computation cost in large, complex multi-D problems when one or more portions of the structure can be conveniently represented by a 1D model.

In setting up a coupled 1-D/multi-D model, the following guidelines should be followed:

1. The multi-D part of the model is meshed as usual and an ED1D interface element is placed at each point of connection between the multi-D domain and the 1-D domain. An arbitrary number of such interface elements can be used in a calculation, according to user needs.

2. Each ED1D element must be so oriented that its first node coincides with the interface (attachment) point, while the second node only defines the orientation of the 1-D structure (i.e., of the interaction force) in multi-D space. The length of such elements is therefore irrelevant.
3. All 1-D parts of the model are represented in a single 1-D model, for which a separate input data set has to be provided.
4. When more than one 1-D part is present, each one of these will form a separate 'level' in the 1-D model (see the EURDYN-1D manual for the definition of level).
5. In setting up the 1-D model, the abscissa of each level should be oriented from the interface point toward the outside of the multi-D body. Thus, the interface node is always the first one (or the 'left' node, according to EDURDYN-1D conventions) of each level.
6. Each ED1D element should be assigned a different VM1D material (see page C.220) and the "PT1D" directive of this material should be used to assign the associated node index in the 1-D model.

Furthermore, note that a few directives, most notably saving for restart, are not available in coupled 1-D/multi-D computations.

Some care should also be taken in specifying the final times for the calculation and times for printouts and data storages. The following procedure is suggested in order to minimize potential problems:

- Assume that the initial and final physical times of interest for the calculations be  $TI$  and  $TF$ , respectively.
- Then, in the 1-D input file choose a final time (see EURDYN-1D input manual)  $FINTIM$  larger than  $TF$ , say:

$$FINTIM = TF + (TF - TI) * K$$

with :  $0.01 < K < 0.1$

- Choose a final printout time:

$$TPRINT(NPRINT) = TF$$

If the final 1-D results should appear on the listing, and choose a final space plots storage time:

$$TSTOR(NSSTOR) = TF$$

If space plots in the final 1-D configuration are desired.

- In the multi-D input file (EUROPLEXUS), choose in the CALCUL directive a final time  $TEND$  larger than  $TF$ , but smaller than the 1-D value ( $FINTIM$ ) specified above:

$$TF < TEND < FINTIM$$

- Choose printout and storage times as appropriate, by including the value  $TF$  if desired.

In this way, the coupled calculation will be stopped at the end by the multi-D part of the code at time `TEND`, and the printouts and storages at `TF` should be correctly produced for both the 1-D and the multi-D models.

The only drawback of this method is that the 'space plots summary' table in the 1-D listing is not produced at the end of the run because the 1-D calculation is stopped before its declared final time is actually reached. However, this is not a real problem, since the 1-D space plots file is nevertheless correctly generated.

### 3.1.2 2-D ELEMENTS

The various elements available in 2D are presented below.

num.	Name	Gibi	Dev	npt	dof	ngp	Remarks
1	COQU	SEG2	CEA	2	3	2	thin shell
2	TRIA	TRI3	CEA	3	2	1	triangle
3	BARR	SEG2	CEA	2	2	1	bar (membrane only)
4	PONC	POI1	CEA	1	2	1	circular bar(axisym.)
5	MEMB	SEG2	CEA	2	2	1	'virole' in membrane (axisym.)
7	CL2D	SEG2	CEA	2	2	1	boundary conditions
8	CAR1	QUA4	CEA	4	2	1	quadrangle with 1 Gauss pt.
9	CAR4	QUA4	CEA	4	2	4	quadrangle with 4 Gauss pt.
10	COQC	SEG2	CEA	2	3	1	thin shell with shear
15	FS2D	RAC2	CEA	4	2	1	F.S. coupling
28	PMAT2D	POI1	CEA	1	2	1	material point
38	Q92	QUA8	JRC	9	2	4	9-node quadrilateral
39	Q93	QUA8	JRC	9	2	9	9-node quadrilateral
42	CL23	SEG3	JRC	3	2	2	3-node b.c.s
43	ED01	SEG2	JRC	2	3	10	beam/conical-shell
45	TVL1	TRI3	CEA	3	2	1	Van Leer triangle
46	CVL1	QUA4	CEA	4	2	1	Van Leer quadrangle
49	Q92A	QUA8	JRC	9	2	4	Q92 on symmetry axis
52	FLU1	QUA4	JRC	4	2	1	fluid quadrilateral
54	PFEM2D	POI1	JRC	1	2	1	particle finite element
56	ED41	SEG2	JRC	4	3	10	old version of ED01
58	ADQ4	QUA4	JRC	4	2	1	advection-diffusion quadrilateral
64	FL23	TRI3	JRC	3	2	1	fluid triangle
65	FL24	QUA4	JRC	4	2	1	fluid quadrilateral
70	CL22	SEG2	JRC	2	2	2	2-node b.c.s
71	Q41	QUA4	JRC	4	2	1	ALE structural quadrilateral
72	Q42	QUA4	JRC	4	2	4	ALE structural quadrilateral
73	Q41N	QUA4	JRC	4	2	1	ALE structural quadrilateral
74	Q42N	QUA4	JRC	4	2	4	ALE structural quadrilateral
75	Q41L	QUA4	JRC	4	2	1	Lagr. structural quadrilateral
76	Q42L	QUA4	JRC	4	2	4	Lagr. structural quadrilateral
77	Q95	QUA8	JRC	9	2	4	test version of Q92
97	MC23	TRI3	JRC	3	2	1	finite volume fluid triangle
98	MC24	QUA4	JRC	4	2	1	finite volume fluid quadrilateral
100	Q42G	QUA4	JRC	4	2	4	ALE structural quadrilateral
105	MS24	QUA4	JRC	4	2	1	spectral "macro" quadrilateral
106	S24	QUA4	JRC	4	2	1	spectral "micro" quadrilateral
109	FUN2	SEG2	JRC	2	2	1	cable element
114	BSHT2D	SEG2	CEA	2	2	-	bushing with translational dof
118	MAP2	—	CEA	3	2	-	Point on solid edge
121	MAP5	—	CEA	3	2	-	Point on 2D shell
124	INT4	—	CEA	4	2	2	quadrilateral interface element
131	T3VF	TRI3	CEA	3	2	1	triangle finite volume
132	Q4VF	QUA4	CEA	4	2	1	quadrangle finite volume
140	DEBR2D	POI1	JRC	1	2	-	debris particle



The specifications for these elements are given hereafter:

## COQU

Reference element for all calculations with 2D thin shells.

## TRIA

This element can equally well represent solids or fluids. However, if the mesh is not very regular, it may give rise to fluctuations in the distribution of masses, especially near the axis of revolution.

The element has 4 components of stress (SIG) and strain (EPST) (when used for solids), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

## BARR

This element is intended for the modeling of steel reinforcement in concrete structures, or of bars that work only in traction-compression.

## PONC

This element should only be used to model circular steel reinforcement in axisymmetric concrete structures.

## MEMB

This element is similar to COQU, but has a purely membrane behaviour.

## CL2D

This element allows to specify a condition of absorbing medium or an impedance.

## CAR1

This element is especially used to represent fluids. However, due to its under-integration, it is strongly advised to add some anti-hourglass numerical damping, unless the boundary conditions themselves prevent the appearance of hourglass motions.

The element has 4 components of stress (SIG) and strain (EPST) (when used for solids), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .

- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

## CAR4

This element is recommended for elastoplastic solids.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

## COQC

Simpler than "COQUE", it also enables to evaluate the shear, if this is not too large.

## FS2D

Incompressible fluid elements which ensure the transmission of forces.

The first face (1-2) is in the fluid, the other (3-4) in the solid. To ensure a proper fluid-structure connection, it is useful that the sides 2-3 and 1-4 be as short as possible, possibly of zero length. In this last case, only the forces normal to faces 1-2 and 3-4 will be transmitted.

These elements are defined in the mesh but they work only through the directive "LIAISON". It is therefore possible to "activate" or "deactivate" this particular connection depending on the problem to be treated, by modifying the "LIAISON" directive during a "REPRISE" (see page SR.40).

## PMAT

This element is particularly aimed at modeling rigid projectiles, in connection with the directive "IMPACT". It can also be used to specify added masses.

## TVL1

This element has been developed for Van Leer fluids.

## CVL1

Similarly to TVL1, this element has been developed for Van Leer fluids.

## BSHT

The availability of the bushing element family allows to define generalized stiffness and damping between two nodes. The implemented model provides in 2D the element BSHT, with only translation degrees of freedom. All the characteristics of the bushing element are defined using "JOINT PROPERTIES" material type.

## MAP2

This element is used in order to glue one slave node to a master side. The slave node should be on the side. 2 kinematic constraints are introduced in order to impose the translation dof of the slave node. These elements are defined in the topology but they work only through the directive "LIAISON".

## MAP5

This element is used in order to glue one slave node to a master shell side. The slave node should be on the side. 2 kinematic constraints are introduced in order to impose the translation dof of the slave node and a kinematic constraint is added on the rotational dof of the slave node. These elements are defined in the topology but they work only through the directive "LIAISON".

## INT4

The INT4 element is a quadrilateral pure displacement interface element (sometimes called *cohesive element*) dedicated to the modeling of interlayers, separating "standard" structural elements. In the particular case of a composite model, this element can be considered as representing a homogeneous resin layer ensuring the interlaminar stress transfer between adjacent plies. This approach is most often referred to as "mesoscopic" laminate modeling.

## T3VF

2D triangle finite volume element. The finite volume is defined as cell centred. Several options for the calculation can be chosen with OPTI VFCC.

## Q4VF

2D quadrangle finite volume element. The finite volume is defined as cell centred. Several options for the calculation can be chosen with OPTI VFCC.

## Q92

This element can be used for precise modelling of continua. It can undergo arbitrarily large deformations. Since it is underintegrated, it is locking-free, but it may occasionally suffer from mechanisms if boundary conditions are too loose. In such cases, use of the Q93 element (which, however, is more expensive) is recommended.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .

- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

### Q93

This is the fully-integrated version of the Q92 element. Its use is only recommended in plane cases, when mechanisms might occur.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

### CL23

This element is mainly used to specify uniform pressure conditions acting on the boundaries of quadratic elements of type Q92, Q93 or Q92A.

### ED01

This element is integrated through the thickness (5 points at each of 2 longitudinal stations) and offers accurate modelling in highly nonlinear cases (spreading of plasticity through the thickness). The effect of arbitrarily large membrane strains over the element thickness is taken into account, unless option "EDSS" is used (see "OPTION").

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy} = 0, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y = 0, \gamma_{xy} = 0, \epsilon_z$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_\theta, \tau_{xy} = 0, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_\theta, \gamma_{xy} = 0, \epsilon_z$ .
- In a plane stress calculation (CPLA), which means uniaxial stress here, so that the element represents a beam in the  $xz$ -plane:  $\sigma_x, \sigma_y \approx 0, \tau_{xy} = 0, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy} = 0, \epsilon_z$ .

Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

### Q92A

This element should (only) be used in place of Q92 in axisymmetric problems, for those elements that have one side along the axis of symmetry (y-axis). It does not suffer from mechanisms.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .

### FLU1

This element offers a specialised treatment which is thought to be particularly effective for fluids, in conjunction with the A.L.E. formulation. An implicit phase for the calculation of pressure is introduced during time integration. The element can be degenerated to represent a triangle by simply repeating one of the nodes in the description of topology.

## PFEM

This element is used to represent a 2D (or 3D) continuum (usually a fluid) by means of the Particle Finite Element method (PFEM).

## ED41

A 4-node version of the ED01 element that facilitates fluid-structure interaction for certain problems. Two nodes are used at each extremity of the element in order to define the element thickness. However, these are really one physical node since displacements, velocities etc. are coincident.

In the element numbering, the first two nodes must define an 'external side' of the element. In other words, they must not be along the element thickness.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy} = 0, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y = 0, \gamma_{xy} = 0, \epsilon_z$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_\theta, \tau_{xy} = 0, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_\theta, \gamma_{xy} = 0, \epsilon_z$ .
- In a plane stress calculation (CPLA), which means uniaxial stress here, so that the element represents a beam in the  $xz$ -plane:  $\sigma_x, \sigma_y \approx 0, \tau_{xy} = 0, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy} = 0, \epsilon_z$ .

## ADQ4

4-node quadrilateral for advection-diffusion problems.

This element is used to model advection-diffusion problems in incompressible fluids with heat transfer, according to JRC's models.

## FL23

3-node triangle for compressible fluids. This is an alternative to the degeneratable FLU1 quadrilateral.

## FL24

4-node quadrilateral for compressible fluids. This is an alternative to the degeneratable FLU1 quadrilateral.

## CL22

2-node boundary condition. This is recommended for use with 2-D Ispra models. This element automatically recognizes the element to which it is attached, and uses the most appropriate pressure discretization.

**Q41**

4-node quadrilateral for structural ALE calculations with reduced integration.  
The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

**Q42**

4-node quadrilateral for structural ALE calculations with full integration.  
The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

**Q41N**

4-node quadrilateral for structural ALE calculations with reduced integration. Uses Godunov algorithm.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

**Q42N**

4-node quadrilateral for structural ALE calculations with full integration. Uses Godunov algorithm. This element doesn't work well in some test cases, so it is advisable to use Q42G instead.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

**Q41L**

4-node quadrilateral for Lagrangian calculations with reduced integration.  
The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

**Q42L**

4-node quadrilateral for Lagrangian calculations with full integration.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

**Q95**

9-node isoparametric quadrilateral with curved sides. This is a special version of the Q9 element under test, that should avoid mechanisms.

Its use is not recommended for the moment.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

**MC23**

3-node finite volume triangle for multicomponent flows.

**MC24**

4-node finite volume quadrilateral for multicomponent flows.

**Q42G**

4-node quadrilateral for structural ALE calculations with full integration. Uses Godunov algorithm.

The element has 4 components of stress (SIG) and strain (EPST), organized as follows.

- In a plane strain calculation (DPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z = 0$ .
- In an axisymmetric calculation (AXIS):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_\theta$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_\theta$ .
- In a plane stress calculation (CPLA):  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$  and  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ .

**MS24**

4-node quadrilateral MACRO spectral element.

The integration points coincide with the Gauss-Lobatto-Legendre points and are determined by specifying the MICRO spectral elements S24.

**S24**

4-node quadrilateral MICRO spectral element This element is used only to specify 'internal' nodes of an MS24.

**FUN2**

2-node cable element.

This is a specialized element for the representation of cables in 2D space, in conjunction with the FUNE material (it resists only in traction, not in compression). When used with the VM23 material, it represents a bar (which resists both in traction and in compression). The element is large-strain.

The element has 4 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y \approx 0, \tau_{xy} = 0, \sigma_z \approx 0$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy} = 0, \epsilon_z$ .

**DEBR**

1-node debris particle element.

This is a specialized element for the representation of flying debris, as e.g. resulting from an explosion or an impact, by means of spherical particles. It may be used both in 2D and in 3D.



### 3.1.3 3-D ELEMENTS

The various elements available in 3D are presented below.

num.	Name	Gibi	Dev	npt	dof	ngp	Remarks
57	ADC8	CUB8	JRC	8	3	1	advection-diffusion brick
32	APPU	POI1	CEA	1	6	1	support
130	ASHB	—	CEA	8	3	5	assumed strain thick shell
79	BILL	POI1	CEA	1	3	1	particle element (NABOR and SPH)
19	BR3D	SEG2	CEA	2	3	1	bar
115	BSHR	—	CEA	2	6	—	bushing with trans. and rot. dof
114	BSHT	—	CEA	2	3	—	bushing with translational dof
144	C272	CU27	JRC	27	3	8	27-node cube
145	C273	CU27	JRC	27	3	27	27-node cube
155	C81L	CUB8	JRC	8	3	1	8-node cube
156	C82L	CUB8	JRC	8	3	8	8-node cube
62	CL32	QUA4	JRC	4	6	4	b.c.s for CQD4
63	CL33	QUA9	JRC	9	6	9	b.c.s for CQD9
18	CL3D	QUA4	CEA	4	3	1	bound. cond. (4-node face)
78	CL3I	TRI3	JRC	3	3	1	3-node b.c.s
99	CL3Q	QUA4	JRC	4	3	1	4-node b.c.s
29	CL3T	TRI3	CEA	3	3	1	bound. cond. (3-node face)
151	CL92	QUA9	JRC	9	3	4	9-node (3D) b.c. for C272
152	CL93	QUA9	JRC	9	3	9	9-node (3D) b.c. for C273
95	CLD3	TRI3	JRC	3	6	3	b.c.s for CQD3
96	CLD6	TRI6	JRC	6	6	4	b.c.s for CQD6
47	CMC3	TRI3	CEA	3	6	2	multilayer shell
12	COQ3	TRI3	CEA	3	6	1	triangular thin shell
14	COQ4	QUA4	CEA	4	6	4	quadrangular thin shell
40	COQI	TRI3	JRC	3	6	15	triangular shell (small strain)
93	CQD3	TRI3	JRC	3	6	15	degenerated shell (Hughes-Liu)
91	CQD4	QUA4	JRC	4	6	20	degenerated shell (Hughes-Liu)
94	CQD6	TRI6	JRC	6	6	20	degenerated shell (Hughes-Liu)
92	CQD9	QUA9	JRC	9	6	45	degenerated shell (Hughes-Liu)
13	CUB6	CUB8	CEA	8	3	6	brick with 6 Gauss pt
30	CUB8	CUB8	CEA	8	3	8	brick with 8 Gauss pt
6	CUBB	CUB8	CEA	8	3	8	brick based on B-bar method
11	CUBE	CUB8	CEA	8	3	1	brick with 1 Gauss pt
133	CUVF	CUB8	CEA	8	3	1	cube finite volume
81	CUVL	CUB8	CEA	8	3	1	Van Leer cube
140	DEBR	POI1	JRC	1	3	—	debris particle
84	DKT3	TRI3	CEA	3	6	15	shell (Mindlin)
83	DST3	TRI3	CEA	3	6	15	shell (Discrete Shear Triangle)
80	ELDI	POI1	CEA	1	6	1	discrete element
66	FL34	TET4	JRC	4	3	1	fluid tetrahedron
67	FL35	PYR5	JRC	5	3	1	fluid pyramid
68	FL36	PRI6	JRC	6	3	1	fluid prism
69	FL38	CUB8	JRC	8	3	1	fluid hexahedron
53	FLU3	CUB8	JRC	8	3	1	fluid brick
16	FS3D	RAC3	CEA	8	3	1	F.S. connection (4-node face)
48	FS3T	PRI6	CEA	6	3	1	F.S. connection (3-node face)
110	FUN3	SEG2	JRC	2	3	1	cable element

153	LIGR	SUPE	CEA	2*	6	-	mechanism (articulated systems)
125	INT6	—	CEA	6	3	1	triangular prism interface element
126	INT8	—	CEA	8	3	4	hexahedron interface element
119	MAP3	—	CEA	4	3	—	point on a triangular facet
120	MAP4	—	CEA	5	3	—	point on a quadrangular facet
122	MAP6	—	CEA	4	6	—	point on a triangular shell facet
123	MAP7	—	CEA	5	6	—	point on a quadrangular shell facet
101	MC34	TET4	JRC	4	3	1	finite volume tetrahedron
102	MC35	PYR5	JRC	5	3	1	finite volume pyramid
103	MC36	PRI6	JRC	6	3	1	finite volume prism
104	MC38	CUB8	JRC	8	3	1	finite volume hexahedron
33	MECA	SEG2	CEA	2	6	1	mechanism (articulated systems)
128	MOY4	—	CEA	4	3	—	node to element mean connector
129	MOY5	—	CEA	5	3	—	node to element mean connector
107	MS38	CUB8	JRC	8	3	1	spectral "macro" quadrilateral
54	PFEM	POI1	JRC	1	3	1	particle finite element
28	PMAT	POI1	CEA	1	3	1	material point
17	POUT	SEG2	CEA	2	6	2	beam
20	PR6	PRI6	CEA	6	3	6	prism with 6 Gauss pt
27	PRIS	PRI6	CEA	6	3	1	prism with 1 Gauss pt
134	PRVF	PRI6	CEA	6	3	1	prism finite volume
82	PRVL	PRI6	CEA	6	3	1	Van Leer prism
136	PYVF	PYR4	CEA	5	3	1	pyramid finite volume
138	Q4MC	QUA4	CEA	4	6	-	multilayered quadrangular shell
90	Q4G4	QUA4	CEA	4	6	4	shell (Batoz)
111	Q4GR	QUA4	CEA	4	6	5	idem Q4G4 (simplified : 1 pt)
112	Q4GS	QUA4	CEA	4	6	20	idem Q4G4 (simplified : 4 pts)
35	QPPS	QUA4	CEA	4	6	5	similar to Q4GR
113	RL3D	SEG2	CEA	2	3	1	two-node spring
108	S38	CUB8	JRC	8	3	1	spectral "micro" quadrilateral
117	SH3D	—	CEA	3	6	—	node to shell connector
127	SH3V	—	CEA	8	3	4	node to element connector
85	SHB8	CUB8	CEA	8	3	5	thick shell
89	SPHC	POI1	CEA	1	6	1	particle element (thick shell)
139	T3MC	TRI3	CEA	4	6	-	multilayered triangular shell
51	T3GS	TRI3	CEA	3	6	5	shell (Reissner-Mindlin)
21	TETR	TET4	CEA	4	3	1	tetrahedron with 1 Gauss pt
135	TEVF	TET4	CEA	4	3	1	tetrahedron finite volume
41	TUBM	SUPE	CEA	1*	3*	1	connection 3D-1D
116	TUYM	SUPE	CEA	1*	3*	1	connection 3D-1D

The specifications for these elements are given hereafter:

## CUBB

Solid element for elasto-plastic computations. This element developed at EDF ([944]) is free from volumetric locking and does not exhibit hourglassing. When the material becomes quasi-incompressible, one should decrease CSTAB coefficient to maintain the calculation stable.

## CUBE

This element is used especially with fluid materials. However, because of its under-integration, "hourglassing" phenomena may appear, if they are not prevented by the boundary conditions. Such phenomena may be contrasted by using the HOURG option (anti-hourglass artificial viscosity).

The element has 6 components of stress (SIG), organized as follows (for solid materials):  $\sigma_x$ ,  $\sigma_y$ ,  $\sigma_z$ ,  $\tau_{xy}$ ,  $\tau_{yz}$ ,  $\tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x$ ,  $\epsilon_y$ ,  $\epsilon_z$ ,  $\gamma_{xy}$ ,  $\gamma_{yz}$ ,  $\gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

### COQ3

This element must be used with care. In order to obtain good results make sure to use a symmetric mesh (ex: British flag).

### CUB6

Solid element for elasto-plastic computations. This element should be used with caution, because being underintegrated it may lead to hourglassing. Use preferentially CUBB or CUB8 elements.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x$ ,  $\sigma_y$ ,  $\sigma_z$ ,  $\tau_{xy}$ ,  $\tau_{yz}$ ,  $\tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x$ ,  $\epsilon_y$ ,  $\epsilon_z$ ,  $\gamma_{xy}$ ,  $\gamma_{yz}$ ,  $\gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

### COQ4

This element is recommended for computations on 3-D shells if the deformations are small. In reality, this element is composed of 4 triangular COQ3 plates which are superimposed and symmetrized.

### FS3D

Same remarks as for FS2D.

### POUT

This element enables the modelling of complex profile beams submitted to a tension or bending stress. The default stability time step for this element is quite conservative (optimized for tubes?). Larger or even much larger values for different cross sections may be obtained by using the option DTBE (see H 20).

### CL3D

Same remarks as for CL2D.

### BR3D

This element is mainly used to model concrete rebars or any other beam submitted to a simple tension.

## PR6

As it is the case of CUB8, this element is recommended for elasto-plastic computations.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \epsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## TETR

This element may be used for fluids or for elasto-plastic computations.

The element has 6 components of stress (SIG), organized as follows (for solid materials):  $\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \epsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## PRIS

This prismatic element is especially used for fluids (see CUBE).

The element has 6 components of stress (SIG), organized as follows (for solid materials):  $\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \epsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## PMAT

Primarily, this element enables the modelling of rigid missiles in connection with the keyword "IMPACT"; but it also enables the introduction of added masses.

## CL3T

Same remarks as for CL2D.

## CUB8

Solid element recommended for elasto-plastic computations (no hourglassing phenomena).

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \epsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## APPU

This element has one node and six degrees of freedom. It allows to use constitutive equations of type non-linear support in a chosen direction. See also page C2.108.

## MECA

This element has two nodes with six degrees of freedom. It allows to use constitutive equations for mechanisms (articulations, etc.).

## TUBM

Connection 3D-1D. Consult the corresponding 'liaison' (connection).

## TUYM

Connection 3D-1D for moving meshes (ALE). Consult the corresponding 'liaison' (connection).

## CMC3

This element is used for the modelling of an eccentric layer in relation to the average plane defined by its 3 nodes. The layer is associated with an orthotropic behaviour in the given plane. Several CMC3 elements are supported by the same nodes, but they are differently eccentric; they represent a multilayer structure.

The geometric and mechanical characteristics of the element (eccentricity, orthotropy associated to a local system) can be defined either when CASTEM2000 generates the mesh (see option CASTEM page A.30) or directly by EUROPLEXUS (see page C.95) in a normal mesh generated by COCO or GIBI.

The local reference of the element is as follows: the first axis is formed by side 1-2, the second is such that the 3rd node lies in the half-plane ( $Y > 0$ ).

## FS3T

Same remarks as for FS2D.

## T3GS

3-node thick shell (Reissner-Mindlin) element with 1 integration point in the plane. It has the same local frame as COQ3. This element developed at EDF ([931]).

It is a predecessor of the Q4G family, uses the same approach for representing the shear strain and is thus the best suited among T3 shell elements to be combined with Q4G shell elements.

The element has 8 components of stress (SIG), organized as follows:  $\sigma_x^m, \sigma_y^m, \tau_{xy}^m, \sigma_x^b, \sigma_y^b, \tau_{xy}^b, \tau_{xz}, \tau_{yz}$ , where the first three components are the membrane contributions, the second three components are the bending contributions and the last two components are the transverse shear contributions (note that the order in which these last two components are given is opposite to what is usually found in 3D continuum elements, for example). The total strains (EPST) follow the same organization:  $\epsilon_x^m, \epsilon_y^m, \gamma_{xy}^m, \epsilon_x^b, \epsilon_y^b, \gamma_{xy}^b, \gamma_{xz}, \gamma_{yz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

**BILL**

This element is primarily aimed at the modeling of fluids or structures by using the method of particles and forces.

**ELDI**

This point-like element has one node with six degrees of freedom. The element is developed at EDF ([935]) to model fragmentation of concrete structures. The discrete element mesh is generated by using a particular geometric padding technique (Jerier 2010) implemented into SpherePadder tool (free software under GNU GPL v2 license) and integrated as a plug-in into SMESH mesher of SALOME plate-form. The DE mesh is available in MED format only. Interactions between these elements allow to model cohesive nature of materials or contact.

**CUVL**

Specific element (hexahedron) for Van Leer fluids in 3D.

**PRVL**

Specific element (prism) for Van Leer fluids in 3D.

**DST3**

3-node shell element (Discrete Shear Triangle).

It is a thick shell element (Mindlin). Same local frame as COQ3.

The element has 8 components of stress (SIG), organized as follows:  $\sigma_x^m, \sigma_y^m, \tau_{xy}^m, \sigma_x^b, \sigma_y^b, \tau_{xy}^b, \tau_{xz}, \tau_{yz}$ , where the first three components are the membrane contributions, the second three components are the bending contributions and the last two components are the transverse shear contributions (note that the order in which these last two components are given is opposite to what is usually found in 3D continuum elements, for example). The total strains (EPST) follow the same organization:  $\epsilon_x^m, \epsilon_y^m, \gamma_{xy}^m, \epsilon_x^b, \epsilon_y^b, \gamma_{xy}^b, \gamma_{xz}, \gamma_{yz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

**DKT3**

3-node shell element (Discrete Kirchhoff Triangle). It is a thick shell element (Mindlin). It has the same local frame as COQ3.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x^m, \sigma_y^m, \tau_{xy}^m, \sigma_x^b, \sigma_y^b, \tau_{xy}^b$ , where the first three components are the membrane contributions and the second three components are the bending contributions. The total strains (EPST) follow the same organization:  $\epsilon_x^m, \epsilon_y^m, \gamma_{xy}^m, \epsilon_x^b, \epsilon_y^b, \gamma_{xy}^b$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

**SHB8**

8-node thick shell element obtained starting from the 8-node brick. The 2 faces of this element are formed by the nodes: 1, 2, 3, 4 for the first face and 5, 6, 7, 8 for the second face.

## SPHC

This thick shell (Mindlin-Reissner) particle element has one node with five degrees of freedom: 3 translations and 2 rotations.

## Q4G4

4-node shell element (Batoz), with 4 integration points in the plane and 5 integration points through the thickness for plasticity.

There are 8 stress components:  $\sigma_x$ ,  $\sigma_y$ ,  $\sigma_{xy}$ ,  $\sigma_x^b$ ,  $\sigma_y^b$ ,  $\sigma_{xy}^b$ ,  $\tau_{xz}$ ,  $\tau_{yz}$ .

It is a thick shell element with 4 nodes (BATOZ formulation) which accounts for the non-coplanarity of the four nodes. It is a complete but expensive version of Batoz's element.

A local frame is defined at each Gauss point: the first vector is tangent to the line ( $csi=cst.$ ) in the sense from node 1 to node 2, the second vector is the vector product of the first by the vector tangent to the line ( $eta=cst.$ ) in the sense from node 1 to node 4. The frame is completed so as to be right-handed.

## Q4GR

4-node shell element (BATOZ) with 1 integration point in the plane and 5 integration points through the thickness for plasticity.

It is a simplified version of Q4G4 with a single integration point in the plane. An incomplete anti-hourglass stiffness (only in rotation) is implemented; an adjusting coefficient for anti-hourglass can be set using the following syntax:

```
"OPTI"  "HGQ4"  hgq4ro
```

The default value of hgq4ro is 0.018.

The element has 10 components of stress (SIG), organized as follows:  $\sigma_x^m$ ,  $\sigma_y^m$ ,  $\tau_{xy}^m$ ,  $\sigma_x^b$ ,  $\sigma_y^b$ ,  $\tau_{xy}^b$ ,  $\tau_{xz}$ ,  $\tau_{yz}$ ,  $\sigma_h^1$ ,  $\sigma_h^2$ , where the first three components are the membrane contributions, the second three components are the bending contributions, the next two components are the transverse shear contributions (note that the order in which these last two components are given is opposite to what is usually found in 3D continuum elements, for example) and the last two components are anti-hourglassing (pseudo-)stresses. The total strains (EPST) follow the same organization:  $\epsilon_x^m$ ,  $\epsilon_y^m$ ,  $\gamma_{xy}^m$ ,  $\epsilon_x^b$ ,  $\epsilon_y^b$ ,  $\gamma_{xy}^b$ ,  $\gamma_{xz}$ ,  $\gamma_{yz}$ ,  $\epsilon_h^1$ ,  $\epsilon_h^2$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## QPPS

This element is similar to Q4GR.

The element has 10 components of stress (SIG), organized as follows:  $\sigma_x^m$ ,  $\sigma_y^m$ ,  $\tau_{xy}^m$ ,  $\sigma_x^b$ ,  $\sigma_y^b$ ,  $\tau_{xy}^b$ ,  $\tau_{xz}$ ,  $\tau_{yz}$ ,  $\sigma_h^1$ ,  $\sigma_h^2$ , where the first three components are the membrane contributions, the second three components are the bending contributions, the next two components are the transverse shear contributions (note that the order in which these last two components are given is opposite to what is usually found in 3D continuum elements, for example) and the last two components are anti-hourglassing (pseudo-)stresses. The total strains (EPST) follow the same organization:  $\epsilon_x^m$ ,  $\epsilon_y^m$ ,  $\gamma_{xy}^m$ ,  $\epsilon_x^b$ ,  $\epsilon_y^b$ ,  $\gamma_{xy}^b$ ,  $\gamma_{xz}$ ,  $\gamma_{yz}$ ,  $\epsilon_h^1$ ,  $\epsilon_h^2$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## Q4GS

4-node shell element (Batoz), with 4 integration points in the plane and 5 integration points through the thickness for plasticity.

It is a simplified version of Q4G4 with 4 integration points in the plane.

The element has 8 components of stress (SIG), organized as follows:  $\sigma_x^m, \sigma_y^m, \tau_{xy}^m, \sigma_x^b, \sigma_y^b, \tau_{xy}^b, \tau_{xz}, \tau_{yz}$ , where the first three components are the membrane contributions, the second three components are the bending contributions and the last two components are the transverse shear contributions (note that the order in which these last two components are given is opposite to what is usually found in 3D continuum elements, for example). The total strains (EPST) follow the same organization:  $\epsilon_x^m, \epsilon_y^m, \gamma_{xy}^m, \epsilon_x^b, \epsilon_y^b, \gamma_{xy}^b, \gamma_{xz}, \gamma_{yz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## RL3D

Two-node nonlinear spring element to model paraseismic supports ([945]). This element has no mass and can have a zero length when using RESG material.

## BSHT and BSHR

The availability of the bushing element family allows to define generalized stiffness and damping between two nodes. The implemented model provides a first type of element, BSHT, with only translation degrees of freedom (available both in 2D and in 3D), and a second type, BSHR, with rotational degrees of freedom too.

All the characteristics of the bushing element are defined using "JOINT PROPERTIES" material type.

## SH3D

This element is used to connect a slave node to a master edge of shell. Three kinematic constraints are introduced on the translational and rotational degrees of freedom of the slave node. The displacements and rotations of the slave node are linearly interpolated between the two master nodes. These elements are defined in the topology but they work only through the "LIAISON" directive.

## SH3V

This element is used to connect a slave node to a master edge of element. It is the same as for the SH3D element except that there is no constraint on rotations. These elements are defined in the topology but they work only through the "LIAISON" directive.

## MAP3 and MAP4

This element is used in order to glue one slave node to a master face. The master face is triangular in the case of the MAP3 and quadrangular in the case of the MAP4. 3 kinematic constraints are introduced in order to impose the translation dof of the slave node. These elements can be used in order to glue 2 volumic meshes. These elements are defined in the topology but they work only through the "LIAISON" directive.



## MAP6 and MAP7

This element is used in order to glue one slave node to a master shell face. The master face is triangular in the case of the MAP6 and quadrangular in the case of the MAP7. 3 kinematic constraints are introduced in order to impose the translation dof of the slave node and 3 kinematic constraints are added on the rotational dof. These elements can be used in order to glue 2 shell meshes. These elements are defined in the topology but they work only through the "LIAISON" directive.

## INT6 and INT8

The INT6 (triangular prism) and INT8 (hexahedron) elements are pure displacement interface elements (also called *cohesive elements*) dedicated to the modeling of interlayers, separating "standard" structural elements. In the particular case of a composite model, these elements can be considered as representing a homogeneous resin layer ensuring the interlaminar stress transfer between adjacent plies. This approach is most often referred to as "mesoscopic" laminate modeling.

## ASHB

8-node thick shell element obtained starting from the 8-node brick. This element is identical as SHB8 but follows the assumed strain formulation. The 2 faces of this element are formed by the nodes: 1, 2, 3, 4 for the first face and 5, 6, 7, 8 for the second face.

## Q4MC

4-node multilayered shell element which is a generalization of the Q4GS element. This element is also multi-material. The number of Gauss point in the thickness depends on the number of plies. The user has to define the total number of Gauss points in the thickness using the parameter NGPZ in COMP (resp. SAND).

## T3MC

3-node multilayered shell element which is a generalization of the DST3 element. This element is also multi-material. The number of Gauss point in the thickness depends on the number of plies. The user has to define the total number of Gauss points in the thickness using the parameter NGPZ in COMP (resp. SAND).

The element has 8 components of stress (SIG), organized as follows:  $\sigma_x^m, \sigma_y^m, \tau_{xy}^m, \sigma_x^b, \sigma_y^b, \tau_{xy}^b, \tau_{xz}, \tau_{yz}$ , where the first three components are the membrane contributions, the second three components are the bending contributions and the last two components are the transverse shear contributions (note that the order in which these last two components are given is opposite to what is usually found in 3D continuum elements, for example). The total strains (EPST) follow the same organization:  $\epsilon_x^m, \epsilon_y^m, \gamma_{xy}^m, \epsilon_x^b, \epsilon_y^b, \gamma_{xy}^b, \gamma_{xz}, \gamma_{yz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## CUVF

3D cubic finite volume element. The finite volume is defined as cell centred. Several options for the calculation can be chosen with OPTI VFCC.

**PRVF**

3D prism finite volume element. The finite volume is defined as cell centred. Several options for the calculation can be chosen with OPTI VFCC.

**TEVF**

3D tetrahedral finite volume element. The finite volume is defined as cell centred. Several options for the calculation can be chosen with OPTI VFCC.

**PYVF**

3D pyramid finite volume element. The finite volume is defined as cell centred. Several options for the calculation can be chosen with OPTI VFCC.

**LIGR**

This element has several nodes with six degrees of freedom. The first node belongs to a shell and the following ones belong to a beam. It allows to use constitutive equations for following two mechanisms :

- ARTI TGGR
- ARTI CRGR

**COQI**

3 node triangular plate element.

This element can be used to model 3D plates or shells (by plane facet approximation). It is integrated through the thickness. The element can undergo large displacements and large rotations as a whole (rigid body), thanks to a co-rotational formulation, but is limited to small strains. In particular membrane strains should remain small (maximum a few %).

The element has 4 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

**FLU3**

8 node specialised element for compressible fluids.

The same remarks apply as for FLU1 in 2D. The element can be degenerated to represent a prism (6 nodes), a pyramid (4 nodes), or a tetrahedron (4 nodes) by suitable repetition of node numbers in the topology.

**PFEM**

This element is used to represent a 2D (or 3D) continuum (usually a fluid) by means of the Particle Finite Element method (PFEM).

**ADC8**

8 node brick for advection-diffusion problems.

This element is used to solve advection-diffusion problems in incompressible fluids with heat transfer according to JRC models.

**CL32**

4-node boundary condition for the CQD4.

These elements must be attached directly to the CQD4, i.e., they share the same nodes.

**CL33**

9-node boundary condition for the CQD9.

These elements must be attached directly to the CQD9, i.e., they share the same nodes.

**FL34**

4-node tetrahedron for compressible fluids. Is an alternative to the degeneratable FLU3 hexahedron.

**FL35**

5-node pyramid for compressible fluids. Is an alternative to the degeneratable FLU3 hexahedron.

**FL36**

6-node prism for compressible fluids. Is an alternative to the degeneratable FLU3 hexahedron.

**FL38**

8-node hexahedron for compressible fluids. Is an alternative to the degeneratable FLU3 hexahedron.

**CL3I**

Boundary conditions of 3 nodes.

Recommended for use with COQI triangular shell elements and in general with all 3D Ispra models. This element automatically recognizes the element to which it is attached and uses the most appropriate pressure discretization.

**CQD4**

4-node quadrilateral degenerated shell element (Hughes-Liu).

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

### CQD9

9-node quadrilateral degenerated shell element (Hughes-Liu).

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

### CQD3

3-node triangular degenerated shell element (Hughes-Liu).

Similar to CQD4 but with a triangular shape.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

### CQD6

6-node triangular degenerated shell element (Hughes-Liu).

Similar to CQD9 but with a triangular shape.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z \approx 0, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

### CLD3

3-node boundary condition element for CQD3.

### CLD6

6-node boundary condition element for CQD6.

### CL3Q

Boundary conditions of 4 nodes.

Recommended for use with 3D Ispra models. This element automatically recognizes the element to which it is attached and uses the most appropriate pressure discretization.

### MC34

Finite-volumes: 4-node tetrahedron for multicomponent flows. This element is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC. For more information on this element, see reference [135].

### MC35

Finite-volumes: 5-node pyramid for multicomponent flows. This element is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC. For more information on this element, see reference [135].

### MC36

Finite-volumes: 6-node prism for multicomponent flows. This element is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC. For more information on this element, see reference [135].

### MC38

Finite-volumes: 8-node hexahedron for multicomponent flows. This element is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC. For more information on this element, see reference [135].

### MS38

Finite-volumes: 8-node hexahedral MACRO spectral element. This element is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC.

The integration points coincide with the Gauss-Lobatto-Legendre points and are determined by specifying the MICRO spectral elements S38.

### S38

8-node hexahedral MICRO spectral element.

This element is used only to specify 'internal' nodes of an MS38.

### FUN3

This is a specialized element for the representation of cables in 3D space, in conjunction with the FUNE material (it resists only in traction, not in compression). When used with the VM23 material, it represents a bar (which resists both in traction and in compression). The element is large-strain.

The element has 4 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y \approx 0, \tau_{xy} = 0, \sigma_z \approx 0$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy} = 0, \epsilon_z$ .

### DEBR

1-node debris particle element.

This is a specialized element for the representation of flying debris, as e.g. resulting from an explosion or an impact, by means of spherical particles. It may be used both in 2D and in 3D.

## C272

This element can be used for precise modelling of continua. It can undergo arbitrarily large deformations. Since it is underintegrated, it is locking-free, but it may occasionally suffer from mechanisms if boundary conditions are too loose. In such cases, use of the C273 element (which, however, is more expensive) is recommended.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## C273

This is the fully-integrated version of the C272 element. Its use is only recommended when mechanisms might occur.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## CL92

9-node boundary condition element for C272.

## CL93

9-node boundary condition element for C273.

## C81L

8-node hexahedron with reduced spatial integration (1 Gauss Point). This element can be used for precise modelling of continua. It can undergo arbitrarily large deformations. Since it is underintegrated, it is locking-free, but it may suffer from mechanisms. In such cases, use of the C82L element (which, however, is more expensive) is recommended.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

## C82L

This is the fully-integrated version (8 Gauss Points) of the C81L element. Its use is recommended when mechanisms might occur.

The element has 6 components of stress (SIG), organized as follows:  $\sigma_x, \sigma_y, \tau_{xy}, \sigma_z, \tau_{yz}, \tau_{xz}$ . The total strains (EPST) follow the same organization:  $\epsilon_x, \epsilon_y, \gamma_{xy}, \epsilon_z, \gamma_{yz}, \gamma_{xz}$ . Note that, as concerns the shear strains, the engineering values  $\gamma$  and not the tensor values  $\epsilon$  are used, with  $\gamma_{ij} = 2\epsilon_{ij}$ .

### 3.2 SANDWICH (MULTI-LAYER) ELEMENTS

Some shell elements developed at Ispra may be defined as a sandwich (an assembly) composed of several layers, each one having its own material. The usual hypothesis that fibers (straight lines across the thickness of an undeformed shell) remain straight during deformation is retained. The fiber may or may not be/remain normal to some ‘mean’ or ‘reference’ shell surface depending on the theory (Kirchhoff or Mindlin) assumed, i.e. on the fact that transverse shear strains are taken into account or not. As a consequence of fibers remaining straight, the deformation assumes a simple pattern through the thickness. In sandwich elements the state of stress may be discontinuous at layer interfaces because the different materials have in general different properties. No detachment (delamination) of the various layers is modelled at present.

These models are useful e.g. for representing reinforced concrete structures, or other composite materials (sandwich structures).

For the moment, this feature is available for elements of type ED01 in 2D and elements of type COQI, CQD3, CQD4, CQD6, CQD9, T3MC, Q4MC, Q4GS, Q4GR, QPPS, T3GS, DKT3 in 3D.

In order to use these models, see the **SAND** directives in the Geometry (page C.45) and in the Materials (page C.1110) Sections of the manual.

#### 3.2.1 LOCATION AND NUMBER OF THE INTEGRATION POINTS

When using sandwich elements, the number of layers and of integration points through the thickness in each layer is specified by the user and may therefore vary from test case to test case. In order to facilitate the use of these elements, the following rule has been chosen:

**For sandwich elements, the numbering of the integration points proceeds along each fiber (through the thickness) first, and from the lower to the upper part of the fiber**

The lower and upper element surfaces are defined by element numbering and the right-hand rule, as usual in EUROPLEXUS. The above numbering scheme is called ‘fiber-first’, as opposed to ‘lamina-first’ numbering schemes.

As an example, consider an element with two fibers, i.e. two integration stations in the element’s plane (sometimes called lamina) and 5 integration points through the thickness. Then, the points numbered 1 to 5 belong to the first fiber, while points 6 to 10 belong to the second fiber. Furthermore, points 1 and 6 are the bottom ones, 3 and 8 the middle ones and 5 and 10 the top ones, and so on.

For ease of reference, the precise numbering schemes for elements susceptible of being multi-layered is given below.

#### ED01 element

The numbering scheme is fiber-first (i.e. identical) for both the old (until August 1995) and the new (homogeneous or multilayered) element.

### **COQI element**

The unlayered element used until August 1995 an unusual numbering rule where the outer integration points were numbered first, then the intermediate points and finally a (single) point in the mean surface (see the Technical Note: “A Triangular Plate Element for the Nonlinear Dynamic Analysis of Thin 3D Structural Components”, reference [87]). The element had 13 points altogether.

The new numbering rule is fiber-first, and is the same for both the unlayered and the layered element. For the unlayered element, 3 fibers of 5 points each (15 points altogether) are assumed, while in the multilayer element the fibers are still 3 but the number of points through the thickness may vary.

### **CQDx elements**

In the versions before August 1995, (unlayered element) a ‘lamina-first’ numbering rule was assumed. Along each lamina, points were numbered along the  $\eta$  direction first, then along the  $\xi$  direction (these directions as well as the lower and upper faces of the elements are uniquely defined by the numbering of the element nodes). The number of points through the thickness was chosen by the user.

In the current version, for both the homogeneous and the multilayer elements, integration points are numbered fiber-first and of course the number of points through the thickness is still variable.

### **Q4MC and T3MC elements**

Those element are a generalization of the Q4GS and DST3 elements. The number of total Gauss point through the thickness must be defined with the NPGZ parameter in the dimensioning section.



## 4 GROUP A—PROBLEM TYPE AND DIMENSIONING

**Object:**

The keywords of this group enable the definition of the problem, and the dimensioning (memory allocation) of the computation.

**Comments:**

These keywords are described in detail in the following pages.

## 4.1 TITLE AND PRELIMINARY INFORMATION

### 4.1.1 TITLE

The title, composed of a maximum of 72 characters, is the first card of the data set and is compulsory.

The contents of this data card is printed at the top of each page edited by EUROPLEXUS together with the date of the run.

### 4.1.2 INPUT DATA ECHO AND INPUT CHECK UP

#### **Object:**

These keywords are used to obtain an echo on the terminal or console window of the input data being precessed and to check up the syntactical correctness and the coherence of the data.

#### **Syntax:**

`< $[ "ECHO" ; "NOEC" ]$ >`

#### **Comments:**

The "ECHO" optional keyword produces an echo on the screen or terminal of the input file directives as they are being processed. If the input file is very long, this may be annoying. By default no echo is produced. See also the option OPTI ECHO, Page H.50, which may be used at any point of the input file after the dimensioning.

The "NOEC" optional keyword disables the echo on the screen or terminal of the input file directives as they are being processed. By default no echo is produced. See also the option OPTI NOEC, Page H.50, which may be used at any point of the input file after the dimensioning.

## 4.2 INTERACTIVE (FOREGROUND) EXECUTION

### Object:

The **CONV** directive can be used to execute EUROPLEXUS **interactively**, i.e. in the foreground (as opposed to the default **batch** or background execution). In this execution mode, the user pilots the advancement of the computation, and the results at selected time instants can be either visualized on graphic screens of various types (**on-screen** rendering), or be stored in graphic files of various types (**off-screen** rendering), including animation files.

### Syntax:

```
< "CONV"    $[ "TEKT" ; "WIN" ; "PS" ; "MIF" ]$ >
```

#### TEKT

Tektronix 4014 screen (PLOT-10 graphics language).

#### WIN

MS-Windows graphics (QuickWin or OpenGL). Note that OpenGL may be supported also on non-Windows platforms, e.g. on Linux.

#### PS

PostScript (but see also the TRAC PS interactive command).

#### MIF

FrameMaker MIF (but see also the TRAC MIF interactive command).

### Comments:

When interactive execution is chosen, EUROPLEXUS reads the input data-set as usual, performs step 0 to initialise the computation, then prompts the user for commands from the keyboard with the phrase: **COMMANDE ?**.

The user can then issue various commands and subcommands **typically from the keyboard** in order to pilot the computation. For example, he can ask the program to perform a certain number of steps, then to pause again for further commands. Each time the calculation is paused, the current computational model can be visualized (e.g. by means of the built-in OpenGL-based visualization module) and information concerning the computation (time step, CPU time, etc.) can be printed. Furthermore, the current time step can be varied by the user.

As an alternative to typing commands by hand from the keyboard, such commands may be included in the regular EUROPLEXUS input file by enclosing them into a special directive **PLAY ... ENDPLAY** as described in Section 13.6 (Page I.24) and in Section 14.7 (Page ED.140).

All EUROPLEXUS interactive commands are described in detail in Section 15 (Group O).

## 4.3 FILE MANAGEMENT

This Section gives some information about the management of files related to the EUROPLEXUS program.

### 4.3.1 DEFAULT FILE NAMES

The code tries whenever possible to use default values both for the file names that it needs during the calculation, and for the associated logical unit numbers.

The idea is that logical unit numbers (a concept specific only to FORTRAN) are irrelevant for the user and should be totally transparent to him or her. What *does* matter for users is file *names*.

In many cases the code helps users by providing default values for such names (the behaviour may depend on the platform, though, see below). Of course, users are free to choose file names (and even unit numbers) if for any reason they find such defaults inconvenient.

#### Default names under MS-Windows

Under the MS-Windows platform default file names are built automatically by the code by using the base name of the (main) input file used in the run.

For example, suppose that the main input file is called `test01.epx` and resides on a directory `D:\Work`. The user may launch the program for example by opening a console window on the directory `D:\Work` (i.e., such that `D:\Work` is the current directory) and by typing a command such as:

```
epx_bench -l test01
```

The actual command may vary depending on the implementation.

The code interprets the name passed on the command line (after removing any options such as the `-l` above) as the name of the main input file. It removes the extension `.epx` from this name, if present, and uses the resulting string as the base name for default file names. Thus, the user may give a full file name, complete with its path, if preferred.

Suppose now that for run `test01` the user needs to read a CASTEM 2000 mesh and wants to produce results for ALICE TEMPS and CASTEM 2000.

The first task may be accomplished by the `CAST` directive (see page A.30). If the mesh file is formatted, and the global (main) mesh object is called `model`, this directive may take any of the following forms:

1. `CAST FORM 'test01.msh' model`
2. `CAST FORM 9 model`
3. `CAST FORM model`

In the first form, the file name containing the mesh is given explicitly. The code associates this name to the default unit number, which for CASTEM mesh reading is number 9.

In the second form, the unit number is given explicitly. The code uses the given number as unit number and opens a file without specifying its name. This results in different behaviour depending on the platform. Under MS-Windows, a file `fort.9` is opened.

In the third form neither the name nor the unit number are specified. The code uses the default file name (`test01.msh` in this case) and the default unit number (9).

It is clear that, whenever possible, the third form is preferable. This requires, however, that the mesh file name matches the main input file name (and is on the same directory).

When either of these conditions is impractical, the first form should be used by preference (name chosen by the user, unit number chosen by default by the code).

The second syntax is obsolete and should be avoided in new inputs.

The task of producing results files may be accomplished similarly, by the ECRI directive (see page G.70). For example, for the ALICE TEMPS output, this directive may take any of the following forms:

1. ECRI FICH ALIC TEMP 'test01.alt' /CTIME/ . . .
2. ECRI FICH ALIC TEMP 11 /CTIME/ . . .
3. ECRI FICH ALIC TEMP /CTIME/ . . .

### Default names under Unix

Under the Unix platform(s), the same concepts apply as seen above.

Let us assume for example that the main input file is called `test01.epx` and resides on a directory `/u/user/My_dir`. The user may run the program starting from his work directory (`My_dir`) by the command:

```
europlexus test01.epx
```

The mesh file will be sought in this directory under the name `test01.msh`. If some other input files are required, they will also be sought in this directory, under the same name but a different extension. The same rule applies to output files, in particular to the listing file.

However, for reasons related to efficient exploitation of disk space, output files may of course be directed to special directories. It is therefore recommended to contact your system administrator to learn about local disk space policies.

### Alternatives under MS-Windows

Under the MS-Windows platform, there exist other ways of launching the program, alternative to the one (command line) assumed in the above example.

For example, a user might double click on the EUROPLEXUS executable or on an icon (shortcut) to this executable, or on the input file `test01.epx` (provided file association is used), etc.

Whenever a file name is not directly available (like in the first two alternative methods listed above), the program prompts interactively the user for such a name. The behaviour is thereafter identical to the case of command-line execution.

### Comments:

Currently, default file names are available for the following directives:

- Reading a CASTEM 2000 or GIBI mesh, GIBI and CAST keywords (see page A.30);

- Writing result files, **ECRI FICH** keywords (see page G.70);
- Reading a results file (post-treatment by EUROPLEXUS), keyword **RESU** (see page ED.10).
- Reading a modal basis file in a multi-domain computation (see **STRU** directive on page I.15).

### 4.3.2 EXPLICIT FILE OPENING

#### Object :

This directive allows to open a file directly from the input data, by specifying its logical unit number and its name.

It may be used whenever either the user wants to override a file choice made by default by the code (example: a very long calculation requires its results file to be written to a remote disk, due to space problems), or to force opening of a unit that is otherwise not opened by the code.

Note, however, that including this directive in input files generally renders them not portable across different platforms or even different machines belonging to the same platform. In fact, absolute (full) file names are usually required.

For this reason, it is preferable to use the short file name syntax (or even the default name syntax) described in the previous sections whenever possible.

#### Syntax

1. Open a generic file:

```
"OPNF" < "FORMAT" > nfic 'nomfic' ;
```

2. Open an XDR file in read or write mode:

```
"OPNF" $ WXDR ; RXDR $ nfic 'nomfic' ;
```

3. Create a results directory:

```
"OPNF" $ "PATH"; "DRST" ; "DPRV" $ $ 'nomdir' ; "PWD0" $
```

#### FORMAT

Specify that a formatted file is required. By default, the file is unformatted.

#### WXDR

Specify that a writeable XDR file is required. For the output "K2000" file for example.

#### RXDR

Specify that a readable XDR file is required. For the input mesh file for example.

#### nfic

Logical unit number of the file.

#### 'nomfic'

File name.

#### PATH, DRST, DPRV

Any of these keywords (which are aliases) specifies that the following name is that of a directory to be created for the storage of calculation results. This allows, among other uses, to un-clutter the current directory where the calculation is performed, for those type of outputs that produce many results files (typically ALIC split format and ParaView).

'nomdir'

Results directory name.

PWD0

The directory will be the current directory.

### Comments :

It is forbidden to open explicitly the logical unit numbers 0, 5, 6 and 7, that are reserved in most operating systems. Other unit numbers reserved by EUROPLEXUS are 15, 16 and from 91 onwards.

The way of coding the file name depends upon the operating system:

- Under Unix, the full name must be given, for example: /u/user/ssrep/fich.sortplex.
- Under MS-Windows the full path name is recommended, like under Unix. For example: D:\Users\Epx\myfile.abc.

For other operating systems, please contact your system administrator.

### Results sub-directory

To unclutter the current directory from results files, a typical usage would be as follows. Suppose we are running a test named `mytest.epx` in a certain directory. The test produces split ALIC files and PVTk files. We want to place all these results files (which are potentially very numerous) in a sub-directory called `mytest.res` of the current directory, rather than directly in the current directory. To this end, the input would be the following:

```
MYTEST
. . .
OPNF PATH 'mytest.res' ! Redundant if sub-directory exists, but safe
. . .
ECRI . . .
FICH SPLI ALIC 'mytest.res\mytest.ali' ...
FICH PVTk 'mytest.res\mytest.pvd' ...
```

The above OPNF command is only necessary if the sub-directory `mytest.res` does not exist yet, and it will create it. Otherwise, the command would do nothing and it can be omitted (but leaving it is no harm).

The presence of back-slash characters as separators in the path names indicates that we are under Windows. At the end of the run, in the sub-directory `mytest.res` one will find the `.ali` files, the `.pvd` file and the `.vtu` files:

```
mytest.res\mytest_0000.ali
mytest.res\mytest_0001.ali
. . .
mytest.res\mytest.pvd
mytest.res\mytest.0001.0001.vtu
mytest.res\mytest.0001.0002.vtu
. . .
```



The ParaView results can be visualized by directly opening with ParaView the `.pvd` file from the sub-directory where it resides.

### 4.3.3 EXPLICIT OUTPUT DIRECTORY DEFINITION

#### Object :

This directive allows to define a directory for output files.

Like for EXPLICIT FILE OPENING, the rules for the name of the given directory may vary from one platform to another.

#### Syntax

```
"OPNF"  "DRST"  |[ 'nomdir' ; "PWD0" ]|
```

#### DRST

Following by the name of directory used for named result files (ALICE, ALICE TEMPS, K2000, INP, VTK-PARAVIEW, ...). Alias : DPRV or PATH.

'nomdir'

Name of directory used for result files.

#### PWD0

Directory name is the current directory.

#### Comments :

In the case of an UNIX system, the defined directory is an absolute path. In the case of a WINDOWS system, the directory is given as a relative path.

In both cases, if given directory does not exist, it is created.

## 4.4 TYPE OF MESH, PROBLEM AND LISTING

### Object:

1/ To define the mesh type that will be used:

- Mesh in free format.
- Mesh generated by GIBI or CAST3M (uses objects).
- Mesh generated by I-DEAS Version 6 or Master Series (uses so-called “permanent groups”).
- Mesh described in a MED file
- Mesh generated LS-PrePost (free pre-processor of LS-Dyna)

2/ To define the general type of computation:

- Axisymmetric, 2-D, 3-D, 1-D, etc.
- Lagrangian or Eulerian or A.L.E. formulation.
- Presence of mechanical rezoning for ALE computations.

3/ To define some general options about the form of printed results (useful to reduce the size of the output listing in extremely large test cases)

### Syntax:

```

$[
  $[ "GIBI" ; "CASTEM" ]$ <$[ "FORM" ; "XDR" ; "BINA" ]$>
                                <$[ ndis ; 'file_name' ]$> 'nomobjet' ;
  "IDEA" <$[ ndidea ; 'file_name' ]$> <"REWR" > <"MAPP" > ;
  "MEDL" 'file_name' ;
  "LECM";
  "KFIL" 'file_name' ;
]$

|[ "DPLA" ; "CPLA" ; "AXIS" <"HOLE" hole> ; "TRID" ]|

< $[ "LAGR" ; "EULE" ; "ALE" ]$ >

< "NAVIER" >    < "HOMO" nhtube >
< "MBETON"      nssc      >
< "FRQR"        nfrqr     > < "SPCO" sphcon >
< "LAGC"        >
< "MBACON"      < "POST" > >
< "SAUVEGARDE" ... >    < "REPRISE" ... >

< "ADDF" < "NAVS" > < "TEMP" > < "TURB" > >

```

```

< "MECA" >
< "MEDE" <"NEPX"> <"ONEF"> >
< "EROS" <ldam> <CROI> <LIMI> >
< "RISK" < "PROB" |[ "FERR" ; "YETP" ]| >
      < "LUNG" |[ "BAKE" ; "LEES" ]| >
      < "SPLI" > >

< "SCLM" ... >
< "BMPI" >
< "ADAP" "MLVL" rlv1 < ( 'nomelm' rle1 ) > >
< "CFVN" >

```

## GIBI

Mesh generated by GIBI (GIBI objects will be read to define the mesh) and stored with the 'SORT' directive (see Comments below).

## CASTEM

Mesh and other characteristics (geometrical, material, champoints) generated by CASTEM2000 and stored with the 'SAUV' directive (see Comments below).

## FORM

The CAST3M generated data are to be read in formatted (ASCII) mode (this is the default).

## XDR

The CAST3M generated data are to be read in XDR mode.

## BINA

The CAST3M generated data are to be read in binary mode.

## ndis

Number of the logical unit of the mesh file.

## 'file\_name'

Complete path localising the mesh file, under Unix operating systems. If both this and the unit number ndis are omitted, the code chooses a name and a unit number by default (see page [GBA\\_0027](#)).

## 'nomobjet'

Name of the whole (main) object meshed by GIBI.

## IDEA

Mesh generated by I-DEAS. At the moment, only formatted IDEAS files are allowed in input.

## ndidea

Number of the logical unit of the mesh file.

`'file_name'`

Complete path localising the mesh file, under Unix operating systems. If both this and the unit number `ndis` are omitted, the code chooses a name and a unit number by default (see page [GBA.0027](#)).

REWR

Write a new I-DEAS file with re-ordered numbering (no holes), see details below.

MAPP

Stop run after re-writing the new I-DEAS file with re-ordered numbering, see below for details.

MEDL

Mesh described in a MED file.

LECM

Allow an automatic deletion of double nodes or double elements in a MED "LECT" procedure. !!! It should be notice that this procedure will perform a new ordination of readen elements or nodes. It is not compatible with the "RACCORD BIFU" method. !!!

`'file_name'`

Full path of the med file (this file contains the mesh). The key word OPEN followed by a unit number is not available to open a MED file.

KFIL

Mesh described in a LS-DYNA k-file.

`'file_name'`

Full path of the k-file file (this file contains the mesh).

AXIS

Axisymmetric computation.

HOLE

Optionally, for axisymmetric cases a central 'hole' may be specified by this keyword: in this case the given hole radius is automatically added to the mesh radial coordinates given in input (this is a shorthand alternative to providing a mesh with the actual hole in it).

DPLA

Two-dimensional plane strain computation.

CPLA

Two-dimensional plane stress computation.

TRID

Three-dimensional computation.

**LAGR**

Computation with the Lagrangian formulation (default). All nodes are Lagrangian. The GRIL directive is not required.

**EULE**

Computation with the Eulerian formulation. All nodes are Eulerian. The GRIL directive is not required.

**ALE**

Computation with the A.L.E. formulation. The GRIL directive may be used to specify the motion of nodes (see [GBINT\\_0018](#)).

**NAVIER**

One-dimensional uncoupled calculation (fixed pipelines), with incompressible or nearly incompressible fluids. The calculation may only be done in Eulerian. Therefore, it is mandatory to specify also the directive "EUL E".

**nhtube**

Maximum number of tubes per unit cell (homogenised material).

**nssc**

Number of layers for the CMC3 element with the BETO material. By default, there is just one layer per element. This number must be between 1 and 20.

**FRQR nfrqr**

Frequency of searching neighbour nodes. By default = 1. This option, valid only for calculations with the NABOR or SPH methods, allows the user to considerably reduce the calculation time.

**SPCO sphcon**

Maximum number of retained structure faces in contact with the same SPH particle. By default it is 1, meaning that only the first contacting face is treated (the penetrated face which is closest to the particle). The value of **sphcon** must be less than or equal to the value of parameter NBCPOS, which is hard coded in MPEF3D (currently NBCPOS = 6).

**LAGC**

This keyword specifies that the contact forces due to impact and sliding will be computed implicitly by the Lagrange multipliers method. This enables the coupling with the permanent connections (relations, boundary conditions, etc.)

**MBACON**

This keyword specifies that the characteristics of multi-layer homogenised elements will be read from a BACON file (see page [GBC\\_0165](#)).

**POST**

This option of MBACON indicates that the rupture criteria will be evaluated in the multi-layer elements during the direct calculation. During post-treatment, this option allows to compute the maximum deformations (upper and lower "skin"), and also the deformations along the directions of strain gauges.

**SAUV**

The directives for saving and restart are described in Section "SR" (see page [GBSR\\_0010](#) and following).

**ADDF**

Advection-diffusion computation. The Eulerian description is used, therefore the "EULE" keyword is compulsory in this case. See description on page [GBA\\_0031](#).

**NAVS**

Solution of Navier-Stokes equations (for fluid velocities) has to be performed in the advection-diffusion computation. See description on page [GBA\\_0031](#).

**TEMP**

Solution of the temperature equation has to be performed in the advection-diffusion computation. See description on page [GBA\\_0031](#).

**TURB**

Solution of the turbulence equations (k, eps) has to be performed in the advection-diffusion computation (this option is still under development).

**MECA**

ALE computation with mechanical rezoning model.

**MEDE**

to create a med file.

**NEPX**

By default, if the input mesh is a MED file, the output MED file uses the MED mesh numbering. The NEPX option forces the output MED file to use the EPX numbering that is different. See description on page [GBG\\_0070](#).

**ONEF**

By default, for a EPX field (CONT, ECRO, EPST), one MED field is created by mesh type and by material. This option allows to creates an single MED field by EPX field.

**EROS**

This keyword activates the "erosion" algorithm of the code. The algorithm has the following characteristics:

1- Those elements whose erosion criterion (mainly number of failed Gauss points where failure is triggered by damage, principal strain, minimum pressure, ...) is beyond a certain level are considered as eroded and are ignored during the rest of the calculation. Erosion can activated in general for all structural elements. The failure criterion of one Gauss point can be defined via a global command in COMP (page [GBC\\_0069](#)), or in some materials.

Erosion can also be defined using displacement erosion (see [GBC\\_0067](#)) or using a minimum time step size of an element (see page [GBI\\_0020](#)).

2- In the case of a calculation with contact by sliding surfaces in 3D (see page [GBD\\_0180](#)), the contact surfaces are updated by eliminating the eroded elements.

3- The list of the eroded elements is stored in the results file. This allows to remove them from the visualization (if so desired) during the post-treatment.

4- The erosion can also be activated for a part of the elements (see page [GBC\\_0069](#)).

5- Obsolete are the keywords `FAIL` and `GHOS`.

#### `ldam`

Optional parameter indicating the number of failed Gauss points that cause erosion of the element, in proportion to the total number of Gauss points of an element. It should lie between 0 and 1. The default value is 1. Negative values indicate no erosion will taken into account for material erosion. The value 1.0 indicates that an element is eroded when *all* its Gauss points have failed. The value 0.5 indicates that an element is eroded whenever approximately half of its Gauss points have failed. The special value 0.0 may be used to indicate that an element is eroded when any one of its Gauss points fails. This value is global and is used (as a default) for all elements in the current calculation. However, specific material types (see e.g. `LEM1`) may contain parameters that allow to override this value. In this way the user may set different values of the erosion threshold in different parts of the model (e.g. low values or even 0.0 for very brittle materials such as glass, and high values for ductile materials such as metals).

#### `CROI`

This option allows the erosion of crossed elements (negative Jacobian matrix/negative volume). The calculation is not stopped in such case.

#### `LIMI`

Special limitation for the element erosion: 2D: only elements, which have not 3 nodes connected to already failed elements can fail. This avoids large zone with eroded elements.

#### `RISK`

This keyword activates the calculation of risk analysis related to explosive events. Note that to perform this type of analysis, it is mandatory to use standard measurement units. In particular, the pressures must be expressed in Pa. This is because the model internally uses some non-dimensionless constants. Furthermore the model assumes that the atmospheric pressure has the standard value of 1.D5 Pa, i.e. 1 bar. The risk estimation is performed according to the two references listed below. In order to compute the total probability from the probit functions, two different approaches can be chosen using the keyword `PROB`. A very conservative probability function from Yet-Pole can be activated with `YETP`. The more realistic probability function of Ferradás can be chosen with `FERR` (this is the default in case no `PROB` is defined). Two different formulations for risk of lung haemorrhage can be activated using the keyword `LUNG`. The default is equation (7, Baker, `BAKE`) from Ferradas paper. Equation (9, Lees, `LEES`) doesn't consider the impulse and is conservative. Note that thre optional sub-directives `PROB ...` and `LUNG ...` must be re-defined in case of results reading from an Alice file. This allows to perform a calculation, say, with the default values (i.e. by specifying only the `RISK` directive, possibly followed by `SPLI` if so desired), and then to do several post-processing of the results each time computing (and visualizing) the risk by using a different set of sub-options (e.g. once by using the Ferradas probits and another time using the Yet-Pole probits), without having



to run again the main calculation (which may be very time-consuming). Therefore, be aware that in calculations with risk it is mandatory to re-define the entire **RISK** directive before the **RESU** directive (which typically reads the results from an Alice file).

#### SPLI

The risk of death is split into three contributions: i) risk of head impact; ii) risk of body impact and iii) risk of lung haemorrhage. The resulting risk can only be visualized starting from a PVTk results file.

#### SCLM

This keyword introduces options related to memory distribution in MPI calculations, see details on Page [GBA\\_0037](#).

#### BMPI

The simulation will only be executed using a parallel MPI version of EUROPLEXUS. Any sequential version running the case will produce a clean stop just after the keyword is read, declaring any qualification as valid.

#### ADAP MLVL

[MPI Only] Automatic dimensioning for ADAPTIVITY: **rlvl** is an estimated average refinement level for all the adaptable elements of the mesh (floating point value)). It is used to compute the size of the extension zones in replacement of the **DIME ADAP** directive. Specific values can also be entered for given element types through the **rlsl** floating point parameter associated to one element name '**nomelm**'.

#### CFVN

Create the central Finite Volume nodes in Cell-Centred Finite Volume meshes.

## References

The “probit” functions for risk estimation are taken from:

- E. González Ferradás, F. Díaz Alonso, M. Doval Miñarro, A. Miñana Aznar, J. Ruiz Gimeno and J.F. Sánchez Pérez: *Consequence analysis by means of characteristic curves to determine the damage to humans from bursting spherical vessels*. Process safety and environmental protection **86** (2008), 121–129.
- I. Yet-Pole, Cheng Te-Lung: *The development of a 3D risk analysis method*. Journal of hazardous materials **153** (2008), 600–608.

## Comments:

The **CASTEM** directive is meant to read data produced by **CAST3M** and saved by the directive “**SAUV**”. It will read also objects of type ‘champoint’ explicitly stored, together with the mesh, by **CAST3M**.

On the other hand, the **GIBI** directive is meant to read only mesh objects (maillage) produced by **CAST3M** and saved by the directive “**SORT**”. No champoints may be stored by **CAST3M** with this directive, therefore they may not be transmitted to EUROPLEXUS.

The syntax in CAST3M is:

### (SORT)

Assume we have an object 'mymesh' of type 'maillage' to write on the file 'myfile.msh' on the current directory. Then :

```
. . .  
OPTI SORT 'myfile.msh' ;  
SORT mymesh ;  
. . .
```

### (SAUV)

Assume we have an object 'mymesh' of type 'maillage' and an object 'mychampnt' of type 'champoin' to write on the file 'myfile.msh' on the current directory. Then :

(formatted mode)

```
. . .  
OPTI SAUV FORM 'myfile.msh' ;  
SAUV FORM mymesh mychampnt ;
```

or:

(XDR mode)

```
. . .  
OPTI SAUV 'myfile.msh' ;  
SAUV mymesh mychampnt ;
```

or:

(binary mode)

```
. . .  
OPTI SAUV BINA 'myfile.msh' ;  
SAUV BINA mymesh mychampnt ;
```

Note that, if the data have been produced by CAST3M in formatted mode, then the keyword FORM may optionally be used in the EUROPLEXUS directive, for clarity, but it is not necessary, since this is the default reading mode for this directive.

On the other hand, if the data have been produced by CASTEM in "XDR" (resp. "BINA") mode, then the keyword XDR (resp. BINA) is mandatory in the EUROPLEXUS directive.

Note also, the XDR mode is the default mode for CAST3M output "SAUVER" file.

In the case of the "GIBI" directive, the mesh file is always formatted.

If neither the keyword "GIBI" nor "CASTEM" appear, the program assumes that the mesh is directly given in the main input file under the form of a list of coordinate values, and as many lists of node index (topology) values as there are element zones. This format is also known as the 'COCO' type format.

This form allows also to use meshes issued from mesh generators other than CASTEM-GIBI or even, in case for example the mesh is quite simple, to enter this data directly, or to generate them by an independent software. For more information, consult the 'GEOM' directive.

The type of problem directive must be specified and contain at least one keyword (at least 2 for non-Lagrangian cases), for example:

"AXIS"

or

"TRID" "ALE"

The various options are mutually incompatible.

## I-DEAS mesh

When the IDEA directive is used, the program reads the mesh from an I-DEAS 'universal file'. This file may contain other information besides the mesh, but the extra information is ignored.

The program interprets the following information: 1) nodal coordinates (dataset 781 or 2411) ; 2) element topology (dataset 780 or 2412); 3) permanent groups (dataset 752 or 2429 or 2430). Permanent groups are identified by a name, which can then be used in the EUROPLEXUS input file (like in the case of CASTEM mesh) to identify the corresponding list of nodes or elements.

Note that normally I-DEAS universal files do not have consecutive node or element numbers, while EUROPLEXUS requires consecutive numbering (and starting from 1). In order to solve this problem, use the optional REWR directive: the code reads the universal file (use extension '.unv'), re-orders the mesh numbering and writes the result in a new universal file (same name but with the characters 'new' appended). Nodes are re-ordered simply by eliminating 'holes' in the numbering. For the elements, however, a subdivision into homogeneous 'blocks' of the

same element type, material and physical properties (geometrical complements) is performed. A summary of the resulting blocks is printed on the listing.

Elements are re-ordered subdividing them into homogeneous ‘blocks’ of the same element type, material and physical properties (geometrical complements). This permits the mapping of the I-DEAS element library onto the EUROPLEXUS one, for the relation between I-DEAS and EUROPLEXUS element libraries is not unique. The ordering criterion followed by the procedure is: [material property number] - [element type number] - [physical property number]; as an example, if some elements have to be declared as the last ones (i.e. CLxx elements when present), they must be associated to the highest material number in the I-DEAS mesh.

A list of the resulting blocks is printed on the listing, containing the number of elements, the element type in I-DEAS mesh and a list of possible choices for the corresponding EUROPLEXUS element type; these informations are useful in order to set up the declaration of the geometry in the EUROPLEXUS input file (see 5.3).

The MAPP optional directive may be used in order to stop the code right after writing the re-ordered universal file.

Example 1: EUROPLEXUS input file:

```
$-----  
Example of use of IDEAS universal file: 1. file re-writing  
IDEA 'myfile.unv' REWR MAPP  
DIME TERM  
FIN
```

This input reads in universal file ‘myfile.unv’, re-orders the mesh and produces a new universal file ‘myfile.unvnew’.

It is important to note that in order to post-process the EUROPLEXUS results with I-DEAS, use should be made of the reordered universal file (myfile.unvnew in the above example).

Example 2: EUROPLEXUS input file:

```
$-----  
Example of use of IDEAS universal file: 2. actual computation  
IDEA 'myfile.unvnew'  
DIME  
...  
(problem definition, using 'permanent group' names)  
FIN
```

In this second example, the geometry is read from the re-ordered universal file, and in any successive input directive the names of permanent groups may be used to define element or node lists. The syntax is the same as with CAST3M objects.

## MED mesh

When the MEDL directive is used, the program reads the mesh from a MED file. This file may contain other information besides the mesh. If the file comes from EPX or Code\_Aster, displacements, velocities, strains, stresses and internal variables could be read and used for the initialization. (See description on page [GBE.0180](#)). Other extra informations are ignored.

From the elements families and from the nodes families the program reconstructs the elements groups and the nodes groups. Groups are identified by a name, which can then be used in the EUROPLEXUS input file (like in the case of CASTEM mesh) to identify the corresponding list of nodes or elements. The elements groups described in a MED file are homogeneous ‘blocks’ of the same geometric support.

Note that the MED elements numbers are not necessarily the EUROPLEXUS elements numbers.

A summary of the resulting blocks is printed on the listing.

### Warning

For an axisymmetric computation, the program considers a sector of ONE RADIAN. Therefore, all the forces, added masses, etc. must be defined correspondingly.

This has to be taken into account in particular when defining the mass associated to a material point: the “true” mass shall be divided by  $2\pi$ .

Example:

If the whole force is  $F_{\text{tot}}$ , the force to be introduced in EUROPLEXUS is:

$$F_{\text{plex}} = \frac{F_{\text{tot}}}{2\pi}.$$

#### 4.4.1 MODELING OF ADVECTION-DIFFUSION PHENOMENA

EUROPLEXUS includes a module for the modeling of advection-diffusion phenomena. This module stems from the TRAFLU-2D and TRAFLU-3D codes developed at JRC Ispra in the late eighties [53, 54].

The module is activated by using elements of type ADQ4 (in 2D) or ADC8 (in 3D), the ADFM material, specific generalised “loads” (see page F.320) and initial conditions (see page E.85), and specific options (see page H.70). Here is a synthetic description of these models, borrowed from [54].

The model uses a quasi-explicit finite element algorithm for the solution of the basic equations describing combined conductive and convective transfer of heat in a liquid. The presence of enclosing solid (rigid) structures is accounted for.

The governing equations in the fluid region are the incompressible Navier-Stokes equations and the thermal energy equation. These equations are treated in an Eulerian frame of reference and they are expressed in terms of primitive variables: velocity, pressure and temperature.

The flow is assumed to be laminar and the fluid Newtonian and incompressible within the Boussinesq approximation. Either the velocity components or the total surface stress are specified as boundary conditions for the Navier-Stokes equations.

The governing equation in the solid is the transient heat conduction equation. Boundary conditions are of prescribed temperature, imposed normal heat flux and heat transfer by convection or radiation.

Spatial discretization is achieved by means of four-node quadrilateral elements in 2D (ADQ4) or eight-node hexahedral elements in 3D (ADC8) with multi-linear velocity and temperature fields. The pressure is assumed uniform over each fluid element.

A fractional step method is employed for time integration of the Navier-Stokes and thermal energy equations. This consists of three distinct steps dealing, respectively, with the advective terms, the viscous/diffusion terms and the pressure/incompressibility terms.

A second-order explicit Taylor-Galerkin method is used in the advection step, where the mass matrix is retained in its consistent form to improve phase accuracy.

A first-order explicit Euler method is used in the viscous-diffusion phase. Here the mass matrix is put into diagonal form.

Finally, a first-order implicit method is used in the pressure phase for the momentum equations. The pressure field itself is obtained as solution of a linear algebraic system arising from the discrete form of the incompressibility condition.

#### 4.4.2 TYPE OF OUTPUT LISTING

**Object:**

To define the type of printed output listing. If nothing is specified, in extremely large model computations the standard listing could be very large. Therefore, it may be useful to selectively reduce the printed information via the following directives.

**Syntax:**

```
< $ "LIST"  
    | "COOR" ; "ELEM" ; "GIBI" ; "GRIL" ; "EPAI" ; "NORM" ; "NONE" |  
    "TERM" $ >
```

**COOR**

the initial nodal coordinates will be printed on the output listing; furthermore, the principal directions of inertia of COQI element nodes will also be printed

**ELEM**

the mesh topology (element nodes) will be printed on the output listing

**GIBI**

the composition of CASTEM2000 objects will be printed on the output listing

**GRIL**

the characteristics of ALE grid motion will be printed on the output listing

**EPAI**

the initial element thicknesses will be printed on the output listing

**NORM**

the FSA and FSR normals will be printed on the output listing

**NONE**

none of the above quantities will be printed on the output listing

**Remarks**

By default, i.e. in the absence of the `LIST` directive, all the above quantities are printed out in the normal way. When the `LIST` directive is encountered, all the above printouts are inhibited, i.e. the effect is the same as with `LIST NONE`. Any of the keywords `COOR ... NORM` may then be used to re-activate the printing of selected quantities. In this case, however, printing of sequences of integer numbers occurs in a “compact” way, in the sense that any sequence of four or more **consecutive** numbers  $n_1, n_2, \dots, n_n$  is listed simply as ' $n_1$  to  $n_n$ '. In many cases this allows important savings in the quantity of output data.

For example, the directive:

```
LIST NONE ELEM EPAI TERM
```

would print only the mesh topology and element thicknesses.

To obtain the most compact listing, use `LIST NONE TERM`.

Another way of obtaining a compact listing is the option `OPTI NOPR`, see Page H.50. However, that directive does not allow selective printout.



## 4.5 MPI GLOBAL OPTIONS

### Object:

Optional global options to set for MPI calculations:

- SCLM toggles aggressive memory distribution, coming with restrictions upon output and qualification options (**still under strong development**),
- BMPI prevents current dataset to be run without MPI.

### Syntax:

```
< "SCLM" <"DTUN"> <"PMET"> <"ROB" <"REGU"> <"CART"> <"CINI">
    <"WFIL" <ndwfil>> <"DACT" /LECDDL/> <"DPRE" ipre>
    <"IOPT" iopt> >

< "BMPI">
```

#### DTUN

Multiple time scales treatment (one per subdomain) is deactivated. Every subdomain has the same time scale (see comment below).

#### PMET

ParMetis library is used to perform domain decomposition.

#### ROB

Recursive Orthogonal Bisection algorithm is used to perform domain decomposition (see comment below).

#### REGU

Activates a regularizaing step for ROB domain decomposition to avoid quasi-orphans (i.e. elements sharing a majority of their faces with elements from another subdomain). The purpose is to optimize interfaces and to provide robustness for geometric operations associated with mesh adaptivity.

#### CART

MPI only. Activates an optimization step improving the splitting of elements located on located on a plane orthogonal to the actual cutting direction during ROB decomposition. Concerned grid cells must then be selectively chosen using the directive "OPTI DOMD CART" after the GEOM directive and before the MATE directive.

#### CINI

Automatic domain decomposition with ROB after a restart is performed using initial coordinates instead of current coordinates.

WFIL

Use of an element weight file for automatic domain decomposition (see comment below).

ndwfil

Number of the logical unit of the weight file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is .wgt.

DACT

Selection of active directions (from 1 to 2 in 2D, from 1 to 3 in 3D) for automatic domain decomposition using ROB.

ipre

Number of the first cutting direction for automatic domain decomposition using ROB (see comment below).

iopt

Level of memory optimization (see comment below).

BMPI

When present, the current dataset can only be run using MPI.

### Comments:

Keywords to be used with SCLM option are very close to the ones dedicated to the STRUCTURE directive with automatic domain decomposition activated (keyword AUTO, see page I.15). Indeed, to provide an optimized memory distribution, the domain decomposition has to be defined and performed before the global data structure is built and initialized, which is not the case when using the STRUCTURE directive just before launching the calculation. Only automatic domain decomposition can be defined this way. See comments on page I.15 for a complete description of keywords DTUN, PMET, ROB, CINI, WFIL, DACT, DPRE.

When activating memory optimization for MPI calculations with SCLM option, **the STRUCTURE directive must not be used**, since domain decomposition has already been defined.

IOPT keyword is used to define the level of memory optimization: the more aggressive the optimization is, the more restrictions upon output and qualification there are.

- Level 0: all centralized outputs are forbidden (ECRI directive), except listing printouts, ALICE and ALICE TEMPS files (without time splitting), distributed PVTk files (MPI keyword).
- Level 1: same as level 0, plus immediate qualification QUAL directive just after CALCUL directive also forbidden.

TIP: The right way to deal with restrictions imposed by SCLM option is to write an ALICE file during the parallel calculation within a series of time-steps of interest and then to generate the desired output files and perform the desired qualifications from this file.

## 4.6 DIMENSIONING

**Object:**

Allocation of memory for the problem variables.

**Syntax:**

"DIMENSION"

**Comments:**

The dimensioning of variables data is specified by keywords, which enable the user to reserve for a given problem only the memory that will be really necessary to perform the computation. All the dimensions are maximum values, their value by default is 0 (unless a different value is specified in the description).

On the following pages, the keywords have been classified according to the data they affect. Actually, they can be provided in any order. These keywords together form the directive "DIMENSION".

#### 4.6.1 DIMENSIONS RELATIVE TO GROUP B (GEOMETRY)

These dimensions concern the geometry (elements) and, in ALE computations, the motion of grid nodes.

##### Overview:

The overall syntax is as follows:

```

< NPOI np    > < NDDL  nd >
< "typ1" n1 "typ2" n2 ... >

< ADAP NPOI np <NIND ni> <NVFI nvfi> <NTHR nthr> <NPIN npin>
    "typ1" n1 "typ2" n2 ... ENDA >

< DECO NPOI np ENDD >

< NALE nale > < NBLE nble >

< NGPZ mxngpz >

< ME1D me1d >

```

## NODES

### Syntax:

< "NPOI" np > < "NDDL" nd >

#### np

Maximum number of mesh points. This parameter **is not necessary**, except some special cases. (Cf. comment below).

#### nd

Total number of degrees of freedom. This parameter **is not necessary**, except some special cases. (Cf. comment below).

### Comments:

Normally EUROPLEXUS detects automatically the exact number of nodes (and the number of degrees of freedom), the exact number of each element types required, from the input file or from the associated mesh file. Therefore, the directives `NPOI`, `NDDL` and `TYPi` are usually not necessary.

These directives are needed only in special cases, whereby EUROPLEXUS has to create additional nodes (not specified in the input nor in the associated mesh file) after the reading of the geometry: for example a pipeline circuit with a bifurcation, a rigid body, or in case of remeshing.

## NUMBER OF ELEMENTS

### Object:

The number of the different elements that will be used in the problem is specified (if necessary).

### Syntax:

```
| "typ1" n1 "typ2" n2 ..... |
```

**typi**

name of an element type (see page INT.80).

**ni**

maximum total number of corresponding elements.

### Comments:

Normally EUROPLEXUS detects automatically the exact number of each element type required, from the input file or from the associated mesh file. Therefore, these directives are usually not necessary. These directives are needed only in special cases, whereby EUROPLEXUS has to create additional elements (not specified in the input nor in the associated mesh file) after the reading of the geometry: for example a pipeline circuit with a bifurcation, a rigid body, in case of remeshing, or in case of flying debris.

The various elements are described on page INT 80.

### Warning :

If you use 1-D elements (except ED1D), the directives:

```
"TRID" "EULE"
```

are mandatory in the definition of the problem type (page A.30).

**ADAPTIVITY (Adaptive Mesh Refinement)****Purpose:**

This optional sub-directive allows to set the dimensions for the automatic mesh refinement during a computation, as required e.g. in adaptivity. The directive syntax is similar to that described in the previous pages for the “base” mesh. The user must define the maximum number of nodes, of degrees of freedom, and of elements (for each element type which can be refined) that are allowed to be “created” during the transient calculation. This is referred to as the “extension” zone as opposed to the “base” zone containing the base (normal) mesh. The optional directive must be terminated by the keyword **ENDA**.

**Syntax:**

```
< ADAP
      NPOI np <NIND ni> <NVFI nvfi> <NTHR nthr> <NPIN npin>
      "typ1" n1 "typ2" n2 ...
  ENDA >
```

**np**

Maximum number of mesh points in the extension zone.

**ni**

Total number of error indicator variable types used in the adaptive calculation. For example, if one wants to use both displacement and velocity as indicators, then it must be **NIND 2**. By default (i.e. if this keyword is omitted) only one error indicator variable is allowed. This quantity is used only in adaptive calculations with the error indicator.

**nvfi**

Maximum number of cell-centred finite volume (VFCC) interfaces in the extension zone. This quantity is used only in adaptive calculations with VFCC fluids.

**nthr**

Total number of threshold indicator variable types used in the adaptive calculation. For example, if one wants to use both displacement and velocity as indicators, then it must be **NTHR 2**. By default (i.e. if this keyword is omitted) only one threshold indicator variable is allowed. This quantity is used only in adaptive calculations with the threshold indicator.

**npin**

Maximum number of parent (0-level) pinballs in the extension zone. Such pinballs are automatically created during the mesh refinement process if the element being split has an attached (parent) pinball (and this recursively).

**n1, n2 ...**

Total number of elements of type “typ1”, “typ2” etc. in the extension zone. For the names of the element types see pages INT.80, INT.90 and INT.100.

**Comments:**

The number of degrees of freedom in the extension memory zone (i.e. the dofs relative to the adaptive nodes) is automatically computed by the code as the number of nodes in the extension zone (`np`) multiplied by the space dimension (2D or 3D). This implies of course that only elements whose nodes do not have any rotational dofs can be used in adaptivity, for the moment.



**DECOHESION (Automatic Separation of the Elements)****Purpose:**

This optional sub-directive allows to set the maximum number of created nodes during a computation as required for this numerical method (Automatic Separation of the Elements). Automatic separation of the elements is only available for CUB8 elements affected by the BOIS (wood) material. More explanations can be found in [\[899\]](#). The optional directive must be terminated by the keyword ENDD.

**Syntax:**

```
< DECO
      NPOI np
      ENDD >
```

**np**

Maximum number of created nodes during a computation.

**Comments:**

For the moment this method (Automatic Separation of the Elements) is only available for CUB8 elements affected by the BOIS (wood) material.

**GRID MOTION (A.L.E.)****Syntax:**

```
< "NALE" nal > < "NBLE" nbl >
```

**nale**

Maximum number of ALE nodes subjected to manual (i.e., non-automatic) rezoning. To be used only in ALE computations. The nodes are specified by the **GRIL** directive. By default (i.e., if not specified) the code assumes **nale** = 1 for ALE calculations.

**nble**

Maximum number of ALE nodes subjected to automatic rezoning. To be used only in ALE computations. The nodes are specified by the **GRIL** directive. By default (i.e., if not specified) the code assumes **nble** = 1, for ALE calculations.

**SPACE INTEGRATION FOR SHELL AND BEAM ELEMENTS****Syntax:**

```
< "NGPZ" mxngpz >
```

**mxngpz**

Maximum number of Gauss Points through the thickness for shell, plate or beam elements which are integrated through the thickness. This value overrides the default value set in INICO1 for each of these element types. This value is global and affects *all* the concerned element types. To set the “true” number of integration points through the thickness for each element (possibly a different value for each element), see the **COMP** directive (Geometrical Complements) on page C.42.

**MEMORY FOR ED1D CALCULATIONS****Syntax:**

```
< "ME1D" mead >
```

**me1d**

Length of memory table (in REAL\*4) for the ED1D calculations, in case of a coupled 1-D/multi-D calculation. The 1-D part is computed by the EURDYN-1D code, now embedded in EUROPLEXUS (see Page I.23). The default value of **me1d** is 50,000.

#### 4.6.2 DIMENSIONS RELATIVE TO GROUP C (MATERIALS)

**Object :**

Dimensions relative to the materials used.

**Syntax :**

```
< "LMAS" lmas >  
< "ECRO" lecr >  
< "PYRO" mpxyro >
```

**lmas**

Size of the consistent mass matrix.

**lecr**

Maximum length of the vector of parameters associated to materials (ECR). This parameter **is not necessary**, except some special cases. (Cf. comment below).

**mpxyro**

Maximum number of distinct oil pyrolysis bubbles (material FLUT ... PYRO, see page C.530). This material is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC.

**Comments :**

Normally EUROPLEXUS detects automatically the exact length of the ECR vector associated to materials, from the input file. Therefore, the directive **ECRO** is usually not necessary. This directive is needed only in special cases, whereby EUROPLEXUS has to create additional elements (not specified in the input nor in the associated mesh file) after the reading of the geometry: for example a pipeline circuit with a bifurcation, a rigid body, or in case of remeshing.

It is compulsory to enter the size of the consistent mass matrix, when the computation includes the material "MHOM".

### 4.6.3 DIMENSIONS RELATIVE TO GROUP D (CONNECTIONS)

#### Object:

Dimensions relative to the couplings.

#### Syntax:

```
< "MXLI" maxlie >          < "LNOD" maxnod>      < "LCOF" maxcof >
< "GLIS" nslid nemax >    < "JONC" njonc >
< "NPEF" nmpef      "NPTS" nomax >          < "SOLI" nsol >
< "MECA" nmeca >
< "FSSA" mxfssa >          < "FSSL" mxfssl >      < "FSSF" mxfssf >
< "NBJE" nbjeux >
< "VCON" mxvcon >
```

#### maxlie

Total number of connections in the 'LIAISONS'. This parameter **is not necessary**, except some special cases. (see comment below).

#### maxnod

Total number of nodes involved in the 'LIAISONS'. This parameter **is not necessary**, except some special cases. (see comment below).

#### maxcof

Total number of coefficients used in the 'LIAISONS'. This parameter **is not necessary**, except some special cases. (see comment below).

#### nslid

Number of couples of sliding lines.

#### nemax

Total number of nodes defining these lines (master AND slave).

#### njonc

Total number of nodes involved in a "TUBM" or "TUYM" type of junction.

#### nmpef

Number of "particle-structure" couples.

#### nomax

Total number of nodes defining these "particle-structure" couples.

#### nsol

Number of rigid solids.

`nmeca`

Total number of mechanisms.

`mxfssa`

Maximum number of nodes or element side couples subjected to fluid-structure sliding of the ALE type according to JRC's model (see directive "FSS" "ALE").

`mxfssl`

Maximum number of nodes or element side couples subjected to fluid-structure sliding of the Lagrangian type according to JRC's model (see directive "FSS" "LAGR").

.

`mxfssf`

Maximum number of nodes or element side couples subjected to fluid-structure sliding of the fixed type according to JRC's model (see directive "FSS" "FIXE").

`nbjeux`

Number of couples of nodes to which an impact with gap is associated.

`mxvcon`

Maximum total number of parameters used to define bilateral constraints (CONT SPHE, CYLI, CONE, TORE) with variable coefficients (OPTI CONT VARI). Each sphere requires 3 parameters, each cylinder or cone requires 6 parameters, and each torus requires 9 parameters).

### Comment:

Normally EUROPLEXUS detects automatically the exact number of LIAISONS parameters required, from the input file. Therefore, the directives `MXLI`, `LNOD`, `LCOF`, ... **are usually not necessary**. These directives are needed only in special cases, whereby EUROPLEXUS has to create additional nodes or additional elements.

#### 4.6.4 DIMENSIONS RELATIVE TO GROUP G (PRINTOUTS)

**Object:**

Dimensions relative to printout and storage keywords.

**Syntax:**

```
< "MTTI" mtime >    < "MNTI" mntime >  
< "NFRO" nfront >   < "NPFR" npfron >  
< "NEPE" nepedi >
```

**mtime**

Maximum number of times for which the printing/storage of the results is requested.

**mntime**

Maximum number of time steps for which the printing/storage of the results is requested.

**nfront**

Number of borders ('frontiere') for which the calculation of resultants is requested.

**npfron**

Total number of nodes involved (by putting all borders together).

**nepedi**

Length of the memory reserved for the vector NEPEDI which is constructed in subroutine edit1. The code normally computes this automatically.



#### 4.6.5 DIMENSIONS RELATIVE TO GROUP I (CALCULATION)

**Object:**

Dimensions relative to the calculation run.

**Syntax:**

```
< "TTHI" mtthis >
```

**mtthis**

Maximum number of time values for which the solution has to be computed, in case of "PAS UTIL" option and "CALC" ... "HIST" (the time marching is imposed by the user).

#### 4.6.6 DIMENSIONS RELATIVE TO ADVECTION-DIFFUSION

**Object:**

Dimensions relative to advection-diffusion problems as declared by keyword "ADDF" above in this section.

**Syntax:**

```
< "ELSN" mxelsn    "BWDT" mxbwdt    "TPOI" mxtpoi  
  "ELGR" mxelgr    "CVEL" mxcvel    "GRPS" mxgrps >
```

**mxelsn**

Maximum number of elements surrounding (i.e., connected to) any given node.

**mxbwdt**

Maximum bandwidth of pressure matrix (for direct solution) or maximum number of elements surrounding an element (for iterative solution).

**mxtpoi**

Maximum number of time points for prescribed time-dependent 'charges' in advection-diffusion problems (temperatures, heat flux, heat generation, heat convection, heat radiation, external pressure, velocities).

**mxelgr**

Maximum number of elements in each group with prescribed time-dependent 'charges'.

**mxcvel**

Maximum number of nodes with constrained velocities.

**mxgrps**

Maximum number of groups with prescribed time-dependent 'charges'.

**4.6.7 END OF DIMENSIONING****Syntax:**

"TERM"

**Comments:**

The word "TERM" marks the end of the dimension, it must appear.

## 5 GROUP B—MESH AND GRID MOTION

### Object:

The following directives enable to define the mesh.

### Syntax:

- Mesh generated by COCO or in free format:

```
"GEOM" <optional mesh manipulation commands> ... "TERM"  
      ... COCO data or free format data ...
```

- Mesh generated by GIBI:

```
"GEOM" <optional mesh manipulation commands>  
      ("nomelm" ('nomobjet') ) "TERM"
```

- Mesh generated by I-DEAS:

```
"GEOM" <optional mesh manipulation commands>  
      (... zone declaration list ...) "TERM"
```

- Mesh generated by LS-DYNA (k-file) (see [GBB\\_0055](#)):

```
"GEOM" <optional mesh manipulation commands>  
      ("nomelm" ('nomobjet') ) "TERM"
```

### Comments:

These directives are described in detail on the following pages.

The **GEOM** directive accepts some simple optional mesh manipulation commands that can be used to scale, shift, etc. the mesh read from an external mesh generator before starting the transient calculation. These commands affect only the nodal coordinates, but not the mesh connectivity. They are described below on page [GBB\\_0015](#).

## 5.1 OPTIONAL MESH MANIPULATION COMMANDS

### Object:

To manipulate the mesh coordinates read from an external mesh generator before starting the transient calculation. For example, the mesh can be scaled, translated, centred, etc. Note that these commands only affect the (initial) nodal coordinates, they do not affect the elements (i.e. the connectivity).

### Syntax:

```
< SCAL $[ FACT fc ;  
          FACX fx ; FACY fy ; <FACZ fz> ]$ >  
< SHIF $[ CENT ;  
          SHIX sx SHIY sy <SHIZ sz> ]$ >
```

#### SCAL

Scale the coordinates of the mesh to be subsequently read in input either isotropically (i.e. by the same factor **fc** along all axes), or anisotropically.

**fc**

Isotropic scaling factor.

**fx**

Scaling factor along the  $x$  direction (by default 1.0).

**fy**

Scaling factor along the  $y$  direction (by default 1.0).

**fz**

Scaling factor along the  $z$  direction (by default 1.0).

#### SHIF

Shift the coordinates of the mesh to be subsequently read in input either in such a way that it is centered around the origin, or by specified amounts in each spatial direction.

#### CENT

Shift the coordinates of the mesh to be read in input in such a way that it is centered around the origin.

**sx**

Shift along the  $x$  direction (by default 0.0).

**sy**

Shift along the  $y$  direction (by default 0.0).

**sz**

Shift along the  $z$  direction (by default 0.0).

**Comments:**

The above commands, in particular the scaling commands, can be useful e.g. in case the geometry has been produced by an external mesh generator in some non-standard units. For example, assume the mesh has been generated in millimetres rather than metres. To convert to metres use the command `GEOM SCAL FACT 0.001 ...`

Note that the mesh manipulation occurs immediately after reading the nodal coordinates, so that the coordinates printed on the listing are the corrected ones, not the ones read from the input file.

Note that in case of simultaneous mesh scaling and shifting, the scaling occurs first, then the shifting is applied. Therefore, the shift amounts should be given in the corrected (scaled) mesh units, not in the original mesh units.

## 5.2 MESH IN COCO-LIKE OR IN FREE FORMAT

### 5.2.1 GEOMETRY

#### Object:

To read the mesh (i.e. the nodal coordinates and the elements connectivity) either in “free” format or in a fixed (COCO-like) format.

In the first case, the mesh data are read directly from the input file.

In the second case, the (fixed) format used for the mesh data must be specified and then the mesh data can either be read directly from the main input file, or from an external file whose unit number (`nl` in the following) is specified by the user. This second possibility can be handy e.g. to un-clutter the main input file in case of large mesh data, or to read a (formatted) mesh data set produced by an exotic mesh generator for which no direct EPX interface exists.

#### Syntax:

```
"GEOM"  $[          < "LIBR" >   < "POLA" >          ;
          < nl > < '(format1)'   '(format2)' > ]$
```

```
"POIN"  npoin
```

#### LIBR

The file describing the geometry will be read in free format.

#### POLA

Nodes are specified by their polar coordinates (by default Cartesian coordinates). In 2D, first enter the radius ( $R$ ), then the angle ( $\theta$ ) in degrees for each node. In 3D, the third coordinate is interpreted as the elevation  $Z$ , thus the coordinate system is cylindrical. The corresponding Cartesian coordinates are computed according to:  $x = R \cos(\theta)$ ,  $y = R \sin(\theta)$  and  $z = Z$  (3D only).

#### nl

Logical number of input unit (file) from which the geometry will be read. By default, `nl=5` (15 at JRC). If it is omitted, then the core will read the mesh data directly from the (main) input file.

#### format1

Reading format of nodal coordinates. By default `format1=6E12.5`.

#### format2

Reading format of the numbers of the nodes composing the elements (i.e. the elements connectivity). By default `format2=18I4`.

**npoin**

Exact number of mesh nodes.

**Comments:**

If nothing is specified after the word "GEOM", the file describing the geometry is assumed to be in COCO format, it is read from logical unit 5 (15 at JRC) with the formats: format1 and format2.

If the formats are modified, they must be enclosed in parentheses AND in apostrophes.

Example:

```
"GEOM"    '(5E20.12)' '(16I5)'  "POIN" 123
```

The option "LIBR" is particularly useful for a simple mesh when the user himself prepares the coordinates and the topology.

In the case of polar coordinates, EUROPLEXUS transforms them into Cartesian coordinates for the following computations. If outputs in polar coordinates are desired, see keyword "OPTION".

The number of nodes npoin must not be greater than the number declared for the dimension (page A.40).

In order to read the mesh data from an external file, specify the **n1** unit number (an integer value) just after the **GEOM** keyword. This is treated as a file without a name. In Fortran, the actual (default) name of the file may vary depending upon the platform. For example, under Windows by choosing unit number 9, the code will try to open a file called **fort.9** for reading and will attempt reading the mesh data from this file. It is the responsibility of the User (or of the code-launching procedure) to make such a file available on the current directory. Under Windows, for example, unit 9 is the unit normally devoted to the **.msh** file, in case of mesh produced by Cast3m. Therefore the launching procedure automatically creates a **fort.9** file by copying the **.msh** file (if this exists). So the simplest way to read the mesh from an external file under Windows is to use the value 9 for **n1** (**GEOM 9 ...**) and to put the formatted mesh data in a file called **<base>.msh** where **<base>** is the base name of the main input file. See also the practical examples below.



### 5.2.2 ELEMENT ZONES

**Object:**

Each of the following keywords defines a zone of elements of the given type, that are sequentially numbered.

**Syntax:**

```
| "typ1" n1 "typ2" n2 ... |  
  
"TERM"  
  
...   COCO data set with its title  
      (title is optional if free format, see "LIBR") ...
```

**typi**

Name of an element type (see page I.80).

**ni**

Number of elements in the zone

**TERM**

Marks the end of the directive GEOMETRY.

**Comments:**

The various elements are described on page INT.80.

The number of the elements announced for a zone must correspond exactly to the elements defined in the COCO data set.

The same type of element can occupy several zones.

The number of zones must be less than or equal to the one given during the dimensioning (p. A.40).

In the COCO data set, the topology of the elements must be read by zones, and these zones are arranged in the order of their definition in the directive "GEOM".

The word "TERM" is compulsory to indicate the end of the keyword GEOMETRY.

The title appearing before the coordinates of the points is not compulsory when reading in FREE format (see "LIBR").

In order to become acquainted with the keyword "GEOM" the user may have a look at the examples on pages EX 10 and on the following ones.

### Warning

There is a mandatory logical order for the 1-D elements (except ED1D). These elements are to be subdivided in 3 groups, which are respectively:

- 1st group: TUBE and TUYA,
- 2nd group: CL1D and CLTU,
- 3rd group: CAVI and BIFU.

Further information allowing to completely define the properties of "CAVI" and "BIFU" elements are given by the "RACCORD" sub-directive of the "COMPLEMENT" directive (**GBC\_0080**).

The elements of type "BIFU" cause the automatic generation of connections between the concerned d.o.f.s. It is therefore mandatory to list them again in the "LIAISON" directive: see this directive.

The junction elements "CAVI" and "BIFU" must possess the same materials as the neighbour elements of which they ensure the continuity.

### 5.2.3 EXAMPLE: MESH FROM FORMATTED EXTERNAL FILE

Hereafter a simple example is given of how to read the mesh data with a fixed (COCO-like) format. In a first case, we will read the data directly from the (main) input file. Then we will show how to read the data from an auxiliary file, so that the (main) input file is more compact and legible.

Here is the complete input file with formatted mesh read directly from the main input file (test00.epx):

TEST00 - MESH READ FROM THE MAIN INPUT FILE WITH FIXED FORMAT

LAGR AXIS

GEOM '(2E22.15)' '(7I10)' POIN 18

Q92 1 Q93 1 ED01 2 TERM

The following is the mesh data in fixed format

1.000000000000000E+00	0.000000000000000E+00					
1.500000000000000E+00	0.000000000000000E+00					
2.000000000000000E+00	0.000000000000000E+00					
2.500000000000000E+00	0.000000000000000E+00					
3.000000000000000E+00	0.000000000000000E+00					
1.000000000000000E+00	0.500000000000000E+00					
1.500000000000000E+00	0.500000000000000E+00					
2.000000000000000E+00	0.500000000000000E+00					
2.500000000000000E+00	0.500000000000000E+00					
3.000000000000000E+00	0.500000000000000E+00					
1.000000000000000E+00	1.000000000000000E+00					
1.500000000000000E+00	1.000000000000000E+00					
2.000000000000000E+00	1.000000000000000E+00					
2.500000000000000E+00	1.000000000000000E+00					
3.000000000000000E+00	1.000000000000000E+00					
0.000000000000000E+00	0.000000000000000E+00					
1.000000000000000E+00	1.000000000000000E+00					
1.000000000000000E+00	2.000000000000000E+00					
1	2	3	8	13	12	11
6	7					
3	4	5	10	15	14	13
8	9					
16	17	17	18			

\* Mesh data is finished, we continue reading the (main) input data

COMP EPAI 1. LECT 1 PAS 1 4 TERM

MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8

TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.

LECT 1 2 TERM

VM23 RO 4000. YOUN 2.D11 NU 0.2 ELAS 4.D8

TRAC 2 4.D8 2.D-3 6.D8 1.

LECT 3 4 TERM

LINK COUP

BLOQ 12 LECT 5 PAS 5 15 TERM

BLOQ 123 LECT 16 TERM

INIT VITE 2 300 LECT 6 PAS 1 9 TERM

VITE 1 -200 LECT 6 PAS 1 8 TERM

```

VITE 2 -100 LECT 17 TERM
VITE 1 200 LECT 18 TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO FREQ 100
FICH ALIC FREQ 1
OPTI PAS UTIL NOTE LOG 1
CALCUL TINI 0. TEND 0.001D0 PASF 1.D-5
FIN

```

The data are read directly from the main input file because no unit number (`n1`) is specified after the `GEOM` keyword. Instead of using the default reading formats we choose a format `2E22.15` for the nodal coordinates and a format `7I10` for the connectivity.

Note that a comment line (reading “The following is the mesh data in fixed format”) must be put before the actual mesh data.

The mesh data start with the nodal coordinates. Two values (in 2D cases) or three values (in 3D cases) must be specified for each node, and this for the exact number of nodes (`POIN`) chosen by the user. Coordinates are read as a single block of data using the user-specified format.

Then, the element connectivity has to be specified, i.e. the nodes of each element. These data are read element zone by element zone, since each element type may have a different number of nodes. The element zones correspond to the declaration of the element types given by the user (`Q92 1 Q93 1 ED01 2`, i.e. three zones in the example).

In this example, the first zone has a single `Q92` element, which has 9 nodes. This takes two lines of input to specify, since we have chosen to put only seven values per line. The second zone has one `Q93` element and is similar to the first one. The third and last zone contains two `ED01` elements, each with two nodes. Only one line (4 values) of input is needed for this.

Now we show how to modify the previous example in order to read the mesh data from an external input file.

The main input file (`test01.epx`) reads:

```

TEST01 - MESH READ FROM AN AUXILIARY INPUT FILE WITH FIXED FORMAT
LAGR AXIS
GEOM 9 '(2E22.15)' '(7I10)' POIN 18
      Q92 1 Q93 1 ED01 2 TERM
* Mesh data is finished, we continue reading the (main) input data
COMP EPAI 1. LECT 1 PAS 1 4 TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
      TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
      LECT 1 2 TERM
      VM23 RO 4000. YOUN 2.D11 NU 0.2 ELAS 4.D8
      TRAC 2 4.D8 2.D-3 6.D8 1.
      LECT 3 4 TERM
LINK COUP
      BLOQ 12 LECT 5 PAS 5 15 TERM
      BLOQ 123 LECT 16 TERM
INIT VITE 2 300 LECT 6 PAS 1 9 TERM
      VITE 1 -200 LECT 6 PAS 1 8 TERM

```

```

VITE 2 -100 LECT 17 TERM
VITE 1 200 LECT 18 TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO FREQ 100
FICH ALIC FREQ 1
OPTI PAS UTIL NOTE LOG 1
CALCUL TINI 0. TEND 0.001D0 PASF 1.D-5
FIN

```

We have chosen 9 as the unit number for the external file from which the mesh data will be read (GEOM 9 ...). Under Windows, this unit is automatically connected to the file `.msh` corresponding to the main input file, i.e. the launching procedure copies the `.msh` file onto a local file called `fort.9`, from which the mesh data will be read. If a different unit number is chosen, say 34, it is the responsibility of the user to provide a file `fort.34` containing the mesh data in the current directory.

The mesh file (test01.msh) reads:

The following is the mesh data in fixed format

```

1.000000000000000E+00 0.000000000000000E+00
1.500000000000000E+00 0.000000000000000E+00
2.000000000000000E+00 0.000000000000000E+00
2.500000000000000E+00 0.000000000000000E+00
3.000000000000000E+00 0.000000000000000E+00
1.000000000000000E+00 0.500000000000000E+00
1.500000000000000E+00 0.500000000000000E+00
2.000000000000000E+00 0.500000000000000E+00
2.500000000000000E+00 0.500000000000000E+00
3.000000000000000E+00 0.500000000000000E+00
1.000000000000000E+00 1.000000000000000E+00
1.500000000000000E+00 1.000000000000000E+00
2.000000000000000E+00 1.000000000000000E+00
2.500000000000000E+00 1.000000000000000E+00
3.000000000000000E+00 1.000000000000000E+00
0.000000000000000E+00 0.000000000000000E+00
1.000000000000000E+00 1.000000000000000E+00
1.000000000000000E+00 2.000000000000000E+00
      1      2      3      8      13      12      11
      6      7
      3      4      5      10     15     14     13
      8      9
     16     17     17     18

```

### 5.3 MESH IN CASTEM FORMAT

**Object:**

To define the objects associated to each element type.

**Syntax:**

```
"GEOM" ( "nomelm" ( 'nomobjet' ) ) "TERM"
```

**nomelm**

Name defining the type of element to be taken from the list of available elements.

**'nomobjet'**

Name of the GIBI object(s)

**TERM**

End of the directive "GEOM"

**Comments:**

The names of the available elements may be found on page [GBINT\\_0080](#), the same keywords are used as in the case of COCO or free-format data (see the chapter ELEMENT ZONES).

The names of the elements cannot be used for other purposes. This explains why the names of the GIBI objects cannot begin with the same first four letters as the name of an element.

Several objects may be associated to a certain type of element. In order to obtain, on the GIBI drawings, the same node and element numbers as on EUROPLEXUS, write in GIBI:

```
"TRAC" ('objet1' "ET" 'objet2' "ET" 'objet3') ...;
```

in the same order as for the instruction "GEOM" of the EUROPLEXUS program.

There is a zone each time a name of an element is specified. Do not forget to sufficiently dimension the number of zones (page A.40).

**Warning**

The elements with variable number of nodes such as "CAVI" and "BIFU" may be simply generated by GIBI by means of so-called super-elements:

```
my_junction = 'MANU' 'SUPE' pt1 pt2 ... ptn ;
```

It is also possible to use topologically equivalent elements:

```
my_junct_1 = 'MANU' 'POI1' pt1 ;  
my_junct_2 = 'MANU' 'SEG2' pt1 pt2 ;  
my_junct_3 = 'MANU' 'TRI3' pt1 pt2 pt3 ;  
my_junct_4 = 'MANU' 'QUA4' pt1 pt2 pt3 pt4 ;  
my_junct_5 = 'MANU' 'PYR5' pt1 pt2 pt3 pt4 pt5 ;  
etc...
```

However, in order to generate a K2000 file correct for post-treatment, it is necessary that such objects be formed by one single element.

The elements of type "BIFU" cause the automatic generation of connections between the concerned d.o.f.s. It is therefore mandatory to list them again in the "LIAISON" directive: see this directive.

### 5.3.1 Superposed elements in a Cast3m mesh (color problem)

A problem may arise with meshes generated by Cast3m containing **superposed elements**, for example a structure made of 3D shells to which a layer of CLxx elements is attached in order to apply an external pressure. In this case, each structural element has the same nodes as the corresponding CLxx element.

EPX treats superposed elements correctly (i.e., in the way that is probably expected by the user) only if the two elements have **different colors**. If the color is the same, the two superposed elements are accepted (without error messages) and are made part of the EPX mesh. However, they are eliminated from the Cast3m object names, and therefore these objects contain the wrong elements (the structural objects end up containing the CLxx elements). The problem seems to occur only when the CLxx object is a complex one (i.e. if it has sub-objects), that is when there is more than one pressure sub-object.

To avoid the problem the following simple rule may be used: to generate CLxx elements (say object **pres**) in Cast3m, starting from a set of existing (shell-like) structural elements (say object **stru**), use the following syntax:

```
pres = stru COUL ROUG;
```

where it is assumed that the **stru** object either has no color or has a color different from **ROUG**. This ensures that the **pres** and **stru** objects have different colors and will be correctly read in by EPX.

The frequently used alternative syntax:

```
pres = stru PLUS (0 0 0);  
ELIM tol (pres ET stru);
```

is strongly discouraged (although as said it seems to work when **stru** is a single object, without sub-objects).



## 5.4 MESH IN I-DEAS FORMAT

### Object:

To read the coordinates of the nodes and the topology of the elements from an I-DEAS universal file. The elements are declared through a list of keywords defining zones of elements of the given type, that are sequentially numbered.

Due to the fact that the I-DEAS format is quite different depending on its version special attention must be given in order to check if the mesh is read in a right way.

### Syntax:

```
"GEOM" | "typ1" n1 "typ2" n2 ... | "TERM"
```

(same as "Mesh by means of GIBI or CASTEM")

### Comments:

Even when reading from an I-DEAS mesh, the topology of the elements has to be read by zones, as defined in the directive **GEOM**. The user has to be sure that the list of zones declared is consistent with the data contained in the I-DEAS universal file used. In order to make this easy, a first run using the **REWR** option (see Page A.30) can be carried out; the informations to be set into the **GEOM** list can be obtained from the table printed on the listing file.

The word **TERM** is compulsory to indicate the end of the keyword **GEOM**.

## 5.5 MESH IN LS-DYNA FORMAT (K-FILE)

### Object:

To read the coordinates of the nodes and the topology of the elements from an LS-DYNA k-file. The elements can be defined in the k-file by using the PART command. No materials or element type must be assigned to the PART in the k-file. The elements are simply defined by a list and can be grouped by node lists. The k-file can be created by using the free LS-PREPOST software.

### Syntax:

```
"GEOM" | "typ1" PART n1 "typ2" PART n2 "typ3" ESET n3 ... | "TERM"
```

n1, n2

Elements taken from PART n1, n2 (also the part name can be used with a maximum length of 4 characters)

n3

Elements taken from element set n3 (also the set name can be used with a maximum length of 4 characters)

In the same way, the geometry objects (PART, ESET, NSET, NODE, ELEM) can be addressed later on e.g. for the material or the boundary conditions (see also [GBINT\\_0050](#)).

### Comments:

The following k-file keywords are interpreted:

- PART
- NODE
- ELEMENT\_SOLID
- ELEMENT\_SHELL
- ELEMENT\_BEAM
- ELEMENT\_MASS
- SET\_NODE = SET\_NODE\_LIST = SET\_NODE\_LIST\_TITLE
- SET\_SOLID = SET\_SOLID\_TITLE = SET\_SOLID\_LIST
- SET\_SHELL = SET\_SHELL\_TITLE = SET\_SHELL\_LIST
- SET\_BEAM = SET\_BEAM\_TITLE = SET\_BEAM\_LIST

The word **TERM** is compulsory to indicate the end of the keyword **GEOM**.

The names of the available elements may be found on [GBINT\\_0080](#).

## 5.6 GRID MOTION IN AN A.L.E. COMPUTATION

### Object:

These keywords enable the user to impose the motion of the mesh under the Arbitrary Lagrangian Eulerian (ALE) formulation. Therefore, this directive can only be used in an ALE computation (see keyword "ALE" on page A.30).

### Attention!

If you use any "RACCORDS" 1D ("CAVI" and "BIFU"), the "GRIL" directive must be placed **after** the directive "COMPLEMENT" (see [GBC\\_0010](#)).

### Syntax:

```
"GRILLE" < "LAGRANGE" /LECTURE/ >
          < "EULE" /LECTURE/ >
          < "FS" /LECTURE/ >
          < "BFIKE" /LECTURE/ >
          < "GRFS" /LECTURE/ >

          < "SUIVRE" ... >
          < "LIGNE" ... >
          < "CONTOUR" ... >
          < "PLAN" ... >
          < "TETR" ... >
          < "HEXA" ... >
          < "PRIS" ... >
          < "PYRA" ... >
          < "SLIP" ... >
          < "AUTO" ... >
          < "MEAN" ... >
          < "DIRE" ... >
          < "QUAD" ... >
          < "SPEC" ... >
          < "MECA" ... >
          < "GLOB" ... >
```

### LAGRANGE

The following nodes are Lagrangian: the mesh is fixed to material particles.

### EULE

List of nodes explicitly declared Eulerian, they are fixed in space but they correspond to different material particles at different times, in general.

**FS**

The user has to mention all the elements of the fluid-structure type in contact with ALE continuum (fluid) elements. In this case, the keyword "SUIVRE", to ensure the continuity of the mesh, is redundant: the keyword "FS" will do it automatically.

**"BFIKE"**

The following nodes will be considered as fixed (purely Eulerian). This directive allows thus to specify all the fixed nodes that will serve as base points for manual rezoning options to be entered successively.

**"GRFS"**

The following elements must be of the CLxx type and their nodes must be geometrically coincident with structural nodes belonging to shell elements. During the calculation, the fluid nodes will be piloted by the corresponding structural nodes like if the "SUIVRE" directive would have been specified. These elements must always be associated to the "IMPE" "GRFS" material.

**Comments:**

If the motion of a node is not specified, then it is supposed to be Eulerian (fixed mesh).

Several options may be set for the fine-tuning of the automatic rezoning algorithms. For more information, please see the OPTI REZO directive in [GBH\\_0150](#).

For the directive GLOB see section [GBB\\_0136](#).

**Warning:**

Do not repeat the fluid-structure couplings in the instruction "LIAISON" (except in very specific cases: perforated plates described on [GBC\\_0330](#)).

Do not forget to dimension sufficiently: "NALE" described on [GBC\\_0040](#) (number of A.L.E. or Eulerian nodes).

The order in which the different directives appear (LAGRANGE, BFIKE, FS, SUIVRE, CONTOUR...) is important: EUROPLEXUS follows the same order during the remeshing operations.

The following rule holds:

- 1) A node that has to be used as base (master) point for the motion of other points must have a defined motion, else it will be considered as fixed.
- 2) When a point is already used as base point, its motion may no longer be re-defined.

The order of instructions use more often is:

- 1) First, the LAGRANGIAN nodes are defined;
- 2) Then, one passes to a first manual rezoning directive among (SUIVRE, LIGNE, ...) by respecting the following rule: a node may be used as base point only if its motion is already defined previously.

The points defined by this directive may then be used as base points for the following directives.

**Restrictions for 1-D problems:**

In the presence of bifurcations or cavities (elements "BIFU" and "CAVI"), their junctions must be defined BEFORE specifying grid motions. For further details see the directive "COMPLEMENT", sub-directive "RACCORD".

It is useless to define a grid motion for elements "TUYA". In fact, they result from the assembly of an element of type "POUT" and one of type "TUBE", and the "grid" for the internal fluid is nothing but the set of nodes that define the tube walls. EUROPLEXUS automatically ensures their motion.

### 5.6.1 AUXILIARY FILE

**Object :**

This directive allows to read the grid remeshing data from an auxiliary file.

**Syntax :**

```
"GRILLE"      < "FICHIER"      'nom.fic'  >
```

In certain cases these data may be bulky. Then it is advisable to store them on an auxiliary file in order to shorten the main data file. The auxiliary file is activated by means of the directive "FICHIER", followed by the name (complete under Unix) of the file. Then, in the main data file remains only the keyword "GRILLE", followed by "FICHIER".

The auxiliary file (in free format) will contain all grid rezoning data, except the "GRILLE" keyword. In order to return to reading from the main input file, the auxiliary file must terminate by the keyword "RETOUR".

### 5.6.2 “SUIVRE”

**Object:**

To force one or more A.L.E. mesh nodes to follow the motion of a ”base” node.

**Syntax:**

```
"SUIVRE"    "BASE"    /LECTURE/  
            "LIST"    /LECTURE/
```

BASE /LECTURE/

Number of the ”base” node to be followed.

LIST /LECTURE/

Numbers of the A.L.E. nodes with an imposed motion.

**Comments:**

The particle which is present at the node is changing all the time. This instruction is therefore very different from imposing a node to be Lagrangian.

**Warning:**

Please read the rule for defining the base points, page B.60.

### 5.6.3 “LIGNE”

**Object:**

To impose the motion of several nodes so that they remain aligned between two “base” nodes . The initial subdivision is maintained: the segments remain in the same relation.

**Syntax:**

```
"LIGNE"      "BASE"    /LECTURE/  
              "LIST"    /LECTURE/
```

"BASE" /LECTURE/

Numbers of the 2 base nodes which will impose the motion.

"LIST" /LECTURE/

Numbers of A.L.E. nodes lying between the two proceeding points and following the motion. The list can safely include also the two base nodes: if present, they are automatically discarded from the list.

**Comments:**

It is possible to have roughly aligned points, but if the basic points are very distant from each other, the computation tends to realign the points (and vice versa).

**Warning:**

Please read the rule for defining the base points, page B.60.



#### 5.6.4 “PLAN”

**Object:**

To impose a homeomorphic motion to several nodes of the grid describing a triangle or a quadrangle. This command is available both in 2D and in 3D. In the 3D case, all slave points should lie at least approximately on the plane defined by the triangle or quadrangle.

**Syntax:**

```
"PLAN"      "BASE"  /LECTURE/  
            "LIST"  /LECTURE/
```

```
"BASE" /LECTURE/
```

Numbers of the base points composing a triangle (3 points) or a quadrangle (4 points).

```
"LIST" /LECTURE/
```

Numbers of A.L.E. points submitted to homeomorphic motion. These nodes must be located inside the basic triangle or quadrangle, or on their boundaries. The list can safely include also the base nodes: if present, they are automatically discarded from the list.

**Comments:**

It is strongly recommended to use quadrangles. Triangles are only useful if the initial mesh already has a triangular shape.

**Warning:**

Please read the rule for defining the base points, page B.60.

### 5.6.5 “TETR”

**Object:**

To impose a homeomorphic motion to several nodes of the grid describing a tetrahedron. This command is available only in 3D.

**Syntax:**

```
"TETR"      "BASE"    /LECTURE/  
            "LIST"    /LECTURE/
```

"BASE" /LECTURE/

Numbers of the 4 (usually Lagrangian) base points defining the tetrahedron. These points should not be coplanar.

"LIST" /LECTURE/

Numbers of A.L.E. points submitted to homeomorphic motion. These nodes must be located inside the basic tetrahedron or along its boundaries. The list can safely include also the base nodes: if present, they are automatically discarded from the list.

**Warning:**

Please read the rule for defining the base points, page B.60.

### 5.6.6 “HEXA”

**Object:**

To impose a homeomorphic motion to several nodes of the grid describing a hexahedron. This command is available only in 3D.

**Syntax:**

```
"HEXA"      "BASE"    /LECTURE/  
            "LIST"    /LECTURE/
```

```
"BASE" /LECTURE/
```

Numbers of the 8 (usually Lagrangian) base points defining the hexahedron.

```
"LIST" /LECTURE/
```

Numbers of A.L.E. points submitted to homeomorphic motion. These nodes must be located inside the basic hexahedron or along its boundaries. The list can safely include also the base nodes: if present, they are automatically discarded from the list.

**Warning:**

Please read the rule for defining the base points, page B.60.

### 5.6.7 “PRIS”

**Object:**

To impose a homeomorphic motion to several nodes of the grid describing a prism. This command is available only in 3D.

**Syntax:**

```
"PRIS"      "BASE"    /LECTURE/  
            "LIST"    /LECTURE/
```

```
"BASE" /LECTURE/
```

Numbers of the 6 (usually Lagrangian) base points defining the tetrahedron.

```
"LIST" /LECTURE/
```

Numbers of A.L.E. points submitted to homeomorphic motion. These nodes must be located inside the basic prism or along its boundaries. The list can safely include also the base nodes: if present, they are automatically discarded from the list.

**Warning:**

Please read the rule for defining the base points, page B.60.

### 5.6.8 “PYRA”

**Object:**

To impose a homeomorphic motion to several nodes of the grid describing a pyramid. This command is available only in 3D.

**Syntax:**

```
"PYRA"      "BASE"    /LECTURE/  
            "LIST"    /LECTURE/
```

```
"BASE" /LECTURE/
```

Numbers of the 6 (usually Lagrangian) base points defining the pyramid.

```
"LIST" /LECTURE/
```

Numbers of A.L.E. points submitted to homeomorphic motion. These nodes must be located inside the basic pyramid or along its boundaries. The list can safely include also the base nodes: if present, they are automatically discarded from the list.

**Warning:**

Please read the rule for defining the base points, page B.60.

### 5.6.9 “CONTOUR”

#### Object:

To impose a homeomorphic motion to the grid nodes inside a given bounded area.

#### Syntax:

```
"CONT"  <"NORM" "NX" nx "NY" ny <"NZ" nz> <"ORDB">>
                                         "BASE" /LECTURE/
                                         "LIST" /LECTURE/
```

#### "NORM"

Introduces the optional definition of the normal direction. This is the direction along which the contour will be deformed. For example, in case of impact on a circular pipe, the contour is initially a circle but later on is squeezed to an ellipsis or even a concave shape. In this case the normal direction coincides with the direction of the impact.

#### "NX"

Component of the normal along  $x$ .

#### "NY"

Component of the normal along  $y$ .

#### "NZ"

Component of the normal along  $z$  (3D only).

#### "ORDB"

Let the code try to order the base nodes given (in random order) in the following **BASE** sub-directive.

#### "BASE" /LECTURE/

Numbers of the base nodes defining the boundary. If no normal is specified, these can be given in any order. However, if a normal is specified, the nodes are assumed to be listed in the order they occur along the contour (starting from any node on the contour). If an ordered list is not available, by specifying the optional **ORDB** keyword, the code itself tries to order the base nodes. Note, however, that the ordering can only succeed if the base nodes lie approximately on a plane and form a **convex** contour.

#### "LIST" /LECTURE/

Numbers of the ALE nodes submitted to the homeomorphic motion (in any order). In principle these nodes should be located inside the bounded area, they may not be on the contour. However, the list can safely include also the base nodes (on the contour): if present, they are automatically discarded from the list.

#### Comments:

It is recommended to use the facilities offered in GIBI by the keywords "CONTOUR" and "ENVELOPPE".

For ease of use, EUROPLEXUS accepts that among the points defined by "LIST" there be also the base points. These points will be then removed by a special treatment within the code.

Otherwise, one can also separate the internal points from those on the contour, as shown in the following example.

Given an object "LIQ", the user wants to distinguish its internal nodes (ALE) from the nodes defining the outline of the surface (Lagrangian).

In GIBI:

```
TLIQ = LIQ changer POI1 ;
CLIQ = contour LIQ      ;
CLIQ = CLIQ changer POI1 ;
ILIQ = TLIQ differ  CLIQ ;
```

In EUROPLEXUS:

```
CONTOUR BASE LECTURE CLIQ TERM
LIST LECTURE ILIQ TERM
```

This directive should be relatively robust for translation and rigid rotation of the contour. It should perform well also for moderate deformation of the contour itself, provided the contour is initially convex (ideally, similar to a circle).

If the contour is (or becomes, due to deformation) concave, then the algorithm does not perform well in general. In this case, if the direction of the (predominant) deformation of the contour is known *a priori*, it is advised to specify it via the optional NORM keyword.

Recall, however, that in this case the list of base nodes (on the contour) **should normally** be given in the order they occur along the contour (starting from an arbitrary node). If an ordered list of the nodes is not available, you can try giving them in random order and specifying the optional ORDB keyword.

### 5.6.10 "SLIP"

#### Object:

Define 2D curves consisting of nodes that are allowed to slip tangentially to the curve itself. This only applies to ALE models in structures or fluids.

#### Syntax:

```
"SLIP"      | [ "NORM" /LECTURE/  ;  
                "EQUI" /LECTURE/ ] |
```

#### "SLIP"

The following nodes belong to a curve (in 2D) that is Lagrangian in the normal direction but ALE in the tangential direction. Examples are free surfaces in fluids, free boundaries in solids treated by the ALE method for structures, or interfaces between different materials that should not be mixed. Sliding in the tangential direction can be of two types: a) no sliding (only normal motion) or b) slide tangentially so as to keep nodes nearly equidistant.

#### "NORM" /LECTURE/

The specified nodes will only move along the normal direction to the curve.

#### "EQUI" /LECTURE/

The specified nodes will move both along the normal and along the tangential direction, so as to remain nearly equidistant from each other.

#### Comments

The nodes have to be listed in /LECTURE/ in the order in which they appear along the curve, and by leaving the body on the left side (for free surfaces).

Each list must include an initial and a final node, not subject to the imposed "SLIP" motion, whose positions are used to evaluate the normal direction for each triple of nodes. Therefore, each /LECTURE/ must contain at least three numbers.

#### Restrictions

The algorithms implemented here are only valid for "nearly straight" curves, in the sense that the angle between successive segments of the curve (element sides) must be close to 180 degrees. If this is not the case, then the mesh should be refined locally.



**5.6.11 “AUTO”****Object:**

Use Giuliani’s (automatic) rezoning algorithm to determine the motion of the nodes specified next.

**Syntax:**

```
"AUTO"      | [ "AUTRES"          ;
                "NOEUDS" /LECTURE/ ] |
```

**"AUTRES"**

All the ‘remaining’ A.L.E. nodes (i.e. those which have not been forced by a ‘manual’ command such as **SUIVRE**, **LIGNE**, ...) will be automatically rezoned.

**"NOEUDS" /LECTURE/**

Allows to list explicitly the nodes which have to be automatically rezoned.

**Comments:**

Use of the automatic rezoning technique is encouraged when tackling a new problem or in cases when the node pattern and the deformation process cannot be described by simple laws such as those provided by the ‘manual’ rezoning commands. However, compatibility is ensured so that manual commands can still be used in conjunction with the automatic option in case some nodes (usually few) are not properly treated by the automatic technique.

The algorithm starts by estimating node by node, and on the basis of purely geometric criteria, the best grid velocity  $W$  that would bring to an optimal rezoning in just one step.

Then, this velocity is projected onto the fluid velocity  $V$  at the node concerned: this in order to take automatically into account possible boundary conditions imposed at the node.

Finally, the resulting module is limited in order to avoid too high remeshing velocities. This limitation is done by a coefficient  $\gamma_0$ , i.e :

$$-\gamma_0 V < W < (1 + \gamma_0) V$$

The coefficient  $\gamma_0$  may be defined by the option **OPTI REZO GAM0**, see page H.150. This parameter does not have a large influence on remeshing, but in any case with small values of  $\gamma_0$  one should get a slightly more effective remeshing. Suggested values are between 0.1 and 0.8, the default is 0.2.

Note that **GAM0** is a global parameter, and hence it is the same for the whole mesh. Consequently, it is recommended to “help” the remeshing algorithm, in case of need, by the manual remeshing directives such as **SUIVRE**, **LIGNE**, etc.

Note that ALE nodes lying on fluid-structure sliding lines of the ALE type (see directive `FSS ALE`) have to be declared as automatically rezoned. The program then automatically applies the correct sliding conditions.

The list of nodes can be given either explicitly, by a `/LECTURE/`, introduced by keyword `NOEU`, or implicitly, by keyword `AUTR`. In the latter case, the nodes considered are all nodes that have not been assigned any rezoning method up to the current point in the input file. The `GAM0` parameter should be specified in the `OPTI` directive, since it applies not only to Giuliani's but also to the other automatic remeshing methods (in fact, it is used in the mesh velocity restriction algorithm).

Several options may be set for the fine-tuning of the automatic rezoning algorithms. For more information, please see the `OPTI REZO` directive in Section 12.

### **Warning:**

To date, the automatic rezoning facility is only implemented in 2D for nodes belonging to elements of type `TRIA`, `CAR1`, `CAR4`, `FLU1`, `FL23`, `FL24`, `Q41`, `Q41N`, `Q42`, `Q42N`, `TRIA`, `CVL1`, `TVL1`, `MC23`, `MC24`. In 3D, it is implemented for nodes belonging to elements of type `FLU3`, `FL34`, `FL35`, `FL36`, `FL38`, `TETR`, `PRIS`, `CUBE`.

**5.6.12 "MEAN"****Object:**

Use the mean algorithm to determine the motion of the nodes specified next. This rezoning method is available for all ALE element types.

**Syntax:**

```
"MEAN"      $[ "NOEU" /LECTURE/  ;  
              "AUTR"              ]$
```

**"NOEU"**

The following (ALE) nodes will be rezoned by the "mean position" algorithm. The position of each node will tend to become the mean of the position of neighbour nodes. For a generic node I, neighbour nodes are those connected to it in the mesh by a straight (two-noded) element side.

**"AUTR"**

All other 'remaining' ALE nodes, i.e. those that have not been forced to move by a 'manual' command such as "SUIVRE", "LIGNE", etc., will be rezoned by the "MEAN" algorithm.

**Comments:**

Several options may be set for the fine-tuning of the automatic rezoning algorithms. For more information, please see the OPTI REZO directive in Section 12.

**5.6.13 “DIRE”****Object:**

Use the direct algorithm to determine the motion of the nodes specified next. Note, however, that this algorithm is experimental and is currently implemented only for 2D quadrilateral ALE finite elements and finite volumes.

**Syntax:**

```
"DIRE"      $[ "NOEU" /LECTURE/  ;  
              "AUTR"              ]$
```

**"NOEU"**

The following (ALE) nodes will be rezoned by the "direct" algorithm.

**"AUTR"**

All other 'remaining' ALE nodes, i.e. those that have not been forced to move by a 'manual' command such as "SUIVRE", "LIGNE", etc., will be rezoned by the "DIRE" algorithm.

**Comments:**

Several options may be set for the fine-tuning of the automatic rezoning algorithms. For more information, please see the OPTI REZO directive in Section 12.

**Warning**

This algorithm is experimental and is currently implemented only for 2D quadrilateral ALE finite elements and finite volumes.

**5.6.14 “QUAD”****Object:**

Use the specific quadrilateral algorithm to determine the motion of the nodes specified next. Note, however, that this algorithm is experimental and is currently implemented only for 2D quadrilateral ALE finite elements and finite volumes.

**Syntax:**

```
"QUAD"      $[ "NOEU" /LECTURE/  ;  
              "AUTR"              ]$
```

**"NOEU"**

The following (ALE) nodes will be rezoned by the "quadrilateral" algorithm.

**"AUTR"**

All other 'remaining' ALE nodes, i.e. those that have not been forced to move by a 'manual' command such as "SUIVRE", "LIGNE", etc., will be rezoned by the "QUAD" algorithm.

**Comments:**

Several options may be set for the fine-tuning of the automatic rezoning algorithms. For more information, please see the OPTI REZO directive in Section 12.

**Warning**

This algorithm is experimental and is currently implemented only for 2D quadrilateral ALE finite elements and finite volumes.

**5.6.15 “SPEC”****Object:**

Use the element-specific algorithm to determine the motion of the nodes specified next. Note, however, that this algorithm is experimental and is currently implemented only for 2D quadrilateral and triangular ALE finite elements and finite volumes.

**Syntax:**

```
"SPEC"      $[ "NOEU" /LECTURE/  ;  
              "AUTR"              ]$
```

**"NOEU"**

The following (ALE) nodes will be rezoned by the "specific" algorithm.

**"AUTR"**

All other 'remaining' ALE nodes, i.e. those that have not been forced to move by a 'manual' command such as "SUIVRE", "LIGNE", etc., will be rezoned by the "SPEC" algorithm.

**Comments:**

Several options may be set for the fine-tuning of the automatic rezoning algorithms. For more information, please see the OPTI REZO directive in Section 12.

**Warning**

This algorithm is experimental and is currently implemented only for 2D quadrilateral and triangular ALE finite elements and finite volumes.

**5.6.16 “MECA”****Object:**

Use the mechanical algorithm to determine the motion of the nodes specified next.

**Syntax:**

```
"MECA"      $[ "NOEU" /LECTURE/  ;  
              "AUTR"              ]$
```

**"NOEU"**

The following (ALE) nodes will be rezoned by the "mechanical" algorithm.

**"AUTR"**

All other 'remaining' ALE nodes, i.e. those that have not been forced to move by a 'manual' command such as "SUIVRE", "LIGNE", etc., will be rezoned by the "MECA" algorithm.

**Comments:**

Several options may be set for the fine-tuning of the automatic rezoning algorithms. For more information, please see the OPTI REZO directive in Section 12.

**Warning**

This algorithm is experimental and is currently implemented only for 2D quadrilateral ALE finite elements and finite volumes. Finally, in order to use this model the problem type keyword MECA must be specified in the initial part of the input file (see Section 4.2).

**5.6.17 “ELAS”****Object:**

Affects a fictitious elastic material to grid elements to control the motion of the ALE nodes.

**Syntax:**

```
"ELAS" "RO" ro "YOUNG" young "NU" nu "DAMP" damp /LECTURE/
```

"ro"

Fictitious material's density.

"young"

Fictitious material's Young modulus.

"nu"

Fictitious material's Poisson ratio.

"damp"

Inertial damping.

/LECT/

List of the fluid *elements* concerned.

**Comments:**

At the moment, this type of rezoning is available for the following element types: TRIA, CAR1, TETR, PRIS, CUBE, T3VF, Q4VF, TEVF, PRVF, CUVF, FL23, FL24, FL34, FL36, FL38. Note that the model is not yet available for the pyramid elements (FL35 for example).

Be aware that a critical time step is computed for the explicit elastic rezoning problem to remain stable. A soft fictitious material should be used so that this time step is not smaller than the "physical" critical time step.

An large inertial damping coefficient (i.e. from 1.E3 to 1.E4) should be used to prevent vibrating oscillations of the deformed grid.

Do not forget to set option OPTI REZO LIAI (see Page H.150, Section [12.16](#)), so that nodes subjected to kinematic links are rezoned accordingly.



**5.6.18 “GLOB”****Object:**

Option which gives the possibility to link a fluid grid to a structure

**Syntax:**

```
"GLOB"  "DACT"  /LECDDL/  "STRU"  /LECTURE/  
        /LECTURE/
```

"DACT"

Activation of the dlls to be linked

/LECDDL/

Reading procedure of the degrees of freedom concerned.

"STRU"

/LECT/

List of the structure *elements* concerned.

/LECT/

List of the fluid *elements* concerned.

**Comments:**

Multiphasic law is not available with this option

### 5.6.19 USER'S ROUTINE "COOGRI"

#### Object:

This routine can be written by the user and exploited in order to specify the motion of fluid nodes when using the ALE description. Its use should be only needed in exceptional cases, because normally the automatic and manual rezoning directives are perfectly appropriate.

Since this routine is called last by routine NVCOOR any motion specified in it will overwrite any other motion, either automatic or by means of manual rezoning directives, specified by the user.

Normally, the routine does nothing.

The listing of the sample routine is included hereafter.

```

      SUBROUTINE COOGRI(V,WG,postp,mvgril)
C-----
C   ---  ROUTINE UTILISATEUR ( vitesses DE LA GRILLE )
C-----
c v      : fluid velocities
c wg     : mesh grid velocities
c postp  : pointer in both v and wg
C MVGRIL(I,1)
c          -1=LAGRANGIAN
c           0=EULERIAN
C          1=A.L.E.,AUTOMATICALLY REZONED (JRC);
C          2=A.L.E.,MANUALLY REZONED (CEA),
c          3=A.L.E, rezoned by FSS ALE (ALE sliding JRC)
c          4=A.L.E, "MEAN" rezoned (JRC)
C MVGRIL(I,2)  NODAL INDEXES IN THE FOLLOWING ORDER :
C               - FIRST LAGRANGIAN NODES (GROWING ORDER)
C               - THEN NON-LAGRANGIAN BASE NODES (USED AS MASTER
C                 NODES FOR MOTION OF SLAVE A.L.E. NODES)
C               - FINALLY ALL OTHER NODES
C MVGRIL(I,3)  IAD (ADDRESS IN <NBALE> AND <CBALE>) IF MVGRIL(I,1)=2
c              -1  IF NODE IS SUBJECT TO "LIAISON" AND MVGRIL(I,1)=1
C              0   IN ALL OTHER CASES
C-----
c
      implicit none
      include 'CONTRO.INC'
C
      double precision V(*), WG(*)
      integer postp, mvgril
C
      dimension postp(*), mvgril(*)
c
c Insert hereafter the user's definition of the appropriate
c grid velocities:

```

c  
C

RETURN  
END

## 5.7 MESH REFINEMENT FOR WAVEFRONT TRACKING

### Object:

This directive enables the user to impose the refinement of the computational mesh grid to follow the propagation of one or more prescribed wave fronts. It should be used in conjunction with mesh adaptivity dimensioning directive **ADAP**, see page A.62. However, note that this directive is incompatible with “true” adaptivity (piloted by an error indicator) which is activated by the **ADAP** directive of page B.210.

Note that this model does not represent a true implementation of adaptivity, since the mesh refinement and de-refinement is entirely piloted by the user with the present directive. However, it may be useful to check the mesh refinement and de-refinement processes in simple test cases, where the propagation of wave fronts is known a priori.

For a true implementation of adaptivity (piloted by an error indicator) see the **ADAP** directive on page B.210.

### Syntax:

```
WAVE  nwav * ($ SPHE ; PLAN ; CYLI $
              X x Y y <Z z> <NX nx NY ny <NZ nz>>
              TO t0 <T1 t1>
              $ C c ; D d $ <FONC nufo>
              MAXL m H1 h1 H2 h2)
```

#### WAVE

Prescribe one or more wave fronts to be tracked.

#### nwav

Number of wave fronts to be defined.

#### SPHE

The wave being defined is a spherical wave.

#### PLAN

The wave being defined is a plane wave.

#### CYLI

The wave being defined is a cylindrical wave.

#### x

X-coordinate of the wave source point.

#### y

Y-coordinate of the wave source point.

**z**

Z-coordinate of the wave source point (0 by default).

**nx**

X-component of the normal vector to the plane (for **PLAN** waves). X-component of the cylinder axis vector (for **CYLI** waves). Note that the vector need not be normalized to unit length.

**ny**

Y-component of the normal vector to the plane (for **PLAN** waves). Y-component of the cylinder axis vector (for **CYLI** waves).

**nz**

Z-component of the normal vector to the plane (for **PLAN** waves). Z-component of the cylinder axis vector (for **CYLI** waves). This is 0 by default.

**t0**

Time instant at which wave propagation starts from the corresponding source point.

**t1**

Time instant at which wave propagation terminates. The mesh is completely un-refined (thus returning to the base mesh) at times greater than this value.

**c**

Propagation speed of the wavefront. For spherical waves, propagation is assumed isotropic in all space directions.

**d**

Propagation displacement of the wavefront. For spherical waves, propagation is assumed isotropic in all space directions.

**nufo**

Nuber of the function describing the variation in time of the propagation speed (if **C** is given) or of the propagation displacement (if **D** is given). If omitted, the propagation speed or displacement is considered constant in time. Note that the function **must** be defined over the entire time interval of the calculation, i.e. between the initial time **TINI** and the final time **TEND** as defined in the **CALC** directive detailed on Page I.20.

**m**

Maximum level of mesh refinement at the wave front.

**h1**

Thickness of maximum refined mesh layer normally to the wave front.

**h2**

Thickness of refined mesh layer normally to the wave front. Refinement passes from level **m** to level 0 (no refinement) linearly when passing from a distance **h1** to a distance **h2** from the wave front.

## 5.8 ADAPTIVITY

### Object:

This directive enables the user to impose the refinement or un-refinement of the computational mesh grid (**adaptivity**) in accordance to some chosen **error indicator**. The error indicator can be:

- a “classical” error indicator quantity, or
- a point cloud-based indicator, or
- a threshold-based indicator (**refinement only**, by default).

The directive should be used in conjunction with mesh adaptivity dimensioning directive ADAP, see [GBA\\_0062](#).

This directive enables automatic mesh refinement and un-refinement based on some criteria chosen by the user. However, note that at the moment this directive is incompatible both with the wave front tracking directive WAVE [GBB\\_0200](#) and with the FSI-related adaptivity directives associated with the FLSR and FLSW directives, see [GBD\\_0143](#) and [GBD\\_0555](#), respectively.

In contrast to the WAVE directive ([GBB\\_0200](#)), the present model represents a “true” implementation of adaptivity.

### Syntax:

```
ADAP < UPDT /CTIME/ >
( INDI |[ DEPL ; VITE ; ACCE ;
      PRES ; DENS ; CONT icon ; ECRO iecr ]|
  < TYPE |[ CURV ; GRAD ]| >
  STRA $[ PERR perr ; PELE pele ALFA alfa ; PEMA pema ]$
  CERR ( cerr /LECTURE/ ) )

( THRS |[ PRES ; DENS ; PEPS ; FAIL ;
      CONT icon ; ECRO iecr; EPST icon ]|
  TMIN tmin TMAX tmax MAXL maxl
  <CRIT crit>
  <UNSP ; NOUN>
  /LECTURE/ )

$[ PCLD ( |[ INDI NIND nind MAXL maxi ELEM /LECTURE/ ;
      THRS MAXL maxt ELEM /LECTURE/ ;
      FSI FLUI /LECTURE/
      STRU /LECTURE/
      MAXL maxf RADF radf ;
      GAP MAST /LECTURE/
      SLAV /LECTURE/
```

```

MAXL maxg RADI radg
EDGE MAST /LECTURE/
SLAV /LECTURE/
MAXL maxe RADI rade
VOFI FLUI /LECTURE/
CLIM clim
MAXL maxv RADI radv ]| ) ]$

```

**ADAP**

Activates true adaptivity according to the error indicator(s) chosen next.

**Directives related to classical error indicators****UPDT**

MPI Only - Introduces an update frequency for the mesh adaptation to save computation time.

**INDI**

Introduces the variable(s) used as “classical” error indicator(s). Note that exactly **ni** variables must be specified next, where **ni** is the number given in the dimensioning (**NIND ni**), see page A.62. If the keyword **NIND** has been omitted in the dimensioning, then by default **ni** = 1.

**DEPL**

Use nodal displacement (norm) as error indicator.

**VITE**

Use nodal velocity (norm) as error indicator.

**ACCE**

Use nodal acceleration (norm) as error indicator.

**PRES**

Use (fluid) element pressure as error indicator. Only **GRAD** type of indicator can be used in this case.

**DENS**

Use (fluid) element density as error indicator. Only **GRAD** type of indicator can be used in this case.

**CONT icon**

Use element stress component **icon** as error indicator. Only **GRAD** type of indicator can be used in this case.

**ECRO iecr**

Use element hardening component **iecr** as error indicator. Only **GRAD** type of indicator can be used in this case.

## TYPE

Introduces the types of error indicator(s). Note that exactly **ni** variables must be specified next, where **ni** is the number given in the dimensioning (**NIND ni**), see page [GBA\\_0062](#). The types must be entered in the same order as the indicator variables, i.e. the first type corresponds to the first indicator variable, and so on. If the keyword **NIND** has been omitted in the dimensioning, then by default **ni** = 1. If the **TYPE** sub-directive is omitted, all indicators are assumed to be of the curvature type.

## CURV

Error indicator is of curvature type, i.e. the curvature of the indicator variable is used to compute the error indicator. This type of error indicator can be used only for the node-based indicator variables listed above (i.e. only for **DEPL**, **VITE** or **ACCE**).

## GRAD

Error indicator is of gradient type, i.e. the gradient of the indicator variable is used to compute the error indicator. This type of error indicator can be used for all indicator variables listed above.

## STRA

Introduces the strategy used for the error indicator. At the moment, two strategies are available: prescribing the error or prescribing (approximately) the number of used elements.

PERR **perr**

Introduces the prescribed error **perr** ( $\tilde{e}$ ). This is then used to compute the prescribed element size  $\tilde{h}_k$ , see formula below.

PELE **pele**

Introduces the prescribed number of adaptive elements **pele** ( $\tilde{n}$ ).

ALFA **alfa**

Coefficient  $\alpha$  used in the formula to estimate the predicted number of elements in memory (see below). This is an empirical value. The suggested value is 4 for 2D calculations.

PEMA **pema**

Introduces the prescribed number of adaptive elements **pema** ( $\tilde{n}$ ). This version of the command is suited for use in conjunction with the **OPTI ADAP MAXL** option, see [GBH\\_0180](#). In fact when this option is specified, the **PELE** strategy respects very badly the prescribed number of elements. Note that the **PEMA** strategy requires additional calculations with respect to **PELE** and is therefore more expensive. Note also that the **PEMA** strategy makes no use of the **ALFA** coefficient.

CERR **cerr**

Introduces the choice of the constant  $C$  (**cerr**) appearing in the expression of the error indicator (see below). The value may vary from element to element (e.g., due to different element types). Each (parent) element in the adaptive mesh must receive a value. Descendent elements inherit the value from their own parent element.

/LECTURE/ (**CERR keyword**)



List of the elements to which the value `cerr` is assigned.

### Directives related to point-cloud indicators

#### PCLD

Introduce point cloud-based indicators.

#### INDI

Base the point-cloud indicator upon one of the “classical” adaptivity indicators that have been listed above.

#### NIND `nind`

Rank `nind` of the concerned indicator in the INDI list of original adaptivity indicators specified above.

#### MAXL `maxi`

Maximum refinement level `maxi` for this indicator as a PCLD indicator.

#### ELEM /LECTURE/

List of the elements concerned by this PCLD indicator.

#### THRS

Base the point-cloud indicator upon the threshold indicator that has to be defined above.

#### MAXL `maxt`

Maximum refinement level `maxi` for this indicator as a PCLD indicator.

#### ELEM /LECTURE/

List of the elements concerned by this PCLD indicator.

#### FSI

Base the point-cloud indicator upon Fluid-Structure Interaction.

#### FLUI /LECTURE/

List of the fluid elements concerned by this PCLD indicator.

#### STRU /LECTURE/

List of the structure elements defining the points of the cloud.

#### MAXL `maxf`

Maximum refinement level for this PCLD indicator.

#### RADI `radf`

Influence radius associated with the structure.

#### GAP

Base the point-cloud indicator upon contact (gap).

MAST /LECTURE/

List of the elements defining the point cloud.

SLAV /LECTURE/

List of the slave elements concerned by this PCLD indicator.

MAXL maxg

Maximum refinement level for this PCLD indicator.

RADI radg

Influence radius associated with the master elements.

EDGE

Base the point-cloud indicator upon distance from a free shell edge (3D only).

MAST /LECTURE/

List of the shell elements defining the point cloud.

SLAV /LECTURE/

List of the slave elements concerned by this PCLD indicator.

MAXL maxg

Maximum refinement level for this PCLD indicator.

RADI radg

Influence radius associated with the free edges of the master elements.

VOFIRE

Base the point-cloud indicator upon VOFIRE anti-dissipation for physical interface tracking.

FLUI /LECTURE/

List of the concerned fluid elements..

CLIM clim

Minimum concentration for any fluid component to identify an interface cell. One cloud point is placed at the centroid of each interface cell.

MAXL maxv

Maximum refinement level for this PCLD indicator.

RADI radv

Influence radius associated with the interface cells.

### Directives related to threshold-based indicators

**THRS**

Introduce threshold-based indicators.

**PRES**

Use element pressure as threshold-based indicator.

**DENS**

Use element density as threshold-based indicator.

**PEPS**

Use element principle strain as threshold-based indicator.

**CONT icon**

Use element stress component **icon** as threshold-based indicator.

**EPST icon**

Use element strain component **icon** as threshold-based indicator.

**ECRO iecr**

Use element hardening component **iecr** as threshold-based indicator.

**TMIN min**

Introduces the minimum threshold value **tmin** above which the mesh starts to be refined.

**TMAX max**

Introduces the maximum threshold value **tmax** at which the mesh refinement reaches the maximum level (specified below).

**MAXL maxl**

Maximum level of mesh refinement, which is reached at the value **tmax** of the threshold specified above.

**CRIT crit**

Optional criterion to be used to compute the monitored quantity's representative value over an element. By default (or by specifying **CRIT 0**) the average value of the monitored quantity over all Gauss points of the element is taken. Alternatively, 1 means taking the maximum value of all the Gauss points of the element, 2 means the minimum value, 3 means the maximum *absolute* value. Note that alternative criteria (other than the default one based on the average) are *not* available at the moment for monitored quantities of type **PEPS** and **FAIL**. Therefore, the value of **crit** is ignored in these two cases.

**UNSP**

Keyword in order to enable unsplitting procedures for the elements concerned.

**NOUN**

Keyword in order to disable unsplitting procedures for the elements concerned. This is the default, so this keyword should be redundant.

/LECTURE/

List of the elements concerned by this type of mesh adaptation.

### Comments for classical indicators:

For a **curvature-based indicator**, the expression used to compute the error indicator is:

$$|e| \approx Ch^2 \max(|k_1|, |k_2|)$$

where  $C$  is the constant `cerr` given in input for the current element,  $h$  is the local mesh size (i.e. the characteristic length of the element under consideration),  $k_1$  and  $k_2$  are the principal curvatures of the variable chosen as error indicator on a patch composed by the element itself and by all its direct neighbors.

For a **gradient-based indicator**, the expression used to compute the error indicator is:

$$|e| \approx Ch ||G||$$

where  $C$  is the constant `cerr` given in input for the current element,  $h$  is the local mesh size (i.e. the characteristic length of the element under consideration),  $||G||$  is the norm of the gradient of the variable chosen as error indicator on a patch composed by the element itself and by all its direct neighbors.

The expression used to compute the prescribed element size  $\tilde{h}_k$  is:

$$\tilde{h}_k = \sqrt{\frac{\tilde{e}}{e_k}} h_k, \quad k = 1, \dots, N$$

where  $N$  is the current number of active elements in the mesh, and  $e_k$  in the estimated error in the  $k$ -th element.

The expression used to estimate the number of elements  $\tilde{n}$  in memory is:

$$\tilde{n} \approx \frac{\alpha}{\tilde{e}} \sum_{k=1}^N e_k$$

This formula is actually used to compute  $\tilde{e}$ :

$$\tilde{e} \approx \frac{\alpha}{\tilde{n}} \sum_{k=1}^N e_k$$

and then the above formula gives  $\tilde{h}_k$ .

### Comments for PCLD indicators:

The PCLD subdirective introduces particular indicators designed to be combined with one another.

They are mainly based on simple distance relations between the centroids of slave elements and some master points attached to chosen elements, defining a cloud of points. The minimum

distance between a slave centroid and any point of the cloud (which can be seen as the projection of the point onto the cloud) defines the refinement level of the slave element, according to the ratio between this distance and the given radius. The given maximum refinement level is applied to the elements closest to the cloud, and then decreasingly for farther elements.

To be combined with other PCLD indicators, original adaptivity indicators must be slightly reformulated. This is the goal of the PCLD INDI sub-option.

#### **Comments for threshold-based indicators:**

The mesh refinement level varies linearly between 1 and `maxl` as the monitored value passes from `tmin` to `tmax`.

By default, threshold-based indicators cause splitting but never cause un-splitting of a mesh. This is because such indicators are typically associated with irreversible quantities, such as damage or plastification in structures. In order to activate possible un-splitting with threshold-based indicators, the optional keyword `UNSP` should be specified.

## 6 GROUP C—GEOMETRIC COMPLEMENTS

**Object:**

These directives enable the user to complete the geometry.

**Syntax:**

"COMPLEMENT"

**Comments:**

These directives are described in detail on the following pages.

The keyword "COMPLEMENT" and the associated data are compulsory only if required by the elements (shells, beams and 1-D elements) or if the user enters added masses.

Do not forget the corresponding dimensioning (page A.70).

## 6.1 AUXILIARY FILE

### Object :

This directive allows to read complementary data from an auxiliary file.

### Syntax :

```
< "FICHIER"    'nom.fic'  >
```

In certain cases the data may be bulky. It is then advisable to store the data on an auxiliary file in order to shorten the main input data file. The auxiliary file is activated by the keyword "FICHIER" that precedes the full name of the file (under Unix). Then, only the keyword "COMPLEMENT" preceding the keyword "FICHIER" remains in the main input file.

The auxiliary file (in free format) will contain the whole set of geometry complement data, with the exception of the keyword "COMPLEMENT" itself. To resume reading from the main input data file, the auxiliary file must be terminated by the keyword "RETOUR".

## 6.2 FS-Core GEOMETRIC COMPLEMENTS

### Object:

This directive allows to enter geometric complements for structures (and possibly fluids in the future) in FS-Core application.

FSCORE STRUCTURE describes a Fuel Assembly by means of rod network characteristics and global properties of the associated equivalent Timoshenko beam.

### Syntax:

```
"FSCORE" ( "STRU" "RDIA"   rdia
           "RNUM"   inum
           "WIDTH"  rwth
           "PITCH"  rptc
           "IROT"   rirt
           /LECTURE/ )
```

**rdia**

Diameter of rods within Fuel Assembly.

**inum**

Number of rods within Fuel Assembly.

**rwth**

Width of Fuel Assembly.

**rptc**

Pitch of a the rod network within Fuel Assembly.

**rirt**

Rotational inertia of the Fuel Assembly seen as an equivalent beam.

**LECTURE**

List of the FSBM concerned elements.



### 6.3 ADDED MASSES

#### Object:

This instruction enables the user to define the masses which are added to certain nodes, along certain degrees of freedom.

#### Syntax:

```
< "MASS" ( /LECDDL/ xm /LECTURE/ ) >
```

#### LECDDL

List of the degrees of freedom concerned.

#### xm

Value of the added mass.

#### LECTURE

List of the nodes concerned.

#### Comments:

1/ Several added masses may be defined without repeating the key-word "MASS".

Example:

```
"MASS" /LECDDL/ xm1 /LECTURE/
        /LECDDL/ xm2 /LECTURE/
        . . .
```

2/ Use:

```
"MASS" 1342 xm "SUIT" 1 2 3 "TERM"
```

The degrees of freedom 1,3,4,2 of the nodes 1,2,3 are modified by the mass xm (this mass is added to the initial one).

In axisymmetric cases, do not forget to divide the "true" mass by  $2\pi$ .

#### Remark:

Material points ("PMAT" elements) may be used too, in order to enter added masses (see page C.200).

## 6.4 Automatic mass distribution : MAPM

### Object:

The automatic mass distribution "MAPM" is a method that allows the user to complete the mass of an EUROPLEXUS model giving the spatial definition of the total mass. It should be notice that that mass can be only added and not removed.

In the data set, the user defines a spatial masses repartition of a model given the value of each mass and its respective coordinates. For each added mass, four words must be written in the data set: the first one is "MASS" and allows to define the mass value and the three others ("POSX", "POSY" and "POSZ") allow to define its x, y, z position.

### Syntax:

```
< "MAPM"  
  "MASS" m "POSX" x "POSY" y "POSZ" z  
/LECTURE/ >
```

m

mass of the current point

x

X coordinate of the current point

y

Y coordinate of the current point

z

Z coordinate of the current point

/LECTURE/

List of concerned nodes.

### Comments:

Several masses may be defined without repeating the key-word "MAPM".

Example:

```
"MAPM"  "MASS" m1 "POSX" x1 "POSY" y1 "POSZ" z1  
         "MASS" m2 "POSX" x2 "POSY" y2 "POSZ" z2  
         . . .
```

## 6.5 THICKNESS OR SECTION

### Object:

1/ Thickness:

By means of this directive the user specifies the thickness of two- and three-dimensional shells. A thickness must also be specified for elements of types Q92, Q93, Q92A, ED01, ED41, COQI, Q41, Q41N, Q42, Q42N, Q41L, Q42L, Q95, CQD4, CQD9, CQD3, CQD6, FUN2, FUN3. The directive is mandatory if the mesh contains any of these elements.

In case of FUN3 element, the thickness means the cross section of the element.

2/ Section:

This directive is similar to the preceding one, but it is only applied to beams and bars.

### Syntax:

```
< |[ "EPAI" ( ep /LECTURE/ ) ;  
      "EPAI" ( "CQDX" /LCHP/ /LECTURE/ ) ]| >
```

Or:

```
< "SECT" ( ep /LECTURE/ ) >
```

ep

Thickness or section.

LECTURE

List of the elements concerned.

### Comments:

Various thicknesses can be defined for different elements without repeating the key-word "EPAI". It is the same for "SECT".

Example:

```
"EPAI" ep1 /LECTURE/  
      ep2 /LECTURE/  
      ep3 /LECTURE/
```

A special syntax is foreseen to define the thickness of degenerated shell elements CQDx. For these elements, the thickness should be defined at the nodes. The most accurate way of doing this is to prepare a 'champoint' object with CASTEM2000 and store it together with the mesh in the CASTEM2000 file (see directive "SAUV" in CASTEM2000). This file is then read by EUROPLEXUS using the directive "CASTEM" (see page A.30) and can then be referred to from other directives.

The keyword "CQDX" introduces this kind of syntax: the reference to the CASTEM2000 champoint object is read by the /LCHP/ procedure (see page INT.57) and the associated geometrical support (object of type 'maillage' containing the shell elements) is indicated by the following /LECT/. Note that /LCHP/ and /LECT/ must be given in this order.

A simpler, but not as precise, way of specifying the thicknesses of these shells is to assign a single value to each element by the standard "EPAI" directive, without using the CASTEM2000 objects for this purpose. In this case, the program itself estimates values for the thicknesses (and fiber orientations) at each node based on the values of the surrounding elements.

## 6.6 GEOMETRICAL PARAMETERS FOR SHELL ELEMENTS

### Object:

This directive allows to choose some geometrical parameters for shell, plate and beam elements. For example, the type of spatial integration through the thickness or in the lamina directions, for certain types of shell/plate elements.

### Syntax:

```
<"NGPZ"  ngpz  /LECT/ > <"INTE"  typ1 /LECT/>
<"ALPH"  alpha /LECT/ > <"BETA"  beta /LECT/>
<"SK"    sk    /LECT/ > <"REFE"  refe /LECT/>
```

#### ngpz

Number of gauss points in the thickness for shell, plate or beam elements. This value must not exceed the maximum value specified in the dimensioning (see DIME ... NGPZ on page A.66). The default value for the CQDx elements is 3.

#### typ1

Type of lamina integration for 3D degenerated shell elements (CQD3, CQD4, CQD6, CQD9), **SELE** means selective reduced, **REDU** means reduced, **FULL** means full, **SELM** means selectively reduced with ‘mean tau’ procedure and **FULM** means full with ‘mean tau’ procedure; the default is reduced.

#### alpha

Participation to bending (only for some shell elements). Default is 2/3. This parameter is only used by elements which adopt a global model: elements integrated through the thickness ignore the value of this parameter.

#### beta

Participation to membrane (only for some shell elements). Default is 1. This parameter is only used by elements which adopt a global model: elements integrated through the thickness ignore the value of this parameter.

#### sk

Shear correction factor for 3D degenerated shell elements (CQD3, CQD4, CQD6, CQD9), default value is 5/6.

#### refe

Location of the reference surface. *refe* = -1, 0, +1 indicates that the surface is located at the bottom, middle, and top surface of the shell, respectively. The shell element is moved in the positive direction of the element normal.

/LECT/

List of the concerned elements.

### Comments:

The number of gauss points through the thickness for a sandwich element is defined by `COMP SAND NGPZ`, see Page C.45. Therefore, an additional `COMP NGPZ` for the *same* element should be avoided.

The ‘mean tau’ procedure may be applied to CQD3, CQD4, CQD6, CQD9 degenerated shell elements. However, it simply sets the transverse shear values to a mean value, not to a linearly variable pattern. This is likely to be too simplistic for the 9-node element CQD9 (and also for the 6-node CQD6).

The parameter `alpha` is used to modify the bending coefficient for the global shell models. The program will use the following criterion:

$$\text{sig*} = \text{SQRT} (\text{sigm} ** 2 + (\text{alph} * \text{sigf}) ** 2)$$

In this formula, `sig*`, `sigm` and `sigf` represent the Von Mises equivalent stress, the membrane stress and the bending stress, respectively. By default,  $\alpha = 0.666$  (i.e.  $2/3$ ). Of course, this parameter only makes sense for shell elements that use a global model (i.e. which are **not** integrated through the thickness).

Each of the above directives may be repeated as needed to associate appropriate values of the parameters to each concerned element.

## 6.7 EXCENTRICITY FOR SHELL ELEMENTS

### Object:

This (optional) directive allows to specify excentricity for thick shell elements.

### Syntax:

```
<"EXCE" exce /LECT/ >
```

#### **exce**

Distance between the shell mean surface and the reference surface used in the calculation.  
The default value is 0 (no excentricity).

#### **/LECT/**

List of the elements concerned.

### Comments:

For the moment, this feature concerns only T3GS and Q4GS shells [\[939\]](#).

## 6.8 SANDWICHES AND LAYERS

### Object:

To define sandwiches, each composed of several layers, for use with some types of shell elements. Each **SAND** directive defines a new sandwich composed of several layers and associates it with a group of elements.

### Syntax:

```
( "SAND"    nl "FRAC" nl*fracl "NGPZ" nl*ngpzl /LECTURE/ )
```

**nl**

Number of layers in the sandwich, which is associated with each of the elements given in the following **/LECT/**.

**fracl**

Thickness fraction of each layer: this is the ratio of the layer thickness to the total thickness of the element.

**ngpzl**

Number of integration points through the thickness in each layer.

**LECTURE**

Elements concerned.

### Comments:

For the moment, only the elements of type ED01, COQI, CQD3, CQD4, CQD6 and CQD9 may be chosen to be multi-layered sandwiches.

The directive **SAND** may be repeated as necessary (by repeating each time also the **SAND** keyword itself) in order to define all the geometrical information related to sandwiches. There may e.g. be a sandwich (and therefore elements) with, say, 3 layers, and another (other elements) with, say, 5 layers, in the same calculation.

Each sandwich stores all the geometrical information related to the layered structure. However, elements within the same sandwich may be made of different materials (or material combinations in the various layers).

Remember to assign a material to each layer, see page C.750.



The total number of integration points through the thickness of each element is defined by the sum of the `ngpz1` values over the layers of its sandwich. This value should not exceed the maximum value available for each element type. The value can be uncreased by using `DIME ... NGPZ`.

The number of Gauss points through the thickness for each layer is defined by `ngpz1`. Therefore, an additional `COMP NGPZ` for the *same* elements should be avoided.

The set of sandwiches defined in the `SAND` directive(s) is stored in an array. Each layer receives an index corresponding to its definition order within the corresponding sandwich. This index is then used in order to assign a material (see page C.1110) or a set of orthotropy directions (see page C.97) to each layer.

For example, suppose that we define two sandwiches, the first with 3 layers and the second with 5 layers. The layers of the first sandwich will be identified by indexes 1 to 3 while those of the second sandwich by indexes 1 to 5. These are the indexes to be used in successive directives to assign materials and/or orthotropy characteristics to each layer within the corresponding sandwich (i.e., within the associated element).

## 6.9 GEOMETRY OF BEAMS

### Object:

Description of the characteristics of beam elements.

There are four possible shapes:

- arbitrary section QUEL;
- rectangular section RECT;
- circular section CIRC;
- annular section (pipe) TUYA.

Moreover, in the case of pipes, it is possible to enter a curvature in order to model the elbows.

### References:

For an example of use of the various cross sections see e.g. reference [709]. For an example of use of annular sections see also reference [773]. Finally, reference [778] gives an overview of pipelines.

### Syntax:

```
"GEOP" |[ "QUEL"  "VX" vx  "VY" vy  "VZ" vz  "AIRE" aire
           "IY" iy  "IZ" iz  "HY" hy  "HZ" hz
           < "J" j >  "R" r  < "EXCE" ex >           ;

"RECT"  "VX" vx  "VY" vy  "VZ" vz  "AY" ay
        "AZ" az  < "GAUC" gauch >  < "J" j >
        < "EXCE" ex >                               ;

"CIRC"  "VX" vx  "VY" vy  "VZ" vz  "DEXT" diam
        < "EXCE" ex >                               ;

"TUYA"  "VX" vx  "VY" vy  "VZ" vz  "DEXT" diam
        "EP" ep  < "COUR" co >  < "RAYC" rayco >
        < "SFY" sfy  "SFZ" sfz  "SFT" sft >
        < "EXCE" ex >                               ; ]|

< "VMIS"  "APRS" aprs  "AMMB" ammb  "ATRS" atrs  "AFLX" aflx >

/LECTURE/
```

**vx vy vz**

Global coordinates **X**, **Y**, **Z** of a vector **v** defining the local system (**oxyz**), see comments below.

**aire**

Area of the cross section of the beam.

**iy iz**

Bending inertias around the local axes **y** and **z**.

**hy**

Distance along **y** used to estimate an “equivalent bending stress” around the local axis **z**.

**hz**

Distance along **z** used to estimate an “equivalent bending stress” around the local axis **y**.

**j**

Torsional inertia, if different from **iy + iz**.

**r**

Distance used to estimate an “equivalent torsional stress”.

**ex**

Excentricity along **y** (optional).

**ay az**

Length of the sides for a rectangular beam section (along **y** and **z**, respectively)

**diam**

External diameter for a beam with a circular section or for a pipe.

**ep**

Thickness of the pipe making up the beam.

**co**

Curvature of the elbows. It is the inverse of the curvature radius. In the case of a straight pipe: **co** = 0.0 This curvature radius stays constant during a calculation.

**rayco**

Curvature radius for the elbows. It is infinite for straight pipes.

**sfy sfz sft**

Coefficients of “surflexion” of the elbow: in the elbow plane (**sfy**), out of the plane (**sfz**) and in torsion (**sft**). The moment of inertia corresponding to the straight pipe is divided respectively by **sfy** (or **sfz** or **sft**) in order to account for the increased flexibility of the elbows (see also the comments below).

**gauch**

Coefficient of “gauchissement” for the cross-section, allowing to compute the torsional inertia starting from  $J_o = I_y + I_z$  by means of a multiplicative coefficient:  $J_p = \text{gauch} * J_o$ .

**VMIS**

This keyword states that the weighting coefficients that allow to compute the Von Mises criterion starting from the different “equivalent stresses” are given by the user.

**aprs**

Weighting coefficient in the Von Mises criterion for the internal pressure of pipes.

**ammb**

Weighting coefficient in the Von Mises criterion for the membrane stress (normal stress).

**atrs**

Weighting coefficient in the Von Mises criterion for the equivalent torsional stress.

**aflx**

Weighting coefficient in the Von Mises criterion for the equivalent bending stress.

**LECTURE**

List of the elements concerned.

**Comments:**

A local reference system **oxyz** is attached to each beam element. The origin **o** of the system is in node 1 of the element. The second node (node 2) of the element then defines the longitudinal direction of the beam, which is assumed to correspond to the local **x** axis.

Then, to complete the definition of the local reference frame, another direction corresponding to the local **y** axis must be specified. This is done by giving a vector **v** (by means of its global components **vX**, **vY** and **vZ**), which is located in the **oxy** plane and completely defines this plane. Thus, the **v** vector may **not** coincide with **x**. The length of the **v** vector is irrelevant (but of course it may not be zero).

The **y** vector is then computed as the vector normal to **x** and lying in the plane defined by **x** and **v**. Finally, the **z** vector is computed as the vector normal to **x** and **y**.

In the case of an elbow, the vector **v** must be in the elbow plane and directed to the inner side of the elbow, however it is not compulsory that **v** is radial (directed exactly towards the “center” of the elbow). Bending around the **y** axis is therefore outside the elbow plane, while bending around the **z** axis is in the plane of the elbow.

In case of arbitrary cross section, the equivalent stress corresponding to the moment around **y** (respectively **z**) is computed for a distance  $h_z$  from the axis (respectively  $h_y$ ) according to the following formula:

$$\sigma_y = M_y \frac{h_z}{I_y}$$

In the other cases, the formula is identical but the distance  $h$  becomes:

- for a rectangle: the corresponding half-side,
- for a cylinder or a tube: the external radius.

In the case of torsion, it is again the same formula, and the  $h$  distance is then:

- for an arbitrary cross-section: the  $r$  parameter,
- for a rectangle: the semi-diagonal,
- for a cylinder or a tube: the external radius.

In the case of a rectangular cross-section, it is possible to give either a “gauchissement” coefficient allowing to compute the torsional inertia starting from the inertia terms  $i_y$  and  $i_z$ , or to give directly the torsional inertia  $J$ .

For a square cross section this “gauchissement” coefficient has the value 0.844.

The bending inertias are modified in the case of elbows in order to account for the flexibility produced by the curvature as :

$$I_{coude} = I_{droit}/k$$

where  $k$  is **sfy**, **sfz** or **sft**.

ATTENTION: these coefficients are associated with the parameters of the von Mises criterion. A modification of the default values of **sfy**, **sfz** and **sft** imposes a different set of input values for **aprs**, **ammb**, **atrs** and **aflx**.

By default, these coefficients are computed as follows, according to RCCMR 3644.31: *i*) it is assumed that there is no change for the torsion (**sft** = 1). *ii*) it is also assumed that the flexibility is the same in the plane

$$\text{sfy} = \text{sfz} = k$$

.

The coefficient  $k$  is a function of the elbow parameter  $\lambda$ , defined as follows:  $k = 1.65/\lambda$ . By definition, the elbow parameter  $\lambda$  is of the form :

$$\lambda = e R_c / R_m^2$$

where  $e$  is the thickness and  $R_m$  the mean radius of the tube, and  $R_c$  the curvature radius of the elbow. In practice, EUROPLEXUS computes and prints the inverse of this value, that vanishes for a straight tube.

The used Von Mises criterion  $\sigma^*$  is of the form :  $\sigma^* = \sqrt{\alpha_p P^2 + \alpha_n \sigma_n^2 + \alpha_t \sigma_t^2 + \alpha_f \sigma_f^2}$

In this formula, coefficients  $\alpha_p$ ,  $\alpha_n$ ,  $\alpha_t$  and  $\alpha_f$  are respectively the parameters **aprs**, **ammb**, **atrs** and **aflx** defined above. By default, these coefficients assume the following values (which are those for a pipeline):  $\alpha_p = 0.75$ ,  $\alpha_n = 1$ ,  $\alpha_t = 3$  and  $\alpha_f = \pi^2/16$ .

In the case of elbows, the coefficients  $\alpha_t$  and  $\alpha_f$  are modified. The default values are:  $\alpha_t = 0.75$  and  $\alpha_f = (\gamma \pi/4)^2$ .

The coefficient  $\gamma$  has the expression:  $\gamma = \max(1; \frac{8}{9}\lambda^{-2/3})$

**Important:** please consult also **GBG\_0025** for the printout of the results.

## 6.10 DIAMETERS

### Object :

Diameters of the one-dimensional elements "TUBE", "TUYA", "CL1D" and "CLTU".

The elements "TUBE" and "TUYA" may be straight (constant diameter) or conical, but the elements "CL1D" et "CLTU" are always straight.

For practical reasons, there are always two diameters per element (at the inlet and at the outlet, respectively).

### Syntax:

```
"DIAM" $[ "ELAS" ;
           "ELAT" "EPAI" epai "YOUN" youn "NU" nu ]$
| [ "DROI" d1 /LECTURE/ ;
    "CONE" "D1" d1 "D2" d2 "ORIG" /LECTURE/
    "LIST" /LECTURE/ ] |
```

#### "ELAS"

This optional key-word activates an elastic correction of the speed of sound in the fluid of the straight (DROI) TUYA elements listed in the followed /LECT/ sequence. By default, the fluid vena section is rigid. For information about this correction see reference [927].

#### "ELAT"

This optional key-word activates an elastic correction of the speed of sound in the fluid of the TUBE elements listed in the followed /LECT/ sequence. When specified, this key-word must be associated with the key-words "EPAI", "YOUN" and "NU" defining the Allievi correction (characteristics of an equivalent pipe structure). By default, the fluid vena section is rigid. For information about this correction see reference [940].

#### epai

Pipe thickness (optional key-word associated with "ELAT" key-word)

#### youn

Young modulus (optional key-word associated with "ELAT" key-word)

#### nu

Poisson coefficient (optional key-word associated with "ELAT" key-word)

#### "DROI"

Introduces the characteristics of a straight tube.

d1

Diameter of the straight tube.

LECTURE

List of the concerned elements.

"CONE"

Introduces the characteristics of a conical tube.

d1

Inlet diameter, corresponding to node "ORIGINE".

d2

Outlet diameter.

"ORIG"

The following directive /LECTURE/ allows to specify the origin node, where the diameter is d1.

"LIST"

The following directive /LECTURE/ lists the elements concerned.

### Comments :

If the element is straight:  $d1 = d2$ . If the element is conical, d1 is different from d2.

If a conical pipe is composed of several elements, EUROPLEXUS automatically computes the inlet and outlet diameters of each of them.

If some friction (material PAROI) is associated to the fluid material, only cones with a vertex angle  $\alpha$  less than 20 degrees are allowed.

In the last case, EUROPLEXUS computes the friction correction factor  $C_{frot}$ , according to the formulas of IDEL'CIK (diagram 5.2), where  $R$  is the ratio of the areas at the inlet and outlet of the cone ( $R < 1$ ), and  $\lambda$  the loss coefficient for a straight tube:

$$C_{frot} = \frac{\lambda}{8 \sin(\alpha)} (1 - R^2)$$

According to IDEL'CIK, the loss coefficients are then,  $C_{div}$  for a divergent pipe and  $C_{conv}$  for a convergent pipe (diagrams 3.7 and 5.2) :

$$C_{div} = C_{frot} + 3.2 \tan(\alpha)^{\frac{5}{4}} (1 - R)^2$$

$$C_{conv} = C_{frot} + 0.45 (1 - R)$$

## 6.11 NAMED ELEMENT GROUPS

### Object:

To define a (new) set of named groups of elements. Note that the **GROU** directive may be repeated as needed within the **COMP** directive. The set of groups defined is *added* to any pre-existing named groups of elements.

### Syntax:

```
( "GROU"  ngro * ('nom_grou' |[ /LECT/ ; STFL FLUI ; STFL CLXS ; DEBR ]| )
                                <conditions>
                                <"CENT" cx cy cz>
                                <"SHRI" sh>
                                <"SHFT" sx sy sz> )
```

**ngro**

Total number of groups that will be defined.

**nom\_grou**

Name associated with the group, enclosed in quotes.

**/LECT/**

List of the concerned elements.

**STFL FLUI**

Instead of the explicit elements list **/LECT/**, all structured fluid elements (defined using **STFL** command, see page C.68) are taken. Of course, if present this command must be specified *after* the definition of the structured fluid mesh.

**STFL CLXS**

Instead of the explicit elements list **/LECT/**, all **CLxS** elements attached to structured fluid elements (defined using **STFL CLxx** command, see page C.68) are taken. Of course, if present this command must be specified *after* the definition of the structured fluid mesh.

**DEBR**

Instead of the explicit elements list **/LECT/**, all **DEBR** elements are taken. Of course, if present this command must be specified *after* the definition of the debris particles.

**<conditions>**

An optional set of conditions that allow to restrict the chosen elements to a subset of the list specified in the preceding **/LECT/**. See below for the syntax of conditional statements.

**CENT**



Introduces the optional definition of a centerpoint for the group, of coordinates **cx**, **cy**, **cz**, to be used in graphical rendering. In particular, this point is used for shrinking operations (see **SHRI** below). If omitted, the code computes it automatically as the (unweighted) average of the center points of the elements belonging to the group.

**SHRI**

Introduces the optional definition of a shrinkage factor for the group, of value **sh**, to be used in graphical rendering. If omitted, a factor 1.0 is assumed.

**SHFT**

Introduces the optional definition of a shift vector for the group, of components **sx**, **sy**, **sz**, to be used in graphical rendering. If omitted, zero shift is assumed.

**Comments:**

Object names are not case-sensitive: they are converted internally to upper-case before being used.

After their definition, group names may be used to specify in input directives lists of elements (or of the associated nodes), in exactly the same way as GIBI object names are, within a **/LECT/** directive. The set of nodes ‘associated’ with a group of elements is the union of all nodes belonging to the elements in the group.

GIBI object names, or universal format groups or I-DEAS groups have the precedence over the present element groups, in case they are present (and in case of homonymy).

Note that, if element groups are to be passed to the OpenGL-based visualization module, they should preferably be disjoint, i.e. such that each element belongs to (at most) one group. This would ensure independence of rendering from the order in which group selection/unselection operations are performed. However, the code does not enforce this requirement, so that the graphical results are under full control (and responsibility) of the user.

The optional center, shrink and shift definitions may be used e.g. to obtain special graphical rendering effects such as an “exploded” view of a geometrical model. Be aware that the code applies shrinkage by the chosen factor around the centerpoint first, then followed by the chosen shift, if any.

**Conditional statements**

Various types of conditions may be imposed. The first one compares the position of the element’s barycenter to a given value. Another one selects the (single) element whose barycenter is nearest to a given node or point. Other directives allow to identify all elements within a certain geometric shape (a box, a sphere, a cylinder, a cone). The next one allows to build up the complement (symmetric difference) of the chosen object with respect to a second object. The last directive allows to extract the elements that share either all nodes or at least one node with a second object.

```

(COND | $ XB ; YB ; ZB $    $ LT ; GT $  val          |
      | NEAR $ NODE /LECT/ ; POIN x y <z> $          |
      | BOX  <X0 x0> <Y0 y0> <Z0 z0> DX dx DY dy <DZ dz> |
      | SPHE <XC xc> <YC yc> <ZC zc>  R  r          |
      | CYLI <X1 x1> <Y1 y1> <Z1 z1> <X2 x2> <Y2 y2> <Z2 z2>  R  r          |
      | CONE <X1 x1> <Y1 y1> <Z1 z1> <X2 x2> <Y2 y2> <Z2 z2>  R1 r1 R2 r2 |
      | COMP /LECT/                                     |
      | APPU $ STRI ; LARG $ /LECT/                    |)

```

**COND**

Introduces a condition. This keyword may be repeated as many times as necessary to specify multiple conditions, which are applied in sequence.

**XB**

X-coordinate of the element's barycenter.

**YB**

Y-coordinate of the element's barycenter.

**ZB**

Z-coordinate of the element's barycenter.

**LT**

Less than operator.

**GT**

Greater than operator.

**val**

Value.

**NEAR**

Selects the (single) element whose centroid is nearest to a given node or point. If there is more than one element at the minimum distance, then only the first one found is retained.

**NODE**

Specify the node by the following /LECT.

**POIN**

Specify the point by its coordinates x, y and z. The last coordinate is needed only in 3D calculations.

**BOX**

Introduces the definition of a "box", (a quadrilateral in 2D or a parallelepiped in 3D) with the sides aligned with the global axes.

$x_0, y_0, z_0$

Coordinates of the ‘origin’ of the box.

$dx, dy, dz$

Lengths of the box sides.

SPHE

Introduces the definition of a sphere (in 3D, or a circle in 2D).

$x_c, y_c, z_c$

Coordinates of the centre of the sphere (or of the circle).

$r$

Radius of the sphere or of the circle.

CYLI

Introduces the definition of a cylinder (3D only). The cylinder is defined by the two extremities of its axis (P1, P2) and its radius.

$x_1, y_1, z_1$

Coordinates of the first extremity P1 of the cylinder axis.

$x_2, y_2, z_2$

Coordinates of the second extremity P2 of the cylinder axis.

$r$

Radius of the cylinder.

CONE

Introduces the definition of a (truncated) cone (3D only). The cone is defined by the two extremities of its axis (P1, P2) and its radii.

$x_1, y_1, z_1$

Coordinates of the first extremity P1 of the cone axis.

$x_2, y_2, z_2$

Coordinates of the second extremity P2 of the cone axis.

$r_1$

Radius of the cone at the first extremity.

$r_2$

Radius of the cone at the second extremity.

COMP

Introduces a second object to be used for the symmetric difference (complement) operation.

/LECT/

List of the concerned elements.

APPU

Extract the elements that share nodes with an object to be defined next. This command is inspired by Cast3m's *appuyé* operator.

STRI

An elements is extracted if it shares *all* his nodes with the object defined next.

LARG

An elements is extracted if it shares *at least one* of his nodes with the object defined next.

/LECT/

Defines the second object, whose nodes are considered for the previously defined elements extraction.

#### Comments:

If any of the above coordinates (x0, y0 etc.) is omitted, it is assumed to be 0.

#### Example:

Suppose that we want to select all the elements of a 2D object **ob1** that lie in the first quadrant. The syntax would be:

```
COMP ... GROU 1 'firqua' LECT ob1 TERM
                        COND XB GT 0
                        COND YB GT 0
```

The group is from now on accessible under the name **firqua**.

Suppose then that we want to do the same thing as in the previous example, but also get access to the parts of **ob1** in the other three quadrants, under the name **others**. The syntax would be:

```
COMP ... GROU 2 'firqua' LECT ob1 TERM
                        COND XB GT 0
                        COND YB GT 0
                        'others' LECT ob1 TERM
                        COND COMP LECT firqua TERM
```

Note that the **firqua** object becomes available immediately after its definition, and may therefore be used in the definition of the **others** group.

## 6.12 NAMED NODE GROUPS

### Object:

To define a (new) set of named groups of nodes. Note that the **NGRO** directive may be repeated as needed within the **COMP** directive. The set of groups defined is *added* to any pre-existing named groups of nodes.

### Syntax:

```
( "NGRO"  nngr * ('nom_grou' /LECT/ <conditions>) )
```

**nngr**

Total number of node groups that will be defined.

**nom\_grou**

Name associated with the group, enclosed in quotes.

**/LECT/**

List of the concerned nodes, except in the particular case of the **ENVE** condition (see <conditions> below), where this is the list of concerned *elements*.

**<conditions>**

An optional set of conditions that allow to restrict the chosen nodes to a subset of the list specified in the preceding **/LECT/**. See below for the syntax of conditional statements.

### Comments:

Object names are not case-sensitive: they are converted internally to upper-case before being used.

After their definition, group names may be used to specify in input directives lists of nodes (or of the associated elements), in exactly the same way as GIBI object names are, within a **/LECT/** directive. An element is considered ‘associated’ with a group of nodes if and only if all its nodes belong to the group.

GIBI object names, or universal format groups or I-DEAS groups have the precedence over the present node groups, in case they are present (and of homonymy). Moreover, named groups of elements (see page C.61) also have the precedence over the present node groups, in case they are present (and of homonymy).

Note that, if node groups are to be passed to the OpenGL-based visualization module, they should preferably be disjoint, i.e. such that each node belongs to (at most) one group. This would ensure independence of rendering from the order in which group selection/unselection operations are performed. However, the code does not enforce this requirement, so that the graphical results are under full control (and responsibility) of the user.

## Conditional statements

Various types of conditions may be imposed. The first one compares the node position to a given value. Other directives allow to identify all nodes within a certain geometric shape (a box, a sphere, a cylinder, a cone). Other directives allow to order a set of nodes according to their position along a straight line (LINE) or along a curve (CURV). The next one allows to build up the complement (symmetric difference) of the chosen object with respect to a second object. Finally, a directive allows to extract all the nodes located on the envelope (in 3D) or the contour (in 2D) of the object (set of elements) strictly insisting on the set of nodes subjected to the condition (an element is strictly insisting on a set of nodes if all nodes of the element belong to the set).

```
(COND | $ X ; Y ; Z $ $ LT ; GT $ val |
      | NEAR $ NODE /LECT/ ; POIN x y <z> $ |
      | BOX <X0 x0> <Y0 y0> <Z0 z0> DX dx DY dy <DZ dz> |
      | SPHE <XC xc> <YC yc> <ZC zc> R r |
      | CYLI <X1 x1> <Y1 y1> <Z1 z1> <X2 x2> <Y2 y2> <Z2 z2> R r |
      | CONE <X1 x1> <Y1 y1> <Z1 z1> <X2 x2> <Y2 y2> <Z2 z2> R1 r1 R2 r2 |
      | LINE X1 x1 Y1 y1 <Z1 z1> X2 x2 Y2 y2 <Z2 z2> TOL tol <DIST d> |
      | CURV $ X1 x1 Y1 y1 <Z1 z1> ; NOD1 /LECT/ $ |
      | <DX dx DY dy <DZ dz>> <CLOS> |
      | COMP /LECT/ |
      | ENVE <LOCA> |)
```

### COND

Introduces a condition. This keyword may be repeated as many times as necessary to specify multiple conditions, which are applied in sequence.

### X

X-coordinate of the node.

### Y

Y-coordinate of the node.

### Z

Z-coordinate of the node.

### LT

Less than operator.

### GT

Greater than operator.

### val

Value.

### NEAR

Selects the (single) node nearest to a given node or point. If there is more than one node at the minimum distance, then only the first one found is retained.

**NODE**

Specify the node by the following **/LECT**. This node is of course excluded from the search of the nearest node to it.

**POIN**

Specify the point by its coordinates **x**, **y** and **z**. The last coordinate is needed only in 3D calculations.

**BOX**

Introduces the definition of a “box”, (a quadrilateral in 2D or a parallelepiped in 3D) with the sides aligned with the global axes.

**x0**, **y0**, **z0**

Coordinates of the ‘origin’ of the box.

**dx**, **dy**, **dz**

Lengths of the box sides.

**SPHE**

Introduces the definition of a sphere (in 3D, or a circle in 2D).

**xc**, **yc**, **zc**

Coordinates of the centre of the sphere (or of the circle).

**r**

Radius of the sphere or of the circle.

**CYLI**

Introduces the definition of a cylinder (3D only). The cylinder is defined by the two extremities of its axis ( $P_1$ ,  $P_2$ ) and its radius.

**x1**, **y1**, **z1**

Coordinates of the first extremity  $P_1$  of the cylinder axis.

**x2**, **y2**, **z2**

Coordinates of the second extremity  $P_2$  of the cylinder axis.

**r**

Radius of the cylinder.

**CONE**

Introduces the definition of a (truncated) cone (3D only). The cone is defined by the two extremities of its axis ( $P_1$ ,  $P_2$ ) and its radii.

**x1**, **y1**, **z1**

Coordinates of the first extremity  $P_1$  of the cone axis.

x2, y2, z2

Coordinates of the second extremity  $P_2$  of the cone axis.

r1

Radius of the cone at the first extremity.

r2

Radius of the cone at the second extremity.

LINE

Introduces the definition of a straight line. The line is defined by the two extremities ( $P_1$ ,  $P_2$ ), a relative tolerance  $\epsilon$  (TOL) and an optional relative spacing  $\delta$  (DIST) between the nodes to be retained. The nodes which fall on the line are extracted, listed in the order in which they occur passing from  $P_1$  to  $P_2$  (included).

In 3D space, the line passing through 2 points  $P_1(x_1, y_1, z_1)$  and  $P_2(x_2, y_2, z_2)$  has the following parametric equations:

$$\begin{aligned}x &= x_1 + dx\,t, & dx &= x_2 - x_1 \\y &= y_1 + dy\,t, & dy &= y_2 - y_1 \\z &= z_1 + dz\,t, & dz &= z_2 - z_1\end{aligned}$$

From each equation a value for  $t$  can be defined:

$$\begin{aligned}t_x &= \frac{x - x_1}{dx} \\t_y &= \frac{y - y_1}{dy} \\t_z &= \frac{z - z_1}{dz}\end{aligned}$$

These 3 quantities are calculated for each point of the geometrical object to which the LINE condition is being applied and, if the 3 following conditions are satisfied, then the point is retained.

- $t_x, t_y, t_z$  are equal or at least their difference (in absolute value) is smaller than or equal to the defined tolerance  $\epsilon$  (for the point to lie on the line defined by  $P_1, P_2$ ).
- The common value of  $t_x, t_y, t_z$  is between  $(0 - \epsilon)$  and  $(1 + \epsilon)$  (for the point to lie between  $P_1$  and  $P_2$ , extremities included).
- If  $\delta$  is not zero, then the value of  $t$  divided by  $\delta$  (DIST) should be in the vicinity of an integer number (the length of the vicinity is defined by the input tolerance  $\epsilon$ ).

x1 y1 z1

Coordinates of the first extremity  $P_1$  of the line.

x2 y2 z2

Coordinates of the second extremity  $P_2$  of the line.



tol

Tolerance (relative)  $\epsilon$  for searching the nodes on the line. The absolute tolerance  $\tau$  is equal to the relative tolerance multiplied by the distance  $L$  between  $P_1$  and  $P_2$ :  $\tau = L\epsilon$ .

d

Parametric (relative) distance  $\delta$  between two consecutive nodes retained on the line. If omitted, all nodes on the line are retained. If specified, then the relative distance between two consecutive retained nodes will be greater or equal to  $\delta$  (and so possibly some nodes on the line will be skipped). The absolute distance  $d$  is equal to the relative distance multiplied by the distance  $L$  between  $P_1$  and  $P_2$ :  $d = L\delta$ .

CURV

The nodes are listed in the order in which they occur along the minimum-distance curve starting at one of them. The first node of the resulting ordered set is either the node closest to a given position in space  $(x_1, y_1, z_1)$  or an explicitly chosen node  $N_1$ . The next node is the closest one to the previous one, and so on, until *all* the selected nodes have been used. Note that this behaviour is different from the **LINE** directive, where only those selected nodes which are actually on the line specified (within the given tolerance) are retained. Optionally, the curve can be closed (**CLOS**), by repeating the first node at the end of the ordered set. Finally, the optional **DX**, **DY**, **DZ** sub-directive allows to specify the preferential direction of search for the *second* node in cases where there might be an ambiguity. This can be useful if there are two or more nodes at the same distance (within a small tolerance) from the first node. Then, among the candidates we choose the one whose position vector with respect to the first node forms the largest dot (scalar) product with the given direction. An example of use of the **DX**, **DY**, **DZ** and **CLOS** optional keywords is as follows. Consider a series of nodes equi-spaced along a circle. We want to construct the curvilinear abscissa along the entire circle, starting at one node. There are two nodes at substantially the same distance to the chosen first node (a *left* one and a *right* one, so to say). The **DX**, **DY**, **DZ** keywords allow to choose whether to move along the circle in one sense or in the other. The **CLOS** keyword allows to obtain a curvilinear abscissa of the same length as the (discretized) circumference, instead of the circumference minus one sub-division.

x1 y1 z1

Coordinates of the first extremity  $P_1$  of the curve. The first node in the ordered set is the node, among the previously selected ones, that is closest to  $P_1$ .

NOD1 /LECT/

Allows to explicitly choose the first node in the ordered set ( $N_1$ ). Note that  $N_1$  must be among the previously selected nodes.

dx dy dz

Preferential direction for the search of the *second* node of the ordered set, in case there are two or more selected nodes that are (approximately) equi-distant from the first node.

CLOS

Close the curve, i.e. close the ordered set. The index of the first node is added (repeated) to the end of the ordered set.

**COMP**

Introduces a second object to be used for the symmetric difference (complement) operation.

**/LECT/**

List of the concerned nodes.

**ENVE**

Extract all the nodes located on the envelope (in 3D) or the contour (in 2D) of the object (set of elements) *strictly insisting* on the set of nodes subjected to the condition. An element is *strictly* insisting on a set of nodes if *all* nodes of the element belong to the set. By default, the *global* envelope/contour is considered. That is, a node is on the envelope/contour of the object if it belongs to an external face of an element of the object, i.e. to a face that has no neighbour *in the whole mesh*. This behaviour may be modified by the **LOCA** keyword, see below. Attention: **CLxx** elements are NOT considered as valid neighbours for the search of envelope/contour.

**LOCA**

The *local* envelope/contour is considered. That is, a node is on the envelope/contour of the object if either it belongs to an external face of an element of the object i.e. to a face that has no neighbour *in the whole mesh*, or it has a neighbour but this neighbour is not part of the elements of the object. Attention: **CLxx** elements are NOT considered as valid neighbours for the search of envelope/contour. Clearly, if the object to which the **ENVE** condition is being applied is the set of all nodes in the mesh (**tous**), then specifying or not the **LOCA** keyword has no effect on the result.

**Comments:**

If any of the above coordinates (**x0**, **y0** etc.) is omitted, it is assumed to be 0.

**Example:**

Suppose that we want to select all the nodes of a 2D object **ob1** that lie (strictly) within the first quadrant. The syntax would be:

```
COMP ... NGRO 1 'firqua' LECT ob1 TERM
          COND X GT 0
          COND Y GT 0
```

The group is from now on accessible under the name **firqua**.

Suppose then that we want to do the same thing as in the previous example, but also get access to the nodes of **ob1** in the other three quadrants, under the name **others**. The syntax would be:

```
COMP ... NGRO 2 'firqua' LECT ob1 TERM
      COND X GT 0
      COND Y GT 0
      'others' LECT ob1 TERM
      COND COMP LECT firqua TERM
```

Note that the `firqua` object becomes available immediately after its definition, and may therefore be used in the definition of the `others` group.

## 6.13 ELEMENT COLORS

### Object:

To define or re-define (e.g. in the case of a mesh generated by Cast3m) the colors of elements, for visualization purposes.

### Syntax:

```
"COUL" (nom_coul /LECT/)
```

`nom_coul`

Name of the color (**not** enclosed in quotes). The valid names are those of Cast3m, i.e. `bleu`, `roug`, `rose`, `vert`, `turq`, `jaun`, `blan` or `noir`, plus the following nine gray levels: `gr10` (almost black), `gr20`, `gr30`, `gr40`, `gr50`, `gr60`, `gr70`, `gr80` and `gr90` (almost white). Elements not assigned a color have a default color.

`/LECT/`

List of the concerned elements.

### Comments:

Repeat as many times as necessary to define all the desired colors.

This directive is particularly useful in conjunction with the definition of element groups (see `GROU`) to assign colors to groups of elements.

If there are several colors to be defined, be sure **not** to repeat the keyword `COUL`, but only the color name `nom_coul` followed by the corresponding `/LECT/`. In fact, each time the keyword `COUL` is encountered, all colors defined so far are reset to the default color (black). For example:

```
COUL roug LECT explosive TERM
      vert LECT structure TERM
```

is correct, while:

```
COUL roug LECT explosive TERM
COUL vert LECT structure TERM
```

would be wrong (the `explosive` object would appear black and not red).

## 6.14 RTM COMPOSITE MATERIALS

### Object:

This directive allows to define values to used by a composite material made by a RTM process ie. the CRTM material (page C264).

### Syntax:

"RTMVF"    vf        /LECTURE/

"RTMRCT"   rct       /LECTURE/

"RTMANGL"   angle    /LECTURE/

vf

Value of the volumic fraction

rct

Value of the ratio between warp and weft

angle

Value of the angle between warp and weft directions.

LECTURE

List of the elements concerned.

## 6.15 PFEM METHOD

### Object:

Warning: the present model is under development at JRC and not all the directives described below are available yet!

This directive sets some parameters used by the PFEM method.

### Syntax:

```
PFEM  H h ALPHA alpha
```

**h**

Expected distance between nodes in the Bowyer-Watson triangulation.

**alpha**

Alpha coefficient for the Alpha-shape method to determine the contour of the triangulation.

## 6.16 FLYING DEBRIS MODEL

### Object:

This directive allows to model flying debris resulting from an explosion or an impact. Each piece of debris is modeled by a particle, optionally embedded in the surrounding fluid and optionally subjected to the gravity force. This latter force, if present, must be specified via the CHAR CONS GRAV directive, see page F.30.

The fluid surrounding the debris particles may be modeled either as a uniform field with constant properties (velocity, density), or as an evolving fluid field, discretized by Finite Elements or Finite Volumes, or even as a combination of the two (e.g., FE fluid field near the explosive source, uniform field far away).

The drag pressure acting on a particle is:

$$\underline{p}_D = -C_D \frac{1}{2} \rho_f ||\underline{w}||^2 \frac{\underline{w}}{||\underline{w}||}$$

where  $C_D$  is the particle's drag coefficient (see below),  $\rho_f$  is the fluid's density and  $\underline{w}$  is the particle's velocity  $\underline{v}$  relative to the fluid velocity  $\underline{v}_f$ :  $\underline{w} = \underline{v} - \underline{v}_f$ .

Then, the drag force  $F_D$  exerted by the fluid on the particle is computed by multiplying the drag pressure by the exposed area  $A$  of the particle:

$$F_D = p_D A = p_D \pi \frac{d^2}{4}$$

where  $A = \frac{\pi}{4} d^2$  is the particle's cross-section, with  $d$  the diameter of the particle (assumed to be spherical).

One may also activate a feed-back mechanism whereby the drag forces generated by the fluid on the particles are applied (with the minus sign) to the fluid itself. This is currently the default when the fluid is discretized by FE (but not by VFCC). Note, however, that in general it is preferable to deactivate the feedback mechanism (see option NOFB below), since it perturbs the fluid flow, while the drag formula (whose coefficient  $C_D$  is determined experimentally) assumes an unperturbed fluid flow. In other words, in order to compute the relative velocity  $\underline{w}$  to be introduced in the formula, the fluid velocity should be taken at a certain distance from the particle, where the effect of the particle is not felt. Furthermore, note that at the moment feedback forces are only applied (by default, i.e. without the NOFB option) when the fluid is discretized by Finite Elements (FE). They cannot be applied (with or without NOFB) to a fluid discretized by Cell-Centred Finite Volumes (VFCC).

The debris particles may either be active from the very beginning of the calculated transient (thus assuming that they result from a fragmentation process that occurred at a previous time), or they may be associated with certain finite elements representing a structure, and be activated automatically by the code only when the element undergoes complete failure.

In any case, since particles are represented by the specialized DEBR elements, and since the topology is basically constant in time in EUROPLEXUS, **all particles must be declared** (and thus they are present in the model) **from the beginning of the calculation**. However, some of them may be already active at the initial time, some not.

The model includes the optional treatment of the impact of debris particles against the surrounding structure. This is accomplished by the pinball method.

### Syntax:

```
DEBR ![ <ROF rof>
      <VFX vfx> <VFY vfy> <VFZ vfz>
      <FLUI /LECT/ <DGRI> $[ HGRI hgri ; NMAX nmax ; DELE dele ]$ >
      < $ FBAC ; NOFB $ > ]!
      (PART particle_description)
      (FILL fill_description)
```

### rof

Density of the (default) uniform fluid field in which the particles are embedded. This value is 0.0 by default, meaning that by default the particles move in vacuum (if they are not coupled with a discretized fluid domain by the **FLUI** keyword). Note that the drag force acting on a particle depends on the density but also on the drag coefficient (**DRAG**, see below). By setting **DRAG** to 0, the drag force vanishes even though the density is not zero.

### vfx, vfy, vfz

Components of the velocity of the surrounding uniform fluid field. These values are 0.0 by default.

### FLUI

Introduces the **/LECT/** of the discretized fluid domain with which the particles motion should be coupled. When a particle traverses this domain, the (local) fluid velocity and density are automatically computed by the code, instead of using the constant user-given values **rof**, **vfx**, **vfy**, **vfz** described above. A fast search algorithm based on a grid of cells (as in bucket sorting) is used to compute the fluid element (if any) encompassing each debris particle.

### DGRI

Dump out the initial grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed, that is, when some active debris are created.

### HGRI

Specifies the size of the grid cell. Each cell has the same size in all spatial directions and is aligned with the global axes. Note that the size of this grid is related to the size of the **fluid** elements specified in **FLUI**, not of the structural elements producing the flying debris.

### NMAX

Specifies the maximum number of cells along one of the global axes.



**DELE**

Specifies the size of the grid cell as a multiple of the diameter of the largest coupled **fluid** element. Element “diameters” are computed only along each global spatial direction and the maximum is taken. For example, by setting **DELE 4** the size of the cell is four times the diameter of the largest coupled fluid element. By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DELE** are specified, the code takes **DELE 3** (this value is probably too large, a value of 1.1 or so should be more appropriate in most cases).

**FBAC**

Activate the feed-back mechanism whereby the drag forces generated by the fluid on the debris particle are also applied (with the minus sign) to the fluid itself. This is the default, although in many cases it is preferable to deactivate the feedback mechanism, since it perturbs the fluid flow while the drag formula assumes an unperturbed fluid flow (the fluid velocity should be taken at a certain distance from the debris particle, where the effect of the debris particle is not felt). Furthermore, note that at the moment feedback forces can only be applied when the fluid is discretized by Finite Elements (FE), not by Cell-Centred Finite Volumes (VFCC).

**NOFB**

Deactivate the feed-back mechanism whereby the drag forces generated by the fluid on the debris particle are also applied (with the minus sign) to the fluid itself. If the fluid is modelled by VFCC, activating or not the feedback mechanism has no influence on the solution because the feedback is not available for this type of fluid discretization.

**PART**

Introduces the description of a single particle, see details below. Such particles are active from the very beginning of the calculation. This directive may be repeated any number of times.

**FILL**

Fill by particles a single finite element or a finite element mesh. Such particles may either be active from the very beginning of the calculation, or be activated upon failure of the associated element(s). See below for the details of this directive. This directive may be repeated any number of times.

**Dimensioning for the flying debris:**

Dimensioning for the flying debris cannot be made fully automatic, because of the **FILL** command which generates a variable number of particles depending upon which finite element type it is applied to and upon how many such elements will actually fail.

The number given in the dimensioning is the maximum number of debris particles that can be generated in the calculation. If the number given is not sufficient, the code will issue a warning message but the calculation will be continued (without generating any more debris particles). At the end of the run, the code will print out the exact number of particles needed (should the calculation be repeated).

The following two-step procedure is suggested:

- Prepare an input file containing the `COMP DEBR ...` directive and a tentative dimensioning for the debris. Run the calculation. If the given dimension is insufficient, the code will print a warning message and it will continue the calculation as explained above. At the end of the run, the code will print the total number of `DEBR` needed, say `n_debr`.
- Add to the input file the directive `DIME DEBR n_debr TERM`. Next time the calculation should run smoothly. If the printed `n_debr` is less than or equal to the number given, of course it is not necessary to re-run the calculation (but you may want to adjust the dimension at the exact value anyway, should you need to re-run the test case for any reason).

### Dimensioning in case of domain decomposition (MPI):

In case of a calculation with domain decomposition (MPI), the dimensioning for the flying debris can assume two forms:

- If one gives `DIME DEBR n_debr` with `n_debr > 0`, then the code will assume this size of the debris *on each processor* (so that the maximum total number of debris will be `n_debr` times the number of processors `n_proc`).
- If one gives `DIME DEBR n_debr` with `n_debr < 0`, the code will assume this size of the debris *in total* (so that the maximum number of debris on each processor will be `n_debr` divided by the number of processors `n_proc`).

### Comments:

Debris particles may be subjected to the gravity force. This latter force, if present, must be specified via the `CHAR CONS GRAV` directive, see page F.30.

The total number of particles described by the `PART` and `FILL` sub-directives must be less than or equal to the number of elements of type `DEBR` that has been reserved in the dimensioning of the problem.

### Describing a single particle

#### Object:

To describe a single particle of debris. The particle is already active at the beginning of the transient calculation. Therefore, it results from the fragmentation of a structure which has occurred at a previous time.

#### Syntax:

```
PART <X x> <Y y> <Z z>    <VX vx> <VY vy> <VZ vz>
      RO ro  D d  DRAG drag
      <COUP> <IMPA> <TRAJ> <RISK> <ERIM>
```

x, y, z

Coordinates of the particle at the initial time. These values are 0.0 by default.

vx, vy, vz

Velocity components of the particle at the initial time. These values are 0.0 by default.

ro

Density of the particle.

d

Diameter of the particle.

drag

Drag coefficient of the particle. This is a number usually between 0.3 and 1.1 for a sphere. In the supersonic region the value is almost constant and close to 1.0, while it drops rapidly in the transonic region.

COUP

Couple the particle's motion with the surrounding fluid domain defined by the FLUI directive above. Note that this keyword is mandatory if one wants to activate risk evaluation (see the RISK keyword below).

IMPA

Treat the impact of the particle against surrounding structures (by the pinball method). One (parent) pinball is embedded in the particle, of diameter equal to that of the particle. Pinballs for the potentially impacted structures must be defined separately by the PINB directive, see page D.480.

TRAJ

Save the particle's trajectory, e.g. for visualization purposes. By default, the particle trajectory will consist of 100 points equispaced in time between the initial and final times of the calculation, but the number of points can be set by the OPTI DEBR NTRA option (see page H.45). Beware that, by default, the trajectory data are **not** stored in the ALIC results file. Therefore, visualization of the particles trajectory can only be done during the main calculation. In order to draw the trajectories when reading back results RESU from an ALIC file, one must activate the OPTI DEBR STTR option (see page H.45), which stores the trajectory data on the ALIC file (beware that the size of the file may increase considerably if there are many particles and many points per trajectory).

RISK

The risk of death due to the impact of the current flying debris particle on the human body is calculated. This keyword is only accepted if the RISK keyword has been specified in the problem type definition (see Page A.30). Furthermore note that, in order to actually compute the risk, the keyword COUP must also be specified as explained above. The debris-related risk calculation is based on the Lewis model and requires the presence of a (discretized) fluid domain. In fact, the risk is evaluated in each fluid element (or volume) as this is traversed by the flying debris particles.

ERIM

Activates the erosion of the debris after an impact by pinball contact.

## Filling an element or a mesh by particles

### Object:

To fill by debris particles an element or a mesh. Particles are automatically generated uniformly within the volume of the element or mesh (element by element). The particles inherit the density of the parent element's material when they are generated.

The particles may either: 1) be already active at the beginning of the transient calculation (in this case they result from the fragmentation of a structure which has occurred at a previous time), or 2) be activated automatically by the code when the associated element(s) undergo complete failure.

In case 1) above, the associated element or mesh is defined at the geometric level only as a geometric support for the particles generation. This element or mesh must be associated with a **FANT** material so as to exclude it from the transient computation. Assign to the **FANT** material the desired density, which will be inherited by the generated particles.

In case 2) above, the associated element or mesh must be assigned a structural material with a failure model (thus **not** the **FANT** material). When the element(s) fail, the associated particles are suddenly activated while at the same time the element is deactivated, so it no longer contributes to the model.

### Syntax:

```
FILL  $<VX vx> <VY vy> <VZ vz> ; <VR vr <CX cx> <CY cy> <CZ cz>>>$
      PLEV plev DRAG drag <AFly afly>
      <COUP> <IMPA> <TRAJ> <RISK <MACR <RMAC rmac>>>
      <ERIM>
      OBJE /LECT/
```

**vx, vy, vz**

Components of the velocity of the particles at the initial time. This value is 0.0 by default.

**vr**

Radial velocity of the particles at the initial time. This value is 0.0 by default.

**cx, cy, cz**

Coordinates of the particles centroid with respect to which the radial velocity is expressed. By default this is the centroid of the geometrical object defined by the **/LECT/** directive given below.

**plev**

Level of hierarchic subdivision of the parent element(s) along each spatial direction in order to generate the particles. The particles' diameter is automatically determined so as to conserve the element's total volume. For example, a level of 3 means that  $2^3 = 8$  subdivisions along each spatial direction to generate the particles. A 2D quadrilateral element would in this case be filled by  $8 \cdot 8 = 64$  particles.

**drag**

Drag coefficient of the particles. This is a number usually between 0.3 and 1.1 for a sphere. In the supersonic region the value is almost constant and close to 1.0, while it drops rapidly in the transonic region.

**aflly**

The drag forces and the AIRB forces depend on the area of the spherical particle, which can be different from the "true" debris cross section. For each shell element and beam element a minimum and a maximum area are estimated. By using the keyword AFLY the minimum (**aflly** = 0.0) or the maximum (**aflly** = 1.0) value is used. Values of **aflly** between 0.0 and 1.0 interpolate linearly between these two values. The default value is 0.5. For solid elements, the cross section of the spherical particle is used and so **aflly** is ignored.

**COUP**

Couple the particles' motion with the surrounding fluid domain defined by the FLUI directive above. Note that this keyword is mandatory if one wants to activate risk evaluation (see the RISK keyword below).

**IMPA**

Treat the impact of the particles against surrounding structures (by the pinball method). One (parent) pinball is embedded in each particle, of diameter equal to that of the particle. Pinballs for the potentially impacted structures must be defined separately by the PINB directive, see page D.480.

**TRAJ**

Save the particle's trajectories, e.g. for visualization purposes. By default, the particle trajectory will consist of 100 points equispaced in time between the initial and final times of the calculation, but the number of points can be set by the OPTI DEBR NTRA option (see page H.45). Beware that, by default, the trajectory data are **not** stored in the ALIC results file. Therefore, visualization of the particles trajectory can only be done during the main calculation. In order to draw the trajectories when reading back results RESU from an ALIC file, one must activate the OPTI DEBR STTR option (see page H.45), which stores the trajectory data on the ALIC file (beware that the size of the file may increase considerably if there are many particles and many points per trajectory).

**RISK**

The risk of death due to the impact of the active flying debris particles on the human body is calculated. This keyword is only accepted if the RISK keyword has been specified in the problem type definition (see Page A.30). Furthermore note that, in order to actually compute the risk, the keyword COUP must also be specified as explained above. The debris-related risk calculation is based on the Lewis model and it is applied to the active debris

particles which are produced after the erosion of an element. The risk calculation requires the presence of a (discretized) fluid domain. In fact, the risk is evaluated in each fluid element (or volume) as this is traversed by the flying debris particles.

#### MACR

This keyword adds to the risk calculation described above (related to the flying debris particles produced **after** erosion of an element) also the calculation of the risk related to the impact of macro fragments (i.e., **before** the erosion of an element). To estimate such risk, “spurious” (inactive) particles (“markers”) are attached to the elements (one particle per element) from the very beginning of the calculation (i.e., as long as the elements do not fail). In a recently proposed strategy which, however, has *not* been implemented yet, the risk estimation would take into account not only the mass of the current particle (current element) but also the mass of the surrounding (unfailed) elements, up to a certain influence radius that can be specified by the user (**RMAC**, see below). If an element fails and is eroded, the corresponding marker particle is suppressed and a suitable number of debris particles are activated (for which the impact risk is then computed normally).

#### RMAC

Attention: this keyword is silently accepted by the code but the corresponding model has *not* being implemented yet, so the keyword has *no effect*. Introduces the influence radius (**r<sub>mac</sub>**) for the evaluation of the impact risk of macro fragments. If a marker impacts (penetrates through) a fluid element, the risk is estimated by using the mass of all (unfailed) elements connected with the current one within a distance equal to **r<sub>mac</sub>**. By specifying **RMAC 0** an infinite radius of influence is taken. This means that the total mass of the macro fragment is considered in the impact.

#### ERIM

Activates the erosion of the debris after an impact by pinball contact.

#### OBJE

Introduces the list of the elements to be filled.

#### /LECT/

List of the elements to be filled.

#### Comments:

The initial velocity of each group of particles defined by the **FILL** directive described above may be defined in two ways:

- Either it is assumed as uniform for all particles: in this case specify the necessary Cartesian components **vx**, **vy** and **vz**;
- Or, it is assumed to be oriented radially from a point: in this case specify the velocity modulus **vr**. By default, the assumed point is the centroid of the geometric object being filled. However, the user may specify a different point by giving **cx**, **cy** and **cz**.

**Correction of the drag formula (June 2019)**

The formula of the drag force used in the initial implementation of the flying debris model was:

$$\underline{F}_D = -C_D \rho_f A ||\underline{w}||^2 \frac{\underline{w}}{||\underline{w}||}$$

that is, the factor 1/2 was originally *not* included. A correction was introduced in June 2019 where the factor 1/2 was added (also for uniformity with the DRAG model described on Page D.630) and at the same time the optional keywords **FBAC** and **NOFB** were implemented.

In order to exactly reproduce with the new code version results once obtained with a version of EPX older than June 2019, it is sufficient to multiply by 2.0 the value of the drag coefficient  $C_D$  given in the input (DRAG keyword). Thus, for example, if the old input used DRAG 1.0, replace it by DRAG 2.0.

## 6.17 DISPLACEMENT EROSION

### Object:

This directive allows to define an erosion (failure) criterion, which uses a maximum displacement of a given node.

The model can be used for calculations of laminated windows. The criterion of the complete erosion of a laminated window can be set to 30% of the span.

The model can be combined with any other erosion criterion.

### Syntax:

```
FAIL ( DISP disp NODE /LECT/ OBJE /LECT/ )  
      ( AUTO rati DIRE disp /LECT/ )
```

#### disp

Displacement of the node given by the keyword **NODE**, which results in an erosion (failure) of the elements given by the keyword **OBJE**.

#### node

Node used for the displacement criterion. The following **/LECT/** must contain just one node index.

#### OBJE

Introduces the **/LECT/** of the elements which are eroded, if the criterion is reached.

#### AUTO

The keyword **AUTO** introduces an automatic development of the nodes used for the displacement criterion and the element which should eroded. The elements given by **/LECT/** are separated to several subsets, the node near the barycentre is used for the criterion, the full subset is used for the elements which could be eroded.

#### rati

Ratio of the minimum span which should be used for the displacement criterion.

#### dire

The maximum span is defined in this direction.

### Remarks:

The set of keywords **DISP ... OBJE** may be repeated as many times as needed to define all the desired displacement-based erosion criteria.



## 6.18 STRUCTURED FLUID GRID MODEL

### Object:

This directive allows to define a structured, Eulerian fluid grid consisting of either Finite Elements (FE) or of Cell Centred Finite Volumes (VFCC) that is added to the mesh specified in the **GEOM** directive. The grid has the form of a rectangular parallelepiped, is aligned along the global axes, and has a uniform spacing in each of the three global directions.

Using a structured fluid grid may substantially speed up the numerical calculations because many operations (especially those related to searching) can be highly optimized. In particular, this model is useful in conjunction with the **FLSR** model for fluid-structure interaction, see page D.143, in the case of fluid FE, or with the **FLSW** model, see page D.555, in the case of fluid VFCC.

If FE are chosen for the fluid, special fluid elements of type **FL2S** (in 2D) or **FL3S** (in 3D) are automatically built up and used to discretize the structured grid. The former is a simplified version of **FL24** while the latter is a simplified version of **FL38**. Alternatively, one may activate use of CEA's finite elements **CAR1** and **CL2D** (in 2D), **CUBE** and **CL3D** (in 3D), by the CEA optional keyword (see below).

If VFCC are chosen for the fluid, fluid volumes of type **Q4VF** (in 2D) or **CUVF** (in 3D) are automatically built up and used to discretize the structured grid.

All nodes of the structured fluid grid (which are also generated automatically) must be declared Eulerian in the **GRIL** directive, see page B.60. Note that nodes not mentioned in the **GRIL** directive are indeed considered Eulerian.

This directive may only be used in ALE or purely Eulerian calculations.

In addition to the fluid elements (or volumes), special boundary condition elements of type **CL2S** (in 2D) or **CL3S** (in 3D) (for the FE fluid case) or of type **CL2D** (in 2D) or **CL3D** (in 3D) (for the fluid VFCC case) may be optionally generated along the appropriate faces of the fluid domain (see **CLij** input directives below). These may be used, for example, to specify absorbing boundary conditions.

### Syntax:

```
STFL <VFCC> <CEA>
      X0 x0 Y0 y0 <Z0 z0>
      LX lx LY ly <LZ lz>
      NX nx NY ny <NZ nz>
      <CLX1> <CLX2> <CLY1> <CLY2> <CLZ1> <CLZ2>
```

### STFL

Introduces the parameters for the generation of a structure fluid mesh.

**VFCC**

Use VFCCs for the structure fluid mesh. If this optional keyword is not specified, by default the structured mesh will be made of fluid FE.

**CEA**

Use CEA's fluid finite elements instead of JRC's fluid finite elements: that is, use CAR1 and CL2D (in 2D), CUBE and CL3D (in 3D). This optional keyword has only effect if FE are selected, i.e. if the previous optional keyword **VFCC** is *not* specified.

**x0, y0, z0**

Coordinates of the origin of the structured fluid grid. The  $z$ -coordinate **z0** is only needed in 3D calculations.

**lx, ly, lz**

Total lengths of the sides of the structured fluid grid. The  $z$ -length **lz** is only needed in 3D calculations.

**nx, ny, nz**

Number of cells of the structured fluid grid in each direction. The  $z$ -number of cells **nz** is only needed in 3D calculations. Cells have a uniform length in each direction.

**CLX1**

Automatically generate CL2S/CL2D elements (CL3S/CL3D in 3D) along the face of the fluid domain of equation  $x = x_0$ .

**CLX2**

Automatically generate CL2S/CL2D elements (CL3S/CL3D in 3D) along the face of the fluid domain of equation  $x = x_0 + l_x$ .

**CLY1**

Automatically generate CL2S/CL2D elements (CL3S/CL3D in 3D) along the face of the fluid domain of equation  $y = y_0$ .

**CLY2**

Automatically generate CL2S/CL2D elements (CL3S/CL3D in 3D) along the face of the fluid domain of equation  $y = y_0 + l_y$ .

**CLZ1**

Useful only in 3D. Automatically generate CL3S/CL3D elements along the face of the fluid domain of equation  $z = z_0$ .

**CLZ2**

Useful only in 3D. Automatically generate CL3S/CL3D elements along the face of the fluid domain of equation  $z = z_0 + l_z$ .

**Comments:**

Each cell (element) of the grid is a rectangle (rectangular parallelepiped in 3D) with sides of length  $l_x/n_x$ ,  $l_y/n_y$  (and  $l_z/n_z$  in 3D).

Nodes and elements in the grid are numbered progressively starting from the chosen origin  $(x_0, y_0, z_0)$ , first along the global  $X$ -direction, then along the  $Y$ -direction (in 3D, finally along the  $Z$ -direction).

Once the additional elements and nodes have been generated by the **STFL** directive, they are considered like any other elements and nodes, in particular as concerns the rest of the input file and the post-processing.

Appropriate materials must be assigned, in the usual way, to all the automatically generated elements. For example, a low-pressure gas to all fluid elements except those in a bubble zone, representing an explosion, in which a high-pressure gas is assigned. In order to identify the concerned elements, use may be made e.g. of directives for the definition of element groups, see page C.61. A special command to choose the **STFL** elements is provided, see **STFL FLUI** or **STFL CLXS** on page C.61.

In the frequent case of absorbing boundaries of the fluid domain, the concerned **CL2S/CL2D** or **CL3S/CL3D** elements must be identified in order to assign an adequate impedance material to them. The rule for automatic numbering of the generated elements is as follows: first, all fluid elements are generated (their number may be computed as specified above). Next, any specified **CL2S/CL2D** or **CL3S/CL3D** elements are generated, in the following order: **CLX1**, **CLX2**, **CLY1**, **CLY2**, **CLZ1**, **CLZ2**.

Appropriate boundary conditions may also be specified (e.g. via **LINK**) at the boundary nodes (e.g. to block a certain face of the fluid domain).

The **STFL** directive requires no dimensioning since the code is able to determine the number of necessary nodes and elements automatically.

The directive is also compatible with fluid mesh adaptivity (**ADAP**). For example, the user may activate FSI-driven fluid mesh adaptivity via the **FLSR** or **FLSW** directives by specifying as fluid domain a domain generated by **STFL**. See pages D.143 and D.555, respectively, for more information.

## 6.19 AUTOMATIC GENERATION OF SPECTRAL MICRO MESH

### Object:

This directive allows to automatically generate a Spectral Element (SE) “micro” mesh starting from an SE “macro” mesh and a given degree ( $N$ ) of the interpolation polynomial. The degree of the polynomial is the same for all spectral elements, and along each of the spatial directions.

The “macro” spectral element mesh is composed of either MS24 4-node quadrilateral elements (in 2D) or of MS38 8-node hexahedral elements (in 3D), and must have been specified in the previous GEOM directive. The generated micro SE mesh will be composed of S24 4-node quadrilaterals in 2D or of S38 8-node hexahedra in 3D.

### Syntax:

SPEC GMIC NSPE *nspe*

### GMIC

Introduces the automatic generation of micro SE elements according to the parameters given in the following.

### *nspe*

Degree  $N$  of the interpolation polynomial for the SE mesh.

### Comments:

Each macro SE generates exactly  $N^2$  micro SE in 2D or  $N^3$  micro SE in 3D.

The number of micro SE nodes generated is roughly (by excess)  $(N+1)^2$  in 2D or  $(N+1)^3$  in 3D, for each macro SE. The exact number of generated nodes is difficult to determine *a priori* because it depends upon the connectivity of the macro SE mesh (coincident nodes of adjacent micro SE and coincident nodes of micro and macro SE are eliminated). After the calculation of the exact number of nodes (and elements) required, the code prints out this information in case the user wants to keep the memory to a minimum (by giving minimum dimensioning commands).

The generated micro SE are available in an automatically created element group named `.S24` if the calculation is 2D, or `.S38` if the calculation is 3D.

Note that, like for other directives which change the mesh topology (by adding new elements and new nodes), the dimensioning related to geometrical data cannot be fully automatic. The user *must* in this case dimension the total number of nodes, the total number of degrees of freedom and the total number of micro SE generated elements (S24 in 2D or S38 in 3D), like in the following example:

. . .  
DIME NPOI 9 NDDL 18 S24 4 TERM  
. . .  
GEOM . . .  
COMP SPEC GMIC NSPE 2  
. . .

## 6.20 ELEMENT-SPECIFIC EROSION

### Object:

This directive allows to define an erosion criterion for a specific subset only of the elements. A global definition of the erosion criterion is given in the definition of the problem (see directive `EROS <ldam>` on [GBA\\_0030](#)). The global value given there can be overridden for one or more subsets of the elements by using the present directive.

### Syntax:

```
EROS  $[ eros ; NOER ]$ /LECT/
```

#### `eros`

Erosion criterion for the elements given by `/LECT/`. If no erosion limit is needed for a set of elements the keyword `NOER` can be taken. Negative erosion limit indicates also no erosions for the elements.

### Remarks:

The set of keywords `EROS ... /LECT/` may be repeated as many times as needed to define all the desired element-based erosion criteria.

## 6.21 MESH ORIENTATION

### Object:

To orient or re-orient those elements of the mesh for which a specific orientation is important. Typically, these are 3D shell elements without a topological thickness. Normally, proper orientation should be done in the mesh generator, but the present directive may be useful to correct any problems in case one uses a mesh whose generator is not available.

This sub-directive should be used only in emergency cases, e.g. when the mesh used in a calculation (especially flat 3D shell elements) has the wrong orientation and comes from a mesh generator that is not available. This command has the last word on the orientation of the elements, since it comes after the automatic re-orientation which is done in the SENS routine (called from the geometry reading routine). The user is therefore fully responsible of the use of this command.

### Syntax:

```
"ORIE"    < "OBJE" /LEC1/ $[ "POIN" x y z ; "NODE" /LECN/ ]$ >
          < "INVE" /LEC2/ >
```

#### OBJE

The elements in object /LEC1/ have to be oriented so that their outwards normal direction points towards a certain point or node in space, to be specified next. By “pointing” we intend here simply that the scalar product of the unit normal with the line joining the element’s center to the given point or node should be positive.

#### POIN

Introduces the coordinates of the point.

x y z

Coordinates of the point. Note that three coordinates should always be given even in 2D cases (but the ORIE directive is only useful in 3D cases anyway).

#### NODE

Introduces the index of the node.

/LECN/

One node index or the name of an object with just one node (e.g. a Cast3m point name if the mesh has been produced by Cast3m).

#### INVE

The orientation of elements in object /LEC2/ has to be inverted without any checking.

**Comments:**

Only some element types admit re-orienting: typically, these are 3-node or 4-node “thin” elements in 3D, such as shell, membrane or CLxx elements.

Note that the **ORIE** sub-directive may be repeated any number of times, if needed. For example, this may be useful to re-orient a randomly oriented closed surface so that it points outwards. Use a first **ORIE** sub-directive to orient the all the surface elements consistently towards an internal point (e.g. its barycenter). Then, use a second **ORIE** sub-directive to invert the orientation:

```
COMP ... ORIE OBJE LECT toto TERM POIN x y z
        ORIE INVE LECT toto TERM
```



## 6.22 AUTOMATICALLY GENERATED SPH PARTICLES

### Object:

To generate automatically SPH particles within user-defined volumes.

### Syntax:

```
"GBIL"  ngen * (RBIL r <RESE rese>
(INSI | BOX <X0 x0> <Y0 y0> <Z0 z0> DX dx DY dy <DZ dz> |
      | SPHE <XC xc> <YC yc> <ZC zc> R r |
      | CYLI <X1 x1> <Y1 y1> <Z1 z1> <X2 x2> <Y2 y2> <Z2 z2> R r |
      | CONE <X1 x1> <Y1 y1> <Z1 z1> <X2 x2> <Y2 y2> <Z2 z2> R1 r1 R2 r2 |
      | MESH /LECT/ |)
(OUTS | BOX <X0 x0> <Y0 y0> <Z0 z0> DX dx DY dy <DZ dz> |
      | SPHE <XC xc> <YC yc> <ZC zc> R r |
      | CYLI <X1 x1> <Y1 y1> <Z1 z1> <X2 x2> <Y2 y2> <Z2 z2> R r |
      | CONE <X1 x1> <Y1 y1> <Z1 z1> <X2 x2> <Y2 y2> <Z2 z2> R1 r1 R2 r2 |
      | MESH /LECT/ |))
```

#### ngen

Total number of groups of SPH particles that will be generated.

#### r

Radius for the particles of this group.

#### rese

Type of spheres packing: 0 means compact hexagonal (default), 1 means compact cubic (to be implemented), 2 means trivial (non-compact) cubic (normally to be used only for tests and debugging).

#### INSI

Introduces an “inside” condition: all particles “within” a certain geometrical shape are to be generated. This keyword can be repeated as many times as necessary to specify multiple conditions, which are applied in sequence. As a result, all the particles “inside” the union of the specified geometrical shapes will be generated.

#### OUTS

Introduces an “outside” condition: from all particles in the set generated by the previously specified **INSI** condition(s), only those “external” to a certain geometrical shape are to be retained. This keyword can be repeated as many times as necessary to specify multiple conditions, which are applied in sequence. As a result, only the particles “outside” the union of the specified geometrical shapes will be retained.

#### BOX

Introduces the definition of a “box”, (a quadrilateral in 2D or a parallelepiped in 3D) with the sides aligned with the global axes.

$x_0, y_0, z_0$

Coordinates of the ‘origin’ of the box.

$dx, dy, dz$

Lengths of the box sides.

SPHE

Introduces the definition of a sphere (in 3D, or a circle in 2D).

$x_c, y_c, z_c$

Coordinates of the centre of the sphere (or of the circle).

$r$

Radius of the sphere or of the circle.

CYLI

Introduces the definition of a cylinder (3D only). The cylinder is defined by the two extremities of its axis (P1, P2) and its radius.

$x_1, y_1, z_1$

Coordinates of the first extremity P1 of the cylinder axis.

$x_2, y_2, z_2$

Coordinates of the second extremity P2 of the cylinder axis.

$r$

Radius of the cylinder.

CONE

Introduces the definition of a (truncated) cone (3D only). The cone is defined by the two extremities of its axis (P1, P2) and its radii.

$x_1, y_1, z_1$

Coordinates of the first extremity P1 of the cone axis.

$x_2, y_2, z_2$

Coordinates of the second extremity P2 of the cone axis.

$r_1$

Radius of the cone at the first extremity.

$r_2$

Radius of the cone at the second extremity.

**MESH /LECT/**

Introduces the definition of an arbitrary volume, represented by a mesh whose elements are specified in the following **/LECT/**. This mesh must have been defined in the **GEOM** directive and may be composed of elements of Cast3m shape CUB8, PRI6, TET4 and PYR5. Since these elements are probably useless for the calculation (they only serve to define the volume), they should be assigned the **FANT** material.

**Comments:**

If any of the above coordinates (**x0**, **y0** etc.) is omitted, it is assumed to be 0.

**Example:**

Suppose that we want to generate SPH particles within a cylinder representing a pipe full of fluid. Then the syntax would be simply:

```
GBIL 1 RBIL 0.001
      INSI CYLI X0 0 Y0 0 Z0 0 X1 0 Y1 0 Z1 10 R 0.1
```

The group of particles is from now on accessible under the name **\_gbil001**.

Suppose then that the pipe of the previous example is submerged in the sea. To generate also the particles in a prismatic sea region around the pipe, the syntax would be:

```
GBIL 2 RBIL 0.001
      INSI CYLI X0 0 Y0 0 Z0 0 X1 0 Y1 0 Z1 10 R 0.1
      RBIL 0.001
      INSI BOX  X0 -1 Y0 -1 Z0 0 DX 2 DY 2 DZ 10
      OUTS CYLI X0 0 Y0 0 Z0 0 X1 0 Y1 0 Z1 10 R 0.1
```

The two group of particles are from now on accessible under the names **\_gbil001** and **\_gbil002**, respectively.

The presence of the (mandatory) **RBIL** keyword starts a new group of particles. Each group must contain at least one **INSI** condition. All **INSI** conditions must be specified before the **OUTS** conditions (if any).

## 6.23 WATER TABLES

### Object :

These directives create tables containing the physical properties of water, according to one of the following textbooks:

- 1) Directive "TEAU" : Properties of water and steam in SI - units

(E.Schmidt Springer Verlag, Berlin 1979)

or

- 2) Directive "TH2O" (letter O) : NBS/NRC Steam Tables 1984

(Extended tables)

### Syntax :

```
$[ "TEAU" ; "TH2O" ]$    "TMIN"  tmin  "TMAX"  tmax
                        "PMIN"  pmin  "PMAX"  pmax
                        "UNIL"  cl    "UNIM"  cm
                        "DBTE"  nbte  "DSAT"  nsat    "DHTE"  nhte
                        < "DPHY"  nhhy >
```

For the tests:

```
< "DESS" >  < "PERF" >  < "TEST" ( "CAS" ... ) "FINT" >
```

With :

```
"CAS"  num    "P1" p1 $ "T1" t1 ; "X1" x1 $
          $ "P2" p2 $ "T2" t2 ; "X2" x2 $ $
          $ "DVS" dvs    "DH" dh          $
```

**tmin**

Minimum temperature in the tables (this must be lower than the saturation temperature corresponding to **pmin**).

**tmax**

Maximum temperature in the tables (this must be higher than the saturation temperature corresponding to **pmax**, in the case of a sub-critical domain, or to 374 degrees Celsius in the case of a hyper-critical domain).

**pmin**

Minimum pressure in the tables.

**pmax**

Maximum pressure in the tables.

**c1**

Conversion factor of the length units adopted, towards metres.

**cm**

Conversion factor of the mass units adopted, towards kilograms.

**nbte**

Number of intervals into which the low-temperature domain. is subdivided.

**nhte**

Number of intervals into which the high-temperature domain. is subdivided.

**nsat**

Number of intervals into which the pressure in the saturation curve is subdivided.

**nhy**

Number of intervals into which the pressure in the hypercritical domain is subdivided.

For the tests:

**"DESS"**

Allows to draw a cross-section of the tables in the plane (temperature, pressure).

**"PERF"**

Allows to output on a file (logical unit 7) the tables of water properties.

**"TEST"**

This keyword provides a trace of the work performed by the algorithm that searches the thermodynamic parameters of the one-phase or two-phase water, by giving an initial and a final state.

**"FINT"**

End of the sequence opened by keyword **TEST**.

**"CAS" nbr**

Number of the treated case. **nbr** is a simple identification number.

**p1,t1 or p1,x1**

Initial state of the water. If this state is two-phase, it is sufficient to specify **p1** and **x1**. Attention: temperatures are expressed in degrees Celsius, pressures in bar and concentrations are per unit mass.

**p2,t2 or p2,x2**

Final state of the water. If this state is two-phase, it is sufficient to specify **p2** and **x2**. Attention: temperatures are expressed in degrees Celsius, pressures in bar and concentrations are per unit mass.

dvs, dh

Instead of **p2** and **t2** (or **p2** and **x2**), it is possible to specify the variation of specific volume (in m<sup>3</sup>/kg) and the variation of specific enthalpy (**dh**) in J/kg.

### Warning :

If one intends just to run a test without a real EUROPLEXUS transient calculation, it is preferable to put the keyword "FIN" immediately after the directive "COMPLEMENT". This test is recommended, because it allows to verify the initial conditions (pressure, temperature, void fraction) and to check the composition of the tables.

### Comments :

A portion of the saturation curve must be included in the Pressure-Temperature domain chosen.

The **nbte** temperature intervals start from the minimum water temperature to the saturation temperature (for the minimum pressure). It is the same for the **nhte** intervals between the saturation temperature and the maximum temperature, for the maximum pressure.

The **nsat** pressure intervals lie between minimum pressure and maximum pressure, if this is in the sub-critic domain ; else, they go from the minimum pressure to the critical pressure.

In the case of a maximum pressure above 221 bars (hypercritical domain), a further parameter is needed : the number **nhy** of intervals between the critical pressure and the maximum pressure.

For low temperatures ("DBTE"), the subdivision is linear in the temperature. Along the saturation curve ("DSAT"), the subdivision is initially linear in the temperature, then linear in the pressure, in order to obtain a regular subdivision along this curve. Beyond the critical point ("DHTE" and "DPHY"), the subdivision is logarithmic in temperature and in pressure.

When the "TEAU" directive is used, the pressure must be between 0.0062 bar and 1000 bar, and the temperature between 0 and 800 degrees Celsius.

If the extended tables are used (directive "TH2O"), the pressure must be between 0.0062 bar and 30000 bar, and the temperature between 0 and 2000 degrees Celsius.

If the user enters his data in the SI unit system, it is **c1=cm=1**. Otherwise, **c1** or **cm** represent the value of the SI unit expressed in the user's unit. For example, if the lengths are in mm, then **c1=1000**.

## 6.24 HELIUM TABLES

### Object :

These directives create tables containing the physical properties of helium, according to CEA/IRF/DPC (1985).

### Syntax :

```
"THEL"      "UNIL"  c1      "UNIM"  cm
```

For the tests:

```
< "TEST" <"DETA" > ( "CAS" ... ) "FINT" >
```

With :

```
"CAS"  num      "P1" p1 $ "T1" t1 ; "X1" x1 $
          $      "P2" p2 $ "T2" t2 ; "X2" x2 $ $
          $      "DVS" dvs      "DH" dh      $
```

c1

Conversion factor of the length units adopted, towards metres.

cm

Conversion factor of the mass units adopted, towards kilograms.

For the tests:

"TEST"

This keyword provides a trace of the work performed by the algorithm that searches the thermodynamic parameters of the one-phase or two-phase helium, by giving an initial and a final state.

"FINT"

End of the sequence opened by keyword TEST.

"CAS" nbr

Number of the treated case. **nbr** is a simple identification number.

p1,t1 or p1,x1

Initial state of helium. If this state is two-phase, it is sufficient to specify **p1** and **x1**. Attention: temperatures are expressed in degrees Kelvin, pressures in bar and concentrations are per unit mass.

p2,t2 or p2,x2

Final state of helium. If this state is two-phase, it is sufficient to specify **p2** and **x2**. Attention: temperatures are expressed in degrees Kelvin, pressures in bar and concentrations are per unit mass.

**dvs, dh**

Instead of **p2** and **t2** (or **p2** and **x2**), it is possible to specify the variation of specific volume (in  $\text{m}^3/\text{kg}$ ) and the variation of specific enthalpy (**dh**) in J/kg.

### **Warning :**

If one intends just to run a test without a real EUROPLEXUS transient calculation, it is preferable to put the keyword "FIN" immediately after the directive "COMPLEMENT". This test is recommended, because it allows to verify the initial conditions (pressure, temperature, void fraction).

### **Comments :**

The pressure must be above 0.042 bar, and the temperature above 2.1 Kelvin.

The critical point of helium is at 2.27 bars and 5.19 Kelvin. Beyond these values there is only one phase.

Only the gas and the liquid are considered. In the case of high pressures and very low temperatures, the latter must be higher than the melting temperature: solid  $\rightarrow$  liquid.

Unlike the water tables (page C.74), the thermodynamic parameters of helium are directly calculated starting from the interpolation polynomials. Therefore, calculations may at times become very time-consuming.

If the user enters his data in the SI unit system, it is **c1=cm=1**. Otherwise, **c1** or **cm** represent the value of the SI unit expressed in the user's unit. For example, if the lengths are in mm, then **c1=1000**.



## 6.25 PIPE JUNCTIONS

### Object :

Join together different branches of a pipeline.

### Syntax:

```
"RACC" |[ "BIFU" ;
          "CAVI" ;
          "BREC" ]|

          $[          n1 ... nk          ;
          "LECT" racname "TERM" "LECT" P1 ....PK "TERM" ]$

          .... "DSOR" d1...dk < "VOLU" volu "DX" dx "AUTO">
```

`n1 ... nk`

Numbers of the nodes connected by the junction (the order is irrelevant).

`racname`

LECTURE of the name of the junction element in GIBI.

`P1....PK`

LECTURE of the names of the connected points as given in GIBI.

`d1 ... dk`

Internal diameters of the pipelines joined together (the order must correspond to that of the node numbers or names given above).

`volu`

Volume of the junction: mandatory for a cavity, useless for a bifurcation or a pipeline rupture.

`dx`

Mean mesh size of elements surrounding a bifurcation. Volume is then computed as  $V = 1/k \sum_i^k d_i^2 \Delta x$

`AUTO`

The volume of the bifurcation is computed automatically from the mean volume of neighbor elements. AUTO is on by default, if no volume or dx is specified. If AUTO is set, VOLU and DX are not considered.

### Comments:

One may distinguish two cases:

"BIFU" or "BREC" : bifurcation with a small volume (acoustic continuity)

"CAVI" : cavity with a large volume (with a law describing its behaviour)

In the case of a bifurcation or a pipeline rupture, EUROPLEXUS re-computes a fictitious volume, which corresponds to the sphere having the same area as the sum of the areas of the branches that arrive at the bifurcation. When the mesh size is much smaller than the pipe diameters this fictitious volume can set the bifurcation as some kind of tank and cause unwanted reflection waves. In order to avoid that, it is recommended to put the "AUTO" keyword.

The exact number of junction elements must be specified in the "GEOM" directive (see page B.30), and the order of junctions is the same as in the "GEOM" directive if GIBI is not used.

Example:

```
"GEOM"      . . .  "CAVI"  2  "BIFU"  1  "BREC"  1  "TERM"
```

corresponds to:

```
"CAVI"  ....  )   2 cavities
"CAVI"  ....  )
"BIFU"  ....  1 bifurcation
"BREC"  ....  1 pipeline rupture
```

If GIBI is used, the number of junction elements specified may be larger than the exact number, and the order is not compulsory, because the name of each junction is specified in the directive.

Example:

```
"GEOM"      ...  "CAVI" cav_one "CAVI" cav_two "BIFU" my_bif "BREC" my_bre ... "TERM"
```

corresponds to (in the RACC directive) :

```
"CAVI"  "LECT" cav_one "TERM" "LECT" P1          "TERM" ....
"CAVI"  "LECT" cav_two "TERM" "LECT" P2 P3        "TERM" ....
"BIFU"  "LECT" my_bif  "TERM" "LECT" P4 P5 P6      "TERM" ....
"BREC"  "LECT" my_bre  "TERM" "LECT" P7 P8        "TERM" ....
```

## 6.26 TUBM (3D-1D JUNCTION)

### Object:

To connect, by means of a "TUBM" element, a pipeline meshed in 1-D with a fluid meshed in 3-D.

### Syntax:

```
"RACC" ( "TUBM" /LECTURE/  "NTUB" /LECTURE/  "DTUB" dtub  ...  
        "FACE" /LECTURE/  "COEF" coef  )
```

"TUBM" /LECTURE/

The /LECTURE/ procedure allows to specify the name of the GIBI object associated with the junction element. In case of a mesh in the MED format, the name of the junction element must follow a specific rule, see comments below.

"NTUB" /LECTURE/

The /LECTURE/ procedure allows to specify the name of the GIBI object or MED group associated with the 1D node of the tube.

dtub

Internal diameter of the connected tube.

"FACE" /LECTURE/

The /LECTURE/ procedure allows to specify the name of the GIBI object or MED group associated with nodes of the 3D face. The object **must** be composed of surface elements (typically CL3D), which are used to identify the 3D part of the junction. These CL3D elements should *not* be assigned any material, since EPX automatically assigns them a FANT material.

coef

This coefficient allows to take into account of possible symmetries in the 3D mesh. The area of the face meshed in 3D is multiplied by **coef** in order to find out the same area as that of a non-symmetrised face.

### Comments :

These elements are created by CASTEM, by means of the following syntax:

```
mon_tubm = MANU SUPERELEMENT (p_tube ET s_face) ;
```

where **p\_tube** is the object corresponding to the 1D point, and **s\_face** the object corresponding to the nodes of the 3D face. All nodes of the 3D face must be coplanar.

In case of a mesh in the MED format, in which the SUPERELEMENT structure does not exist, the required procedure is the following:

- In the mesh, MED groups corresponding to the 1D point and the faces of the 3D face must be based on the same name, e.g. "raccord", to which the suffixes "\_n" and "\_s" must respectively be added.
- The name of the junction element to be used is then "splm\_raccord". It is automatically created when a TUBM element is declared.

"TUBM" connects the fluid of the continuum elements (3D) with the fluid of a "TUBE" element (continuity of the mass flow rate). The velocities of nodes belonging to the 3D face are all equal and normal to the face itself.

The type of elements whose face(s) participate in forming the 3D face is irrelevant: therefore it is possible to use cubes, prisms or even tetrahedrons for the mesh.

A material must be associated to the "TUBM" element, although this has no behaviour law.

### **Warning:**

It is mandatory to specify in the dimensioning the parameter "JONC", in order to reserve the space indispensable for the relations associated to the junction (see page A.80).

Do not forget to mention "TUBM" also in the "LIAISON" directive (page D.200).

## 6.27 TUYM (3D-1D JUNCTION)

### Object:

To connect, by means of a "TUYM" element, a pipeline meshed ("TUYA" element) in 1-D with a fluid meshed in 3-D for moving meshes (A.L.E computation).

### Syntax:

```
"RACC" ( "TUYM" /LECTURE/  "NTUB" /LECTURE/  "DTUB" dtub  ...
        "FACE" /LECTURE/  "COEF" coef  )
```

"TUYM" /LECTURE/

The /LECTURE/ procedure allows to specify the name of the GIBI object associated with the junction element. In case of a mesh in the MED format, the name of the junction element must follow a specific rule, see comments below.

"NTUB" /LECTURE/

The /LECTURE/ procedure allows to specify the name of the GIBI object or MED group associated with the 1D node of the tube ("TUYA" element).

dtub

Internal diameter of the connected tube ("TUYA" element).

"FACE" /LECTURE/

The /LECTURE/ procedure allows to specify the name of the GIBI object or MED group associated with nodes of the 3D face. The object **must** be composed of surface elements (typically CL3D), which are used to identify the 3D part of the junction. These CL3D elements should *not* be assigned any material, since EPX automatically assigns them a FANT material.

coef

This coefficient allows to take into account of possible symmetries in the 3D mesh. The area of the face meshed in 3D is multiplied by **coef** in order to find out the same area as that of a non-symmetrised face.

### Comments :

These elements are created by CASTEM, by means of the following syntax:

```
mon_tuym = MANU SUPERELEMENT (p_tuya ET s_face) ;
```

where `p_tuya` is the object corresponding to the 1D point, and `s_face` the object corresponding to the nodes of the 3D face. All nodes of the 3D face must be coplanar.

In case of a mesh in the MED format, in which the `SUPERELEMENT` structure does not exist, the required procedure is the following :

- In the mesh, MED groups corresponding to the 1D point and the faces of the 3D face must be based on the same name, e.g. `"raccord"`, to which the suffixes `"_n"` and `"_s"` must respectively be added.
- The name of the junction element to be used is then `"splm_raccord"`. It is automatically created when a TUYM element is declared.

"TUYM" connects the fluid of the continuum elements (3D) with the fluid of a "TUYA" element (continuity of the mass flow rate). The velocities of nodes belonging to the 3D face are all equal and normal to the face itself.

The type of elements whose face(s) participate in forming the 3D face is irrelevant: therefore it is possible to use cubes, prisms or even tetrahedrons for the mesh.

A material must be associated to the "TUYM" element, although this has no behaviour law.

### Warning:

It is mandatory to specify in the dimensioning the parameter `"JONC"`, in order to reserve the space indispensable for the relations associated to the junction (see page A.80).

Do not forget to mention "TUYM" also in the "LIAISON" directive (page D.200).

## 6.28 CORRESPONDENCE BETWEEN NODES

### Object:

The purpose of this directive is to define a one-to-one correspondence between couples of nodes. This user-defined correspondence may be useful in various situations, in which the code needs to find a one-to-one correspondence between nodes in the mesh and the automatic determination of such a correspondence is impossible. For example, this might happen under exceptional circumstances in the following cases:

- In the search for structural (Lagrangian) nodes correspondent to fluid nodes in the **FSA** fluid-structure interaction directive, see page D.450.
- In the search for structural nodes corresponding to fluid nodes in the model for perforated plates **IMPE PPLT**, see page C.760.
- In the search for structural nodes corresponding to fluid nodes in the model for rupture disks **IMPE RUDI**, see page C.770.
- In the search for structural nodes corresponding to fluid nodes in the model for rupture disks for the MC formulation, **IMPE RDMC**, see page C.790.

In such cases, the code tries to automatically determine the structural (or other Lagrangian) node “corresponding” to a certain fluid node. This node is defined as the Lagrangian node having the same initial coordinates as the fluid node under consideration, within a certain small tolerance (that may be changed via the **OPTI TOLC**, page H.40). If there is no such node or if more than one candidate node is found (e.g. because there are several superposed structures in the mesh), then the automatic search would fail. In this case, the user may assume control by explicitly specifying the corresponding Lagrangian node to each “ambiguous” fluid node.

It is advised to use this directive only in case of necessity. First, an input without this directive should be prepared. Then, in case the code produces some error messages related to the impossibility of automatically determining the node correspondence, the present directive may be added to resolve the identified conflicts.

### Syntax:

```
"CNOD"   "NODF"   /LECT1/   "NODS"   /LECT2/
/LECT1/
```

List of first nodes of each node couple. Typically, these are fluid nodes.

```
/LECT2/
```

List of second nodes of each node couple. These nodes must be Lagrangian. Typically, these are structure nodes, but Lagrangian fluid nodes are also accepted.

### Comments:

The order in which nodes are listed in `/LECT1` or `/LECT2` is retained. To the  $i$ -th node of `/LECT1` corresponds the  $i$ -th node of `/LECT2`. The number of nodes in `/LECT1` and `/LECT2` must be the same.

Note that the directive `CNOD` may be specified only once in each calculation (i.e. it should not be repeated). In other words, all correspondent nodes should be specified in just one `/LECT1/` and `/LECT2/`.

In case of problems with the `FSA` directive, please note that another way of resolving node conflicts, alternative to the present `CNOD` directive, is the `STRU` sub-directive of `FSA`, see page D.450, which is more practical in case there is a large number of conflicting nodes.



## 6.29 SPH SHELL ELEMENT (SPHC)

### Object :

This instruction introduces characteristics for the SPH shell elements (SPHC) which allow discretizing shell structures with a single layer of particles.

### Syntax:

```
"CSPH"  "RAYO" rbille  "EPAI" ep  
        "ORX" orx  "ORY" ory  "ORZ" orz  
        < "LINE" cl >  < "QUAD" cq >  
        < "RLIM" rlim > < "RESEAU" ires >  
        < "VOIS" nvoi >  
        ( "STRP" istrp /LECT/ )
```

#### rbille

Radius of the SPH shell particles.

#### ep

Thickness of the shell particles.

#### orx, ory, orz

x,y,z co-ordinates of a point used to orient normals of the SPH shell particles.

#### cl

Linear damping coefficient.

#### cq

Quadratic damping coefficient.

#### rlim

Multiplicative coefficient for the search radius.

#### ires

Type of particles lattice (1: cubic, 0: hexagonal).

#### nvoi

Number of neighbouring particles sought.

#### istrp

Type of stress points (1: free, 2: clamped) read in the following /LECTURE/ sequence (see comment below).

**Comments :**

For the quadratic damping, it is advised to take  $cq=4$ .

To damp out the high-frequency oscillations it is advisable to use a value of  $cl$  between 0.1 and 0.5.

At least one set of stress points must be entered. Several sets can be entered by repeating the STRP keyword.

Two types of particle lattice are possible: for  $ires = 1$  a cubic lattice is adopted; in the case  $ires = 0$  (default value), a compact hexagonal lattice is adopted.

The number of sought neighbouring particles is by default 12. This number may not be changed for the PEF algorithm. Its modification is accepted only for the SPH method.

For a given particle, the search considers the neighbours whose center is within a distance of  $r_{lim} \cdot r_{bille}$  from its center. By default,  $r_{lim}=1.3$ .

### 6.30 DISCRETE ELEMENT MODEL (ELDI)

#### Object :

This instruction is mandatory in the input file when using discrete elements (ELDI). It allows printing out to the output listing the value of the radius of each discrete element and to impose the correct masses of different parts of the discrete element model (element density will be corrected).

This directive is used to define a bridging (recovering) zone allowing to couple a set of discrete elements (ELDI) with a 3D finite element model (meshed with the CUB8 elements only) or a shell model (Q4GS elements only).

#### Syntax:

```
"CELDI"  < "IMPR" >
          < "MASS" nval
              nval*(val /LECTURE/ ) >
          < "ARMA" /LECTURE/ >
          < "LTM"  nbse nbse*(beta plas /LECTURE/) >

          "TYPL" nbtypl*([ [ "COHE" <"IMPR"> <"COEF" val> /LECTURE/ ;
                              "BIMA" <"IMPR">
                              "MAT1" <"COEF" val> /LECTURE/
                              "MAT2" <"COEF" val> /LECTURE/ ;
                              "CONT" <"IMPR"> <"COEF" val> /LECTURE/ ] | )

          < "CSTE"  coef >
          < "EDEF" nbcoup
              nbcoup*("NCOU" ncouches
                      "ELDI" /LECTURE/
                      "FRON" /LECTURE/ ) >
          < "CBOX" xmin xmax ymin ymax zmin zmax >
```

#### "IMPR"

This optional keyword allows printing out in the output listing the value of the radius of each discrete element.

#### "MASSE"

This optional keyword enables the user to impose the masses of discrete elements lists.

#### nval

Number of imposed masses.

#### val

Value of the imposed mass.

"ARMA"

Indicates the presence of steel reinforcement modeled with aligned steel discrete elements. Caculates the main direction of the reinforcement used for steel-concrete interaction.

LECTURE

List of the discrete elements concerned.

"LTM"

Indicates the presence of bending properties (rotation stiffnesses for discrete elements).

nbse

Number of sequences with different bending properties.

beta

Coefficient used to calculate the bending stiffness:  $K_r = \beta EI/R$ .

plas

Coefficient used to calculate the plastic torque:  $M_p = \text{plas} \cdot \sigma \cdot I/R$ . In elastic calculations, one should use  $\text{plas} = 0$  (no test on  $M_p$ ) or put  $\text{plas} \gg 1$  to guarantee  $M_p$  is very high.

"TYPL"

This keyword defines different types of links (interactions) between discrete elements (ELDI) within one or several sets of particles. Links may be of two kinds : cohesive links and contacts. The interaction forces between the discrete elements are then computed with respect to the types of material used.

nbtypl

Number of sequences beginning from one of the following words : "COHE" or "BIMA" or "CONT".

"COHE"

This keyword initializes the search of cohesive interactions within a set of discrete elements.

<"COEF" val>

Interaction range. The default value of the interaction range *val* is 1.

"BIMA"

This keyword initializes the search of cohesive interactions between two sets of discrete elements (permanent contact of two materials).

"MAT1" <"COEF" val>

This keyword allows to define the first set of discrete elements and its interaction range *val*. The default value is 1.

"MAT2" <"COEF" val>

This keyword allows to define the second set of discrete elements and its interaction range *val*. The default value is 1.

"CONT"

This keyword initializes the search of contact interactions inside one or several sets of discrete elements specified in /LECTURE/. In this case, the value of the interaction range is 1.

<"IMPR">

This optional keyword allows to print out in the output listing the result of the interactions search.

"CSTE"

This optional keyword enables the user to define the security coefficient of the time step.

coef

Security coefficient (by default 0.1) :  $dt = coef * dt_{crit}$

nbcoup

Number of combined finite/discrete zones to connect.

"NCOU"

Number of finite element range defining the combined finite/discrete element zone.

"ELDI" /LECTURE/

List of the discrete elements concerned to research in the combined finite/discrete element zone.

"FRON" /LECTURE/

List of nodes forming the border of the finite elements mesh in the bridging finite/discrete element zone.

"CBOX"

Allows defining a box restraining search for DE contacts. Six reals must be given: xmin, xmax, ymin, ymax, zmin, zmax.

### Comments :

To guarantee the masses of different parts of the discrete element model are correct, each discrete element should belong to one group only.

To identify the interacting neighbors, a grid subdivision method is used.

An interaction between elements  $a$  and  $b$  of radius  $R^a$  and  $R^b$  respectively, is defined within an interaction range *val* and does not necessarily imply that two elements are in contact (for cohesive interactions). Then, these elements will interact if,

$$val * (R^a + R^b) > or = D^{a,b}$$

where  $D^{a,b}$  is the distance between the centroids of element  $a$  and  $b$  and where  $val$  is the interaction range.  $val$  is mandatory and must be  $> or = 1$ .

### 6.31 MULTILAYER ELEMENT CMC3

#### Object:

The characteristics of CMC3 elements are described when they have not been defined by CASTEM2000.

#### Syntax:

```
"CORTHO"  "EPAISSEUR"  ep      "EXCENTREMENT"  ex
           $[  "ANGLE"    angle      ;
              "VECTEUR"   vx  vy  vz  ]$          /LECTURE/
```

#### ep

Thickness of the element.

#### ex

Element eccentricity with respect to the plane defined by the 3 nodes of the mesh.

#### angle

Angle (in degrees) formed by the first side of the element and the first axis of the orthotropic system.

#### vx, vy, vz

The 3 components in the global frame of the vector that defines the first orthotropy direction.

#### LECTURE

List of the elements concerned.

#### Comments:

The sign of the excentricity is defined by the orientation of the normal. This depends on the numbering of the nodes of the CMC3 element (see Maxwell's cork-screw rule).

The first side of the element is the one formed by the first 2 nodes.

## 6.32 ORTHOTROPY

### Object:

Description of the orthotropy directions for continuum elements in 2D and 3D.

### Syntax:

```
"MORTHO"  $[ "ALPHA" angle1                      ;
              "TETHA" angle2                      ;
              "AXE1"  e11 e12 e13  "AXE2" e21 e22 e23      ;
              "COCY"  "POINT" $[ /LECTURE1/  ;
                                xx yy zz    ]$ "VECT" v1 v2 v3 ;
              "V1LC"  v1x v2x v3x  "V2LC" v2x v2y v2z      ]$
              /LECTURE/
```

#### angle1

Angle (in degrees) formed by the  $Ox$  axis (in 2D plane cases or 3D) or the  $Or$  axis (in axisymmetric) and the first axis of the orthotropy reference frame.

#### angle2

Angle (in degrees) formed by the first side of the element and the first orthotropy axis (in 2D or axisymmetric). The first side of the element is the segment connecting the first two nodes declared for the element in the GEOM directive.

#### e11, e12, e13

First vector defining the orthotropy plane of the material.

#### e21, e22, e23

Second vector defining the orthotropy plane of the material.

#### COCY

This directive allows to define a “cylindrical” type of orthotropy, that may be used for example by the BOIS (wood) material. The first axis of orthotropy is parallel to the vector defined by the **VECT** directive described above. The second axis of orthotropy (perpendicular to the first one) lies on the plane formed by a straight line passing through the **POINT** defined below and parallel to **VECT**, and the barycenter of the element.

#### POINT



This directive allows to define a point which is either a node of the mesh (option /LECTURE1/), or a geometric point defined by its three coordinates (**xx yy zz**).

**v1 v2 v3**

First vector defining the orthotropy reference frame of the **COCY** directive.

**v1x, v1y, v1z**

First vector defining the orthotropy plane of the material in the local repere of the element.

**v2x, v2y, v2z**

Second vector defining the orthotropy plane of the material in the local repere of the element.

**LECTURE**

List of the elements concerned.

### Comments:

One can define several orthotropy directions by repeating each time the keyword **MORT**. It is also possible to repeat it starting from different items.

The **ALPHA** or **TETHA** keywords are used in 2D, the **AXE1 ... AXE2** or **COCY** directive are used in 3D.

The vectors **V1(e11,e12,e13 or v1x,v1y,v1z)** and **V2(e21,e22,e23 or v1x,v1y,v1z)** are not necessarily unit vectors, and **V2** is not necessarily normal to **V1**.

Starting from these input data, **EUROPLEXUS** computes and stores the values in the local reference frames relatives to each element. These local values will be utilised during the transient calculation. For this reason, the calculation remains valid also for large rotations.

### 6.33 ORTHOTROPY FOR 3D SHELLS

#### Object:

Description of the orthotropy characteristics for 3D (layered) shell elements using JRC's "sandwiches" and "layers" model (see **SAND** and **LAYE** keywords).

The directive defines the orthotropy characteristics related to the following **material types**: HILL (3), ORTS (46), ORPE (55), COMM (88), GLRC (92), and to the following **element types**: QPPS (35), COQI (40), T3GS (51), DST3 (83), DKT3 (84), CQD4 (91), CQD9 (92), CQD3 (93), CQD6 (94), Q4GR (111), Q4GS (112), Q4MC (138), T3MC (139).

The orthotropy characteristics are stored in the ECR() table of each GP of each concerned element. If the element has layers (**LAYE** directive) then the characteristics may vary from layer to layer. Here is how the characteristics are stored, for each material type:

```
HILL (3)  ! ECR(8) = ANGLE
ORTS (46) ! ECR(3) = ANGLE
ORPE (55) ! ECR(21) = ANGLE
COMM (88) ! ECR(6:8) = VX, VY, VZ
GLRC (92) ! ECR(11:13) = VX, VY, VZ
```

Note that the GLRC material may *not* be associated with a layer because this is a global material model.

For historical reasons, two alternative *input syntaxes* can be used:

1. ORTS vx vy vz /LECT/ <LAYE /LECT\_LAY/>: This syntax specifies directly vx, vy, vz i.e. a vector whose projection on the lamina (local) reference of the (shell) element is the first orthotropy direction of the material which is sufficient to define the orthotropy for shell elements. This syntax applies directly to elements (or element layers) using the COMM or GLRC material, since such materials require the values of VX, VY, VZ. For elements (or element layers) using the HILL, ORTS or ORPE material the vx, vy, vz are converted internally to an angle, (by projecting the vector onto the shell's lamina plane), which is then stored in the ECR() table.
2. ORTS \$ ANGL angl ; AXE1 a1x a1y a1z ; CIRC cx cy cz \$ /LECT/ <LAYE /LECT\_LAY/>: This syntax specifies either an angle angl, a vector a1x, a1y, a1z or a (central) point cx, cy, cz.
  - **ANGL**: The angl parameter is the angle between the first axis of the shell's lamina (local) reference frame and the first orthotropy direction of the material (also lying on the lamina plane). The angle is then stored in the ECR() table, if the material is HILL, ORTS or ORPE. For the other two materials (COMM or GLRC) an error message is currently raised because these materials require a vector (vx, vy, vz) and not an angle. (This might be corrected in a future release by computing the vector internally from the given angle.)
  - **AXE1**: The vector a1x, a1y, a1z has the same meaning as the vx, vy, vz in the first syntax. Therefore, if the material is COMM or GLRC, the values a1x, a1y, a1z are stored directly in the ECR() table. For the other materials (HILL, ORTS or ORPE), the vector defined by ax1, ax2 and ax3 is converted internally to an angle (by projecting the vector onto the shell's lamina plane), which is then stored in the ECR() table.
  - **CIRC**: cx, cy, cz define a center point. For each element, the first orthotropic direction is the projection of the vector [center point/element gravity center]. It allows to treat circular geometry.

**Syntax:**

```
"ORTS" |[ vx vy vz                ;
          $[ "ANGL" alpha ; "AXE1" a1x a1y a1z ]$
          ]|
          /LECT/    < "LAYE" /LECT_LAY/>
```

**vx vy vz**

Components, in the global reference frame, of a vector whose projection on the lamina (local) coordinate system of the 3D shell element indicates the orthotropy direction (one such direction is sufficient, for shell elements).

**ANGL alpha**

Angle between the first direction of the shell element and the first direction of the orthotropic frame (in radians). It can only be used with Q4GS, Q4GR, Q4MC, DKT3, DST3 or T3MC elements associated either with HILL or with ORTS material.

**AXE1 a1x a1y a1z**

Components, in the global reference frame, of a vector whose projection on the lamina (local) coordinate system of the 3D shell element indicates the orthotropy direction (one such direction is sufficient, for shell elements). It can only be used with Q4GS, Q4GR, Q4MC, DKT3, DST3 or T3MC elements associated either with HILL or with ORTS material

**/LECT/**

Concerned elements.

**/LECT\_LAY/**

Concerned layers. Layers are identified by their indexes, as described in the SAND directive on page C.45.

**Comments:**

Note that the directive **COMP ORTS** must be specified *after* the definition of the material characteristics (**MATE** directive). If other quantities (e.g. thickness, etc.) are to be specified via the **COMP** directive, then *two* **COMP** directives should be used: the first one, immediately after the **GEOM** directive, and the second one (**COMP ORTS**) immediately after the **MATE** directive.

## 6.34 PARTICLE ELEMENT (BILLE)

### Object :

Description of the characteristics of the BILLE element (particle element).

### Syntax:

```
"CBILLE"  "RAYON"  rbille  < "LINEAIRE"  cl  >  ...  
... < "QUADRATIQUE"  cq  > < "RESEAU"  ires  >  ...  
... < "VOISIN"      nvoi  >  
... < "RLIM"        rlim  >
```

**rbille**

Radius of the particles.

**cl**

Linear damping coefficient.

**cq**

Quadratic damping coefficient.

**ires**

Type of particles lattice.

**nvoi**

Number of neighbouring particles sought.

**rlim**

Multiplicative coefficient for the search radius.

### Comments :

For the quadratic damping, it is advised to take  $cq=4$ .

To damp out the high-frequency oscillations it is advisable to use a value of  $cl$  between 0.1 and 0.5.

Two types of particle lattice are possible: for  $ires = 1$  a cubic lattice is adopted; in the case  $ires = 0$  (default value), a compact hexagonal lattice is adopted.

The number of sought neighbouring particles is by default 12. This number may not be changed for the PEF algorithm. Its modification is accepted only for the SPH method.

For a given particle, the search considers the neighbours whose center is within a distance of  $\text{rlim} \times \text{Diameter}$  from its center. By default,  $\text{rlim}=1.3$ .

## 6.35 RIGID BODIES (JRC Implementation)

### Object :

To define one or more rigid (non-deformable) bodies. The geometrical characteristics of each rigid body are specified here. The material characteristics (basically, the density) are specified by assigning to each rigid body a **RIGI** (rigid) material, see Page C.295.

### Syntax:

```
RIGI nrigi * ( <MASS mass <MTOT mtot> >
               <BARY bary <GX gx GY gy <GZ gz>> >
               <INNER iner <JXX jxx JYY jyy JZZ jzz
                   JYZ jyz JXZ jxz JXY jxy> >
               /LECT/ )
```

#### nrigi

Total number of rigid bodies. Each rigid body is geometrically defined by a set of elements.

#### MASS

Optional specification of the method that should be used to compute the total mass of the body. The value **0** (default) means using the **nodal masses** as computed by EPX using standard procedures. The value **1** means using the **element masses** as computed by EPX using standard procedures. The value **2** means decomposing each element into **simplexes** (triangles or tetrahedrons) and then using analytical formulas over each simplex in order to assemble the total mass of the whole body. The value **3** means that the **user** specifies the total mass by the following **MTOT** keyword. In this latter case the value of the **RIGI** material density is not used, but it must be specified anyway for input completeness, see Page C.295.

#### BARY

Optional specification of the method that should be used to compute the center of gravity (barycenter) of the body. The value **0** (default) means using the **nodal masses**, located at the rigid nodes composing the body. The value **1** means using the **element masses**, located at the centroids of the rigid elements composing the body. An element's centroid is computed as the (non-weighted) arithmetic average of the element's nodal positions. In general, this is only an approximation of the real center of gravity (barycenter) of the element. The value **2** means decomposing each element into **simplexes** (triangles or tetrahedrons) and then using analytical formulas over each simplex in order to compute the barycenter of the whole body. The value **3** means that the **user** specifies the coordinates of the barycenter, by the following **GX**, **GY** and **GZ** (3D only) keywords. In this latter case the value of the **RIGI** material density is not used, but it must be specified anyway for input completeness, see Page C.295.

#### INNER

Optional specification of the method that should be used to compute the tensor of inertia of the body with respect to three mutually perpendicular axes parallel to the global axes and passing through the barycenter of the body. The value **0** (default) means using the **nodal masses**, located at the rigid nodes composing the body. The value **1** means using the **element masses**, located at the centroids of the rigid elements composing the body. An element's centroid is computed as the (non-weighted) arithmetic average of the element's nodal positions. In general, this is only an approximation of the real center of gravity (barycenter) of the element. The value **2** means decomposing each element into **simplexes** (triangles or tetrahedrons) and then using analytical formulas over each simplex in order to assemble the inertia tensor of the whole body. The value **3** means that the **user** specifies the tensor of inertia by the following **JXX**, **JXY**, **JZZ**, **JYZ**, **JXZ** and **JYX** keywords. In this latter case the value of the **RIGI** material density is not used, but it must be specified anyway for input completeness, see Page C.295.

/LECT/

List of the elements defining the rigid body.

#### Comments :

The elements belonging to a rigid body *must* be assigned a rigid (**RIGI**) material, which is used to define the density of the rigid body and thus to compute the mass, the barycenter and the inertia moments of the body (unless they are specified by the user).

*Only* elements belonging to a rigid body **RIGI** can be assigned a rigid material **RIGI**.

A named elements group **\_RIGI<nnn>** is automatically created for each rigid body. The **<nnn>** is the rigid body index (001, 002, ..., **nrigi**) in the order of definition of the rigid bodies. Each group contains only one element and only one node: the “lumped” element and the “lumped” node that represent the rigid body *as a whole*. These names can be used to apply external loads, boundary conditions, etc., to a rigid body *as a whole*.

## 7 GROUP C1—MATERIALS

### Object:

This instruction enables the user to enter the properties of various materials.

### Syntax:

```
"MATE"    (  < "LOI" numldc > . . . )
```

#### LOI

This keyword announces that a number will be assigned to the constitutive law whose definition follows.

#### numldc

Number of the constitutive law.

### Comments:

The word "MATE" is compulsory and may only be used once, at the beginning of the data sequence relative to the instruction MATERIALS.

The numbers introduced by the "LOI" directive may be in arbitrary order, and some numbers may be missing. This is very useful in the case of multiple materials: one can add or move material data in the input file without changing the number of the corresponding material law (see "MULT", page C.380).

If the "LOI" directive is absent, the number automatically attributed to the law by EUROPLEXUS is the index of the material in the order its constitutive law is listed in the input data.

Do not forget the corresponding dimensioning ([GBA\\_0070](#)).



## 7.1 LIST OF MATERIALS

The material models are (in alphabetical order):

number	name	ref	law of behaviour
74	ABSE		
53	ADCJ	7.8.25	hypothetical core disruptive accident with law of type JWL for the bubble
34	ADCR	7.8.19	containment accident (fast neutrons)
47	ADFM	7.8.32	advection-diffusion fluid
71	APPU	7.5	Material for elements of type PPUI
32	ASSE		motor asservissement (meca)
11	BETO	7.7.13	concrete (NAHAS model)
57	BILL	7.8.26	LIBRE (user's free particle material), or FLUIDE (isothermal fluid particle: $c = cte$ )
29	BL3S	7.7.62	reinforced concrete for discrete elements
20	BLMT	7.7.14	DYNAR LMT Concrete
75	BOIS	7.7.28	wood for shock adsorbing (only compression)
121	BPEL	7.7.14	model for prestressing cable-concrete friction
114	BREC	7.8.12	data for pipeline break
59	BUBB	7.8.38	Balloon model for air blast simulations
89	CAMC	7.7.45	Modified Cam-clay material
8	CAVI		isothermal fluid with cavitation
68	CDEM	7.8.39	Discrete Equation Method for Combustion
64	CHAN	7.7.23	Multi-layer with the CHANG-CHANG criterion
51	CHOC	7.8.22	Shock waves, Rankine-Hugoniot equation
21	CLVF	7.9.34	Boundary conditions for finite volumes
90	CLAY	7.7.46	Modified Cam-clay material (backward fully implicit algorithm, viscoplastic regularization)
88	COMM	7.7.44	Composite material (linear orthotropic), Ispra implementation
113	CREB		
58	CRIT	7.7.19	damage criteria calculation : PY (damage of type P/Y), DUCTile (ductile damage)
100	CRTM	7.7.58	Composite manufactured by RTM process
109	DADC	7.7.17	Dynamic Anisotropic Damage Concrete
110	DEMS	7.8.40	Discrete Equation Method for Two Phase Stiffened Gases
38	DONE	7.7.41	viscoplastic material
111	DPDC	7.7.18	dynamic plastic damage concrete
87	DPSF	7.7.43	Drucker Prager with softening and viscoplastic regularization
83	DRPR	7.7.52	Drucker Prager Ispra model
12	DRUC	7.7.6	Drucker-Prager
19	DYNA	7.7.7	dynamic Von Mises isotropic rate-dependent
22	EAU	7.8.9	two-phase water (liquid + vapour)
115	ENGR	7.7.21	elastic gradient damage material
105	EOBT	7.7.20	anisotropic damage of concrete
49	EXVL	7.8.20	hydrogen explosion Van Leer
17	FANT	7.7.31	phantom: ignore the associated elements
27	FLFA	7.8.15	rigid tube bundles
86	FLMP	7.8.35	Fluid multi-phase
7	FLUI	7.8.2	isothermal fluid ( $c = cte$ )

36	FLUT	7.8.30	fluid, to be specified by the user
93	FOAM	7.7.53	Aluminium foam (for crash simulations)
80	FUNE	7.7.47	specialized cable material (no compression resistance)
9	GAZP	7.8.4	perfect gas
118	GGAS	7.8.1	generic ideal gas material
44	GLAS	7.7.61	glass with strain-rate effect
116	GLIN	7.7.3	generic linear material
92	GLRC	7.7.54	Plasticity with kinematic softening for orthotropic shells. Global plastic criterion.
52	GPDI	7.8.23	diffusive perfect gas Van Leer
117	GPLA	7.7.4	generic plastic material
48	GVDW	7.8.28	Van Der Waals gas
40	GZPV	7.8.24	perfect gas for Van Leer
28	HELI	7.8.10	helium
3	HILL	7.7.60	Isotropic plasticity associated with a HILL criterion and with a orthotropic elastic behaviour
95	HYPE	7.7.55	hyperelastic material (Model of Mooney-Rivlin, Hart-Smith and Ogden)
16	IMPE	7.9	impedance
43	IMPV	7.9.21	impedance Van Leer
4	ISOT	7.7.7	isotropic Von Mises
108	JCLM	7.7.66	Johnson-Cook with Damage Lemaitre-Chaboche for SPHC
91	JPRP	7.12	for bushing elements
50	JWL	7.8.21	explosion (Jones-Wilkins-Lee model)
66	JWLS	7.8.29	Explosion (Jones-Wilkins-Lee for solids)
72	LEM1	7.7.10	Von Mises isotropic coupled with damage (type Lemaitre)
13	LIBR	7.7.32	free (material defined by the user)
1	LINE	7.7.1	linear elasticity
23	LIQU	7.8.14	incompressible (or quasi-) fluid
70	LMC2	7.7.12	Von Mises isotropic coupled with damage (Lemaitre) with strain-rate sensitivity
63	LSGL	7.7.63	laminated security glass material
26	MASS	7.7.30	mass of a material point
85	MAZA	7.7.16	Mazars-linear elastic law with damage
82	MCFF	7.8.34	multicomponent fluid material (far-field)
81	MCGP	7.8.33	multicomponent fluid material (perfect gas)
60	MCOU	7.7.22	Linear multi-layer homogenised through the thickness
45	MECA	7.10	mechanism associated to articulated systems
33	MHOM	7.8.16	homogenization
97	MINT	7.7.56	Material for interface element
31	MOTE		motor force or couple (meca)
25	MULT	7.8.13	multiple materials (coupled monodim.)
10	NAH2	7.8.7	sodium-water reaction
18	ODMS	7.7.27	nonlinear damage with orthotropy (ODM)
134	OPFM	??	Onera Progressive Failure Model (Composite materials)
139	ORFM	??	Onera Rate dependent Failure Model (Composite materials)
132	ORSR	7.7.29	Rate dependent linear elastic orthotropic with local refer- ence frame
42	ORTE	7.7.26	linear damage with orthotropy
41	ORTH	7.7.24	linear orthotropic in user system
46	ORTS	7.7.25	linear elastic orthotropic with local reference frame

2	PARF	7.7.7	perfectly plastic Von Mises
56	PARO	7.8.11	friction and heat exchange for pipeline walls
96	PBED		particle bed
6	POST		post-rupture (beton)
69	PRGL	7.8.27	Porous jelly for the particles
39	PUFF	7.8.17	equation of state of type "PUFF"
123	RESG	7.7.4	Material for RL3D spring element in the global reference frame
61	RESL	7.7.1	Material for RL3D spring element in the local reference frame
125	RIGI	7.7.68	Rigid material (for rigid bodies)
54	RSEA	7.9.13	reaction sodium-water with three constituents
103	SG2P		
112	SGBN		
104	SGMP		
107	SLIN	7.7.65	Linear Damage for SPHC
99	SLZA	7.7.57	Steinberg-Lund-Zerilli-Armstrong
106	SMAZ	7.7.64	Mazars Damage for SPHC
24	SOUR	7.8.6	imposed pressure in a continuum element
30	STGN	7.7.8	Steinberg - Guinan
102	STIF		
94	SUPP	7.6	support
101	TAIT		
5	TETA	7.7.7	Von Mises dependent upon temperature
98	TVMC	7.7.59	elastoplastic short fibres with damage
37	VM1D	7.7.40	material for elements of type "ED1D"
35	VM23	7.7.39	Von Mises elasto-plastic radial return
2/4/5/19	VMIS	7.7.7	Von Mises materials
124	VMGR	7.7.9	Von Mises orthotropic reinforcement grid model
76	VMJC	7.7.48	Johnson-Cook
78	VMLP	7.7.49	Ludwig-Prandtl
79	VMLU	7.7.50	Ludwik
84	VMSF	7.7.42	Von Mises with softening and viscoplastic regularization
77	VMZA	7.7.51	Zerilli-Armstrong
120	VPJC	7.7.67	visco-plastic Johnson-Cook
67	ZALM	7.7.11	Zerilli-Armstrong with damage Lemaitre-Chaboche

The "FANT" material may be allocated to any element, with the effect of 'eliminating' it from the mesh, as far as mechanical resistance is concerned.

The different elements may use the following materials (defined by their numbers):

## 7.2 AVAILABLE MATERIALS FOR ELEMENT TYPES

=====

NO. | ELEMENT | AVAILABLE MATERIALS

----|-----|-----

1		COQU		LINE	PARF	ISOT	TETA	DYNA	ORTH										
2		TRIA		LINE	PARF	ISOT	TETA	POST	FLUI	CAVI	GAZP	NAH2	BETO						
				DRUC	DYNA	EAU	LIQU	SOUR	MULT	FLFA	STGN	ADCR	VM23						
				PUFF	ORTH	JWL	CHOC	ADCJ	RSEA	CRIT	BUBB	JWLS	ZALM						
				LMC2	LEM1	BOIS	VMJC	VMZA	VMLP	VMLU	DRPR	VMSF	DPSF						
				CAMC	CLAY	SGMP	ENGR	VPJC											
3		BARR		LINE	PARF	ISOT	DYNA												
4		PONC		LINE	PARF	ISOT													
5		MEMB		LINE															
6		CUBB		LINE	PARF	ISOT													
7		CL2D		IMPE	CLVF	IMPV													
8		CAR1		LINE	PARF	ISOT	TETA	POST	FLUI	CAVI	GAZP	NAH2	BETO						
				DRUC	DYNA	EAU	LIQU	SOUR	MULT	FLFA	STGN	ADCR	VM23						
				DONE	PUFF	ORTH	JWL	CHOC	ADCJ	RSEA	CRIT	BUBB	JWLS						
				ZALM	LMC2	LEM1	BOIS	VMJC	VMZA	VMLP	VMLU	DRPR	VMSF						
				DPSF	CAMC	CLAY	SGMP	VPJC											
9		CAR4		LINE	PARF	ISOT	TETA	POST	FLUI	CAVI	GAZP	BETO	DRUC						
				DYNA	EAU	MULT	FLFA	STGN	VM23	DONE	PUFF	ORTH	GLAS						
				CHOC	ADCJ	CRIT	BUBB	LSGL	ZALM	LMC2	LEM1	BOIS	VMJC						
				VMZA	VMLP	VMLU	DRPR	VMSF	DPSF	CAMC	CLAY	HYPE	ENGR						
				VPJC															
10		COQC		LINE	PARF	ISOT	ORTH												
11		CUBE		LINE	ISOT	TETA	FLUI	GAZP	NAH2	DRUC	DESM	DYNA	BLMT						
				EAU	LIQU	SOUR	MULT	FLFA	STGN	ADCR	VM23	PUFF	ORTH						
				ORTE	GLAS	ORTS	JWL	CHOC	ADCJ	RSEA	ORPE	CRIT	BUBB						
				LSGL	JWLS	ZALM	LMC2	LEM1	BOIS	VMJC	VMZA	VMLP	VMLU						
				DRPR	VMSF	MAZA	DPSF	CAMC	CLAY	FOAM	HYPE	PBED	TVMC						
				SLZA	CRTM	SGMP	EOBT	DADC	DPDC	BDBM	VPJC	ORTP	OPFM						
				ORFM	ORSR														
12		COQ3		LINE	ISOT	TETA	DYNA	MCOU	CHAN										
13		CUB6		LINE	ISOT	TETA	FLUI	GAZP	DRUC	DYNA	BLMT	MULT	STGN						
				VM23	PUFF	ORTH	ORTE	GLAS	ORTS	CHOC	ORPE	CRIT	LSGL						
				BOIS	VMJC	DRPR	VMSF	DPSF	CAMC	CLAY	FOAM	HYPE	SLZA						
				CRTM	VPJC	ORSR													
14		COQ4		LINE	ISOT	TETA	DYNA	MCOU	CHAN										
15		FS2D																	
16		FS3D																	
17		POUT		LINE	ISOT	DYNA													
18		CL3D		IMPE	CLVF														
19		BR3D		LINE	PARF	ISOT	DYNA												
20		PR6		LINE	HILL	ISOT	TETA	FLUI	GAZP	BETO	DRUC	DESM	DYNA						
				BLMT	MULT	VM23	PUFF	ORTH	ORTE	ORTS	ORPE	CRIT	ZALM						
				LMC2	LEM1	BOIS	VMJC	DRPR	VMSF	MAZA	DPSF	CAMC	CLAY						
				FOAM	HYPE	SLZA	CRTM	EOBT	DADC	VPJC	DPDC								
21		TETR		LINE	HILL	ISOT	TETA	FLUI	GAZP	NAH2	DRUC	DESM	DYNA						
				BLMT	EAU	LIQU	SOUR	MULT	FLFA	ADCR	VM23	PUFF	ORTH						
				ORTE	ORTS	JWL	CHOC	ADCJ	RSEA	ORPE	CRIT	BUBB	JWLS						
				ZALM	BOIS	VMJC	DRPR	VMSF	MAZA	DPSF	CAMC	CLAY	FOAM						
				HYPE	SLZA	CRTM	SGMP	EOBT	DADC	ENGR	VPJC	DPDC							
22		TUBE		FLUI	GAZP	NAH2	EAU	LIQU	SOUR	MULT	HELI	ADCR	GVDW						
				JWL	RSEA	PARO	JWLS												

23		TUYA		LINE ISOT FLUI GAZP NAH2 EAU LIQU SOUR MULT HELI
				ADCR GVDW RSEA PARO
24		CL1D		IMPE CLVF
25		BIFU		FLUI GAZP NAH2 EAU LIQU SOUR ADCR GVDW RSEA
26		CAVI		FLUI GAZP NAH2 EAU LIQU SOUR MULT ADCR GVDW RSEA
				PARO
27		PRIS		LINE HILL ISOT TETA FLUI GAZP NAH2 DRUC DESM DYNA
				BLMT EAU SOUR MULT FLFA ADCR VM23 PUFF ORTH ORTE
				ORTS JWL CHOC ADCJ RSEA ORPE CRIT BUBB JWLS ZALM
				LMC2 LEM1 BOIS VMJC DRPR VMSF DPSF CAMC CLAY FOAM
				HYPE CRTM SGMP EOBT VPJC DPDC
28		PMAT		LINE MASS
29		CL3T		IMPE CLVF
30		CUB8		LINE HILL ISOT TETA FLUI GAZP DRUC DESM ODMS DYNA
				BLMT MULT STGN VM23 PUFF ORTH ORTE GLAS ORTS CHOC
				ORPE CRIT LSGL ZALM LMC2 LEM1 BOIS VMJC VMZA VMLP
				VMLU DRPR VMSF MAZA DPSF CAMC CLAY FOAM HYPE SLZA
				CRTM EOBT DADC DPDC ENGR BDBM VPJC ORTP OPFM ORFM
				ORSR
31		CLTU		IMPE CLVF
32		APPU		APPU SUPP
33		MECA		MOTE ASSE MECA ABSE
34		QAX1		FLUI GAZP BUBB
35		QPPS		LINE ISOT DYNA VM23 DONE GLAS MCOU LSGL CHAN LEM1
				VMJC VMZA VMLP VMLU VMSF DPSF GLRC SLZA VPJC
36		FHQ2		MHOM
37		FHT2		MHOM
38		Q92		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU DRPR VMSF
				DPSF CAMC CLAY VPJC
39		Q93		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU DRPR VMSF
				DPSF CAMC CLAY VPJC
40		COQI		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU VMSF DPSF
				COMM VPJC
41		TUBM		FLUI GAZP NAH2 EAU LIQU ADCR GVDW RSEA GAZD TAIT
				STIF SG2P SGMP SGBN
42		CL23		IMPE
43		ED01		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU VMSF DPSF
				VPJC
44		ED1D		VM1D
45		TVL1		GZPV GVDW EXVL GPDI
46		CVL1		GZPV GVDW EXVL GPDI
47		CMC3		LINE ISOT BETO ORTH
48		FS3T		
49		Q92A		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU DRPR VMSF
				DPSF CAMC CLAY VPJC
50		CL3L		
51		T3GS		LINE ISOT TETA DYNA VM23 DONE GLAS LSGL LEM1 VMJC
				VMZA VMLP VMLU VMSF DPSF GLRC VPJC
52		FLU1		FLUT BUBB
53		FLU3		FLUT BUBB
54		PFEM		FLUI

55		FL2S		FLUT BUBB FLMP
56		ED41		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU VMSF DPSF
				VPJC
57		ADC8		ADFM
58		ADQ4		ADFM
59		FL3S		FLUT BUBB FLMP
60		CL2S		IMPE FLUT
61		CL3S		IMPE FLUT
62		CL32		IMPE
63		CL33		IMPE
64		FL23		FLUT BUBB FLMP
65		FL24		FLUT BUBB FLMP
66		FL34		FLUT BUBB FLMP
67		FL35		FLUT BUBB FLMP
68		FL36		FLUT BUBB FLMP
69		FL38		FLUT BUBB FLMP
70		CL22		IMPE FLUT IMPV MCFF
71		Q41		VM23 DONE LSGL VMSF DPSF
72		Q42		VM23 DONE LSGL VMSF DPSF
73		Q41N		VM23 DONE LSGL VMSF DPSF
74		Q42N		VM23 DONE LSGL VMSF DPSF
75		Q41L		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU DRPR VMSF
				DPSF CAMC CLAY VPJC
76		Q42L		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU DRPR VMSF
				DPSF CAMC CLAY VPJC
77		Q95		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU DRPR VMSF
				DPSF CAMC CLAY VPJC
78		CL3I		IMPE FLUT MCFF
79		BILL		LINE ISOT PUFF BILL PRGL MAZA
80		ELDI		BL3S
81		CUVL		GZPV GVDW EXVL GPDI
82		PRVL		GZPV GVDW EXVL GPDI
83		DST3		LINE HILL ISOT TETA DYNA VM23 DONE GLAS ORTS MCOU
				LSGL CHAN LEM1 VMJC VMZA VMLP VMLU VMSF DPSF HYPE
				SLZA VPJC
84		DKT3		LINE ISOT DYNA VM23 DONE GLAS MCOU LSGL CHAN LEM1
				VMJC VMZA VMLP VMLU VMSF DPSF GLRC SLZA VPJC
85		SHB8		LINE ISOT DYNA ZALM LMC2 LEM1 SLZA
86		XCUB		LINE PARF ISOT ODMS ORTE VMJC VMSF
87		XCAR		LINE PARF ISOT VMJC VMSF
88		PROT		LINE ISOT
89		SPHC		LINE ISOT LEM1 SMAZ SLIN JCLM
90		Q4G4		LINE ISOT
91		CQD4		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU VMSF DPSF
				COMM VPJC
92		CQD9		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU VMSF DPSF
				COMM VPJC
93		CQD3		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU VMSF DPSF
				COMM VPJC
94		CQD6		VM23 DONE GLAS LSGL VMJC VMZA VMLP VMLU VMSF DPSF
				COMM VPJC

95		CLD3		IMPE									
96		CLD6		IMPE									
97		MC23		MCGP									
98		MC24		MCGP									
99		CL3Q		IMPE	FLUT	MCFF							
100		Q42G		VM23	DONE								
101		MC34		MCGP									
102		MC35		MCGP									
103		MC36		MCGP									
104		MC38		MCGP									
105		MS24		LINE									
106		S24		LINE									
107		MS38		LINE									
108		S38		LINE									
109		FUN2		VM23	DONE	VMJC	VMZA	VMLP	VMLU	FUNE	VPJC		
110		FUN3		VM23	DONE	VMJC	VMZA	VMLP	VMLU	FUNE	VPJC		
111		Q4GR		LINE	HILL	ISOT	TETA	DYNA	VM23	DONE	GLAS	MCOU	LSGL
				CHAN	LEM1	VMJC	VMZA	VMLP	VMLU	VMSF	DPSF	GLRC	SLZA
				VPJC	VMGR	DPDC							
112		Q4GS		LINE	HILL	ISOT	TETA	DYNA	VM23	DONE	GLAS	ORTS	MCOU
				LSGL	CHAN	LEM1	VMJC	VMZA	VMLP	VMLU	VMSF	DPSF	GLRC
				HYPE	SLZA	VPJC	VMGR						
113		RL3D		RESL	RESG								
114		BSHT		JPRP									
115		BSHR		JPRP									
116		TUYM		FLUI	GAZP	NAH2	EAU	LIQU	ADCR	GVDW	RSEA	GAZD	TAIT
				STIF	SG2P	SGMP	SGBN						
117		SH3D		JPRP									
118		MAP2											
119		MAP3											
120		MAP4											
121		MAP5											
122		MAP6											
123		MAP7											
124		INT4		LINE	MINT								
125		INT6		LINE	MINT								
126		INT8		LINE	MINT								
127		SH3V											
128		MOY4											
129		MOY5											
130		ASHB		LINE	ISOT	DYNA	ZALM	LMC2	LEM1	SLZA			
131		T3VF		FLUI	GAZP	EAU	ADCR	GVDW	JWL	ADCJ	BUBB	JWLS	CDEM
				GAZD	TAIT	STIF	SG2P	SGMP	DEMS	SGBN	CREB		
132		Q4VF		FLUI	GAZP	EAU	ADCR	GVDW	JWL	ADCJ	BUBB	JWLS	CDEM
				GAZD	TAIT	STIF	SG2P	SGMP	DEMS	SGBN	CREB		
133		CUVF		FLUI	GAZP	EAU	ADCR	GVDW	JWL	ADCJ	BUBB	JWLS	CDEM
				GAZD	TAIT	STIF	SG2P	SGMP	DEMS	SGBN	CREB		
134		PRVF		FLUI	GAZP	EAU	ADCR	GVDW	JWL	ADCJ	BUBB	JWLS	CDEM
				GAZD	TAIT	STIF	SG2P	SGMP	DEMS	SGBN	CREB		
135		TEVF		FLUI	GAZP	EAU	ADCR	GVDW	JWL	ADCJ	BUBB	JWLS	CDEM
				GAZD	TAIT	STIF	SG2P	SGMP	DEMS	SGBN	CREB		

136		PYVF		FLUI	GAZP	EAU	ADCR	GVDW	JWL	ADCJ	BUBB	JWLS	CDEM
				GAZD	TAIT	STIF	SG2P	SGMP	DEMS	SGBN	CREB		
137		COQ2		LINE	ISOT	TETA							
138		Q4MC		LINE	HILL	ISOT	DYNA	ORTS	ORPE				
139		T3MC		LINE	HILL	ISOT	DYNA	VM23	ORTS	ORPE	LSGL	VPJC	
140		DEBR											
141		INS6											
142		INS8											
143		P3ZT		LINE	HILL	ISOT	DYNA	ORTS	ORPE	PIEZ			
144		C272		VM23	DONE	GLAS	LSGL	VMJC	VMZA	VMLP	VMLU	DRPR	VMSF
				DPSF	CAMC	CLAY	VPJC						
145		C273		VM23	DONE	GLAS	LSGL	VMJC	VMZA	VMLP	VMLU	DRPR	VMSF
				DPSF	CAMC	CLAY	VPJC						
146		BREC		EAU	MULT	BREC							
147		TUVF		FLUI	GAZP	EAU	MULT	ADCR	GVDW	JWL	ADCJ	PARO	JWLS
				GAZD	TAIT	STIF	SG2P	SGMP	SGBN				
148		TYVF		LINE	ISOT	FLUI	GAZP	EAU	MULT	ADCR	GVDW	JWL	ADCJ
				PARO	JWLS	GAZD	TAIT	STIF	SG2P	SGMP	SGBN		
149		BIVF		FLUI	GAZP	EAU	ADCR	GVDW	JWL	ADCJ	JWLS	GAZD	TAIT
				STIF	SG2P	SGMP	SGBN						
150		CAVF		FLUI	GAZP	EAU	MULT	ADCR	GVDW	JWL	ADCJ	JWLS	GAZD
				TAIT	STIF	SG2P	SGMP	SGBN					
151		CL92		IMPE									
152		CL93		IMPE									
153		LIGR		MECA									
154		RNFR		BPEL									
155		C81L		VM23	DONE	GLAS	LSGL	VMJC	VMZA	VMLP	VMLU	DRPR	VMSF
				DPSF	CAMC	CLAY	VPJC						
156		C82L		VM23	DONE	GLAS	LSGL	VMJC	VMZA	VMLP	VMLU	DRPR	VMSF
				DPSF	CAMC	CLAY	VPJC						

## AVAILABLE ELEMENTS FOR EACH MATERIAL

=====

E AFTER MATERIAL INDICATES ERODIBLE.

NO.		MATERIAL		AVAILABLE ELEMENTS
----		-----		-----
1		LINE E		COQU TRIA BARR PONC MEMB CUBB CAR1 CAR4 COQC CUBE
				COQ3 CUB6 COQ4 POUT BR3D PR6 TETR TUYA PRIS PMAT
				CUB8 QPPS CMC3 T3GS BILL DST3 DKT3 SHB8 XCUB XCAR
				PROT SPHC Q4G4 MS24 S24 MS38 S38 Q4GR Q4GS INT4
				INT6 INT8 ASHB COQ2 Q4MC T3MC P3ZT TYVF
2		PARF E		COQU TRIA BARR PONC CUBB CAR1 CAR4 COQC BR3D XCUB
				XCAR
3		HILL		PR6 TETR PRIS CUB8 DST3 Q4GR Q4GS Q4MC T3MC P3ZT
4		ISOT E		COQU TRIA BARR PONC CUBB CAR1 CAR4 COQC CUBE COQ3
				CUB6 COQ4 POUT BR3D PR6 TETR TUYA PRIS CUB8 QPPS
				CMC3 T3GS BILL DST3 DKT3 SHB8 XCUB XCAR PROT SPHC
				Q4G4 Q4GR Q4GS ASHB COQ2 Q4MC T3MC P3ZT TYVF
5		TETA E		COQU TRIA CAR1 CAR4 CUBE COQ3 CUB6 COQ4 PR6 TETR



				PRIS	CUB8	T3GS	DST3	Q4GR	Q4GS	COQ2				
6	POST			TRIA	CAR1	CAR4								
7	FLUI			TRIA	CAR1	CAR4	CUBE	CUB6	PR6	TETR	TUBE	TUYA	BIFU	
				CAVI	PRIS	CUB8	QAX1	TUBM	PFEM	TUYM	T3VF	Q4VF	CUVF	
				PRVF	TEVF	PYVF	TUVF	TYVF	BIVF	CAVF				
8	CAVI			TRIA	CAR1	CAR4								
9	GAZP			TRIA	CAR1	CAR4	CUBE	CUB6	PR6	TETR	TUBE	TUYA	BIFU	
				CAVI	PRIS	CUB8	QAX1	TUBM	TUYM	T3VF	Q4VF	CUVF	PRVF	
				TEVF	PYVF	TUVF	TYVF	BIVF	CAVF					
10	NAH2			TRIA	CAR1	CUBE	TETR	TUBE	TUYA	BIFU	CAVI	PRIS	TUBM	
				TUYM										
11	BETO	E		TRIA	CAR1	CAR4	PR6	CMC3						
12	DRUC	E		TRIA	CAR1	CAR4	CUBE	CUB6	PR6	TETR	PRIS	CUB8		
14	IFS													
15	DESM			CUBE	PR6	TETR	PRIS	CUB8						
16	IMPE			CL2D	CL3D	CL1D	CL3T	CLTU	CL23	CL2S	CL3S	CL32	CL33	
				CL22	CL3I	CLD3	CLD6	CL3Q	CL92	CL93				
18	ODMS			CUB8	XCUB									
19	DYNA	E		COQU	TRIA	BARR	CAR1	CAR4	CUBE	COQ3	CUB6	COQ4	POUT	
				BR3D	PR6	TETR	PRIS	CUB8	QPPS	T3GS	DST3	DKT3	SHB8	
				Q4GR	Q4GS	ASHB	Q4MC	T3MC	P3ZT					
20	BLMT			CUBE	CUB6	PR6	TETR	PRIS	CUB8					
21	CLVF			CL2D	CL3D	CL1D	CL3T	CLTU						
22	EAU			TRIA	CAR1	CAR4	CUBE	TETR	TUBE	TUYA	BIFU	CAVI	PRIS	
				TUBM	TUYM	T3VF	Q4VF	CUVF	PRVF	TEVF	PYVF	BREC	TUVF	
				TYVF	BIVF	CAVF								
23	LIQU			TRIA	CAR1	CUBE	TETR	TUBE	TUYA	BIFU	CAVI	TUBM	TUYM	
24	SOUR			TRIA	CAR1	CUBE	TETR	TUBE	TUYA	BIFU	CAVI	PRIS		
25	MULT			TRIA	CAR1	CAR4	CUBE	CUB6	PR6	TETR	TUBE	TUYA	CAVI	
				PRIS	CUB8	BREC	TUVF	TYVF	CAVF					
26	MASS			PMAT										
27	FLFA			TRIA	CAR1	CAR4	CUBE	TETR	PRIS					
28	HELI			TUBE	TUYA									
29	BL3S			ELDI										
30	STGN			TRIA	CAR1	CAR4	CUBE	CUB6	CUB8					
31	MOTE			MECA										
32	ASSE			MECA										
33	MHOM			FHQ2	FHT2									
34	ADCR			TRIA	CAR1	CUBE	TETR	TUBE	TUYA	BIFU	CAVI	PRIS	TUBM	
				TUYM	T3VF	Q4VF	CUVF	PRVF	TEVF	PYVF	TUVF	TYVF	BIVF	
				CAVF										
35	VM23	E		TRIA	CAR1	CAR4	CUBE	CUB6	PR6	TETR	PRIS	CUB8	QPPS	
				Q92	Q93	COQI	ED01	Q92A	T3GS	ED41	Q41	Q42	Q41N	
				Q42N	Q41L	Q42L	Q95	DST3	DKT3	CQD4	CQD9	CQD3	CQD6	
				Q42G	FUN2	FUN3	Q4GR	Q4GS	T3MC	C272	C273	C81L	C82L	
36	FLUT			FLU1	FLU3	FL2S	FL3S	CL2S	CL3S	FL23	FL24	FL34	FL35	
				FL36	FL38	CL22	CL3I	CL3Q						
37	VM1D			ED1D										
38	DONE			CAR1	CAR4	QPPS	Q92	Q93	COQI	ED01	Q92A	T3GS	ED41	
				Q41	Q42	Q41N	Q42N	Q41L	Q42L	Q95	DST3	DKT3	CQD4	
				CQD9	CQD3	CQD6	Q42G	FUN2	FUN3	Q4GR	Q4GS	C272	C273	

			C81L	C82L									
39	PUFF		TRIA	CAR1	CAR4	CUBE	CUB6	PR6	TETR	PRIS	CUB8	BILL	
40	GZPV		TVL1	CVL1	CUVL	PRVL							
41	ORTH		COQU	TRIA	CAR1	CAR4	COQC	CUBE	CUB6	PR6	TETR	PRIS	
			CUB8	CMC3									
42	ORTE		CUBE	CUB6	PR6	TETR	PRIS	CUB8	XCUB				
43	IMPV		CL2D	CL22									
44	GLAS	E	CAR4	CUBE	CUB6	CUB8	QPPS	Q92	Q93	COQI	ED01	Q92A	
			T3GS	ED41	Q41L	Q42L	Q95	DST3	DKT3	CQD4	CQD9	CQD3	
			CQD6	Q4GR	Q4GS	C272	C273	C81L	C82L				
45	MECA		MECA	LIGR									
46	ORTS		CUBE	CUB6	PR6	TETR	PRIS	CUB8	DST3	Q4GS	Q4MC	T3MC	
			P3ZT										
47	ADFM		ADC8	ADQ4									
48	GVDW		TUBE	TUYA	BIFU	CAVI	TUBM	TVL1	CVL1	CUVL	PRVL	TUYM	
			T3VF	Q4VF	CUVF	PRVF	TEVF	PYVF	TUVF	TYVF	BIVF	CAVF	
49	EXVL		TVL1	CVL1	CUVL	PRVL							
50	JWL		TRIA	CAR1	CUBE	TETR	TUBE	PRIS	T3VF	Q4VF	CUVF	PRVF	
			TEVF	PYVF	TUVF	TYVF	BIVF	CAVF					
51	CHOC		TRIA	CAR1	CAR4	CUBE	CUB6	TETR	PRIS	CUB8			
52	GPDI		TVL1	CVL1	CUVL	PRVL							
53	ADCJ		TRIA	CAR1	CAR4	CUBE	TETR	PRIS	T3VF	Q4VF	CUVF	PRVF	
			TEVF	PYVF	TUVF	TYVF	BIVF	CAVF					
54	RSEA		TRIA	CAR1	CUBE	TETR	TUBE	TUYA	BIFU	CAVI	PRIS	TUBM	
			TUYM										
55	ORPE		CUBE	CUB6	PR6	TETR	PRIS	CUB8	Q4MC	T3MC	P3ZT		
56	PARO		TUBE	TUYA	CAVI	TUVF	TYVF						
57	BILL		BILL										
58	CRIT		TRIA	CAR1	CAR4	CUBE	CUB6	PR6	TETR	PRIS	CUB8		
59	BUBB		TRIA	CAR1	CAR4	CUBE	TETR	PRIS	QAX1	FLU1	FLU3	FL2S	
			FL3S	FL23	FL24	FL34	FL35	FL36	FL38	T3VF	Q4VF	CUVF	
			PRVF	TEVF	PYVF								
60	MCOU		COQ3	COQ4	QPPS	DST3	DKT3	Q4GR	Q4GS				
61	RESL		RL3D										
62	PIEZ		P3ZT										
63	LSGL		CAR4	CUBE	CUB6	CUB8	QPPS	Q92	Q93	COQI	ED01	Q92A	
			T3GS	ED41	Q41	Q42	Q41N	Q42N	Q41L	Q42L	Q95	DST3	
			DKT3	CQD4	CQD9	CQD3	CQD6	Q4GR	Q4GS	T3MC	C272	C273	
			C81L	C82L									
64	CHAN		COQ3	COQ4	QPPS	DST3	DKT3	Q4GR	Q4GS				
65	MORI												
66	JWLS		TRIA	CAR1	CUBE	TETR	TUBE	PRIS	T3VF	Q4VF	CUVF	PRVF	
			TEVF	PYVF	TUVF	TYVF	BIVF	CAVF					
67	ZALM	E	TRIA	CAR1	CAR4	CUBE	PR6	TETR	PRIS	CUB8	SHB8	ASHB	
68	CDEM		T3VF	Q4VF	CUVF	PRVF	TEVF	PYVF					
69	PRGL		BILL										
70	LMC2		TRIA	CAR1	CAR4	CUBE	PR6	PRIS	CUB8	SHB8	ASHB		
71	APPU		APPU										
72	LEM1	E	TRIA	CAR1	CAR4	CUBE	PR6	PRIS	CUB				

[illegible]

```

103 | SG2P | TUBM TUYM T3VF Q4VF CUVF PRVF TEVF PYVF TUVF TYVF
    |      | BIVF CAVF
104 | SGMP | TRIA CAR1 CUBE TETR PRIS TUBM TUYM T3VF Q4VF CUVF
    |      | PRVF TEVF PYVF TUVF TYVF BIVF CAVF
105 | EOBT | CUBE PR6 TETR PRIS CUB8
106 | SMAZ | SPHC
107 | SLIN | SPHC
108 | JCLM | SPHC
109 | DADC | CUBE PR6 TETR CUB8
110 | DEMS | T3VF Q4VF CUVF PRVF TEVF PYVF
111 | DPDC | CUBB CUBE CUB8 PRIS PR6 TETR Q4GR
112 | SGBN | TUBM TUYM T3VF Q4VF CUVF PRVF TEVF PYVF TUVF TYVF
    |      | BIVF CAVF
113 | CREB | T3VF Q4VF CUVF PRVF TEVF PYVF
114 | BREC | BREC
115 | ENGR | TRIA CAR4 TETR CUB8
116 | GLIN E |
117 | GPLA |
118 | GGAS |
119 | BDBM | CUBE CUB8
120 | VPJC E | TRIA CAR1 CAR4 CUBE CUB6 PR6 TETR PRIS CUB8 QPPS
    |      | Q92 Q93 COQI ED01 Q92A T3GS ED41 Q41L Q42L Q95
    |      | DST3 DKT3 CQD4 CQD9 CQD3 CQD6 FUN2 FUN3 Q4GR Q4GS
    |      | T3MC C272 C273 C81L C82L
121 | BPEL | RNFR
122 | ORTP | CUBE CUB8
123 | RESG | RL3D
124 | VMGR | Q4GR Q4GS
132 | ORSR | CUBE CUB6 CUB8
134 | OPFM | CUBE CUB8
139 | ORFM | CUBE CUB8

```

To print out (on the log file!) an up-to-date version of the above element and material tables, just run EUROPLEXUS with any input data file by adding the option `OPTI DPEM` (see also page [GBH\\_0090](#)).

### 7.3 AUXILIARY FILE

**Object:**

This directive allows to read the material data from an auxiliary file.

**Syntax:**

```
"MATE"      < "FICHIER"    'nom.fic'  >
```

In certain cases the data may be bulky. It is then advised to store the data on an auxiliary file in order to shorten the main input data file. The auxiliary file is activated by means of the keyword "FICHIER", followed by the full name (under Unix) of the file. Therefore, only the words "MATE" "FICHIER" 'nom.fic' remain in the main input file.

The auxiliary file (in free format) will contain the whole set of material data, with the exception of the "MATE" keyword. To return to the main input data file, the auxiliary file must be terminated by the keyword "RETOUR".

## 7.4 LOCALISED DAMPING

### Object:

This directive allows adding a localised damping on some d.o.f.s of some particular nodes.

### Syntax:

```
"AMORTISSEMENT" ( /LECDDL/  "BETA" beta  "FREQ" freq  /LECTURE/ )
/LECDDL/
    Concerned degrees of freedom.
beta
    Reduced damping  $\beta$ .
freq
    Frequency  $f$  of the global mode to be damped out.
/LECTURE/
    List of the concerned nodes.
```

### Comments:

The value  $\beta = 1$  corresponds to the critical damping for the frequency  $f$ . All frequencies are damped. The components with a frequency lower than the cut-off frequency:  $f_c = \beta f$  will be damped in a pseudo-periodic manner while those having higher frequencies will be damped in an aperiodic manner.

This damping is proportional to the mass  $M$  and to the particles velocity  $v$ , and may be used in order to damp out preferably the structures without influence on the internal fluid, for example.

One adds an external force  $F_{\text{amort}}$  of the form:

$$F_{\text{amort}} = -2\beta\omega Mv$$

where  $\omega = 2\pi f$ .

It is evident that the work of external forces will be modified by the damping forces.

This directive differs from the global damping directive (OPTI AMOR . . . , see page H.30) mainly by the fact that here the region to which damping is applied may be specified by the user, while in the other case the damping applies to the whole model (but limitedly to some element types, see page H.30).

## 7.5 NON-LINEAR SUPPORTS : "APPUI"

### Object :

This directive allows to model non-linear supports of type spring or damper. It may be used only for the elements of type "APPUI" (material points with 6 d.o.f.s). The user gives the evolution curve of the force applied by the support as a function of its displacement (for the springs) or of its velocity (for the dampers). These supports work in translation or in rotation.

### Syntax :

```
"APPUI" |[ "RESS" ; "AMOR" ] | |[ "TRAN" ; "ROTA" ] |
          "CMPX" cmpx      "CMPY" cmpy      "CMPZ" cmpz
          "COEF" coef      "NUFO" nufo      <"MASS" mass>
          <"INCR" incr>    <"DECX" decx>    <"DECY" decy> /LECTURE/
```

#### "RESS"

The support is of type spring.

#### "AMOR"

The support is of type damper.

#### "TRAN"

The support works in translation.

#### "ROTA"

The support works in rotation.

#### cmpx

Component in X of the translation or rotation axis of the support.

#### cmpy

Component in Y of the translation or rotation axis of the support.

#### cmpz

Component in Z of the translation or rotation axis of the support.

#### coef

Multiplicative coefficient of the function.

#### nufo

Number of the function.

#### mass

Inertia of the support along its working direction.

**incr**

Increment of the velocity or displacement for the calculation of the local stiffness.

**decx**

Offset of the abscissas of the force/displacement or force/velocity curve.

**decy**

Offset of the ordinates of the force/displacement or force/velocity curve.

**/LECTURE/**

List of the concerned nodes.

### Comments :

The user must define a vector corresponding to the rotation axis or translation axis of the support. This vector does not need to be normalised, just its direction matters. This direction defines the local reference frame of the support: it is the projection of the displacement (or of the velocity) of the concerned node onto this axis that allows to determine the reaction force.

An APPUI element may not work simultaneously as a spring AND as a damper, nor in translation AND in rotation. Therefore it will be sometimes necessary to define several APPUI elements, geometrically coincident, in order to correctly define the local stiffness.

The function defining the force generated by the support in response to displacement or velocity of its application point on the supported structure is of the form:

$$F = coef f(D) \quad or \quad F = coef f(V)$$

with  $f(D)$  or  $f(V)$  given by the user. Warning: these values have a sign. Do not forget to give the force with the opposite sign as the displacement (this is a reminder).

For the estimation of the stability step, it is necessary to know the local slope of the behaviour curve. To this end, the user must specify the keyword "INCR". The computation of the local stiffness will then be (by default,  $incr=1.E-4$ ):

$$K = (F(D + incr) - F(D))/incr \quad or \quad C = (F(V + incr) - F(V))/incr$$

In the case that the structure is not in equilibrium for a zero displacement at the beginning of the calculation, the user may impose a translation of vector (decx, decy) of the behaviour curve. The computed force will then be (by default, decx and decy are zero):

$$F = (coef f(D + decx)) - decy$$

( in fact :  $decy = coef f(decx)$  )

### Outputs :

The components of the ECR vector are:



ECR(1): Force (resp. moment) along X.

ECR(2): Force (resp. moment) along Y.

ECR(3): Force (resp. moment) along Z.

ECR(4): Current stiffness.

ECR(5): Current velocity (or angular velocity).

ECR(6): Total displacement (or rotation).

ECR(7): Applied force (or moment) to the node (reaction force).

## 7.6 NON-LINEAR SUPPORTS : "SUPP"

### Object :

This directive allows to model a complex non-linear support, having arbitrary stiffness and damping values along the 6 dofs of the concerned node. It may only be used in conjunction with elements of type "APPUI" (material point with 6 dofs). The user gives the evolution curve of the reaction force generated by the support as a function of the displacement or of the velocity of the associated node.

### Syntax :

```
"SUPP"  "MASS" m
        < |[ "KX" kx ; "KY" ky ; "KZ" kz ]|
          |[
            "NFKT" nufo1 ;
            "NFKX" nufokx "NFKY" nufoky "NFKZ" nufokz
          ]| >
        < |[ "AX" ax ; "AY" ay ; "AZ" az ]| "NFAT" nufo2 >
        <"IRX" irx> <"IRY" iry> <"IRZ" irz>
        < |[ "KRX" krx ; "KRY" kry ; "KRZ" krz ]| "NFKR" nufo3 >
        < |[ "ARX" arx ; "ARY" ary ; "ARZ" arz ]| "NFAR" nufo4 >
        /LECTURE/
```

m

Additional translational mass (optional).

kx, ky, kz

Translational stiffnesses along the global axes.

nufo1

Index of the function associated with translational stiffnesses.

nufokx, nufoky, nufokz

Indexes of the functions associated with 3 translational stiffnesses.

ax, ay, az

Translational dampings along the global axes.

nufo2

Index of the function associated with translational dampings.

irx, iry, irz

Additional rotational inertias along the global axes (optional).

krx, kry, krz

Rotational stiffnesses along the global axes.

**nufo3**

Index of the function associated with rotational stiffnesses.

**arx, ary, arz**

Rotational dampings along the global axes.

**nufo4**

Index of the function associated with rotational dampings.

**/LECTURE/**

List of the concerned nodes.

### Comments :

The stiffnesses and the dampings are given along the global (fixed) axes of the problem. Each of the 4 associated functions applies to the 3 corresponding stiffnesses (or dampings). For translational stiffnesses one can prescribe three different functions.

If a key-word is missing, the corresponding value is zero, and the order in which the parameters are specified is irrelevant.

The reaction force generated by the support has the form (e.g., assuming translation along Ox):

$$F_x = k_x f_1(D_x) + a_x f_2(V_x)$$

If the displacement (or the velocity) is positive, the function f1 (or f2) must be negative in order to obtain a correct reaction.

### Outputs :

The components of the ECR vector are:

ECR(1): Reaction of the support along X.

ECR(2): Reaction of the support along Y.

ECR(3): Reaction of the support along Z.

ECR(4): Reaction of the support along RX.

ECR(5): Reaction of the support along RY.

ECR(6): Reaction of the support along RZ.

## 7.7 SOLID MATERIALS

### 7.7.1 LINEAR ELASTICITY

**Object:**

This option enables materials with a linear elastic behaviour to be used.

**Syntax:**

```
"LINE"  ![ "R0" rho  "YOUN" young  "NU" nu  
          <"VISC" visc "KRAY" kray "MRAY" mray> ]!  /LECTURE/
```

**rho**

Density of the material.

**young**

Young's modulus.

**nu**

Poisson's ratio.

**visc**

Viscosity coefficient (decay factor), used only by spectral elements (MS24, MS38) and finite elements of the following types: TRIA, CAR1, CAR4, CUBE, CUB6, CUB8, TETR, PR6, PRIS.

**kray, mray**

Rayleigh's stiffness and mass proportional damping coefficients, used only by finite elements of the following types: POUT, TUYA, DKT3, T3GS, Q4GS. Default values: kray=0, mray=0. For information about Rayleigh's damping see reference [\[924\]](#).

**LECTURE**

List of the elements concerned.

**Comments:**

This option may be repeated as many times as necessary.

**Outputs:**

The components of the ECR table are as follows:

Solid elements:

ECR(1): pressure

ECR(2): Von Mises criterion

Shells:

ECR(1): Von Mises criterion (membrane)

ECR(2): Von Mises criterion (membrane + bending)

Beams (3D):

ECR(1): Von Mises criterion (bending)

ECR(2): Von Mises criterion (membrane + bending + torsion)

Bars (BARR, PONC, BR3D):

ECR(1): elastic strain

ECR(2): Von Mises criterion

### 7.7.2 RESL: NONLINEAR SPRING IN THE LOCAL REFERENCE FRAME

#### Object:

This directive allows to model a complex non-linear two-node spring, having arbitrary stiffness and damping values along the 3 dofs of the two concerned nodes. It may only be used in conjunction with RL3D elements (two-node spring). Stiffness and damping are given along local axes. The first local axe (xloc) is defined by the direction of the element that why the element must have a non-zero length. Second (yloc) and third (zloc) axes are defined by the user. The user gives the evolution curve of the reaction force generated by the spring as a function of the displacement or of the velocity.

#### Syntax:

```
"RESL"
  <| [ "KL" k1 ; "KT1" kt1 ; "KT2" kt2 ] |
    | [
      "NFKT" nufo1 ;
      "NFKL" nufokl "NFKS" nufoks
    ] | >
  <| [ "AL" al ; "AT1" at1 ; "AT2" at2 ] |
    | [
      "NFAT" nufo2 ;
      "NFAL" nufoal "NFAS" nufoas
    ] | >
  <| [ "VX" vx ; "VY" vy ; "VZ" vz ] | >
/LECTURE/
```

`k1, kt1, kt2`

Translational stiffnesses along the local axes: xloc, yloc, zloc.

`nufo1`

Index of the function associated with translational stiffnesses.

`nufokl, nufoks`

Indexes of the functions associated with longitudinal and transverse stiffnesses.

`al, at1, at2`

Translational dampings along the local axes.

`nufo2`

Index of the function associated with translational dampings.

`nufoal, nufoas`

Indexes of the functions associated with longitudinal and transverse dampings.

`vx, vy, vz`

Coordinates of the vector  $v$ . Projection of  $v$  on the "orthogonal to  $xloc$ " plane gives  $yloc$ .

/LECTURE/

List of the concerned elements.

**Comments:**

The stiffnesses and the dampings are given along the local axes of the problem in the initial configuration. If a single function is specified for stiffnesses (dampings), it applies to the 3 corresponding stiffnesses (dampings).

If a key-word is missing, the corresponding value is put to zero. The order in which the parameters are specified is irrelevant.

The reaction force generated by the spring has the form (e.g., assuming translation along the  $xloc$  axis):

$$F_x = k_x f_1(D_x) + a_x f_2(V_x)$$

If the displacement (or the velocity) is positive, the function  $f_1$  (or  $f_2$ ) must be negative in order to obtain a correct reaction.

**Outputs:**

The components of the ECR vector are:

ECR(1): Force in the spring along  $xloc$ .

ECR(2): Force in the spring along  $yloc$ .

ECR(3): Force in the spring along  $zloc$ .

### 7.7.3 GENERIC LINEAR ELASTICITY

**Object:**

This option enables materials with a linear elastic behaviour to be used. It is an interface to convert the input to the appropriate material (LINE 7.7.1, VM23 7.7.39) for the elements used.

**Syntax:**

```
"GLIN"  ![ "RO" rho  "YOUN" young  "NU" nu ]!  /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

LECTURE

List of the elements concerned.

**Outputs:**

The output variables are according to the material in which the generic material is converted.



### 7.7.4 GENERIC PLASTICITY

**Object:**

This option enables materials with a linear elastic behaviour to be used. It is an interface to convert the input to the appropriate material (VMIS ISOT 7.7.7, VM23 7.7.39) for the elements used.

**Syntax:**

```
"GPLA"  ! [ "R0" rho  "YOUN" young  "NU" nu ]!  "ELAS" sige ...  
      ...      "TRAC"  npts*( sig  eps )      /LECTURE/
```

**rho**

Density of the material.

**young**

Young's modulus.

**nu**

Poisson's ratio.

**sige**

Elastic limit.

**"TRAC"**

This key-word introduces the yield curve.

**npts**

Number of points (except the origin) defining the yield curve.

**sig**

Stress.

**eps**

Total strain (elastic + plastic)

**LECTURE**

List of the elements concerned.

**Outputs:**

The output variables are according to the material in which the generic material is converted.

### 7.7.5 RESG: NONLINEAR SPRING IN THE GLOBAL REFERENCE FRAME

#### Object:

This directive allows to model a complex non-linear two-node spring having arbitrary stiffness and damping values along the 3 dofs of the two concerned nodes. It may only be used in conjunction with RL3D elements (two-node spring). The user gives the evolution curve of the reaction force generated by the spring as a function of the displacement or of the velocity.

#### Syntax:

```
"RESG"
    <| [ "KX" kx ; "KY" ky ; "KZ" kz ] | "NFKT" nufo1 >
    <| [ "AX" ax ; "AY" ay ; "AZ" az ] | "NFAT" nufo2 >
    /LECTURE/
```

**kx, ky, kz**

Translational stiffnesses along the global axes.

**nufo1**

Index of the function associated with translational stiffnesses.

**ax, ay, az**

Translational dampings along the global axes.

**nufo2**

Index of the function associated with translational dampings.

**/LECTURE/**

List of the concerned elements.

#### Comments:

The stiffnesses and the dampings are given along the global (fixed) axes of the problem. Each of the 2 associated functions applies to the 3 corresponding stiffnesses (or dampings).

If a key-word is missing, the corresponding value is put to zero. The order in which the parameters are specified is irrelevant.

The reaction force generated by the spring has the form (e.g., assuming translation along Ox):

$$F_x = k_x f_1(D_x) + a_x f_2(V_x)$$

If the displacement (or the velocity) is positive, the function f1 (or f2) must be negative in order to obtain a correct reaction.

**Outputs:**

The components of the ECR vector are:

ECR(1): Force in the support along X.

ECR(2): Force in the support along Y.

ECR(3): Force in the support along Z.

### 7.7.6 DRUCKER-PRAGER

**Object:**

This option enables to specify materials with a perfect elasto-plastic behaviour (Drucker-Prager criterion).

**Syntax:**

```
"DRUC"  "RO"  rho  "YOUNG"  young  "POISSON"  nu  ...  
...  "TRACTION"  sigt  "COMPRESSION"  sigc  ...  
...  < "FRACTURE"  pf  >  /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

sigt

Maximum stress under tension (without confinement).

sigc

Maximum stress under compression (without confinement).

FRACTURE pf

The material is fractured (does no longer resist tension) as soon as the criterion is reached for the first time. Then, the domain changes and the new D.P. criterion corresponds to vanishing cohesion whereas the slope is equivalent to the one in the previous case. The parameter 'pf' is compulsory and represents the maximum pressure of fracturing under compression. If the criterion is reached for the first time when the pressure is superior to pf, the domain does not change and the criterion is the same as initially.

/LECTURE/

Numbers of the elements concerned.

**Comments:**

The values of sigt, sigc and pf are absolute values.

If P defines the pressure (positive under tension) and SIG\* the Von Mises criterion, the Drucker-Prager criterion is defined by:

$$\text{Criterion} = \text{SIG*} - \text{cohe} + P * \text{pente} \quad ( \text{always} \leq 0 )$$

The 2 parameters : cohe and pente (slope), are calculated from sigt and sigc values, they are printed after the reading of the data. The parameter cohe (cohesion) corresponds to a maximum Von Mises under non-existent pressure. The slope is the straight line limiting the domain, in the coordinate system (P,SIG\*).

In the space of the principal stresses the criterion determines a cone the axis of which is the straight line of equation:  $\text{sig}(1) = \text{sig}(2) = \text{sig}(3)$ .

The maximum stresses: sigt and sigc correspond to the values observed during uniaxial tests without confinement. These two points enable the Drucker-Prager domain to be defined.

The value of the parameter "FRACTURE" enables the behaviour of concrete to be represented in a very simplified way. Two domains may be distinguished:

- Brittle rupture
- Ductile rupture (strong compressions)

Most often one may take  $\text{pf} = \text{sigc}/3$ . A great value for pf delay the fracturation.

## Outputs:

The different components of the ECR table are as follows:

ECR(1): pressure

ECR(2): Von Mises

ECR(3): equivalent plastic strain

ECR(4): D.P. criterion ( always  $\leq 0$  )

ECR(5): cohesion (becomes non-existent in the case of brittle rupture)

### 7.7.7 VON MISES MATERIAL

**Object:**

This sub-directive enables materials with an elasto-plastic behaviour to be used. There are four options:

- "VMIS" "PARF" : perfectly plastic Von Mises material;
- "VMIS" "ISOT" : isotropic Von Mises material;
- "VMIS" "DYNA" : isotropic Von Mises material depending on strain rate;
- "VMIS" "TETA" : isotropic Von Mises material depending on temperature.

**Syntax:**

```
"VMIS"  
  $ [  
    "PARF" . . . ;  
    "ISOT" . . .  
    "DYNA" . . .  
    "TETA" . . .  
  ]$
```

**Comments:**

This sub-instruction may be repeated as many times as necessary with different options each time (if need be). The word "VMIS" cannot be separated from the option which follows.

**PERFECTLY PLASTIC VON MISES****Object:**

Perfectly plastic Von Mises material.

**Syntax:**

```
"VMIS"  "PARF"  "RO" rho  "YOUN" young  "NU" nu  "ELAS" sige  ...  
        ...  /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

sige

Elastic limit.

ncrit

LECTURE

List of the elements concerned.

**Comments:**

The law of behaviour is described by the following diagram of stresses and strains:



Figure 4: VMIS - stress-strain relation

**Outputs:**

The components of the ECR table are as follows:

Solid elements:

- ECR(1): pressure
- ECR(2): Von Mises criterion
- ECR(3): plastic strain

Shells integrated through the thickness:

- ECR(1): pressure
- ECR(2): Von Mises criterion
- ECR(3): plastic strain

Global model shells:

- ECR(1): Von Mises criterion (membrane)
- ECR(2): Von Mises criterion (membrane + bending)
- ECR(3): plastic strain

Bars (BARR, PONC, BR3D):

- ECR(1): elastic strain
- ECR(2): Von Mises criterion
- ECR(3): plastic strain



**ISOTROPIC VON MISES****Object:**

Isotropic Von Mises material.

**Syntax:**

```
"VMIS" "ISOT" "R0" rho "YOUN" young "NU" nu "ELAS" sige ...  
      <FAIL fail LIM1 limi>  
      ...      "TRAC" npts*( sig eps )      /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

sige

Elastic limit.

FAIL

Optional keyword: introduces an element failure model represented by a failure criterion and a by failure limit value. Two failure criteria only available for POUT and bar (BR3D, BARR, PONC) elements are:

fail = 1 for a criterion based upon Von Mises stress (membrane + bending + torsion),

fail = 2 for a criterion based upon plastic strain.

limi

Optional parameter, indicates the failure limit for the chosen criterion.

"TRAC"

This key-word introduces the yield curve.

npts

Number of points (except the origin) defining the yield curve.

sig

Stress.

eps

Total strain (elastic + plastic).

/LECTURE/

List of the elements concerned.

**Comments:**

1/ - The young parameter defines Young's modulus during an elastic phase.

2/ - The points (sig,eps) may have any position; however, concerning the first point, there must be a compatibility between the coordinates, Young's modulus and the elastic limit.

**Outputs:**

The components of the ECR table are as follows:

## Solid elements

ECR(1): pressure  
ECR(2): Von Mises criterion  
ECR(3): plastic strain  
ECR(7): new elastic limit

## Shells integrated through the thickness:

ECR(1): pressure  
ECR(2): Von Mises criterion  
ECR(3): plastic strain  
ECR(7): new elastic limit

## Global model shells:

ECR(1): Von Mises criterion (membrane)  
ECR(2): Von Mises criterion (membrane + bending)  
ECR(3): plastic strain  
ECR(7): new elastic limit

## Beams (3D):

ECR(1): Von Mises criterion (bending)  
ECR(2): Von Mises criterion (membrane + bending + torsion)  
ECR(3): plastic strain  
ECR(7): new elastic limit  
ECR(10): failure flag (0=virgin Gauss Point, 1=failed Gauss Point)

## Bars (BARR, PONC, BR3D):

ECR(1): elastic strain  
ECR(2): Von Mises criterion  
ECR(3): plastic strain  
ECR(7): new elastic limit ECR(10): failure flag (0=virgin Gauss Point, 1=failed Gauss Point)

**DYNAMIC VON MISES****Object:**

Isotropic Von Mises material depending on strain rate.

**Syntax:**

```
"VMIS"  "DYNA"  "RO" rho  "YOUN" young  "NU" nu  ...
...  "TRAC"  npts*( sig  eps )  ...

...  $[ "SYMO"  "D" d  "P" p  ;
      "ISPR" "VITE"  a  b  c  d  e  f  ;
      "LIBR"  num  "PARA"  /LECPARA/  :
      "ARMA" "ALFAY" alfay "ALFAU" alfau
      <"FAIL" nfail "LIMI" limi >  ]$  /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

"TRAC"

This key-word introduces the yield curve.

npts

Number of points (except the origin) defining the yield curve (in the case of the ARMA model only 3 points should be used : the yield point, the onset of hardening and the ultimate point before softening. Yield curve is defined analytically thanks to the assumption that it is a portion of parabola beginning at the onset of hardening and reaching a maximum at the ultimate point).

sig

Stress

eps

Total strain (elastic + plastic).

"SYMO"

Constitutive relation of Symonds and Cowper.

d

First coefficient of the Symonds and Cowper law.

p

Second coefficient of the Symonds and Cowper law.

"VITE"

This key-word introduces the parameters of the dynamic yield curve.

a,b,c,d,e,f

The 6 parameters of the dynamic yield curve.

"LIBRE"

Introduces the utilisation of a user's subroutine to compute the dynamic amplification coefficient.

num

Identification number of the free material.

"PARA"

Keyword that can be used to introduce a series of parameters for the free material. The number of parameters is arbitrary, because the /LECTURE/ procedure signals the termination of the list.

"ARMA"

This key-word introduces the parameters of the dynamic yield curve for steel reinforcing bars.

alfay

Coefficient to obtain the DIF for the yield stress.

alfau

Coefficient to obtain the DIF for the ultimate stress.

FAIL

Optional keyword: introduces an element failure model represented by a failure criterion and a by failure limit value. Two failure criteria only available for POUT and bar (BR3D, BARR) elements are:

fail = 1 for a criterion based upon Von Mises stress (membrane + bending + torsion),

fail = 2 for a criterion based upon plastic strain.

limi

Optional parameter, indicates the failure limit for the chosen criterion.

/LECTURE/

List of the elements concerned.

**Comments:**

For the Symonds and Cowper law, the dynamic traction curve is derived from the static one through a multiplicative coefficient which depends upon the strain rate (EPSP):

$$\text{SIG}(\text{dyna}) = \text{SIG}(\text{stat}) * ( 1 + ( \text{EPSP} / D ) ** (1/P) )$$

Indicatively, for the stainless steel 304 L, experimental results suggest:  $D = 100 \text{ s}^{-1}$  and  $P = 10$  (Forrestal and Sagartz 1978). For ordinary steel, it is usually assumed:  $D = 40 \text{ s}^{-1}$  and  $P = 5$  (Symonds 1965).

For titanium TI-50A, the values suggested are:  $D = 120 \text{ s}^{-1}$  and  $P = 9$  (Symonds et Chon 1974).

For aluminum alloys, some authors use  $D = 6500 \text{ s}^{-1}$  and  $P = 4$  (Symonds 1965).

In the case of the ISPRA law, the formulation is similar, but the multiplying coefficient depends upon the strain (EPS) as well as on the strain rate (EPSP):

$$\text{SIG}(\text{dyna}) = \text{SIG}(\text{stat}) * ( 1 + ( \text{EPSP} / K ) ** M )$$

with the K and M coefficients of the form:

$$K = \text{EXP} ( ( A + B * \text{EPS} ) / ( 1 + C * \text{EPS} ) )$$

$$M = ( D + E * \text{EPS} ) / ( 1 + F * \text{EPS} )$$

Examples of data (source ISPRA-CADARACHE):

Material	a	b	c	d	e	f
Steel 304	5.82	168.76	9.62	0.242	2.263	12.77
Steel 316	6.388	86.215	6.457	0.233	0.0	0.0

For the ARMA model, the dynamic yield curve is obtained as follows:  
The dynamic increase factor for the yield stress is given by:

$$\text{SIG}_Y(\text{dyna}) = \text{SIG}_Y(\text{stat}) * \text{DIF}_Y$$

$$\text{DIF}_Y = ( \text{EPSP} / 10^{-4} ) ** \text{ALFAY}$$

The dynamic increase factor for the ultimate stress is given by:

$$\text{SIG}_U(\text{dyna}) = \text{SIG}_U(\text{stat}) * \text{DIF}_U$$

$$\text{DIF}_U = ( \text{EPSP} / 10^{-4} ) ** \text{ALFAU}$$

Then, the yield curve is defined analytically thanks to the assumption that it is a portion of parabola beginning at the onset of hardening and reaching a maximum at the ultimate point.

This model is suited for steel reinforcing bars, so it can be used only with bar and beam elements.

It is suggested in "Dynamic Increase Factors for Steel Reinforcing Bars, L. J. Malvar and J. E. Crawford, Twenty-Eighth DDESB Seminar, Orlando, Florida, USA, August 1998" that ALFAY and ALFAU can be estimated by the expressions:

$$\text{ALFAY} = 0.074 - (0.040 * f_y / 414. )$$

$$\text{ALFAU} = 0.019 - (0.009 * f_y / 414. )$$

where  $f_y$  is the bar yield strength in MPa.

This formulation is valid for bars with yield stress between 290 and 710 MPa and for strain rates between  $10^{-4}$  and  $225 \text{ s}^{-1}$ .

### IMPORTANT POINT:

For all formulations, the strain rate EPSP is filtered with a first order low-pass filter:

$$d\text{EPSP}/dt = (\text{EPSPC} - \text{EPSP})/\tau$$

with EPSPC the current value of the strain rate and TAU the filter time constant:

$$\text{TAU} = 1 / (2 * \pi * f_c)$$

with  $f_c$  the cutoff frequency of the filter.

By time integration, we obtains:

$$\text{EPSP}(n+1) = ( \text{EPSP}(n) + (\text{DELTAT} / \text{TAU}) * \text{EPSPC}(n+1) ) / ( 1 + (\text{DELTAT} / \text{TAU}) )$$

$$\text{DELTAT} = t(n+1) - t(n)$$

Furthermore, we supposed that:

$$\text{BETA} = \text{DELTAT} / \text{TAU} = \text{Cte}$$

and introduced FEPS1:

$$\text{FEPS1} = \text{BETA} / (1 + \text{BETA}) = \text{DELTAT} / (\text{DELTAT} + \text{TAU})$$

FEPS1 is defined in OPTI FVIT.

Default value is 1, meaning  $\text{TAU} = 0$ , no filter is applied.

Advised value for FEPS1 is 0.01.

### Outputs:

The components of the ECR table are as follows:

Solid elements:

- ECR (1): pressure
- ECR (2): Von Mises criterion in dynamics
- ECR (3): equivalent plastic strain
- ECR (7): new elastic limit in statics
- ECR (8): equivalent strain rate
- ECR (9): total equivalent deformation
- ECR(11): elastic limit in dynamics

Shells integrated through the thickness:

- ECR (1): pressure
- ECR (2): Von Mises criterion in dynamics
- ECR (3): equivalent plastic strain
- ECR (7): new elastic limit in statics
- ECR (8): equivalent strain rate
- ECR (9): total equivalent deformation
- ECR(11): elastic limit in dynamics

Global model Shells:

- ECR (1): Von Mises criterion (membrane)
- ECR (2): global Von Mises criterion (membrane + bending)
- ECR (3): equivalent plastic strain
- ECR (7): new elastic limit
- ECR (8): equivalent strain rate
- ECR (9): total equivalent deformation
- ECR(11): elastic limit in dynamics

Beams (3D) for ARMA only:

- ECR(1): Von Mises criterion (bending)
- ECR(2): Von Mises criterion (membrane + bending + torsion)
- ECR(3): plastic strain
- ECR(7): new elastic limit
- ECR(10): failure flag (0=virgin Gauss Point, 1=failed Gauss Point)
- ECR(11): elastic limit in dynamics

Bars (BARR, BR3D) for ARMA only:

- ECR(1): elastic strain
- ECR(2): Von Mises criterion
- ECR(3): plastic strain
- ECR(7): new elastic limit
- ECR(10): failure flag (0=virgin Gauss Point, 1=failed Gauss Point)
- ECR(11): elastic limit in dynamics

**TEMPERATURE-DEPENDENT VON MISES****Object:**

Von Mises isotropic material dependent upon the temperature.

**Syntax:**

```
"VMIS"  "TETA"  "RO" rho < "NU" nu >...  
...  "NBCOURBE"  nc*( "TETA" ti "YOUNG" yg <"NUT"> nut ...  
...  "TRAC" npts*( sig eps ) ) /LECTURE/
```

rho

Density.

nu

Poisson coefficient. Only if NU does not depend on the temperature.

nc

Number of traction curves that allow the interpolation as a function of temperature.

ti

Temperature associated with the following traction curve.

yg

Young's modulus.

nut

Poisson Poisson. If NU depend on temperature.

"TRAC"

Introduces the traction curve.

npts

Number of points (excluding the origin) which define the traction curve.

sig

Stress.

eps

Total strain (elastic + plastic).

/LECTURE/

List of the elements concerned.



**Comments:**

Each element is isothermal, i.e. its temperature remains constant during the whole calculation.

Depending upon temperature, the Young's modulus, the poisson coefficient and the traction curve are interpolated starting from the values associated to the known temperatures.

Note that it is possible to define either a temperature-dependant Poisson coefficient or not which can be sufficient in case of steels for example.

**Outputs:**

The components of the ECR table are as follows:

Continuum elements:

- ECR(1) : pressure
- ECR(2) : Von Mises criterion
- ECR(3) : equivalent plastic strain

Shells integrated through the thickness:

- ECR(1) : pressure
- ECR(2) : Von Mises criterion
- ECR(3) : equivalent plastic strain

Global model shells:

- ECR(1) : Von Mises criterion (membrane)
- ECR(2) : global Von Mises criterion (membrane + bending)
- ECR(3) : equivalent plastic strain

**7.7.8 STEINBERG-GUINAN****Object:**

This is a Von Mises isotropic material whose Young's modulus and elastic limit are a function of hydrostatic pressure, temperature increase and strain rate.

**Syntax:**

```
"STGN"  "RO"  rhoz  "YOUN"  youngz  "NU"    nu      ...
...     "SIGE" sigez "SIGD"  sigd   "CHSP"  cv      ...
...     "TF"   tfus  "TINI"  tini   "B"     b       ...
...     "H"   h      "BETA"  beta   "N"     n       /LECTURE/
```

**rhoz**

Density at the initial temperature.

**youngz**

Young's modulus at the initial temperature.

**nu**

Poisson coefficient (constant).

**sigez**

Static elastic limit at the initial temperature.

**sigd**

Dynamic elastic limit at the initial temperature.

**cv**

Specific heat capacity of the solid.

**tfus**

Melting temperature of the material.

**tini**

Initial temperature of the material.

**b,h,beta,n**

Coefficients of the STEINBERG and GUINAN law.

/LECTURE/

List of the elements concerned.

### Comments:

The STEINBERG and GUINAN law uses the Young's modulus  $E$ , and an elastic limit  $Y$ , which vary according to the following expressions:

$$E = \text{youngz} * P1$$

$$Y = \text{yield} * P1$$

with:

$$P1 = 1 + b * P / K^{1/3} + h * dteta$$

$$\text{yield} = \text{MIN} ( \text{sigd} , \text{sigez} * P2 )$$

$$P2 = ( 1 + \text{beta} * \text{EPSP} )^n$$

where:

$P$  is the hydrostatic pressure;

$K$  is the compression ratio (ratio between the current density and the initial density);

$\text{EPSP}$  is the total equivalent strain rate;

$dteta$  is the temperature increase with respect to the initial temperature.

On the other hand, when the current temperature ( $teta = t_{ini} + dteta$ ) exceeds the melting temperature of the material ( $t_{fus}$ ), it is assumed that the material is liquefied: the Young's modulus and the elastic limit are then taken as zero.

### Outputs:

The various components of the ECR table are as follows:

ECR(1) : hydrostatic pressure

ECR(2) : Von Mises

ECR(3) : equivalent plastic strain

ECR(4) : temperature increase ( $dteta$ )

ECR(5) : current elastic limit

ECR(6) : current Young's modulus

ECR(7) : equivalent plastic strain rate

### 7.7.9 VON MISES ORTHOTROPIC GRID MODEL

**Object:**

This orthotropic, perfect Von-Mises model allows to assign a grid behavior to a shell element. It is typically used to model steel reinforcement in concrete panels. The first orthotropic direction is declared in COMP ORTS directive.

**Syntax:**

```
"VMGR"  "RO" rho  "OM1" om1  "YG1" yg1  "SP1" sp1  
        "OM2" om2  "YG2" yg2  "SP2" sp2  
... /LECTURE/
```

rho

Density of the reinforcement material.

om1

Reinforcement ratio in the first orthotropic direction.

yg1

Young's modulus of the reinforcement material the first orthotropic direction.

sp1

Elastic limit of the reinforcement material in the first orthotropic direction.

om2

Reinforcement ratio in the second orthotropic direction.

yg2

Young's modulus of the reinforcement material in the second orthotropic direction.

sp2

Elastic limit of the reinforcement material in the second orthotropic direction.

LECTURE

List of the elements concerned.

**Comments:**

This law is available with or without layers (SANDWICH directive).

**Outputs:**

The components of the ECR table are as follows:

ECR(1): current plastic strain in the first orthotropic direction

ECR(2): current plastic strain in the second orthotropic direction

ECR(3): current grid stress in the first orthotropic direction (averaged shell stress divided by  $om1$ )

ECR(4): current grid stress in the second orthotropic direction (averaged shell stress divided by  $om2$ )

ECR(5): angle between local shell system and orthotropic system

#### **IMPORTANT POINT:**

With VMGR material, the stress CONT corresponds to an averaged stress over element width and layer thickness, while ECR(3-4) provide the actual stresses within reinforcement bars.

For this reason, to avoid confusion, it is strongly advised to specify a layer thickness fraction equal to the reinforcement ratio, so that the above quantities can coincide.

**7.7.10 LEM1****Object :**

This directive allows to describe the behaviour of an elasto-plastic material that may undergo some damage, according to the Lemaitre model. There is coupling between damage and plasticity, represented by the Von Mises criterion. The damage evolution rate is a function of the triaxiality ratio of stresses and of the equivalent plastic strain rate. A failure criterion is implicitly contained within the model: rupture occurs when the damage exceeds a critical value. Two optional parameters allow to introduce a limitation of the damage rate (thanks to the delayed damage model) in order to avoid the mesh dependency.

**Syntax:**

```
"LEM1"  "R0" rho "YOUN" young "NU" nu "ELAS" sige ...
        "EPSD" epsd "S0" s0 "DC" dc ...
        <"CSTA" csta "TAUC" tauc "NOCO" noco> ...
        "TRAC" npts*( sig eps ) /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's coefficient.

sige

Elastic limit.

epsd

Damage threshold (i.e. equivalent plastic strain, weighted by a function of stress triaxiality, within which damage vanishes).

s0

Parameter driving the damage evolution rate.

dc

Critical damage defining the failure criterion.

csta

Parameter of the delayed damage model

tauc

Characteristic time of the delayed damage model.  $(1/\text{tauc})$  represents the maximum damage rate.

**noco**

Optional parameter indicating what to do when no convergence is reached in the material routine. The value 0 is the default and means that an error message is issued and the calculation is stopped. The value 1 indicates that the element (or more precisely, the element's current Gauss point) is made to fail (eroded). The value -1 indicates that subcycling is activated in an attempt to reach convergence, by subdividing the load step into smaller sub-cycles.

**"TRAC"**

Introduces the traction curve.

**npts**

Number of points (except the origin) defining the traction curve.

**sig**

Stress.

**eps**

Total strain (elastic + plastic).

**LECTURE**

List of concerned elements.

**Comments:**

A detailed description of the model can be found in the report DMT/98-026A, available on request.

**Outputs:**

The components of the ECR table are as follows for **Continuum elements**:

ECR(1) : pressure

ECR(2) : Von Mises criterion

ECR(3) : equivalent plastic strain

ECR(4) : plasticity multiplier

ECR(5) : damage

ECR(7) : new elastic limit

When the "erosion" algorithm is activated (see page A.30, Section 4.4, keyword **EROS**), an integration point is considered as failed if **damage**  $\geq$  **dc**. It will be eroded concerning the rules for **EROS**.

### 7.7.11 ZALM

#### Object :

This directive allows to describe the behaviour of an Zirelli-Armstrong material that may undergo some damage, according to the Lemaitre model. There is coupling between damage and plasticity, represented by the Von Mises criterion. The damage evolution rate is a function of the triaxiality ratio of stresses and of the equivalent plastic strain rate. A failure criterion is implicitly contained within the model: rupture occurs when the damage exceeds a critical value. Two optional parameters allow to introduce a limitation of the damage rate (thanks to the delayed damage model) in order to avoid the mesh dependency.

#### Syntax:

```
"ZALM"  "RO" rho "YOUN" young "NU" nu "ELAS" sige ...
        "EPSD" epsd "S0" s0 "DC" dc ...
        "ZAC0" zac0 "ZAC1" zac1 "ZAC2" zac2 "ZAC3" zac3 ...
        "ZAC4" zac4 "ZAC5" zac5 "ZAN" zan ...
        <"CSTA" csta "TAUC" tauc> ...
        "TRAC" npts*( sig eps ) /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's coefficient.

sige

Elastic limit.

epsd

Damage threshold (i.e. equivalent plastic strain, weighted by a function of stress triaxiality, within which damage vanishes).

s0

Parameter driving the damage evolution rate.

dc

Critical damage defining the rupture criterion.

csta

Parameter of the delayed damage model

zac0



Parameter of zerilli-armstrong model c0

zac1

Parameter of zerilli-armstrong model c1

zac2

Parameter of zerilli-armstrong model c2

zac3

Parameter of zerilli-armstrong model c3

zac4

Parameter of zerilli-armstrong model c4

zac5

Parameter of zerilli-armstrong model c5

zan

Parameter of zerilli-armstrong model n

tauc

Characteristic time of the delayed damage model.  $(1/\text{tauc})$  represents the maximum damage rate.

"TRAC"

Introduces the traction curve.

npts

Number of points (except the origin) defining the traction curve.

sig

Stress.

eps

Total strain (elastic + plastic).

LECTURE

List of concerned elements.

### Outputs:

The components of the ECR table are as follows for **Continuum elements**:

ECR(1) : pressure  
ECR(2) : Von Mises criterion  
ECR(3) : equivalent plastic strain  
ECR(4) : plasticity multiplier  
ECR(5) : damage  
ECR(7) : new elastic limit

When the “erosion” algorithm is activated (see page A.30, Section 4.4, keyword **FAIL**), an element is considered as failed if **damage**  $\geq$  **dc**.

## 7.7.12 LMC2

**Object:**

This directive allows to describe the behaviour of an elasto-plastic material that may undergo some damage, according to the Lemaitre-Chaboche model. There is coupling between damage and plasticity, represented by the Von Mises criterion. The damage evolution rate is a function of the triaxiality ratio of stresses and of the equivalent plastic strain rate. A failure criterion is implicitly contained within the model: rupture occurs when the damage exceeds a critical value. Unlike model LEM1, the material properties may depend upon the strain rate. Two optional parameters allow to introduce a limitation of the damage rate (thanks to the delayed damage model) in order to avoid the mesh dependency

**Syntax:**

```
"LMC2" "RO" rho ...
      "YOUN" young < "FONC" nfyou ...
            $[ "TABL" nptyou*( para vyou ) ; "ROUT" ; "DONE" ]$ > ...
      "NU" nu      < "FONC" nfnu ...
            $[ "TABL" nptnu*( para vnu ) ; "ROUT" ; "DONE" ]$ > ...
      "ELAS" sigE < "FONC" nfela ...
            $[ "TABL" nptela*( para vela ) ; "ROUT" ; "DONE" ]$ > ...
      "EPSD" epsd < "FONC" nfepd ...
            $[ "TABL" nptepd*( para vepd ) ; "ROUT" ; "DONE" ]$ > ...
      "SO" s0      < "FONC" nfs0 ...
            $[ "TABL" npts0*( para vs0 ) ; "ROUT" ; "DONE" ]$ > ...
      "DC" dc      < "FONC" nfdc ...
            $[ "TABL" nptdc*( para vdc ) ; "ROUT" ; "DONE" ]$ > ...
      <"CSTA" csta "TAUC" tauc> ...
```

If the traction curve is given by a table:

```
"TRAC" ctra "FTRA" nftra ...
      $ "TABL" npt*( sig eps ) ; "ROUT" ; "DONE" $ ...
```

If the traction curve is given by an abaque:

```
"TRAC" ctra "ATRA" natra $ "SET" npara ...
      "NPTM" nptm*( "PARA" para "TABL" npt*( sig eps )); ...
      "DONE" $ ...
```

/LECTURE/

rho

Density.

young

Young's modulus if it is constant or multiplicative coefficient of Young's modulus if it is defined by a function.

**nfyou**

Number of the function defining the variation of the Young's modulus with the strain rate.

**nptyou**

Number of point defining the variation of the Young's modulus with the strain rate.

**para**

Parameter (here the strain rate).

**vyou**

Value of the Young's modulus corresponding to the parameter.

**nu**

Poisson's coefficient if it is constant or multiplicative coefficient of Poisson's coefficient if it is defined by a function.

**nfnu**

Number of the function defining the variation of the Poisson's coefficient with the strain rate.

**nptnu**

Number of point defining the variation of the Poisson's coefficient with the strain rate.

**vnv**

Value of the Poisson's coefficient corresponding to the parameter.

**sige**

Elastic limit if it is constant or multiplicative coefficient of the elastic limit if it is defined by a function.

**nfela**

Number of the function defining the variation of the elastic limit with the strain rate.

**nptela**

Number of point defining the variation of the elastic limit with the strain rate.

**vela**

Value of the elastic limit corresponding to the parameter.

**epsd**

Damage threshold (i.e. equivalent plastic strain, weighted by a function of triaxiality rate of stresses, below which the damage is zero) if it is constant or multiplicative coefficient of the damage threshold if it is defined by a function.

**nfepd**

Number of the function defining the variation of the damage threshold with the strain rate.

**nptepd**

Number of point defining the variation of the damage threshold with the strain rate.

**vepd**

Value of the damage threshold corresponding to the parameter.

**s0**

Parameter driving the evolution rate of damage if it is constant or multiplicative coefficient of the parameter driving the evolution rate of damage if it is defined by a function.

**nfs0**

Number of the function defining the variation of the parameter driving the evolution rate of damage with the strain rate.

**npts0**

Number of point defining the variation of the parameter driving the evolution rate of damage with the strain rate.

**vs0**

Value of the dparameter driving the evolution rate of damage corresponding to the parameter.

**dc**

Critical damage defining the rupture criterion if it is constant or multiplicative coefficient of critical damage if it is defined by a function.

**nfdc**

Number of the function defining the variation of the critical damage with the strain rate.

**nptdc**

Number of point defining the variation of the critical damage with the strain rate.

**vdc**

Value of the critical damage corresponding to the parameter.

**csta**

Parameter of the delayed damage model

**tauc**

Characteristic time of the delayed damage model.  $(1/\text{tauc})$  represents the maximum damage rate.

**"TRAC"**

Introduces the traction curve.

**ctra**

Multiplicative coefficient of the stress in the traction curve or curves.

**"FTRA"**

Introduces the single traction curve for all strain rates.

**nftra**

Number of the function defining the traction curve.

**npt**

Number of point (except the origin) defining the traction curve.

item[**sig**]

Stress. item[**eps**]

Strain (elastic+plastic).

**"ATRA"**

Introduces an abaque giving the traction curve for different strain rates.

**natra**

Number of the abaque defining the traction curves.

**npara**

Number of the set of parametrised functions that associate to each strain rate the corresponding traction curve.

**nptm**

Maximum number of point (except the origin) defining the traction curve amongst the set of parametrised functions.

**LECTURE**

List of the concerned elements.

### **Comments:**

In the case of traction curve, parametrised or not, the origin is always omitted.

If both the Young's modulus and the traction curve are parametrised, the strain rate parameter should be identical.

Dans le cas de la courbe de traction parametree, il faudra fournir les vitesses de deformation de maniere croissante.

In the case of a component dependent upon strain rate, it is mandatory to give its value for a zero velocity (static case) and for a very large velocity.

A detailed description of the model may be found in the report DMT/98-036A, available on request.

**Outputs:**

The components of the ECR table are as follows for **Continuum elements** :

ECR(1) : pressure

ECR(2) : Von Mises criterion

ECR(3) : equivalent plastic strain

ECR(4) : plasticity multiplier

ECR(5) : damage

ECR(7) : new elastic limit

ECR(8) : strain rate

ECR(11): = 1 critical damage reached, otherwise < 1

When the “erosion” algorithm is activated (see page A.30, Section 4.4, keyword FAIL), an element is considered as failed if  $ECR(11) > 0.99$ .

**7.7.13 CONCRETE: Old version****Object:**

This option is used to define materials such as concrete, soil, rock, etc.

**Comments:**

The law of behaviour used in this model is based on plasticity; it takes into account three modes of damaging the material:

- 1) Damage due to traction;
- 2) Damage due to shear;
- 3) Damage due to hydrostatic pressure.

A material of this type possesses 38 input parameters; however, only some of them are compulsory. Each parameter is entered into the input file by means of a key-word, these words can be entered in any order. Just remember that the data placed between angle brackets are not compulsory, for example: <"PREC" prec>.

The numerical values of the different parameter are entered in absolute value. Moreover the following conventions have been adopted for the outputs:

- positive values: tension stresses;
- negative values: compression stresses.

The option "BETON" can be repeated as many times as necessary.

**Syntax:**

The data can be classified in 4 groups.

**- 1) Generic data**

```
"BETON"  "RO" rho  "YOUN" young  "NU" nu
          < "ALPH" alph >  < "PREC" prec >
```

rho



Density of the material.

young

Elasticity modulus.

nu

Poisson's ratio.

alpha

Coefficient of thermal expansion.

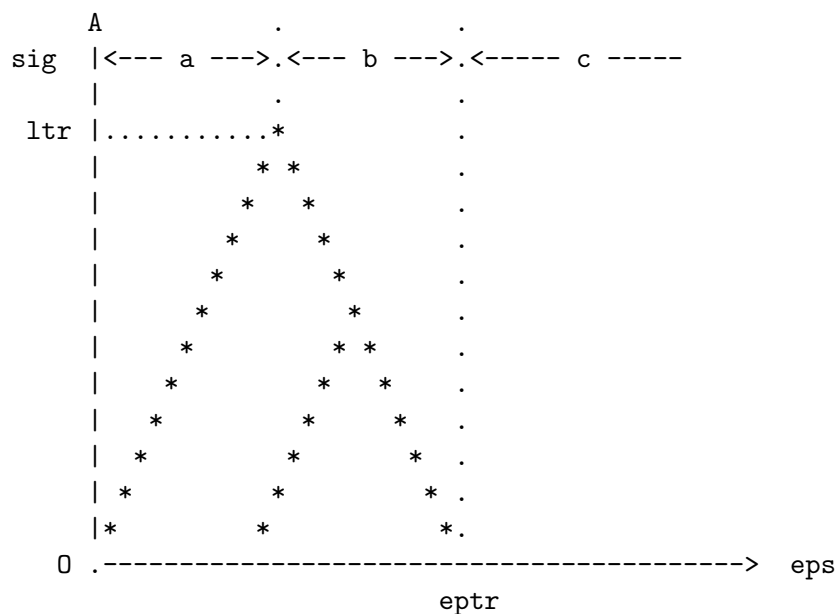
prec

Precision of the computation on the internal iterations.

## - 2) Data concerning the damage due to traction:

This kind of damage occurs in 3 phases:

- a) elastic behaviour;
- b) cracked elastic behaviour;
- c) perfectly plastic behaviour.



< "BETA" cisail >

\* initially isotropic material:

"LTR" ltr  
 "EPTR" eptr

\* initially anisotropic material:

< "IFIS" ifis >

< "LT1" lt1 > < "LT2" lt2 > < "LT3" lt3 >  
 < "EPT1" ept1 > < "EPT2" ept2 > < "EPT3" ept3 >

< "OUV1" ouv1 > < "OUV2" ouv2 > < "OUV3" ouv3 >

< "ANGL" angle > or < "V1X" v1x "V1Y" v1y "V1Z" v1z >  
 < "V2X" v2x "V2Y" v2y "V2Z" v2z >  
 < "V3X" v3x "V3Y" v3y "V3Z" v3z >

cisail

Value of residual shear after cracking, in comparison with the initial status (value between 0 and 1).

ltr

Limit in traction in the case of an initially isotropic material.

eptr

Rupture strain in the case of an initially isotropic material.

ifis

Cracking index (0: no cracking, 1: one crack only, 2: two cracks, 3: three cracks).

lt1, lt2, lt3

Traction limits along the directions 1, 2 and 3 in the case of an initially anisotropic material.

ept1, ept2, ept3

Rupture strains along the directions 1, 2 and 3 in the case of an initially anisotropic material.

ouv1, ouv2, ouv3

Opening of the cracks along the directions 1, 2 and 3 in the case of an initially cracked material (deformations).

angle

Crack angle in the (X-Y) plane, in degrees, in the case of a plane stress analysis.

v1x, v1y, v1z

Components of the vector defining direction 1.

$v2x, v2y, v2z$

Components of the vector defining direction 2.

$v3x, v3y, v3z$

Components of the vector defining direction 3.

The model takes into account the anisotropy induced by the cracking.

The opening and closing of cracks is managed by the model.

For an axisymmetric or three-dimensional analysis, the user can enter different characteristics for the three directions.

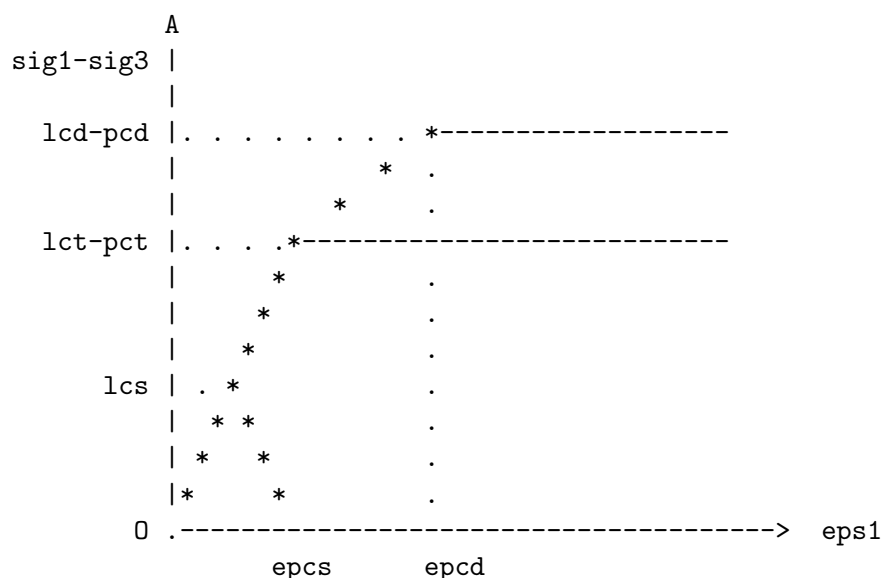
In the case of an initially cracked material, one can input the opening of cracks by means of initial deformations along the cracked direction.

### - 3) Data relative to shear damage:

Triaxial tests, carried out at different confinement levels, are necessary to determine the various parameters of the model. The results are then linearized and entered onto the diagram (sig1-sig3, eps1). The user may distinguish two different domains:

- a) brittle behaviour corresponding to the confinement levels, i.e. low sig3. This behaviour can be schematized by a decreasing branch and a negative work-hardening.
- b) ductile behaviour corresponding to the high confinement levels, i.e. high sig3. They can be schematized by a decrease in the elastic modulus and the appearance of irreversible strains and work-hardening.

Hence, the existence of a threshold stress of confinement, sig3, has been assumed. It corresponds to the border between the two domains: sig3 = PCT.



"LCS" lcs      "EPCS" epcs

```
< "LBIC" lbic > or < "LCT" lct "PCT" pct >
                  < "LCD" lcd "PCD" pcd "EPCD" epcd >
```

1cs

Uniaxial compression limit.

epcs

Strain at rupture in uniaxial compression.

**lbic**

Limit in biaxial compression in the case of a plane stress analysis.

1ct

Compression limit under a confinement pressure equal to the threshold confinement value (sig3).

pct

Threshold confinement pressure.

1cd

Compression limit under the pressure of ductile confinement.

pcd

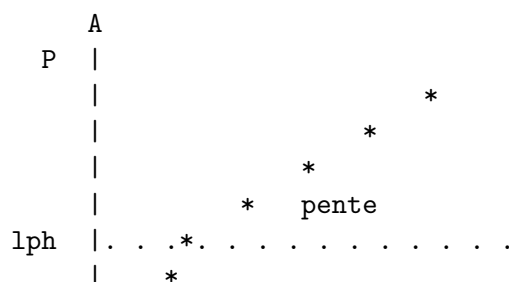
Pressure of ductile confinement.

epcd

Strain corresponding to the beginning of the perfectly plastic behaviour in the ductile domain.

- 4) Data relative to damage due to hydrostatic pressure:

A test is carried out where the sample is submitted to a hydrostatic pressure. The results are then linearized and entered onto the diagram: (P,  $Dv/v$ )





"LPH" lph    "PENT" pente

lph

Limit under hydrostatic pressure.

pente

Slope of the plastic branch on the diagram.

### Outputs:

The different components of the ECR table are as follows:

ECR(1): hydrostatic pressure

ECR(2): Von Mises criterion

ECR(3): equivalent plastic strain

ECR(4): crack angle in the (X-Y) plane (in degrees)

ECR(5): yield limit in traction along direction 1

ECR(6): yield limit in traction along direction 2

ECR(7): yield limit in traction along direction 3

ECR(8): crack opening in direction 1

ECR(9): crack opening in direction 2

ECR(10): crack opening in direction 3

ECR(11): X component of the vector defining direction 1

ECR(12): Y component of the vector defining direction 1

ECR(13): Z component of the vector defining direction 1

ECR(14): lambda(1) damage due to hydrostatic pressure

ECR(15): lambda(2) damage due to the steady ductile Drucker criterion

ECR(16): lambda(3) damage due to Von Mises criterion with hardening

ECR(17): lambda(4) damage due to the steady brittle Drucker criterion

ECR(18): lambda(5) damage due to the brittle Drucker criterion with hardening

ECR(19): index of the damage criterion (0: no shear, 1: ductile shear, 2: brittle shear, 3: both).

ECR(19): crack index (0: no crack, 1: one crack only, 2: two cracks, 3: three cracks).

**Default values for an ordinary concrete:**

All values are given in S.I. units.

**- 1) Generic data:**

RO	=	2.400E+03	Kg / m <sup>3</sup>
YOUN	=	37000E+06	Pa
NU	=	0.2100000	
ALPH	=	1.200E-05	
PREC	=	1.000E-03	

**- 2) Data for the traction damage:**

BETA	=	0.1000000	
LTR	=	4.440E+06	Pa
EPTR	=	3.600E-04	

**- 3) Data for shear damage:**

LCS	=	44.400E+06	Pa
EPCS	=	1.200E-02	
LBIC	=	111.000E+06	Pa
LCT	=	243.312E+06	Pa
PCT	=	71.040E+06	Pa
LCD	=	255.406E+06	Pa
PCD	=	79.920E+06	Pa
EPCD	=	6.000E-02	

**- 4) Data for the hydrostatic pressure damage:**

LPH	=	134.887E+06	Pa
PENT	=	7088.120E+06	Pa

**7.7.14 CONCRETE: DYNAR LMT (BLMT)****Object:**

Isotropic visco-damage and viscoplastic concrete material.

**References:**

- Gatuingt F. and Pijaudier-Cabot G., **Coupled damage and plasticity modelling in transient dynamic analysis of concrete**, *Int. J. Numer. Anal. Meth. Geomec.*, Vol 26, pp 1–24, 2002.
- Gatuingt F., **”Modèle de comportement BETON DYNAR LMT”**. Internal Report.

**Syntax:**

```
"BLMT"  "RO" rho  "YOUN" young  "NU" nu  "FO" f0
         "Q1" q1  "Q2" q2  "Q3" q3  "SGM0" sigM0  "XN" n
         "NVP" nvp  "MVP" mvp  "K" k  "MDT" mDt  "NDT" nDt
         "MDC" mDc  "NDC" nDc  "ED0" epsD0
         "AC" ac  "BC" bc  "AT" at  "BT" bt  /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

f0

Initial porosity of the concrete (0.3)

q1

Parameter of the modified Gurson plasticity criterion (0.5 to 2.)

q2

Parameter of the modified Gurson plasticity criterion (0.5 to 2.)

q3

Parameter of the modified Gurson plasticity criterion (0.5 to 2.)

sigM0

Resistance of the cement paste without pores (70 Mpa)

n

---

	Exponent of the viscoplasticity threshold (15.)
nvp	Parameter of the Perzyna type viscoplasticity (1.5)
mvp	Parameter of the Perzyna type viscoplasticity (1.D-2)
k	Influence the porosity evolution (15 to 60)
mDt	Tension damage viscosity parameter (0.5D-4)
nDt	Tension damage viscosity parameter (5.)
mDc	Compression damage viscosity parameter (0.5D-3)
nDc	Compression damage viscosity parameter (20.)
epsD0	Strain tension threshold (1.D-04)
ac	Parameter for the compression (3000)
bc	Parameter for the compression (4.)
at	Parameter for the tension (20000)
bt	Parameter for the tension (1.6)

**Comments:**

1/ - BE CAREFUL the initial porosity influence the real young modulus  
 $K_m = \text{YOUNG} / (3 * (1 - 2 * \text{NU}))$   
 $G_m = \text{YOUNG} / (2 * (1 + \text{NU}))$

2/ - Compressibility and shear moduli with porosity f (Mori-Tanaka)  
 $K_{\text{poro}} = 4 * X_{K_m} * X_{G_m} * (1 - f) / (4 * X_{G_m} + 3 * X_{K_m} * f)$   
 $G_{\text{poro}} = X_{G_m} * (1 - f) / (1 + f * (6 * X_{K_m} + 12 * X_{G_m}) / (9 * X_{K_m} + 8 * X_{G_m}))$



3/ - Plasticity criterion FNT:

$$F = 3 \cdot J_2(\text{SIG}) / \text{SGM}^{**2} + 2 \cdot Q1 \cdot f \cdot \cosh(Q2 \cdot I1 / 2 \cdot \text{SGM}) - (1 + (Q3 \cdot f)^{**2})$$

4/ - Plastic strain evolution:

$$\text{EPSP} = 1 / (1 - D) \cdot (\text{FNT} / \text{MVP})^{**\text{NVP}} \cdot d\text{FNT} / d\text{SIG}$$

5/ - Porosity evolution:

$$Df = K \cdot f / (1 - f) \cdot (\text{FNT} / \text{MVP})^{**\text{NVP}}$$

$$f(t+dt) = f(t) + df$$

6/ - Damage threshold function in tension and compression:

$$\text{FDi} = (\text{EPSE} - \text{ED0} - 1 / \text{Ai} \cdot (\text{Di} / (1 - \text{Di}))^{**}(1 / \text{Bi}))$$

7/ - Damage evolution in tension and compression:

$$\text{Di} = (\text{FDi} / \text{MDi})^{**\text{NDi}}$$

### Outputs:

The components of the ECR table are as follows:

- ECR(1) : pressure
- ECR(2) : Von Mises criterion
- ECR(3) : Isotropic damage variable
- ECR(4) : Material porosity
- ECR(5) : xx plastic strain
- ECR(6) : yy plastic strain
- ECR(7) : zz plastic strain
- ECR(8) : xy plastic strain
- ECR(9) : yz plastic strain
- ECR(10): zx plastic strain
- ECR(11): Stress in the matrix without pores
- ECR(12): Tension damage variable
- ECR(13): Compression damage variable
- ECR(14): Mazars threshold

**7.7.15 BPEL: MODEL FOR PRESTRESSING CABLE-CONCRETE FRICTION****Object:**

This material allows modelling friction between a prestressing cable and concrete according to BPEL rools (Prestressed Concrete with Borderlines). In French, BPEL stands for Beton Precontraint aux Etats Limites. This is a particular Coulomb-type friction law where the friction force threshold depends on tension in the cable. At each time step, the tension in a cable node is calculated first (mean tension between those in two cables elements using the considered node), then the friction force is calculated and compared with a threshold.

**Syntax:**

```
"BPEL" "FRLI" phil "FRCO" phic /LECTURE/
```

**phil**

Friction coefficient for rectilinear motion, by unit length (1/m)

**phic**

Friction coefficient for curvilinear motion, by unit angle (1/rad)

**LECTURE**

List of the elements concerned.

**Comments:**

This material can be used with RNFR element (nonlinear frictional spring) only.

**Outputs:**

The components of the ECR table are as follows:

ECR(1): Tangential friction force.

ECR(2): Total relative tangential displacement between cable and concrete.

ECR(3): State indicator: 0 if sliding, 1 if adherence.

**7.7.16 CONCRETE: MAZARS-LINEAR ELASTIC LAW WITH DAMAGE****Object:**

Isotropic linear elastic with a modified Mazars damage for concrete and brittle rupture materials.

**References:**

1- Jacky MAZARS, "Application de la mécanique de l'endommagement au comportement non linéaire et à la rupture du béton de structure", Thèse de doctorat, Université Pierre et Marie Curie - Paris 6, 1984.

2- Yann CHUZEL-MARMOT, "Caractérisation expérimentale et simulation numérique d'impacts de glace à haute vitesse", Thèse de doctorat, Université MEGA de Lyon - INSA Lyon, 2009.

**Syntax:**

```
"MAZA" "RO"    rho  "YOUN" young  "NU"    nu    "EPSD" epsd
      "DCRI" dcri "AT"    at      "AC"    ac    "BT"    bt
      "BC"    bc  "LCAR" lcar    "CSTA" csta "DCOE" dcoe
      "VCRI" vcri "VIMP" vimp                                     /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

epsd

Initial strain threshold.

dcri

Critical value of damage (=1 per default).

at

Parameter of the tension law (asymptote of the curve stress-strain)

ac

Parameter of the compression law (asymptote of the curve stress-strain)

bt

Parameter of the tension law (shape of the curve stress-strain)

**bc**

Parameter of the compression law (shape of the curve stress-strain)

**lcar**

Length parameter of the delay-damage

**csta**

Parameter of the delay-damage (=1 per default)

**dcoe**

Exponent of the sensitivity to the strain rate in tension ( $=\frac{1}{3}$  per default)

**vcrit**

Critical velocity in tension (=1 per default)

**vimp**

Velocity impact of the body (or strain rate if it's not an impact)

### Comments:

You can deactivate the delay effect with a negative value for the parameter **lcar**.

You can also deactivate the damage (so you obtain a linear material) with a negative value for the parameter **epsd**.

### Outputs:

The components of the ECR table are as follows:

ECR(1) : Pressure

ECR(2) : Von Mises criterion

ECR(3) : Equivalent deformation

ECR(4) : Global Damage

ECR(5) : Level "traction/compression"

ECR(6) : Strain rate

ECR(7) : Threshold damage

ECR(8) : Damage in traction

ECR(9) : Damage in compression

ECR(10): Factor of dynamic amplification in traction

ECR(11): Bc parameter eventually corrected

### 7.7.17 DADC: Dynamic Anisotropic Damage Concrete

#### Object:

Concrete material with induced anisotropic damage represented by one damage variable and modelling biaxial behaviour.

#### Reference:

Armand Leroux, *Modèle multiaxial d'endommagement anisotrope: Gestion numerique de la rupture et application à la ruine des structures en béton armé sous impacts*. Thèse LaMSID-UMR EDF/CNRS/CEA (2012)[[863](#)]

#### Syntax:

```
"DADC"  "RO" rho      "YOUN" young    "NU" nu      "SIGT" sigyt
        "SIGC" sigyc <"SGBC" sigybc>  "ALPH" alpha
        "BETA" beta  "BT" bt      "DC" dc
        <"XINF" xinf>      <"BV" bv>      <"DTFI" dtfi>
        <"TCS" tcs>      /LECTURE/
```

rho

Density

young

Young's modulus

nu

Poisson's ratio

sigyt

Elastic limit for the tension

sigyc

Elastic limit for the compression in absolute value

sigybc

Elastic limit for the bi-compression in absolute value. Default value is taken from Kupfer diagram as  $1.1 \cdot SIGC$ .

alpha

Damage parameter ALPH. This parameter allows to modify peak values in tension and compression strengths.

beta

Damage parameter BETA. This parameter allows to modify the post-peak behaviour in compression and bi-compression

**bt**

Parameter of the function  $b(T_x)$ . It is used for Hillerborg regularization.

**dc**

Critical value of the damage for the numerical control of rupture. (0.9 to 1)

*Optional parameters*

**dinf**

Delay damage parameter (suggested value: 50000. s-1)

**bv**

Delay damage parameter (suggested value: 1.)

**dtfi**

Activating calculation of time step in the behaviour law with a value of first time step (recommended value, 1E-8 s.). The parameter is considered when the option "PAS AUTO" is used.

**tcs**

Formulation of the selected function  $b(T_x)$  (1: Formulation TCS1(default), 2: Formulation TCS2 )

## Outputs:

The components of the ECR table are as follows:

ECR(1) : pressure  
 ECR(2) : Von Mises criterion  
 ECR(3) : Damage Dxx  
 ECR(4) : Damage Dyy  
 ECR(5) : Damage Dzz  
 ECR(6) : Damage Dxy  
 ECR(7) : Damage Dyz  
 ECR(8) : Damage Dzx  
 ECR(9) : Rotation matrix for the eigenvector basis damage matrix (xx)  
 ECR(10): Rotation matrix for the eigenvector basis damage matrix (xy)  
 ECR(11): Rotation matrix for the eigenvector basis damage matrix (xz)  
 ECR(12): Rotation matrix for the eigenvector basis damage matrix (yx)  
 ECR(13): Rotation matrix for the eigenvector basis damage matrix (yy)  
 ECR(14): Rotation matrix for the eigenvector basis damage matrix (yz)  
 ECR(15): Rotation matrix for the eigenvector basis damage matrix (zx)  
 ECR(16): Rotation matrix for the eigenvector basis damage matrix (zy)  
 ECR(17): Rotation matrix for the eigenvector basis damage matrix (zz)  
 ECR(18): Critical state damage flag  
 ECR(19): Damage rate  
 ECR(20): Equivalent effective stress  
 ECR(21): 1st eigen value basis damage matrix  
 ECR(22): 2nd eigen value basis damage matrix

ECR(23): 3rd eigen value basis damage matrix  
ECR(24): The biggest three eigen values basis damage matrix  
ECR(25): Proposed time step  
ECR(26): Filtered stress tensor after five time steps  
ECR(27): Time step first flag  
ECR(28): Estimation error flag (0=ok,1=error)  
ECR(29): stress triaxiality  
ECR(30): Component of filtered stress tensor (xx)  
ECR(31): Component of filtered stress tensor (yy)  
ECR(32): Component of filtered stress tensor (zz)  
ECR(33): Component of filtered stress tensor (xy)  
ECR(34): Component of filtered stress tensor (yz)  
ECR(35): Component of filtered stress tensor (zx)  
ECR(36): Number of times that the damage criterion (for the calculation of the time step in the behaviour law) is not respected  
ECR(37): Largest components (absolute values) of the strain rate tensor

### 7.7.18 DPDC: Dynamic Plastic Damage Concrete

#### Object:

DPDC A three-invariant cap model with isotropic damage for concrete material. Perfect plasticity with isotropic hardening cap model, brittle and ductile damage, crack closing and strain rate effect.

The version 9 of DPDC can be used in shell elements, with layers or not. It is then possible to take transverse steel reinforcement into account, in an averaged fashion. To this aim, the transverse strain is computed from a fixed-point loop.

#### Reference:

Damage Plastic Model for Concrete Failure Under Impulsive Loadings, Daniel Guilbaud, XIII International Conference on Computational Plasticity - Fundamentals and Applications, COMPLAS XIII (2015), E. Oñate, D.R.J. Owen, D. Peric and M. Chiumenti (Eds)

#### Comments:

All values must be given in SI units.

#### Syntax:

```
"DPDC"  "RO" rho      "YOUN" young  "NU" nu      "FC" fc
"DAGG" dagg
<"GFT" gft      "GFC" gfc      "GFS" gfs>
<"PWRC" pwrc>  <"PWRT" pwrt>
<"B" b>        <"D" d>
<"OVEC" overc> <"OVET" overt> <"SRAT" srate>
<"R" r>        <"XO" xo>      <"W" w>
<"D1" d1>      <"D2" d2>      <"PMOD" pmod>
<"TXCA" txca "TXCT" txct "TXCL" txcl "TXCB" txcb>
<"TXEA" txea "TXET" txet "TXEL" txel "TXEB" txeb>
<"FTR" ftr  "FBCR" fbcr "I1CR" i1cr "RJCR" rjcr>
<"NC" nc "NOC" noc> <"NT" nt "NOT" not>
<"REPW" repow> <"RECO" recov>
<"PRED" pred> <"COPP" copp> <"EXCT" excent>
<"LC" lc "DINF" dpinf>
<"VERS" vers>
<"EFVI">
<"EFVN">
<"EROD" <"ENDT" endt> <"ENDC" endc> <"DVOL" dvol> <"AVOL" avol>>
<"REIN" <"ROST" rost> <"YGST" ygst> <"SIST" sist>>
/LECTURE/
```

rho

Density

youn



	Young's modulus
<b>nu</b>	
	Poisson's ratio
<b>fc</b>	
	Uniaxial compressive strength (Pa)
<b>dagg</b>	
	Maximum aggregate size (m)
<b>vers</b>	
	Version number (8 = old version intended to be replaced by version 9) (9 = new version). Version 9 is used by default.
<i>Optional parameters</i>	
<b>gft</b>	
	Tensile fracture energy at $fc = 10\text{MPa}$ (interpolated as a function of the maximum aggregate size) (J/m <sup>2</sup> )
<b>gfc</b>	
	Compressive fracture energy at $fc = 10\text{MPa}$ (default value: 200 gft) (J/m <sup>2</sup> )
<b>gfs</b>	
	Shear fracture energy at $fc = 10\text{MPa}$ (default value: gft) (J/m <sup>2</sup> )
<b>pwr c</b>	
	Shear-to-compression transition parameter (default value: 1.) (advised value: 3.3) (without unit)
<b>pwr t</b>	
	Shear-to-tension transition parameter (default value: 1.) (without unit)
<b>b</b>	
	Ductile shape softening parameter (default value: 100) (without unit)
<b>d</b>	
	Brittle shape softening parameter (default value: 0.1) (without unit)
<b>over c</b>	
	Maximum overstress allowed in compression (interpolated as a function of the material strength in compression) (Pa)
<b>over t</b>	
	Maximum overstress allowed in tension (interpolated as a function of the material strength in compression) (Pa)

**srate**

Ratio of effective shear stress to tensile stress fluidity parameter (default value: 1.) (without unit)

**r**

Cap aspect ratio (default value: 5.) (advised value: 1.) (without unit)

**xo**

Cap initial location (interpolated as a function of the material strength in compression) (advised value: -10.E6) (Pa)

**w**

Maximum plastic volume compaction (default value: 0.05) (without unit)

**d1**

Linear shape parameter of the cap (default value: 2.5E-10 Pa<sup>-1</sup>)

**d2**

Quadratic shape parameter of the cap (default value: 3.49E-19 Pa<sup>-2</sup>)

*parameters for meridians (for all versions until 7)*

**txca**

TXC surface constant term (TXC: triaxial compression) (interpolated as a function of the material strength in compression) (Pa)

**txct**

TXC surface linear term (interpolated as a function of the material strength in compression) (without unit)

**txcl**

TXC surface nonlinear term (interpolated as a function of the material strength in compression) (Pa)

**txcb**

TXC surface exponent (interpolated as a function of the material strength in compression) (Pa<sup>-1</sup>)

**txea**

TXE surface constant term (TXE: triaxial extension) (interpolated as a function of the material strength in compression) (without unit)

**txet**

TXE surface linear term (interpolated as a function of the material strength in compression) (Pa<sup>-1</sup>)

**txel**

TXE surface nonlinear term (interpolated as a function of the material strength in compression) (without unit)

txeb

TXE surface exponent (interpolated as a function of the material strength in compression) ( $\text{Pa}^{-1}$ )

*parameters for meridians (for versions 8 and 9)*

ftr

ratio  $f_t/f_c$  with  $f_t$  uniaxial tensile strength (default value: 0.1)

Be aware! ftr should be such that:  $0.06 < \text{ftr} < 0.11$

fbcr

ratio  $f_{bc}/f_c$  with  $f_{bc}$  biaxial compressive strength (default value: 1.16)

i1cr

ratio  $i_1/f_c$  : horizontal coordinate of a point belonging to the compressive meridian (default value: -8.806)

rjcr

ratio  $\sqrt{J_2}/f_c$  : vertical coordinate of the same point (default value: 2.4985)

pmod

Modify moderate pressure softening parameter (default value: 0.) (without unit)

nc

Rate effects power for uniaxial compressive strength (default value: 0.78) (without unit)

noc

Rate effects parameter for uniaxial compressive strength (interpolated as a function of the material strength in compression). Default unit:  $\text{s}^{-0.22}$ . The unit depends on the value of nc

nt

Rate effects power for uniaxial tensile strength (default value: 0.48) (without unit)

not

Rate effects parameter for uniaxial tensile strength (interpolated as a function of the material strength in compression). Default unit:  $\text{s}^{-0.52}$ . The unit depends on the value of nt

repow

Power which increases fracture energy with rate effects (default value: 1.) (without unit)

recov

Option to recover stiffness in compression from tensile damage (default value: 0.) (without unit)

**pred**

Damage level for predamaged concrete (default value: 0.) (without unit)

**copp**

Coefficient for potential surface (default value: 1. associated plasticity)

**excent**

Constant excentricity (default value: excentricity function of J1)

**lc**

Characteristic length for damage (m) (first parameter for EFVN option)

**dpinf**

Maximum rate of damage (1/s) (second parameter for EFVN option)

**EFVI**

Strain rate effect option (default value : strain rate effect excluded)

**EFVN**

Bounded damage rate effect option available with version 9 only

**EROD**

Mandatory keyword to introduce different failure criteria (damage, plastic strain). The keyword "EROS" must be added in the problem description of the data file to activate the "erosion" algorithm of the code. When one uses EROD, it is required to indicate at least one "erosion" criterion either devol or avol (version 9 only). When the two criterions are indicated, both are taken into account.

**endt**

Brittle damage threshold

**endc**

Ductile damage threshold

**dvol**

Volumetric strain threshold. The element gets eroded when the volumetric strain reaches 1+dvol.

**avol**

Optional keyword (version 9 only): Absolute volumetric strain threshold. The element gets eroded when the volumetric strain reaches 1+avol.

**REIN**

Mandatory keyword to use DPDC with shell elements, layered or not. Triggers the use of the plane stress version of DPDC and supplies the following characteristics of the transverse steel reinforcement.

**rost**

Transverse reinforcement ratio (without unit)

**ygst**

Young modulus of the steel used (Pa)

**sist**

Yield stress of the steel used (Pa)

### Outputs:

The components of the ECR table are as follows:

ECR(1) : Pressure  
ECR(2) : Von Mises criterion  
ECR(3) : Equivalent plastic strain  
ECR(4) : Cube root of initial element volume (if version8)  
ECR(5) : Lode angle  
ECR(6) : Total variation of the isotropic hardening parameter  
ECR(7) : Volumetric strain  
ECR(8) : Plastic volumetric strain  
ECR(9) : Ductile damage parameter  
ECR(10): Brittle damage parameter  
ECR(11): Ductile damage threshold  
ECR(12): Brittle damage threshold  
ECR(13): Current damage (not used in version 9)  
ECR(14): Initial damage threshold in compression  
ECR(15): Initial damage threshold in tension  
ECR(16): filtered effective strain rate  
ECR(17-22): Components of elastoplastic stress tensor  
ECR(23-28): Components of viscoplastic stress tensor

ECR(29-34): Back stress if version 2  
else:

ECR(29): Effective strain rate  
ECR(30): Triaxiality  
ECR(31): Static ductile damage threshold  
ECR(32): Static brittle damage threshold

In addition, for shell elements:

ECR(57): Transverse steel plastic strain  
ECR(58): True stress in the transverse reinforcing bars

### 7.7.19 DAMAGE

**Object:**

This option allows to associate to the materials VON MISES ISOTROPE and VON MISES PARFAIT different damage laws, and to request the calculation of several fracture criteria. Now, only one criterion (Tuler-Butcher) is available.

**Syntax:**

```
"CRIT"  $[ "TULE" <"SIGL" sigl > <"EPSL" epsl > <"TAUL" taul >
          "SIGS" sigs      "LAMB" lamb  <"KER"  ker  > ]$
/LECTURE/
```

"CRIT"

Indicates that the calculation of different damage criteria is required.

"TULE"

The TULER-BUTCHER's criterion is selected.

sigl

Maximum principal stress criterion.

epsl

Maximum volumetric deformation criterion.

taul

Octahedral shear stress criterion.

sigs

First parameter of the Tuler-Butcher law.

lamb

Second parameter of the Tuler-Butcher law.

ker

Third parameter of the Tuler-Butcher law.

LECTURE

List of the concerned elements.

**Comments:**

The element types accepting these materials are: in 2D elements TRIA, CAR1 and CAR4 and in 3D elements CUBE, CUBE6, CUBE8, PRIS, TETR and PRI6.

Currently the damage model is only available in association with the materials Isotropic Von Mises, Steinberg-Guinan and dynamic Von Mises.

The isotropic Von Mises material must appear first in the input file, before the calculation of the damage criteria, if any, and one of the two damage laws, if necessary.

The Tuler-Butcher is given by the following expression where  $\sigma_1, \sigma_2, \sigma_3$  representing the principal stresses:

$$\int_0^t (Max(\sigma_1, \sigma_2, \sigma_3) - \sigma_s)^\lambda dt < \text{ker}$$

The results stored in the ECR table (5 values) are, according to the material type:

Tuler-Butcher :

- ECR(1) = Maximum principal stress
- ECR(2) = Maximum principal deformation
- ECR(3) = Octahedral shear stress
- ECR(4) = Volume deformation
- ECR(5) = Tuler-Butcher criterion

### Example:

A criterion "LOI 3", is associated with the principal material "LOI 1".

The corresponding data will be for example:

```

LOI 1    VMIS ISOT R0 7800.  YOUN 74020E6  NU .3  ELAS 350E6
          ENDO 3
          TRAC 4
              350.E6          .472845E-2
              476.26E6        7.2835E-2
              518.51E6        15.700E-2
              538.03E6        21.607E-2
              550.24E6        26.083E-2
          LECT TOUS

LOI 3    CRIT TULE  SIGS 1E7 LAMB 1.0
          LECT TOUS

```

**7.7.20 EOBT: ANISOTROPIC DAMAGE OF CONCRETE (EDF)****Object:**

Concrete material with induced anisotropic damage.

**Reference:**

V. Godard, Modélisation de l'endommagement anisotrope du béton avec prise en compte de l'effet unilatéral : Application à la simulation numérique des enceintes de confinement, Thèse de l'Université Paris VI, 2005.

M. Bottoni, Loi de comportement ENDO\_ORTH\_BETON, Manuel de référence de Code\_Aster, R7.01.09.

**Syntax:**

```
"EOBT"  "R0" rho  "YOUN" young  "NU" nu  "K0" k0  
        "K1" k1  "K2" k2  "ECRB" ecrb  "ECD" ecd  
        < "DC" dc >  < "DM" dm >  /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

k0

Threshold in stress for the tension.

k1

Parameter for the threshold in stress in compression.

k2

Parameter for the threshold in stress in compression.

ecrb

Parameter driving the evolution of the loading surface while the damage tensor B is growing.

ecd

Parameter driving the evolution of the loading surface while the damage scalar d is growing.



**Optional parameters:****dc**

Limit value for the eigenvalues of the damage tensor  $B$  and for the damage scalar  $d$ . When this limit is reached, the material is considered to be broken and damage can not grow anymore. The damage is set to  $dm$ . By default,  $dc$  is set to 0.999

**dm**

Imposed value for damage when it reaches its limit value. By default,  $dm$  is set to 0.999

**Outputs:**

The components of the ECR table are as follows:

ECR(1) : pressure  
ECR(2) : Damage in compression  $D$   
ECR(3) : Damage  $D_{xx}$   
ECR(4) : Damage  $D_{yy}$   
ECR(5) : Damage  $D_{zz}$   
ECR(6) : Damage  $D_{xy}$   
ECR(7) : Damage  $D_{yz}$   
ECR(8) : Damage  $D_{zx}$   
ECR(9) : Rotation matrix for the eigenvector basis damage matrix (xx)  
ECR(10): Rotation matrix for the eigenvector basis damage matrix (xy)  
ECR(11): Rotation matrix for the eigenvector basis damage matrix (xz)  
ECR(12): Rotation matrix for the eigenvector basis damage matrix (yx)  
ECR(13): Rotation matrix for the eigenvector basis damage matrix (yy)  
ECR(14): Rotation matrix for the eigenvector basis damage matrix (yz)  
ECR(15): Rotation matrix for the eigenvector basis damage matrix (zx)  
ECR(16): Rotation matrix for the eigenvector basis damage matrix (zy)  
ECR(17): Rotation matrix for the eigenvector basis damage matrix (zz)  
ECR(21): 1st eigen value of the damage tensor  $B$   
ECR(22): 2nd eigen value of the damage tensor  $B$   
ECR(23): 3rd eigen value of the damage tensor  $B$

### 7.7.21 ENGR: ELASTIC GRADIENT DAMAGE MATERIAL

#### Object:

This section describes an elastic-damage material with gradient regularization. The development is still in progress and a more detailed presentation can be found in [951], [952].

This model can be used to predict crack initiation and propagation in a quasi-brittle medium (such as glass or concrete) under dynamic loading conditions. In particular, no plasticity is currently accounted for in this model. It can be seen as a variational approach to fracture in the sense of [Francfort and Marigo 1998, Revisiting brittle fracture as an energy minimization problem]. Crack nucleation, kinking, branching, coalescence or arrest can be automatically predicted through energy minimization. It is known that traditional approaches (with or without X-FEM numerical schemes) in fracture mechanics may fail in the case of crack initiation from a perfectly regular domain or complex crack topological changes. The variational approach can thus be considered as a unified and complete framework of fracture.

An additional scalar nodal field  $0 \leq \alpha \leq 1$ , called damage, is introduced to the model. This field depicts a continuous transition between the undamaged part  $\alpha = 0$  and the crack  $\alpha = 1$ . Spurious mesh dependency observed in traditional damage mechanics is suppressed by gradient regularization  $\nabla \alpha$ . As a consequence, a material characteristic internal length  $\ell$  naturally appears, which determines the size of the damage process zone. This parameter is linked to the maximal tensile stress  $\sigma_m$  that can be supported by the material.

Only two mandatory fracture-type material parameters need to be entered. One is the fracture toughness  $G_c$  defined as the energy needed to create a crack of unit area. Through Irwin's formula this quantity  $G_c$  can be related to the criterion in stress intensity factors  $K_{IC}$ . Another parameter is the internal length  $\ell$  or equivalently the maximal tensile stress  $\sigma_m$ .

Two resolution methods are available.

In the first one (called by default), the damage problem is solved with an implicit resolution method. From a computational point of view, we need to solve at every time step a bound-constrained (quadratic or convex) minimization problem for damage. For that reason, the parallel linear algebra library PETSc is used to manipulate the Hessian matrix and various vectors. For various options of PETSc used for this material ENGR, it is advised to refer to 12.23.

In the second one (called if "MOBI" parameter is given), the damage problem is solved with an explicit resolution method. This method is numerically more efficient but results depend on the value of "MOBI" parameter. There isn't currently a formula to get the optimal value for "MOBI".

#### Syntax:

```
"ENGR"      "RO"    rho    "YOUN" young  "NU"    nu    "GC"    gc
            | [ "ELL"  ell ; "SIGM" sigm ] |
            < "LAW"  law  "TC"   tc      "AC"    ac    "MOBI" mob > /LECTURE/
```

#### Mandatory parameters:

rho

Density  $\rho$ .

young

Young's modulus  $E$ .

nu

Poisson's ratio  $\nu$ .

gc

Fracture toughness  $G_c$ .

**ONE additional mandatory fracture parameter from two possibilities:**

ell

Material characteristic (internal) length  $\ell$ .

sigm

Maximal tensile stress  $\sigma_m$ .

**Optional parameters:**

law

Damage constitutive law describing local stiffness degradation due to damage  $a(\alpha)$  and local damage dissipation evolution function  $w(\alpha)$ . 3 choices are currently implemented.

- Integer 1: this *default* damage law ensures a brittle material behavior by providing an elastic limit  $\sigma_c$  which equals to the maximal stress  $\sigma_m$  possibly entered by the user. The computational cost of this law is also smaller. *We recommend using this damage constitutive law.*

$$a(\alpha) = (1 - \alpha)^2, \quad w(\alpha) = w_1 \alpha$$

where  $w_1 = 3G_c/(8\ell)$ .

- Integer 2: this damage law is widely used in phase-field modeling for fracture mechanics.

$$a(\alpha) = (1 - \alpha)^2, \quad w(\alpha) = w_1 \alpha^2$$

where  $w_1 = G_c/(2\ell)$ .

- Integer 3: this damage law named PB depicts a *Perfectly Brittle* material. During a homogeneous traction test, the stress will immediately drops to zero when the elastic limit is reached.

$$a(\alpha) = (1 - \alpha)^2, \quad w(\alpha) = w_1 (1 - (1 - \alpha^2))$$

where  $w_1 = G_c/(\pi\ell)$ .

tc

This parameter determines the tension-compression asymmetry formulation. For brittle materials such as glass or concrete, the material can be easily damaged or cracked under tension. It is no longer the case under compression. Currently 6 formulations are available.

- Integer 1: by *default* we do not distinguish tension and compression in this gradient damage model. It means cracks may be developed in compressive zones. Use this formulation only if you are sure that material non-interpenetration will not occur.

- Integer 2: the formulation proposed in [Amor et al. 2009, Regularized formulation of the variational brittle fracture with unilateral contact: Numerical experiments] stipulates that under compression flagged by a negative strain trace  $\text{tr} \varepsilon < 0$ , only deviatoric part of the elastic energy contributes to damage.
- Integer 3: this formulation is slightly different from the 2nd model and is proposed in [Zouari et al. 2012, Prise en compte de la différence traction/compression pour les lois d'endommagement. Tests sur les structures en béton]. This formulation prohibits damage evolution as long as the strain trace is negative  $\text{tr} \varepsilon < 0$ .
- Integer 4: this formulation proposed in [Lancioni et al. 2009, The Variational Approach to Fracture Mechanics. A Practical Application to the French Panthéon in Paris] permits only deviatoric-type damage.
- Integer 5: [Miehe et al. 2010, A phase field model for rate-independent crack propagation] initially proposed this model. It uses the principal values of the strain tensor via eigen-decomposition and only positive ones contribute to damage. However this model presents a peculiar behavior under uniaxial compression, where homogeneous damage occurs accompanied by an increasing stress. For this reason, use the 6th model of [Freddi et al. 2010].
- Integer 6: this model is initially formulated by [Freddi et al. 2010, Regularized variational theories of fracture: A unified approach]. The problem with the [Miehe et al. 2010] model under compression is overcome. *We recommend using this formulation for all brittle materials.*

ac

This parameter can be used with the erosion mechanism to define a critical stiffness degradation. Its default value is  $10^{-3}$ . If this value is reached, the Gauss point will be considered as eroded.

mob

This parameter of "mobility", inverse of a viscosity, activates the explicit method to solve the problem of damage. This parameter affects the results, so it is necessary to provide an appropriate value. However, there is no method yet to determine this value. Work is ongoing on this subject.

## Outputs:

No public ECR components are available for output.

## 7.7.22 LINEAR MULTI-LAYER

### Object

This directive allows to define materials obtained by homogenisation through the thickness of different layers (or plies) each having a linear orthotropic behaviour.

### Syntax:

```
"MCOU"      | [  "BACON"      ibacon      ;
               "NBCOUCHE"  ...           /LECTURE/  ] |
```

For the user data option (NBCOUCHE) :

```
... "NBCOUCHE"  nbcouche  "ZMIN"  zmin

nbcouche times    :

|  "ZMAX"  zmax  "TETA"  teta  "ROCO"  roco  "YG1"  yg1  |
|  "YG2"   yg2   "NU12"  nu12  "G12"   g12                |
| < "G13"  g13 > < "G23"  g23 > "TERM"                |
```

### ibacon

Logical unit number of the BACON file from which the characteristics of this material will be read. Using this option implies the necessity to introduce the keyword "MBACON" in part A of the input file (see page A.30) in order to dimension the arrays used by this model.

### NBCOUCHE

The characteristics will be listed below.

### nbcouche

Numbers of layers of the composite.

### zmin

Minimum side of the first layer.

### zmax

Maximum side of the current layer.

### teta

Angle (in degrees) of the first vector of the orthotropy frame of the current layer with respect to the first side of the element.

roco

Density of the current layer.

yg1

Young's modulus along direction 1 of the current layer.

yg2

Young's modulus along direction 2 of the current layer.

nu12

Poisson's coefficient among directions 1-2.

g12

Shear modulus among directions 1-2.

g13

Shear modulus among directions 1-3.

g23

Shear modulus among directions 2-3.

TERM

Indicates that the data for layer i are terminated.

LECTURE

List of the concerned elements.

### Comments:

When the BACON option is used, EUROPLEXUS reads the numbers of the elements associated with this material directly from the BACON file: the /LECTURE/ procedure is redundant. Currently, one may read only one type of laminated material per calculation. On the other hand, EUROPLEXUS will write on the logical unit (ibacon+1):

- 1) the element number (1 value)
- 2) the angle (in degrees) between the first side and the first direction of the laminated (1 value)
- 3) the components of the symmetric matrices A, B and D  
( A(1,1), A(2,1),A(2,2), A(3,1),A(3,2),A(3,3)... )  
(3x6= 18 values).

For the NBCOUCHE option, the various layers must be described in growing order of z. In particular,  $z_{\max}(\text{couche}_i) = z_{\min}(\text{couche}_{i+1})$ .

The value  $z=0$  corresponds to the neutral fiber of the element. This material allows to take excentricity into account.

For the shells that consider transverse shears, i.e. DST3, Q4G4, Q4GR, Q4GS, it is necessary to give the values of G23 and G13.

**Outputs:**

The various components of the ECR table (values computed in the local reference of the shell element) are as follows:

Element COQ3:

- ECR(1) : Von Mises on the lower face of the shell
- ECR(2) : Von Mises on the upper face of the shell
- ECR(3) :  $-d^3w/dx^3$  at the integration point
- ECR(4) :  $-d^3w/dy^3$  at the integration point
- ECR(5) :  $-d^3w/dx^2dy$  at the integration point
- ECR(6) :  $-d^3w/dxdy^2$  at the integration point

Elements DKT3 and DST3:

- ECR(1) : Von Mises on the lower face of the shell
- ECR(2) : Von Mises on the upper face of the shell
- ECR(3) :  $d^2\beta_x/dx^2$  at the first integration point
- ECR(4) :  $d^2\beta_x/dy^2$  at the first integration point
- ECR(5) :  $d^2\beta_x/dxdy$  at the first integration point
- ECR(6) :  $d^2\beta_y/dx^2$  at the first integration point
- ECR(7) :  $d^2\beta_y/dy^2$  at the first integration point
- ECR(8) :  $d^2\beta_y/dxdy$  at the first integration point

Recall that the table of deformations EPST is composed by the following parameters (computed at the integration point):

- EPST(1) :  $du/dx$  (membrane deformation  $e_{xx}$ )
- EPST(2) :  $dv/dy$  (membrane deformation  $e_{yy}$ )
- EPST(3) :  $du/dy + dv/dx$  (membrane deformation  $2e_{xy}$ )
- EPST(4) :  $d\beta_x/dx$  ( $= -d^2w/dx^2$  if thin shell)
- EPST(5) :  $d\beta_y/dy$  ( $= -d^2w/dy^2$  if thin shell)
- EPST(6) :  $d\beta_y/dx + d\beta_x/dy$
- EPST(7) :  $2\epsilon_{xz}$  (eventually)
- EPST(8) :  $2\epsilon_{yz}$  (eventually)

**Example:**

We assume a composite formed by 6 layers regularly spaced on a thickness of 0.12 m. The corresponding data will be:

```
MCOUCH      NBCOUCHE  6          ZMIN -0.06
      ZMAX -0.04  TETA  5.  ROC0 2.5E3  YG1 40E9 YG2 20E9
                        NU12 0.2 G12 16.66666667E9 TERM
      ZMAX -0.02  TETA 36.  ROC0 2.5E3  YG1 40E9 YG2 25E9
                        NU12 0.2 G12 16.66666667E9 TERM
      ZMAX -0.00  TETA 48.  ROC0 2.5E3  YG1 40E9 YG2 20E9
                        NU12 0.2 G12 16.66666667E9 TERM
      ZMAX  0.02  TETA 135  ROC0 2.5E3  YG1 40E9 YG2 20E9
                        NU12 0.2 G12 16.66666667E9 TERM
      ZMAX  0.04  TETA 33.  ROC0 2.5E3  YG1 40E9 YG2 20E9
                        NU12 0.2 G12 16.66666667E9 TERM
      ZMAX  0.06  TETA 15.  ROC0 2.5E3  YG1 40E9 YG2 40E9
                        NU12 0.2 G12 16.66666667E9 TERM
                                LECT 3  4 5 TERM
```



### 7.7.23 CHANG-CHANG MULTI-LAYER MODEL

#### Object:

This directive allows to define composite materials using the CHANG-CHANG criterion, as described in:

A Progressive Damage Model of Laminated Composites  
Containing Stress Concentrations  
by F.-K. CHANG and K.-Y. CHANG  
in Journal of Composite Materials, Vol. 21, Sept. 1987.

#### Syntax:

```
"CHANG"      |[  "BACON"      ibacon      ;
                $  "NBCOUCHE"  ... PBASE .... /LECTURE/      $
```

For the user data option (NBCOUCHE) :

```
... "NBCOUCHE" nbcouche "ZMIN" zmin
```

```
nbcouche times :
```

```
| "ZMAX" zmax "TETA" teta "ROCO" roco "YG1" yg1 |
| "YG2" yg2 "NU12" nu12 "G12" g12 |
| "XT" xt "XC" xc "YT" yt "YC" yc |
| "SC" sc "AO" a0 "BETA" beta |
| "TERM"
```

Once described the layers, one gives 2 points in order to define a reference direction:

```
"PBASE" "LECTURE" nod1 nod2 "TERM"
```

#### ibacon

Logical unit number of the BACON file from which the characteristics of this material will be read. Using this option implies the necessity to introduce the keyword "MBACON" in part A of the input file (see page A.30) in order to dimension the arrays used by this model.

#### NBCOUCHE

The characteristics will be listed below in the main input file.

#### nbcouche

Number of layers in the composite.

**zmin**

Minimum side of the first layer.

**zmax**

Maximum side of the current layer.

**teta**

Angle (in degrees) of the first vector of the orthotropy frame of the current layer with respect to the reference direction.

**roco**

Density of the current layer.

**yg1**

Young's modulus along direction 1 of the current layer.

**yg2**

Young's modulus along direction 2 of the current layer.

**nu12**

Poisson's coefficient among directions 1-2.

**g12**

Shear modulus among directions 1-2.

**xt**

Traction limit along direction 1 of the orthotropy frame.

**xc**

Compression limit along direction 1 of the orthotropy frame.

**yt**

Traction limit along direction 2 of the orthotropy frame.

**yc**

Compression limit along direction 2 of the orthotropy frame.

**sc**

Shear limit 1-2 of the orthotropy frame.

**a0**

Critical area a0 of the CHANG-CHANG criterion.

**beta**

Weibull coefficient.

**TERM**

Indicates that the data for layer  $i$  are terminated.

`nod1,nod2`

Numbers of 2 nodes defining the reference direction.

LECTURE

List of the concerned elements.

### Comments:

When the BACON option is used, EUROPLEXUS reads the numbers of the associated elements directly from the BACON file: the procedure /LECTURE/ is redundant. For this material, the number of laminas is unlimited. However, one should give the adequate numbers in the dimensioning section of the input file: see the key-words MATE and ECRO in the DIMENSIONS directive.

For the NBCOUCHE option, the various layers must be described along increasing order of  $z$ . In particular,  $z_{\max}(\text{couche}_i) = z_{\min}(\text{couche}_{i+1})$ .

The value  $z=0$  corresponds to the neutral fiber of the element. This material allows to account for excentricity.

### Outputs:

This constitutive law computes the damages appearing in each plie of the laminated structure. To this end, it is necessary to define the damage parameters in each layer. In each ply, the ECR table is dimensioned at 10, and the main parameters are:

DIMENSION ECR(10,NPLIS)

ECR(2,ipli) : Von Mises of the ply

ECR(3,ipli) : Rupture criterion of the matrix in traction

ECR(4,ipli) : Rupture criterion of the matrix in compression

ECR(5,ipli) : Rupture criterion of the fiber, or fiber-matrix delamination.

**7.7.24 LINEAR ORTHOTROPY****Object:**

The directive is used to enter materials with a linear orthotropic behaviour into a coordinate system defined by the user. The model is described in: *Mécanique des Matériaux Solides* (J-Lemaitre, L-Chaboche. Ed: Dunod, 1986).

**Syntax:**

```
"ORTH"  "R0"  rho    "YG1"  yg1    "YG2"  yg2    "YG3"  yg3
          "NU12" nu12  "NU13" nu13  "NU23" nu23
          "G12"  g12   "G13"  g13   "G23"  g23      /LECTURE/
```

rho

Density of the material.

yg1

Young's modulus along direction 1.

yg2

Young's modulus along direction 2.

yg3

Young's modulus along direction 3.

nu12

Poisson's ratio between directions 1 and 2.

nu13

Poisson's ratio between directions 1 and 3.

nu23

Poisson's ratio between directions 2 and 3.

g12

Shear modulus between directions 1 and 2.

g13

Shear modulus between directions 1 and 3.

g23

Shear modulus between directions 2 and 3.

#### LECTURE

List of the elements concerned.

#### Comments:

This option may be repeated as many times as necessary.

The associated coordinate system is defined by the option "CORTHO" (see page C1.95) for the multilayer element CMC3.

The associated coordinate system is defined by the option "MORTHO" (see page C1.96) for the continuum elements in 3D and in plane strain.

The associated coordinate system is defined by the directive "COMPLEMENT" (pages C1.95 and C1.96):

- "COMPLEMENT" "CORTHO" for the shells;
- "COMPLEMENT" "MORTHO" for the continuum elements 3D and 2D plane strain and axisymmetric.

Verify that this material is available for your elements, by means of the tables of page C.100.

#### Outputs:

The different components of the ECR table are as follows, for the CMC3 element:

ECR(1): Von mises criterion on the lower face of the multilayer element CMC3.

ECR(2): Vom mises criterion on the upper face of the multilayer element CMC3.

The different components of the ECR table are as follows, for the continuum elements:

ECR(1): pressure.

ECR(2): Vom mises criterion.

**7.7.25 ORTS : LINEAR ORTHOTROPY (Local basis)****Object:**

Attention: The description of the material is not showing all capabilities of the material. The material allows erosion and has several more input and output variables.

The directive is used to enter materials with a linear orthotropic behaviour into a coordinate system defined by the user. The model is described in: *Mécanique des Matériaux Solides* (J-Lemaitre, L-Chaboche. Ed: Dunod, 1986).

Stress and strain are expressed in the user coordinate system.

**Syntax:**

```
"ORTS"  "R0"  rho    "YG1"  yg1    "YG2"  yg2    "YG3"  yg3
        "NU12" nu12  "NU13" nu13  "NU23" nu23
        "G12"  g12   "G13"  g13   "G23"  g23
        <"XT1"  xt1   "XT2"  xt2   "XT3"  xt3
        "XC1"  xc1   "XC2"  xc2   "XC3"  xc3
        "RST1" rst1  "RST2" rst2  "RST3" rst3
        "CRIT" icrit>          /LECTURE/
```

rho

Density of the material.

yg1

Young's modulus along direction 1.

yg2

Young's modulus along direction 2.

yg3

Young's modulus along direction 3.

nu12

Poisson's ratio between directions 1 and 2.

nu13

Poisson's ratio between directions 1 and 3.

nu23

Poisson's ratio between directions 2 and 3.

g12

Shear modulus between directions 1 and 2.

g13

Shear modulus between directions 1 and 3.

g23

Shear modulus between directions 2 and 3.

xt1

Failure criterion for tension in x-direction.

xt2

Failure criterion for tension in y-direction.

xt3

Failure criterion for tension in z-direction.

xc1

Failure criterion for compression in x-direction.

xc2

Failure criterion for compression in y-direction.

xc3

Failure criterion for compression in z-direction.

rst1

Failure criterion for shear in xy-direction.

rst2

Failure criterion for shear in yz-direction.

rst3

Failure criterion for shear in xz-direction.

icrit

Erosion criteria type (e.g.: tsai-hill)

LECTURE

List of the elements concerned.

**Comments:**

This option may be repeated as many times as necessary.

The associated coordinate system is defined by the directive "COMP" (see [GBC\\_0095](#) and [GBC\\_0096](#)):

- "COMP" "CORTHO" for the shells and the multilayer element CMC3;
- "COMP" "MORTHO" for the continuum elements in 3D and in 2D plane strain and axisymmetric.

### **Outputs:**

The different components of the ECR table are as follows, for the CMC3 element:

ECR(1): Von Mises criterion on the lower face of the multilayer element CMC3.

ECR(2): Von Mises criterion on the upper face of the multilayer element CMC3.

The different components of the ECR table are as follows, for the continuum elements:

ECR(1): pressure.

ECR(2): Von Mises criterion.



**7.7.26 ORTE : ELASTIC DAMAGE ORTHOTROPY (only in 3D)****Object:**

The directive is used to enter materials with an orthotropic (local) behaviour into a coordinate system defined by the user, coupling with damage. There is coupling between damage (as material LEM1) and orthotropy (as material ORTS). There are 6 damages :  $d_1, d_2, d_3, d_{12}, d_{13}, d_{23}$ . Each damage evolution rate is a function of strain tensor. A failure criterion is implicitly contained within the model: rupture occurs when a damage exceeds a critical value. Two parameters (for each damage) allow to introduce a limitation of the damage rate (thanks to the delayed damage effect) in order to avoid the mesh dependency.

$$D_{nc} = dc < \frac{\epsilon - ep}{s_0 - ep} >$$

$$\dot{d} = \frac{1}{to} \left( 1 - e^{-a < D_{nc} - d >} \right)$$

**References:**

This material behavior has been studied in [960].

**Syntax:**

```

"ORTE"  "R0"    rho    "YG1"   yg1    "YG2"   yg2    "YG3"   yg3
        "NU12"  nu12   "NU13"  nu13   "NU23"  nu23
        "G12"   g12    "G13"   g13    "G23"   g23
        "EP1"   ep1    "EP2"   ep2    "EP3"   ep3
        "EP12"  ep12   "EP13"  ep13   "EP23"  ep23
        "S01"   s01    "S02"   s02    "S03"   s03
        "S012"  s012   "S013"  s013   "S023"  s023
        "DC1"   dc1    "DC2"   dc2    "DC3"   dc3
        "DC12"  dc12   "DC13"  dc13   "DC23"  dc23
        "<"A1"   a1>   "<"A2"   a2>   "<"A3"   a3>
        "<"A12"  a12>  "<"A13"  a13>  "<"A23"  a23>
        "<"T01"  to1>  "<"T02"  to2>  "<"T03"  to3>
        "<"T012" to12> "<"T013" to13> "<"T023" to23>
        /LECTURE/

```

rho

Density of the material.

yg1

Young's modulus along direction 1.

yg2

	Young's modulus along direction 2.
yg3	
	Young's modulus along direction 3.
nu12	
	Poisson's ratio between directions 1 and 2.
nu13	
	Poisson's ratio between directions 1 and 3.
nu23	
	Poisson's ratio between directions 2 and 3.
g12	
	Shear modulus between directions 1 and 2.
g13	
	Shear modulus between directions 1 and 3.
g23	
	Shear modulus between directions 2 and 3.
ep1	
	Strain threshold for damage in direction 1.
ep2	
	Strain threshold for damage in direction 2.
ep3	
	Strain threshold for damage in direction 3.
ep12	
	Strain threshold for damage in direction 12.
ep13	
	Strain threshold for damage in direction 13.
ep23	
	Strain threshold for damage in direction 23.
dc1	
	Critical damage defining the rupture criterion, in direction 1.
dc2	
	Critical damage defining the rupture criterion, in direction 2.

dc3

Critical damage defining the rupture criterion, in direction 3.

dc12

Critical damage defining the rupture criterion, in direction 12.

dc13

Critical damage defining the rupture criterion, in direction 13.

dc23

Critical damage defining the rupture criterion, in direction 23.

a1

Parameter of the delayed damage model, for the direction 1.

a2

Parameter of the delayed damage model, for the direction 2.

a3

Parameter of the delayed damage model, for the direction 3.

a12

Parameter of the delayed damage model, for the direction 12.

a13

Parameter of the delayed damage model, for the direction 13.

a23

Parameter of the delayed damage model, for the direction 23.

to1

Characteristic time of the delayed damage model in direction 1.  $(1/to)$  represents the maximum damage rate.

to2

Characteristic time of the delayed damage model in direction 2.  $(1/to)$  represents the maximum damage rate.

to3

Characteristic time of the delayed damage model in direction 3.  $(1/to)$  represents the maximum damage rate.

to12

Characteristic time of the delayed damage model in direction 12.  $(1/to)$  represents the maximum damage rate.

to13

Characteristic time of the delayed damage model in direction 13. (1/to) represents the maximum damage rate.

to23

Characteristic time of the delayed damage model in direction 23. (1/to) represents the maximum damage rate.

s01

Critical strain for damage in direction 1.

s02

Critical strain for damage in direction 2.

s03

Critical strain for damage in direction 3.

s012

Critical strain for damage in direction 12.

s013

Critical strain for damage in direction 13.

s023

Critical strain for damage in direction 23.

LECTURE

List of the elements concerned.

### Comments:

This option may be repeated as many times as necessary.

The associated coordinate system is defined by the directive "COMP" (see [GBC\\_0095](#)) and [GBC\\_0096](#)):

- "COMP" "CORTHO" for the shells;
- "COMP" "MORTHO" for the continuum elements in 3D and in 2D plane strain and axisymmetric.

### Outputs:

The different components of the ECR table are as follows, for the continuum elements:

ECR(1): Pressure ( $1/3\sigma_{kk}$ ).

ECR(2): Von Mises criterion.

ECR(3:7) : Unused.

ECR(8) : d1 - damage in direction 1.

ECR(9) : d2 - damage in direction 2.

ECR(10): d3 - damage in direction 3.

ECR(11): d12 - damage in direction 12.

ECR(12): d13 - damage in direction 13.

ECR(13): d23 - damage in direction 23.

ECR(14): d1 - damage not corrected (before delay effect) in direction 1.

ECR(15): d2 - damage not corrected (before delay effect) in direction 2.

ECR(16): d3 - damage not corrected (before delay effect) in direction 3.

ECR(17): d12 - damage not corrected (before delay effect) in direction 12.

ECR(18): d13 - damage not corrected (before delay effect) in direction 13.

ECR(19): d23 - damage not corrected (before delay effect) in direction 23.

ECR(20): T - time.

### 7.7.27 ODMS : ONERA DAMAGE MODEL (only in 3D)

#### Object:

The directive is used to enter materials with an orthotropic (local) behaviour into a coordinate system defined by the user, coupling with damage. There is coupling between damage (as material LEM1) and orthotropy (as material ORTS). There are 3 matrix damages :  $d_1^m, d_2^m, d_3^m$  (direction 1, 2 and 3), and 4 fiber damages :  $d_1^f, d_2^f, d_3^f, d_4^f$  (traction direction 1, compression direction 1, traction direction 2, compression direction 2). Each damage evolution rate is a function of strain tensor following the ONERA Damage Mechanics law.

This material behaviour is intended for braided and woven composite materials.

#### References:

The Onera damage model (ODM) is theorized in [959]. It's implementation in Europlexus is studied in [960].

Two parameters (for each damage) allow to introduce a limitation of the damage rate (thanks to the delayed damage effect) in order to avoid the mesh dependency.

$$\dot{d} = \frac{1}{\tau} \left( 1 - e^{-a \langle D_{nc} - d \rangle} \right)$$

The constitutive law is :

$$C_{eff}^{-1} = S_{eff} = S_0 + \sum_{i=1}^3 \eta_i d_i^m H_{0i}^m + \sum_{j=1}^4 d_j^f H_{0j}^f = C_{matrix}^{-1} + \sum_{j=1}^4 d_j^f H_{0j}^f$$

$$[\sigma] = C_{matrix} \cdot [\epsilon] - C_{eff} \cdot [\epsilon_{residual}]$$

where the matrix 6x6  $S_0$  and  $H_0^i$  are defined as:

Compliance  $S_0$ :  $S_0(1,1) = 1/E_1^0$ ,  $S_0(2,2) = 1/E_2^0$ ,  $S_0(3,3) = 1/E_3^0$ ;  $S_0(1,2) = -\nu_{12}/E_1^0$ ,  $S_0(1,3) = -\nu_{13}/E_1^0$ ,  $S_0(2,3) = -\nu_{23}/E_2^0$ ;  $S_0(4,4) = 1/G_{12}^0$ ,  $S_0(5,5) = 1/G_{23}^0$ ,  $S_0(6,6) = 1/G_{13}^0$ , and 0 otherwise.

Matrix damage effect tensors  $H_0^{m1}, H_0^{m2}, H_0^{m3}$ :

$H_0^{m1}$ :  $H_0^{m1}(1,1) = h_1^n/E_1^0$ ,  $H_0^{m1}(4,4) = h_1^p/G_{12}^0$ ,  $H_0^{m1}(6,6) = h_1^{pn}/G_{13}^0$ , and 0 otherwise.

$H_0^{m2}$ :  $H_0^{m2}(2,2) = h_2^n/E_2^0$ ,  $H_0^{m2}(4,4) = h_2^p/G_{12}^0$ ,  $H_0^{m2}(5,5) = h_2^{pn}/G_{23}^0$ , and 0 otherwise.

$H_0^{m3}$ :  $H_0^{m3}(3,3) = h_3^n/E_3^0$ ,  $H_0^{m3}(5,5) = h_3^p/G_{23}^0$ ,  $H_0^{m3}(6,6) = h_3^{pn}/G_{13}^0$ , and 0 otherwise.

Fiber damage effect tensors  $H_0^{f1}, H_0^{f2}, H_0^{f3}, H_0^{f4}$ :

$H_0^{f1}$ :  $H_0^{f1}(1,1) = hf_1^i/E_1^0$ ,  $H_0^{f1}(2,2) = hf_2^i/E_2^0$ ,  $H_0^{f1}(3,3) = hf_3^i/E_3^0$ ,  $H_0^{f1}(1,2) = H_0^{f1}(2,1) = hf_4^i$ ,  $S_0(1,2)$ ,  $H_0^{f1}(1,3) = H_0^{f1}(3,1) = hf_5^i \cdot S_0(1,3)$ ,  $H_0^{f1}(2,3) = H_0^{f1}(3,2) = hf_6^i \cdot S_0(2,3)$ ,  $H_0^{f1}(4,4) = hf_7^i/G_{12}^0$ ,  $H_0^{f1}(5,5) = hf_8^i/G_{23}^0$ ,  $H_0^{f1}(6,6) = hf_9^i/G_{13}^0$ .

Then, the matrix thermodynamic forces  $y_i^n, y_i^t$  are computed in function of positive strain as following:  $y_i^n = \frac{1}{2} C_{ii}^0 \cdot \epsilon_i^+ \cdot \epsilon_i^+$  for  $i \in 1, 2, 3$  and  $y_1^t = (b_1 \cdot C_{66}^0 \cdot \epsilon_{13}^+ \cdot \epsilon_{13}^+ + b_2 \cdot C_{44}^0 \cdot \epsilon_{12}^+ \cdot \epsilon_{12}^+)$ ,  $y_2^t = (b_3 \cdot C_{55}^0 \cdot \epsilon_{23}^+ \cdot \epsilon_{23}^+ + b_4 \cdot C_{44}^0 \cdot \epsilon_{12}^+ \cdot \epsilon_{12}^+)$ ,  $y_3^t = (b_5 \cdot C_{66}^0 \cdot \epsilon_{13}^+ \cdot \epsilon_{13}^+ + b_6 \cdot C_{55}^0 \cdot \epsilon_{23}^+ \cdot \epsilon_{23}^+)$ .

The fiber thermodynamic forces  $y_f^i$  are computed in function of strain as following:  $y_f^1 = \frac{1}{2} C_0(1,1) \cdot \epsilon_1^+ \cdot \epsilon_1^+$ ,  $y_f^2 = \frac{1}{2} C_0(1,1) \cdot \epsilon_1^- \cdot \epsilon_1^-$ ,  $y_f^3 = \frac{1}{2} C_0(2,2) \cdot \epsilon_2^+ \cdot \epsilon_2^+$ ,  $y_f^4 = \frac{1}{2} C_0(2,2) \cdot \epsilon_2^- \cdot \epsilon_2^-$ .

The damage law is the following :

$$d_i = \max(g_i(y_i), d_i^0)$$

where

$$g_i = d_{ci} \left( 1 - e^{-\left( \frac{\langle \sqrt{y_i} - \sqrt{y_{0i}} \rangle}{\sqrt{y_{ci}}} \right)^{p_i}} \right)$$

and if :  $\Delta\epsilon_i^f \leq \bar{\epsilon}_i$ ,

$$\eta_i = 1$$

if :  $-\Delta\epsilon_i^f \leq \bar{\epsilon}_i \leq \Delta\epsilon_i^f$ ,

$$\eta_i = \frac{1}{2} \left( 1 - \cos \left( \frac{\Pi \bar{\epsilon}_i + \Delta\epsilon_i^f}{2 \Delta\epsilon_i^f} \right) \right)$$

if :  $\bar{\epsilon}_i \leq -\Delta\epsilon_i^f$ ,

$$\eta_i = 0$$

where

$$\Delta\epsilon_i^f = (1 + a_{if} d_i^m) \Delta\epsilon_i^0$$

**Syntax:**

"ODMS"	"RO"	rho	"YG1"	yg1	"YG2"	yg2	"YG3"	yg3
	"NU12"	nu12	"NU13"	nu13	"NU23"	nu23	"G12"	g12
	"G13"	g13	"G23"	g23	"DCN1"	dcn1	"DCN2"	dcn2
	"DCN3"	dcn3	"DCT1"	dct1	"DCT2"	dct2	"DCT3"	dct3
	"YON1"	yon1	"YON2"	yon2	"YON3"	yon3	"YCN1"	ycn1
	"YCN2"	ycn2	"YCN3"	ycn3	"YOT1"	yot1	"YOT2"	yot2
	"YOT3"	yot3	"YCT1"	yct1	"YCT2"	yct2	"YCT3"	yct3
	"PN1"	pn1	"PN2"	pn2	"PN3"	pn3	"PT1"	pt1
	"PT2"	pt2	"PT3"	pt3	"HN1"	hn1	"HN2"	hn2
	"HN3"	hn3	"HP1"	hp1	"HP2"	hp2	"HP3"	hp3
	"HHP1"	hhp1	"HHP2"	hhp2	"HHP3"	hhp3	"XSI1"	xsi1
	"XSI2"	xsi2	"XSI3"	xsi3	"AIF1"	aif1	"AIF2"	aif2
	"AIF3"	aif3	"DE01"	deo1	"DE02"	deo2	"DE03"	deo3
	"B1"	b1	"B2"	b2	"B3"	b3	"B4"	b4
	"B5"	b5	"B6"	b6	"TAU1"	tau1	"TAU2"	tau2
	"TAU3"	tau3	"A1"	a1	"A2"	a2	"A3"	a3
	"GHOS"	ghos	"LATE"	late				
<	"DCF1"	dcf1	"DCF2"	dcf2	"DCF3"	dcf3	"DCF4"	dcf4
	"YF01"	yfo1	"YF02"	yfo2	"YF03"	yfo3	"YF04"	yfo4
	"PF1"	pf1	"PF2"	pf2	"PF3"	pf3	"PF4"	pf4
	"HF11"	hf11	"HF21"	hf21	"HF31"	hf31	"HF41"	hf41
	"HF51"	hf51	"HF61"	hf61	"HF71"	hf71	"HF81"	hf81
	"HF91"	hf91	"HF12"	hf22	"HF32"	hf32	"HF42"	hf42
	"HF52"	hf52	"HF62"	hf62	"HF72"	hf72	"HF82"	hf82
	"HF92"	hf92	"HF13"	hf23	"HF33"	hf33	"HF43"	hf43
	"HF53"	hf53	"HF63"	hf63	"HF73"	hf73	"HF83"	hf83
	"HF93"	hf93	"HF14"	hf14	"HF34"	hf34	"HF44"	hf44
	"HF54"	hf54	"HF64"	hf64	"HF74"	hf74	"HF84"	hf84
	"HF94"	hf94	"HF24"	hf24	"AF1"	af1	"AF2"	af2
	"AF3"	af3	"AF4"	af4	"TOF1"	tof1	"TOF2"	tof2
	"TOF3"	tof3	"TOF4"	tof4	"RDC1"	rdc1	"RDC2"	rdc2

```

"RDC3"  rdc3  "RDC4"  rdc4  "EPS1"  eps1  "EPS2"  eps2
"EPS3"  eps3  "EPS4"  eps4  >
/LECTURE/

```

rho

Density of the material.

yg1

Young's modulus along direction 1.

yg2

Young's modulus along direction 2.

yg3

Young's modulus along direction 3.

nu12

Poisson's ratio between directions 1 and 2.

nu13

Poisson's ratio between directions 1 and 3.

nu23

Poisson's ratio between directions 2 and 3.

g12

Shear modulus between directions 1 and 2.

g13

Shear modulus between directions 1 and 3.

g23

Shear modulus between directions 2 and 3.

dcn1

Normal critical damage defining the rupture criterion, in direction 1.

dcn2

Normal critical damage defining the rupture criterion, in direction 2.

dcn3

Normal critical damage defining the rupture criterion, in direction 3.

dct1

Tangential critical damage defining the rupture criterion, in direction 1.



dct2

Tangential critical damage defining the rupture criterion, in direction 2.

dct3

Tangential critical damage defining the rupture criterion, in direction 3.

yon1

Normal damage threshold in direction 1.

yon2

Normal damage threshold in direction 2.

yon3

Normal damage threshold in direction 3.

ycn1

Critical normal damage threshold in direction 1.

ycn2

Critical normal damage threshold in direction 2.

ycn3

Critical normal damage threshold in direction 3.

yot1

Tangential damage threshold in direction 1.

yot2

Tangential damage threshold in direction 2.

yot3

Tangential damage threshold in direction 3.

yct1

Critical tangential damage threshold in direction 1.

yct2

Critical tangential damage threshold in direction 2.

yct3

Critical tangential damage threshold in direction 3.

pn1

Parameter in normal damage law for direction 1.

pn2

	Parameter in normal damage law for direction 2.
pn3	
	Parameter in normal damage law for direction 3.
pt1	
	Parameter in tangential damage law for direction 1.
pt2	
	Parameter in tangential damage law for direction 2.
pt3	
	Parameter in tangential damage law for direction 3.
hn1	
	Parameter in damaged constitutive law for normal direction 1.
hn2	
	Parameter in damaged constitutive law for normal direction 2.
hn3	
	Parameter in damaged constitutive law for normal direction 3.
hp1	
	Parameter in damaged constitutive law for tangential direction 1.
hp2	
	Parameter in damaged constitutive law for tangential direction 2.
hp3	
	Parameter in damaged constitutive law for tangential direction 3.
hhp1	
	Parameter in damaged constitutive law for tangential direction 1.
hhp2	
	Parameter in damaged constitutive law for tangential direction 2.
hhp3	
	Parameter in damaged constitutive law for tangential direction 3.
xsi1	
	Parameter to take into account residual deformation.
xsi2	
	Parameter to take into account residual deformation.

xsi3

Parameter to take into account residual deformation.

aif1

Parameter to evaluate limit deformation for activation index 1 calculus.

aif2

Parameter to evaluate limit deformation for activation index 2 calculus.

aif3

Parameter to evaluate limit deformation for activation index 3 calculus.

deo1

Parameter to evaluate limit deformation for activation index 1 calculus.

deo2

Parameter to evaluate limit deformation for activation index 2 calculus.

deo3

Parameter to evaluate limit deformation for activation index 3 calculus.

b1

Parameter for thermodynamic forces calculus.

b2

Parameter for thermodynamic forces calculus.

b3

Parameter for thermodynamic forces calculus.

b4

Parameter for thermodynamic forces calculus.

b5

Parameter for thermodynamic forces calculus.

b6

Parameter for thermodynamic forces calculus.

tau1

Characteristic time of the delayed damage model in direction 1.  $(1/\tau)$  represents the maximum damage rate.

tau2

Characteristic time of the delayed damage model in direction 2.  $(1/\tau)$  represents the maximum damage rate.

tau3

Characteristic time of the delayed damage model in direction 3.  $(1/\tau)$  represents the maximum damage rate.

a1

Parameter of the delayed damage model, for the direction 1.

a2

Parameter of the delayed damage model, for the direction 2.

a3

Parameter of the delayed damage model, for the direction 3.

ghos

Parameter driving the post damage evolution: case 0. : nothing special happens (default case is 0.); case 1. : if a fiber damage is almost critical (ratio  $rdc1$ ,  $rdc2$ ,  $rdc3$   $rdc4$ ) at a Gauss point of the element, then the element becomes Ghost; case 2. : if a strain reaches its given limit value at a Gauss point of the element, then the element becomes Ghost; case 3. : if case 1 or case 2 is reached, then the element becomes Ghost; case 4. : if there is an almost critical damage at each Gauss points of the element, then the element becomes Ghost; case 5. : if there is a critical strain at each Gauss points of the element, then the element becomes Ghost; case 6. : if case 4 or case 5, then the element becomes Ghost.

late

Parameter driving the delay effect : case 0. : delay effect not active (default case is 0.) ; case 1. : delay effect is active in calculus, for the 3 matrix damages (and the 4 fiber damages).

dcf1

Normal critical damage defining the rupture criterion of fiber in tension, in direction 1.

dcf2

Normal critical damage defining the rupture criterion of fiber in compression, in direction 1.

dcf3

Normal critical damage defining the rupture criterion of fiber in tension, in direction 2.

dcf4

Normal critical damage defining the rupture criterion of fiber in compression, in direction 2.

yfo1

Damage threshold in tension, in direction 1.

yfo2

Damage threshold in compression, in direction 1.

yfo3

Damage threshold in tension, in direction 2.

yfo4

Damage threshold in compression, in direction 2.

yfc1

Critical damage threshold in tension, in direction 1.

yfc2

Critical damage threshold in compression, in direction 1.

yfc3

Critical damage threshold in tension, in direction 2.

yfc4

Critical damage threshold in compression, in direction 2.

pf1

Parameter in tangential fiber damage law for direction 1, in tension.

pf2

Parameter in tangential fiber damage law for direction 1, in compression.

pf3

Parameter in tangential fiber damage law for direction 2, in tension.

pf4

Parameter in tangential fiber damage law for direction 2, in compression.

hf11

Fiber parameter in damaged constitutive law for tensile direction 1.

hf21

Fiber parameter in damaged constitutive law for tensile direction 1.

hf31

Fiber parameter in damaged constitutive law for tensile direction 1.

hf41

Fiber parameter in damaged constitutive law for tensile direction 1.

hf51

Fiber parameter in damaged constitutive law for tensile direction 1.

hf61

	Fiber parameter in damaged constitutive law for tensile direction 1.
hf71	
	Fiber parameter in damaged constitutive law for tensile direction 1.
hf81	
	Fiber parameter in damaged constitutive law for tensile direction 1.
hf91	
	Fiber parameter in damaged constitutive law for tensile direction 1.
hf12	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf22	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf32	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf42	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf52	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf62	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf72	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf82	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf92	
	Fiber parameter in damaged constitutive law for compressive direction 1.
hf13	
	Fiber parameter in damaged constitutive law for tensile direction 2.
hf23	
	Fiber parameter in damaged constitutive law for tensile direction 2.
hf33	
	Fiber parameter in damaged constitutive law for tensile direction 2.

hf43

Fiber parameter in damaged constitutive law for tensile direction 2.

hf53

Fiber parameter in damaged constitutive law for tensile direction 2.

hf63

Fiber parameter in damaged constitutive law for tensile direction 2.

hf73

Fiber parameter in damaged constitutive law for tensile direction 2.

hf83

Fiber parameter in damaged constitutive law for tensile direction 2.

hf93

Fiber parameter in damaged constitutive law for tensile direction 2.

hf14

Fiber parameter in damaged constitutive law for compressive direction 2.

hf24

Fiber parameter in damaged constitutive law for compressive direction 2.

hf34

Fiber parameter in damaged constitutive law for compressive direction 2.

hf44

Fiber parameter in damaged constitutive law for compressive direction 2.

hf54

Fiber parameter in damaged constitutive law for compressive direction 2.

hf64

Fiber parameter in damaged constitutive law for compressive direction 2.

hf74

Fiber parameter in damaged constitutive law for compressive direction 2.

hf84

Fiber parameter in damaged constitutive law for compressive direction 2.

hf94

Fiber parameter in damaged constitutive law for compressive direction 2.

tof1

Characteristic time of the delayed damage model in tensile direction 1. ( $1/\tau$ ) represents the maximum damage rate.

tof2

Characteristic time of the delayed damage model in compressive direction 1. ( $1/\tau$ ) represents the maximum damage rate.

tof3

Characteristic time of the delayed damage model in tensile direction 2. ( $1/\tau$ ) represents the maximum damage rate.

tof4

Characteristic time of the delayed damage model in compressive direction 2. ( $1/\tau$ ) represents the maximum damage rate.

af1

Parameter of the delayed damage model, for the tensile direction 1.

af2

Parameter of the delayed damage model, for the compressive direction 1.

af3

Parameter of the delayed damage model, for the tensile direction 2.

af4

Parameter of the delayed damage model, for the compressive direction 2.

rdc1

Fiber damage (tensile direction 1) ratio for which element becomes ghost.

rdc2

Fiber damage (compressive direction 1) ratio for which element becomes ghost.

rdc3

Fiber damage (tensile direction 2) ratio for which element becomes ghost.

rdc4

Fiber damage (compressive direction 2) ratio for which element becomes ghost.

eps1

Strain limit (tensile direction 1) for which element becomes ghost.

eps2

Strain limit (compressive direction 1) for which element becomes ghost.

eps3

Strain limit (tensile direction 2) for which element becomes ghost.



eps4

Strain limit (compressive direction 2) for which element becomes ghost.

#### LECTURE

List of the elements concerned.

#### Comments:

This option may be repeated as many times as necessary.

The associated coordinate system is defined by the directive "COMPLEMENT" (see pages C.95, section 6.31 and C.96, section 6.32):

- "COMPLEMENT" "CORTHO" for the shells;
- "COMPLEMENT" "MORTHO" for the continuum elements in 3D and in 2D plane strain and axisymmetric.

Verify that this material is available for your elements in the tables of page C.100, section 7.

#### Outputs:

The different components of the ECR table are as follows, for the continuum elements:

- ECR(1): Pressure.
- ECR(2): Von Mises criterion.
- ECR(3): d1 - damage in direction 1.
- ECR(4): d2 - damage in direction 2.
- ECR(5): d3 - damage in direction 3.
- ECR(6): eta1 - activation damage index in direction 1.
- ECR(7): eta2 - activation damage index in direction 2.
- ECR(8): eta3 - activation damage index in direction 3.
- ECR(9): epsilon s 11 - residual deformation 11.
- ECR(10): epsilon s 22 - residual deformation 22.
- ECR(11): epsilon s 33 - residual deformation 33.
- ECR(12): epsilon s 12 - residual deformation 12.
- ECR(13): epsilon s 23 - residual deformation 23.
- ECR(14): epsilon s 13 - residual deformation 13.
- ECR(15): epsilon r 11 - residual deformation 11.
- ECR(16): epsilon r 22 - residual deformation 22.
- ECR(17): epsilon r 33 - residual deformation 33.
- ECR(18): epsilon r 12 - residual deformation 12.

ECR(19): epsilon r 23 - residual deformation 23.

ECR(20): epsilon r 13 - residual deformation 13.

ECR(21) : df1t - fiber tensile damage in direction 1.

ECR(22) : df1c - fiber compressive damage in direction 1.

ECR(23): df2t - fiber tensile damage in direction 2.

ECR(24): df2c - fiber compressive damage in direction 2.

ECR(25): T - time.

## 7.7.28 WOOD

**Object:**

This directive allows to define the BOIS (wood) material, that is used for example for packaging and transportation as a shock absorber. Only the compressive behaviour of the material is considered, while the material response in traction is approximated as perfectly linear (or linear perfectly plastic) because of the lack of experimental data.

**Syntax:**

```

"BOIS"  "R0"  rho    "YG1"  yg1    "YG2"  yg2    "YG3"  yg3
        "NU12" nu12  "NU13" nu13  "NU23" nu23
        "G12"  g12   "G13"  g13   "G23"  g23
        "SY_1" sy1   "SY_2" sy2   "SY_3" sy3
        "ED_1" ed1   "ED_2" ed2   "ED_3" ed3
        < "TR_1" tr1 > < "TR_2" tr2 > < "TR_3" tr3 >
        < "COE1" coe1 > < "COE2" coe2 > < "COE3" coe3 >
        < "EDCV" edcv > < "DIRF" idir >
        < "CI23" ci23 > < "CI31" ci31 > < "CI12" ci12 >
        < $[ "RUPT" ; "DECO"]$
          < "CONT" |[ "TR1M" tr1m ; "TR2M" tr2m ; "TR3M" tr3m ;
                     "T23M" t23m ; "T31M" t31m ; "T12M" t12m ;
                     "CO1M" co1m ; "CO2M" co2m ; "CO3M" co3m ]|> ;
          < "DPLA" |[ "EP23" ep23 ; "EP31" ep31 ; "EP12" ep12 ]|> >

/LECTURE/

```

rho

Density of the material.

yg1

Young's modulus along direction 1.

yg2

Young's modulus along direction 2.

yg3

Young's modulus along direction 3.

nu12

Poisson's ratio between directions 1 and 2.

nu13

Poisson's ratio between directions 1 and 3.

nu23

Poisson's ratio between directions 2 and 3.

g12

Shear modulus between directions 1 and 2.

g13

Shear modulus between directions 1 and 3.

g23

Shear modulus between directions 2 and 3.

sy1

Elastic limit in compression along the first orthotropy direction.

sy2

Elastic limit in compression along the second orthotropy direction.

sy3

Elastic limit in compression along the third orthotropy direction.

ed1

Limit deformation before reconsolidation along the first orthotropy direction.

ed2

Limit deformation before reconsolidation along the second orthotropy direction.

ed3

Limit deformation before reconsolidation along the third orthotropy direction.

tr1

Optional keyword : Limit elastic stress in traction along the first orthotropy direction.  
If omitted, an elastic behaviour is assumed along this direction.

tr2

Optional keyword : Limit elastic stress in traction along the second orthotropy direction.  
If omitted, an elastic behaviour is assumed along this direction.

tr3

Optional keyword : Limit elastic stress in traction along the third orthotropy direction.  
If omitted, an elastic behaviour is assumed along this direction.

coe1

Optional keyword : Scale factor between initial Young's modulus and Young's modulus  
after reconsolidation in direction 1.

**coe2**

Optional keyword : Scale factor between initial Young's modulus and Young's modulus after reconsolidation in direction 2.

**coe3**

Optional keyword : Scale factor between initial Young's modulus and Young's modulus after reconsolidation in direction 3.

**edcv**

Optional keyword : Threshold deformation in all directions starting convergence of the material towards a consolidated linear elastic material (see comment below).

**idir**

Optional keyword : Spatial direction of wood fibers, if not the first orthotropy direction (see comment below).

**ci23**

Optional keyword : Limit elastic shear stress in the (second orthotropy direction,third orthotropy direction) plane. If omitted, an elastic behaviour is assumed. Only available in 3D.

**ci31**

Optional keyword : Limit elastic shear stress in the (third orthotropy direction,first orthotropy direction) plane. If omitted, an elastic behaviour is assumed. Only available in 3D.

**ci12**

Optional keyword : Limit elastic shear stress in the (first orthotropy direction,second orthotropy direction) plane. If omitted, an elastic behaviour is assumed. Only available in 3D.

**RUPT**

Optional keyword : Introduces an element failure model represented by a failure criterion and by a failure limit value. Only available in 3D.

**DECO**

Optional keyword : Introduces an model of automatic separation of elements defined by a failure criterion and by a failure limit value. Only available in 3D. Only available for CUB8 element. More explanations can be found in [\[899\]](#).

**CONT**

Failure criterion based upon stress associated to RUPT keyword or DECO keyword (but not the both). Only available in 3D.

**DPLA**

Failure criterion based upon plastic strain associated to RUPT keyword or DECO keyword (but not the both). Only available in 3D.

tr1m

Optional keyword : Failure limit value for stress in traction along the first orthotropy direction.

tr2m

Optional keyword : Failure limit value for stress in traction along the second orthotropy direction.

tr3m

Optional keyword : Failure limit value for stress in traction along the third orthotropy direction.

t23m

Optional keyword : Failure limit value for shear stress in the (second orthotropy direction,third orthotropy direction) plane.

t31m

Optional keyword : Failure limit value for shear stress in the (third orthotropy direction,first orthotropy direction) plane.

t12m

Optional keyword : Failure limit value for shear stress in the (first orthotropy direction,second orthotropy direction) plane.

co1m

Failure limit value for stress in compression along the first orthotropy direction.

co2m

Optional keyword : Failure limit value for stress in compression along the second orthotropy direction.

co3m

Optional keyword : Failure limit value for stress in compression along the third orthotropy direction.

ep23

Optional keyword : Failure limit value for shear plastic strain in the (second orthotropy direction,third orthotropy direction) plane.

ep31

Optional keyword : Failure limit value for shear plastic strain in the (third orthotropy direction,first orthotropy direction) plane.

ep12

Optional keyword : Failure limit value for shear plastic strain in the (first orthotropy direction,second orthotropy direction) plane.

LECTURE

List of the elements concerned.

### Comments:

This material model is taken from the thesis of P. François: "Plasticité du bois en compression multi-axiale : Application à l'absorption d'énergie mécanique". Doctoral Thesis of the Bordeaux I University (October 1992).

The associated coordinate system is defined by the option **MORTH0** (see page C.96) for the continuum elements in 2D and 3D.

Verify that this material is available for your elements, by means of the tables of page C.100.

Compression instability may be encountered after reconsolidation, since the material becomes very stiff in the consolidated direction and is still very soft in the other directions. To overcome this problem, it may be considered that the orthotropy of the material is lost when reconsolidation is achieved, since the microstructure has been completely crushed and all the voids filled. This assumption is taken from the observations made in the PhD Thesis of C. Adalian: "The behaviour of wood under multiaxial dynamic compression - Use for the modelling of crashes of containers", PhD Thesis of Bordeaux I University (1998)

Such a behaviour is activated using **EDCV** keyword. The floating value associated to this keyword defines the level of deformation above which the process of convergence towards an elastic isotropic material is started. It should be less than the reconsolidation limit given in each orthotropy direction.

The convergence process is such that material characteristics of the element converge continuously towards the isotropic consolidated values, whatever the first direction is in which reconsolidation is achieved.

Isotropic elastic parameters of the consolidated materials are deduced from elastic parameters given in the wood fibers direction. If this direction is not the first orthotropy direction, it can be specified using **"DIRF"** keyword. Isotropic consolidated parameters are then obtained by the formulae (assuming the fibers direction is direction 1):

$$E_c = E_1 \cdot coe1$$

$$\nu_c = \frac{\nu_{12} + \nu_{13}}{2}$$

$$G_c = \frac{E_c}{2 \cdot (1 + \nu)}$$

### Outputs:

The different components of the ECR table are as follows, for the continuum elements:

ECR(1) : Pressure.

ECR(2) : Von mises criterion.

- ECR(3) : Plastic strain along the first orthotropy direction.
- ECR(4) : Plastic strain along the second orthotropy direction.
- ECR(5) : Plastic strain along the third orthotropy direction.
- ECR(6) : Principal stress along the first orthotropy direction.
- ECR(7) : Principal stress along the second orthotropy direction.
- ECR(8) : Principal stress along the third orthotropy direction.
- ECR(9) : Total strain along the first orthotropy direction.
- ECR(10): Total strain along the second orthotropy direction.
- ECR(11): Total strain along the third orthotropy direction.
- ECR(12): Flag for isotropic elastic converged material (0: not fully converged, 1: fully converged)
- ECR(13): Convergence level in the first orthotropy direction.
- ECR(14): Convergence level in the second orthotropy direction.
- ECR(15): Convergence level in the third orthotropy direction.
- ECR(16): Failure flag (0=virgin Gauss Point, 1=failed Gauss Point) (only in 3D).
- ECR(17): Shear stress in the (second orthotropy direction,third orthotropy direction) plane (only in 3D).
- ECR(18): Shear stress in the (third orthotropy direction,first orthotropy direction) plane (only in 3D).
- ECR(19): Shear stress in the (first orthotropy direction,second orthotropy direction) plane (only in 3D).
- ECR(20): Shear Plastic strain in the (second orthotropy direction,third orthotropy direction) plane (only in 3D).
- ECR(21): Shear Plastic strain in the (third orthotropy direction,first orthotropy direction) plane (only in 3D).
- ECR(22): Shear Plastic strain in the (first orthotropy direction,second orthotropy direction) plane (only in 3D).
- ECR(23): Shear total strain in the (second orthotropy direction,third orthotropy direction) plane (only in 3D).
- ECR(24): Shear total strain in the (third orthotropy direction,first orthotropy direction) plane (only in 3D).
- ECR(25): Shear total strain in the (first orthotropy direction,second orthotropy direction) plane (only in 3D).



**7.7.29 ORSR : RATE DEPENDENT LINEAR ORTHOTROPY (Local basis)****Object:**

Attention: The description of the material is not showing all capabilities of the material. The material allows erosion and has several more input and output variables.

The directive is based on ORTS material law, with the possibility to define rate dependent material properties. 9 material properties can be made rate dependent:  $E_{11}$ ,  $E_{22}$ ,  $E_{33}$ ,  $\nu_{12}$ ,  $\nu_{13}$ ,  $\nu_{23}$ ,  $G_{12}$ ,  $G_{13}$ ,  $G_{23}$ . The rate dependency is based on the following polynomial law:

$$X_{ij}(\dot{\epsilon}) = \sum_{k=0}^5 X_{ij}^k (\log(\dot{\epsilon}))^k \quad (1)$$

As for ORTS directive, stress and strain are expressed in the user coordinate system.

**Syntax:**

```

"ORSR"  "R0"  rho    "YG10"  yg10  "YG11"  yg11
        "YG12"  yg12  "YG13"  yg13  "YG14"  yg14
        "YG15"  yg15  "YG20"  yg20  "YG21"  yg21
        "YG22"  yg22  "YG23"  yg23  "YG24"  yg24
        "YG25"  yg25  "YG30"  yg30  "YG31"  yg31
        "YG32"  yg32  "YG33"  yg33  "YG34"  yg34
        "YG35"  yg35  "N120"  n120  "N121"  n121
        "N122"  n122  "N123"  n123  "N124"  n124
        "N125"  n125  "N130"  n130  "N131"  n131
        "N132"  n132  "N133"  n133  "N134"  n134
        "N135"  n135  "N230"  n230  "N231"  n231
        "N232"  n232  "N233"  n233  "N234"  n234
"N235"  n235  "G120"  g120  "G121"  g121
        "G122"  g122  "G123"  g123  "G124"  g124
        "G125"  g125  "G130"  g130  "G131"  g131
        "G132"  g132  "G133"  g133  "G134"  g134
        "G135"  g135  "G230"  g230  "G231"  g231
        "G232"  g232  "G233"  g233  "G234"  g234
        "G235"  g235  "MINE"  MINE  "MAXE"  MAXE
<"XT1"  xt1  "XT2"  xt2  "XT3"  xt3
        "XC1"  xc1  "XC2"  xc2  "XC3"  xc3
        "RST1"  rst1  "RST2"  rst2  "RST3"  rst3
        "CRIT"  icrit>  /LECTURE/

```

rho

Density of the material.

yg10

0th order coefficient for the rate dependency of the Young's modulus along direction 1.  
Without rate dependency, Young's modulus along direction 1.

yg11

1st order coefficient for the rate dependency of the Young's modulus along direction 1.

yg12

2nd order coefficient for the rate dependency of the Young's modulus along direction 1.

yg13

3rd order coefficient for the rate dependency of the Young's modulus along direction 1.

yg14

4th order coefficient for the rate dependency of the Young's modulus along direction 1.

yg15

5th order coefficient for the rate dependency of the Young's modulus along direction 1.

yg20

0th order coefficient for the rate dependency of the Young's modulus along direction 2.  
Without rate dependency, Young's modulus along direction 2.

yg21

1st order coefficient for the rate dependency of the Young's modulus along direction 2.

yg22

2nd order coefficient for the rate dependency of the Young's modulus along direction 2.

yg23

3rd order coefficient for the rate dependency of the Young's modulus along direction 2.

yg24

4th order coefficient for the rate dependency of the Young's modulus along direction 2.

yg25

5th order coefficient for the rate dependency of the Young's modulus along direction 2.

yg30

0th order coefficient for the rate dependency of the Young's modulus along direction 3.  
Without rate dependency, Young's modulus along direction 3.

yg31

1st order coefficient for the rate dependency of the Young's modulus along direction 3.

yg32

2nd order coefficient for the rate dependency of the Young's modulus along direction 3.

yg33

3rd order coefficient for the rate dependency of the Young's modulus along direction 3.

yg34

4th order coefficient for the rate dependency of the Young's modulus along direction 3.

yg35

5th order coefficient for the rate dependency of the Young's modulus along direction 3.

n120

0th order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 2. Without rate dependency, Poisson's ratio between directions 1 and 2.

n121

1st order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 2.

n122

2nd order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 2.

n123

3rd order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 2.

n124

4st order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 2.

n125

5st order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 2.

n130

0th order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 3. Without rate dependency, Poisson's ratio between directions 1 and 3.

n131

1st order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 3.

n132

2nd order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 3.

n133

3rd order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 3.

n134

4st order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 3.

n135

5st order coefficient for the rate dependency of the Poisson's ratio between directions 1 and 3.

n230

0th order coefficient for the rate dependency of the Poisson's ratio between directions 2 and 3. Without rate dependency, Poisson's ratio between directions 2 and 3.

n231

1st order coefficient for the rate dependency of the Poisson's ratio between directions 2 and 3.

n232

2nd order coefficient for the rate dependency of the Poisson's ratio between directions 2 and 3.

n233

3rd order coefficient for the rate dependency of the Poisson's ratio between directions 2 and 3.

n234

4st order coefficient for the rate dependency of the Poisson's ratio between directions 2 and 3.

n235

5st order coefficient for the rate dependency of the Poisson's ratio between directions 2 and 3.

g120

0th order coefficient for the rate dependency of the shear modulus between directions 1 and 2. Without rate dependency, shear modulus between directions 1 and 2.

g121

1st order coefficient for the rate dependency of the shear modulus between directions 1 and 2.

g122

2nd order coefficient for the rate dependency of the shear modulus between directions 1 and 2.

g123

3rd order coefficient for the rate dependency of the shear modulus between directions 1 and 2.

g124

4th order coefficient for the rate dependency of the shear modulus between directions 1 and 2.

g125

5th order coefficient for the rate dependency of the shear modulus between directions 1 and 2.

g130

0th order coefficient for the rate dependency of the shear modulus between directions 1 and 3. Without rate dependency, shear modulus between directions 1 and 3.

g131

1st order coefficient for the rate dependency of the shear modulus between directions 1 and 3.

g132

2nd order coefficient for the rate dependency of the shear modulus between directions 1 and 3.

g133

3rd order coefficient for the rate dependency of the shear modulus between directions 1 and 3.

g134

4th order coefficient for the rate dependency of the shear modulus between directions 1 and 3.

g135

5th order coefficient for the rate dependency of the shear modulus between directions 1 and 3.

g230

0th order coefficient for the rate dependency of the shear modulus between directions 2 and 3. Without rate dependency, shear modulus between directions 2 and 3.

g231

1st order coefficient for the rate dependency of the shear modulus between directions 2 and 3.

g232

2nd order coefficient for the rate dependency of the shear modulus between directions 2 and 3.

g233

3rd order coefficient for the rate dependency of the shear modulus between directions 2 and 3.

g234

4th order coefficient for the rate dependency of the shear modulus between directions 2 and 3.

g235

5th order coefficient for the rate dependency of the shear modulus between directions 2 and 3.

MINE

Lower boundary of the strain rate interval used for the polynomial function identification.

MAXE

Upper boundary of the strain rate interval used for the polynomial function identification.

xt1

Failure criterion for tension in x-direction.

xt2

Failure criterion for tension in y-direction.

xt3

Failure criterion for tension in z-direction.

xc1

Failure criterion for compression in x-direction.

xc2

Failure criterion for compression in y-direction.

xc3

Failure criterion for compression in z-direction.

rst1

Failure criterion for shear in xy-direction.

rst2

Failure criterion for shear in yz-direction.

rst3

Failure criterion for shear in xz-direction.

icrit

Erosion criteria type (e.g.: tsai-hill)

## LECTURE

List of the elements concerned.

**Comments:**

This option may be repeated as many times as necessary.

The associated coordinate system is defined by the directive "COMP" (see [GBC\\_0095](#) and [GBC\\_0096](#)):

- "COMP" "CORTHO" for the shells and the multilayer element CMC3;
- "COMP" "MORTHO" for the continuum elements in 3D and in 2D plane strain and axisymmetric.

**Outputs:**

The different components of the ECR table are as follows, for the continuum elements:

- ECR(1): pressure.
- ECR(2): Von Mises criterion.
- ECR(16): v1 - Strain rate in direction 1.
- ECR(17): v2 - Strain rate in direction 2.
- ECR(18): v3 - Strain rate in direction 3.
- ECR(19): v4 - Strain rate for shear in 12-direction.
- ECR(20): v5 - Strain rate for shear in 23-direction.
- ECR(21): v6 - Strain rate for shear in 13-direction.
- ECR(22): Apparent modulus in direction 1.
- ECR(23): Apparent modulus in direction 2.
- ECR(24): Apparent modulus in direction 3.
- ECR(25): Apparent Poisson's ratio between directions 1 and 2.
- ECR(26): Apparent Poisson's ratio between directions 1 and 3.
- ECR(27): Apparent Poisson's ratio between directions 2 and 3.
- ECR(28): Apparent shear modulus between directions 1 and 2.
- ECR(29): Apparent shear modulus between directions 2 and 3.
- ECR(30): Apparent shear modulus between directions 1 and 3.

**7.7.30 MASS****Object:**

This directive enables the masses of the material points **PMAT** to be entered.

Optionally, the Young's modulus  $E$  and the Poisson's coefficient  $\nu$  of the material may also be specified. These are used in order to determine the material's bulk modulus

$$\kappa = \frac{E}{3(1 - 2\nu)}$$

when the **PMAT** associated with the **MASS** material has a (*nodal*) pinball attached to it. In this case, it is allowed to specify a zero mass (i.e. a 0 value for **xm**) in order to avoid adding an extra mass to the structure if so desired.

**Syntax:**

```
"MASS" xm
    < "YOUN" youn> < "NU" nu >
    /LECTURE/
```

**xm**

Mass.

**youn**

Young's modulus. If not specified, it is assumed to be 0.

**nu**

Poisson's coefficient. If not specified, it is assumed to be 0.

**LECTURE**

Numbers of the elements concerned.

**Comments:**

If the node corresponding to the material point belongs also to another element, the added mass (**xm**) may be zero. This is very useful in the case of unilateral junctions or of added *nodal* pinballs for contact.

In this way, added masses may be entered too.

In axisymmetric, the real masses must be divided by  $2\pi$ .



**Outputs:**

The different components of the ECR table are as follows:

ECR(1): integrated impulse from the origin

ECR(2): sum of the instantaneous reaction forces.

**7.7.31 PHANTOM****Object:**

This directive enables the elimination of elements in a mesh.

**Syntax:**

```
"FANT" rho /LECTURE/
```

rho

Density.

LECTURE

List of the elements concerned.

**Comments:**

The EUROPLEXUS program considers that all these elements do not exist.

However, the nodes are always present; as their masses may not be zero, it is necessary to give (very low) densities to "FANT" elements.

### 7.7.32 FREE (USER'S MATERIAL)

#### Object:

The directive enables the user to enter his own constitutive laws.

#### Syntax:

```

$ "STRU" ... $
$ "FLUI" ... $
"LIBR" $ "PMAT" ... $ < "PARA" a b c ... > /LECTURE/
$ "MECA" ... $
$ "CLIM" ... $

```

"STRU" ...

Indicates that the free material is of type "STRUCTURE".

"FLUI" ...

Indicates that the free material is of type "FLUIDE".

"PMAT" ...

Indicates that the free material is of type "POINT MATERIEL".

"MECA" ...

Indicates that the free material is of type "MECANISME".

"CLIM" ...

Indicates that the free material is of type "CONDITION AUX LIMITES".

"PARA" ...

Key-word used to introduce a series of additional parameters.

LECTURE

List of the elements concerned.

#### Comments:

The distinction between structure and fluid is due to the processing differences in A.L.E. In fact, there are transport terms for the fluid, whereas the structure is always Lagrangian.

Similarly, the cases of material points, mechanisms and boundary conditions are so peculiar that a dedicated syntax is provided.

These directives are described in detail on the following pages.

**Remarks:**

In the examples proposed in the following pages, there are some calls to utility routines that are available within EUROPLEXUS:

1. ERRMSS(String1,String2) :

Subroutine named 'String1' generates the error message 'String2', increments the error counter and triggers the calculation stop. 'String1' and 'String2' are two character strings.

2. TILT :

This subroutine without arguments triggers the calculation stop at the end of the current time step, and passes control to the next input directive in the input data set, which is normally either "SUIT" or "FIN", following the directive "CALCUL".

3. QUIDNE(LOOP,NUM,LON,VAL) :

This subroutine extracts values relative to a node or to an element (of index "NUM"), and places them in the array "VAL".

If the quantity to be extracted is a vector (e.g. a velocity), the length of the extracted vector is in "LON", and the array "VAL" must be dimensioned sufficiently ( $\text{DIM}(\text{VAL}) \geq \text{LON}$ ).

Argument "LOOP" allows to select the values to be extracted.

For a node :

LOOP = 0 : Coordinates of node "NUM",  
LOOP = 1 : Displacements,  
LOOP = 2 : Velocities,  
LOOP = 5 : Nodal masses.

For an element :

LOOP = 21 : Stresses in element "NUM",  
LOOP = 22 : Total deformations,  
LOOP = 23 : Internal variables (ECR),  
LOOP = 24 : Internal energy.

### 7.7.33 FREE MATERIAL OF TYPE STRUCTURE

**Object:**

This directive introduces a user-defined constitutive behaviour of structural type (“STRUCTURE”).

**Syntax:**

```
"LIBR"  "STRU" num  "RO" rho  "YOUN" young  "NU" nu  ...  
          ...  < "PARA" /LECPARA/ >  /LECTURE/
```

**"STRU" num**

Indicates that the free material of type “STRUCTURE” has the user-specified index **num**.

**"RO" rho**

Density. This value is mandatory in order to compute the element mass.

**"YOUN" young**

Young’s modulus.

**"NU" nu**

Poisson’s ratio.

**"PARA" ...**

Key-word used to introduce a series of additional parameters.

**LECTURE**

List of the elements concerned.

**Comments:**

The number **num** enables several materials chosen by the user to be recognized. The three parameters **rho**, **young** and **nu** are compulsory.

The user specifies his material’s parameters after the keyword “PARAM”. When EUROPLEXUS finds the keyword “LECTURE”, it considers that the list of parameters is terminated, whatever the number of values that have been read.

However, the total number of parameters for this material may not exceed 100, including the three mandatory values (**rho**, **young** and **nu**).

If there are no additional parameters besides the three mandatory ones, the keyword “PARAM” may be omitted.

The parameters are used within the subroutine “MSLIBR” that must be written by the user, compiled and linked with the code libraries to produce a special code executable before launching the run.

The elements that accept the free material of type “STRUCTURE” are the following:

2D : TRIA, CAR1, CAR4.

3D : CUBE, CUB6, CUB8, PRIS, PR6, TETR.

Be careful to respect the conventions chosen to rank the tensor components according to the 2-D plane, 2-D axisymmetric or 3-D cases. See page G.20 for further explanations.

The user can store for each element (and each integration point), the values he wants (up to 10) in the ECR table. For homogeneity with the other materials, the following data will be stored in the first two locations of the ECR table :

ECR(1) = Pressure

ECR(2) = Von Mises

The eight other locations are free.

### Examples:

The following example, taken from the standard benchmark ”bm\_str\_2d.lib”, concerns the traction of an axisymmetric cylinder.

The material data are as follows:

```
MATERIAUX  LIBRE  STRUCTURE 901
              RO 7800. YOUNG 210E9 NU 0.    TOUS
```

In this particular case there is just one material, identified by the user-supplied index 901. There are no additional parameters besides the three mandatory ones, and the Poisson coefficient is zero. All the elements in the mesh possess this material (keyword “TOUS”).

### Programming example relative to MSLIBR:

```

SUBROUTINE MSLIBR(NLGEOM,NUM,TT,XMAT,SIG,DEPS,EDOT,RO,PI,
&                CSON,ECR,X,IEL,IDIM,NBN)
*
* -----
*
*      materiau libre (structure)          m.lepareux 11.86
*
* -----

```

```

*
*   entree :
*       num      = numero de reperage du materiau utilisateur
*       tt       = temps du calcul
*       xmat(1)  = masse volumique initiale
*       xmat(2)  = young
*       xmat(3)  = poisson
*       xmat(4: ) = autres parametres du materiau
*       sig      = contraintes au debut du pas
*       deps     = accroissement des deformations
*       edot     = vitesse de deformation
*       ro       = masse volumique courante
*       x        = coordonnees des nbn noeuds
*       iel      = numero de l'element
*       idim     = dimension (2=2d ou axis , 3=3d)
*
*   sortie :
*       nlgeom   = 0 (dans inico1.ff matal(13)=1)
*       pi       = contraintes a la fin du pas (a calculer)
*       cson     = vitesse du son (a calculer pour la stabilite)
*       ecr(1)   = pression (a calculer)
*       ecr(2)   = critere de von mises (a calculer)
*       ecr(3:7) = emplacements libres
*
*   attention ! le materiau 901 est utilise par le benchmark
*               "bm_str_2d_libr.epx"
*
*   IMPLICIT NONE
*
*---   variables globales :
*   INTEGER, INTENT(IN) :: NUM,IEL,NBN,IDIM
*   INTEGER, INTENT(OUT) :: NLGEOM
*   REAL(8), INTENT(IN) :: TT,XMAT(*),SIG(*),DEPS(*),EDOT(*),
*   &                      RO,X(IDIM,NBN)
*   REAL(8), INTENT(OUT) :: CSON,PI(*)
*   REAL(8), INTENT(INOUT) :: ECR(*)
*
*---   variables locales :
*   REAL(8) :: AMU,ALAMB,AUX,YOUNG,POISS,C1
*
*
*-----   on active les nonlinearites geometriques : nlgeom=0
*   NLGEOM = 0
*
*   SELECT CASE (NUM)
*   CASE(901)
*-----   l'exemple qui suit concerne un mat. elastique (en 2d axis.)
*   YOUNG=XMAT(2)
*   POISS=XMAT(3)
*---   coefficients de lame :
*   C1=YOUNG/(1.+POISS)
*   AMU=0.5D0*C1
*   ALAMB=C1*POISS/(1-2*POISS)
*   AUX=ALAMB*(DEPS(1)+DEPS(2)+DEPS(4))
*---   nouveau tenseur des contraintes :
*   PI(1) = SIG(1) + AUX + C1*DEPS(1)
*   PI(2) = SIG(2) + AUX + C1*DEPS(2)
*   PI(3) = SIG(3) + AMU*DEPS(3)
*   PI(4) = SIG(4) + AUX + C1*DEPS(4)
*---   pression :
*   ECR(1)=(PI(1)+PI(2)+PI(4))/3.D0
*---   critere de von mises :
*   ECR(2)=SQRT(PI(1)*(PI(1)-PI(2))+PI(2)*(PI(2)-PI(4))
*   &          +PI(4)*(PI(4)-PI(1))+3*PI(3)*PI(3))
*---   vitesse du son (stabilite) :
*   CSON=DSQRT(YOUNG/RO)
*
*   CASE DEFAULT
*       CALL ERRMSS('MSLIBR','ROUTINE UTILISATEUR NON PROGRAMMEE')
*       STOP ' "MSLIBR" ABSENT'
*   END SELECT
*
*   END SUBROUTINE MSLIBR

```

### 7.7.34 FREE MATERIAL OF TYPE FLUID

**Object:**

This directive introduces a user-defined constitutive behaviour of fluid type (“FLUIDE”).

**Syntax:**

```
"LIBR" "FLUI" num "RO" rho "PINI" pini "PREF" pref "EINT" ei ...  
... < "PARA" /LECPARA/ > /LECTURE/
```

**"FLUI" num**

Indicates that the free material of type “FLUIDE” has the user-specified index **num**.

**"RO" rho**

Density. This value is mandatory in order to compute the element mass.

**"PINI" pini**

Initial pressure.

**"PREF" pref**

Reference pressure.

**"EINT" ei**

Initial internal energy per unit mass.

**"PARA" ...**

Key-word used to introduce a series of additional parameters.

**LECTURE**

List of the elements concerned.

**Comments:**

The number **num** enables several materials chosen by the user to be recognized. The four parameters **rho**, **pini**, **pref** and **ei** are compulsory.

The user specifies his material’s parameters after the keyword “PARAM”. When EUROPLEXUS finds the keyword “LECTURE”, it considers that the list of parameters is terminated, whatever the number of values that have been read.

However, the total number of parameters for this material may not exceed 100, including the four mandatory values (**rho**, **pini**, **pref** and **ei**).



If there are no additional parameters besides the four mandatory ones, the keyword "PARAM" may be omitted.

The parameters are used within the subroutine "MFLIBR" that must be written by the user, compiled and linked with the code libraries to produce a special code executable before launching the run.

The elements that accept the free material of type "FLUIDE" are the following:

1D : TUBE, TUYA, CAVI.

2D : TRIA, CAR1.

3D : CUBE, PRIS, TETR.

Be careful to respect the conventions chosen to rank the tensor components according to the 2-D plane, 2-D axisymmetric or 3-D cases. See page G.20 for further explanations.

The user can store for each element (and each integration point), the values he wants (up to 10) in the ECR table. For homogeneity with the other materials, the following data will be stored in the first two locations of the ECR table :

ECR(1) = Pressure

ECR(2) = Density.

The eight other locations are free.

### Examples:

The following example, taken from the standard benchmark "bm.flu\_1d.lib", concerns a shock tube with a perfect gas.

The material data are as follows:

#### MATERIAUX

```
LIBRE FLUIDE 903 R0 13. PINI 1e6 PREF 1e5 EINT 192.3077e3
PARAM 1.4 640. LECT tub_1 TERM
```

```
LIBRE FLUIDE 903 R0 1.3 PINI 1e5 PREF 1e5 EINT 192.3077e3
PARAM 1.4 640. LECT tub_2 TERM
```

In this case there are two materials, whose user index is the same (903), but which have different initial conditions.

Note the value of the initial internal energy, which is mandatory because the behaviour of the perfect gas depends both on the density  $\rho$  and on the specific internal energy  $e$ :

$$P = (\gamma - 1)\rho e$$

These two variables  $\rho$  and  $e$  change during the transient, as a function of mass and energy transfer among the neighbouring elements. EUROPLEXUS computes these transfers automatically.

There are two additional parameters besides the four mandatory ones. These are respectively the ratio of specific heats ( $\gamma$ ), and the specific heat at constant volume ( $C_v$ ), which allow to obtain the temperature ( $\theta$ ).

$$\theta = \frac{e}{C_v}$$

### Programming example relative to MFLIBR:

```

      SUBROUTINE MFLIBR(NUM,TT,XMAT,SIG,DEPS,EDOT,RO,EINT,DSIG,CSON,
&
      ECR,X,IEL,IDIM,NBN)
*
* -----
*
*      matieriau libre (fluide)                m.lepareux 11.86
*
* -----
*
*  entree :
*      num      = numero de repereage du matieriau utilisateur
*      tt       = temps du calcul
*      sig      = contraintes au debut du pas
*      deps     = accroissement des deformations
*      edot     = vitesse de deformation
*      ro       = masse volumique courante
*      eint     = energie interne massique courante
*      x        = coordonnees des nbn noeuds
*      iel      = numero de l'element
*      idim     = dimension (2=2d ou axis , 3=3d)
*      nbn      = nombre de noeuds de l'element
*      xmat(1)  = masse volumique initiale
*      xmat(2)  = pression initiale
*      xmat(3)  = pression de reference
*      xmat(4)  = energie interne massique initiale
*      xmat(5:) = parametres de l'utilisateur
*
*  sortie :
*      dsig     = increments de contraintes
*      cson     = vitesse du son (pour la stabilite)
*      ecr(1)   = pression
*      ecr(2)   = masse volumique
*      ecr(3:7) = emplacements libres
*
*  attention ! le matieriau 903 est utilise par le benchmark
*              "bm_flu_1d_libr.epx"
*
*      IMPLICIT NONE
*
* ----  variables globales :
*      INTEGER, INTENT(IN) :: NUM,IEL,NBN,IDIM
*      REAL(8), INTENT(IN) :: TT,XMAT(*),SIG(*),DEPS(*),EDOT(*),RO,EINT,
&
*              X(IDIM,NBN)
*      REAL(8), INTENT(OUT) :: DSIG(*),CSON
*      REAL(8), INTENT(INOUT) :: ECR(*)
*
* ----  variables locales :
*      REAL(8) :: ROZR,PZER,PREF,PABS,PR,CV,GAMA,TRE
*
*
*      SELECT CASE (NUM)
*      CASE(903)
* --      cas d'un gaz parfait :
*          PREF = XMAT(3)                ! PRESSION DE REFERENCE
*          GAMA = XMAT(5)                ! GAMMA DU GAZ
*          CV   = XMAT(6)                ! CHALEUR MASSIQUE
*          PABS = RO * (GAMA -1D0) * EINT ! PRESSION ABSOLUE
*          CSON = SQRT(GAMA*PABS/RO)     ! VITESSE DU SON
*          PR   = PABS - PREF            ! PRESSION RELATIVE
*          TRE  = EINT/CV - 273.15D0     ! TEMPERATURE
*
* --      increments de contrainte (1d)
*          DSIG(1) = -PR -SIG(1)
*
* --      remplissage des "ecr" :
*          ECR(1) = PABS
*          ECR(2) = RO
*          ECR(3) = CSON

```

```
      ECR(4) = TRE
*
CASE DEFAULT
*----- routine use to write
      CALL ERRMSS('MFLIBR','ROUTINE UTILISATEUR NON PROGRAMMEE')
      STOP ' "MFLIBR" ABSENT'
END SELECT
*
END
```

### 7.7.35 FREE MATERIAL OF TYPE MATERIAL POINT

**Object:**

This directive introduces a user-defined constitutive behaviour of type material point (“POINT MATERIEL”).

**Syntax:**

```
"LIBR" "PMAT" num "MASS" m < "PARA" a b c ... > /LECTURE/
```

**"PMAT" num**

Indicates that the free material of type “POINT MATERIEL” has the user-specified index **num**.

**"MASS" m**

Mass of the material point element.

**"PARA" ...**

Key-word used to introduce a series of additional parameters.

**LECTURE**

List of the elements concerned.

**Comments:**

The number **num** enables several materials chosen by the user to be recognized. The single parameter “MASS” is mandatory.

The user specifies his material’s parameters after the keyword “PARAM”. When EUROPLEXUS finds the keyword “LECTURE”, it considers that the list of parameters is terminated, whatever the number of values that have been read.

However, the total number of parameters for this material may not exceed 100, including the single mandatory value (**m**).

If there are no additional parameters besides the mandatory one, the keyword “PARAM” may be omitted.

The parameters are used within the subroutine “MPLIBR” that must be written by the user, compiled and linked with the code libraries to produce a special code executable before launching the run.

The only element that accepts the free material of type "POINT MATERIEL" is "PMAT", that is always 3-D.

Be careful to respect the conventions chosen to rank the tensor components according to the 3-D cases. See page G.20 for further explanations.

The user can store for each element (and each integration point), the values he wants (up to 10) in the ECR table. The ten locations are free.

### Examples:

The following example, taken from the standard benchmark "bm\_str\_terlun", treats the case of two pointwise masses that attract each other according to the universal gravitation law.

The material data are as follows:

#### MATERIAUX

```
!--                                cte G    nbr    pt_lune
LIBRE  PMAT 101  MASS 1.00    PARAM 1.14e4    1      2
                                LECT  terre TERM

!--                                cte G    nbr    pt_terre
LIBRE  PMAT 101  MASS 0.0123  PARAM 1.14e4    1      2
                                LECT  lune  TERM

!                                param(1) = constante de gravitation
!                                param(2) = nbr de noeuds attires par cet element
!                                param(3) = liste des noeuds (ici un seul)
```

In this case there are two materials, whose user index is the same (101), but which have different parameters. Lines starting by a "!" are comments.

The used units are adapted to the treated problem. For masses, the reference is the earth mass, for lengths the earth radius and for times the day.

There are three additional parameters besides the mandatory one, that are respectively the gravity constant, the number of nodes subjected to gravity (here just one) and the index of the concerned node.

### Programming example relative to MPLIBR:

```
SUBROUTINE MPLIBR(NUM,T,PARAM,AMAS,ECR,X,U,F,V,DTSTAB)
*
* -----
*
*      materiau libre pour les points mat.      m.lepareux 08-95
*
* -----
*
*  entree :
*
*      num      : numero de reperage pour l'utilisateur
*      t        : temps
*      param    : tableau des parametres du mat. libre
```

```

*          amas      : masse de l'element
*          x          : coordonnees
*          u          : déplacements
*          v          : vitesses
*
*  sortie :
*          ecr        : tableau des parametres du materiau
*          f          : forces internes
*
*  attention ! le materiau 101 est utilise par le benchmark
*              "bm_str_terlun.epx"
*
*  IMPLICIT NONE
*
*  --- variables globales :
*  INTEGER, INTENT(IN) :: NUM
*  REAL(8), INTENT(IN) :: T,PARAM(*),AMAS,X(3),U(3),V(3)
*  REAL(8), INTENT(OUT) :: F(3)
*  REAL(8), INTENT(INOUT) :: ECR(*),DTSTAB
*
*  --- variables locales :
*  INTEGER LON,NOD,NBR,II
*  REAL(8) GG,FF,R(3),R2,R3,AMB,XB(3),AUX,EPSI,BM(3)
*  REAL(8) CX,SS,ROL,V2,VV,COEF
*
*
*  SELECT CASE (NUM)
*  CASE (101)
*  !--- loi de gravitation universelle :
*  !-- (l'element affecte de ce materiau agit sur les noeuds designes)
*  !-- param(2+1:2+nbr) ! numeros des noeuds designes
*  GG = PARAM(1) ! CONSTANTE G DE GRAVITATION
*  NBR = PARAM(2) ! NOMBRE DE NOEUDS SOUMIS A L'ATTRACTION
*  DTSTAB = 1000.
*  EPSI = 0.01
*
*  F(:) = ODO
*  DO II=1,NBR
*    NOD = PARAM( 2 + II )
*    CALL QUIDNE( 0,NOD,LON,XB ) ! XB = POSITION DU NOEUD CIBLE
*    CALL QUIDNE( 5,NOD,LON,BM )
*    AMB = BM(1) ! MASSE DU NOEUD CIBLE
*
*    R(:) = X(:) - XB(:)
*    R2 = R(1) * R(1) + R(2) * R(2) + R(3) * R(3)
*    R3 = R2 * SQRT(R2)
*    FF = GG * AMAS * AMB / R3
*    AUX = SQRT( R3 / ( GG * AMB ) )
*    DTSTAB = MIN( DTSTAB , AUX * EPSI )
*
*    F(:) = F(:) + R(:) * FF
*  END DO
*  CASE DEFAULT
*    CALL ERRMSS('MPLIBR','NUMERO DE MATERIAU LIBRE INCONNU')
*    STOP ' MPLIBR'
*  !
*  END SELECT
*
*  END

```

### 7.7.36 FREE MATERIAL OF TYPE MECHANISM

**Object:**

This directive introduces a user-defined constitutive behaviour of mechanism type (“MECANISME”).

**Syntax:**

```
"LIBR" "MECA" num    < "PARA"  a b c ... > /LECTURE/
```

**"MECA" num**

Indicates that the free material of type “MECANISME” has the user-specified index **num**.

**"PARA" ...**

Key-word used to introduce a series of additional parameters.

**LECTURE**

List of the elements concerned.

**Comments:**

The number **num** enables several materials chosen by the user to be recognized. There are no mandatory parameters.

The user specifies his material’s parameters after the keyword “PARAM”. When EUROPLEXUS finds the keyword “LECTURE”, it considers that the list of parameters is terminated, whatever the number of values that have been read.

However, the total number of parameters for this material may not exceed 100.

If there are no parameters the keyword “PARAM” may be omitted.

The parameters are used within the subroutine “MMLIBR” that must be written by the user, compiled and linked with the code libraries to produce a special code executable before launching the run.

The only element that accepts the free material of type “MECANISME” is ”MECA”, that is an element with two nodes and 6 degrees of freedom per node.

The main interest of this free material is to allow the user to specify arbitrary relations (in matricial form) between the displacements of the two nodes and the applied forces. For example, it is possible to enter a symmetric stiffness matrix in order to model a complicated support (78

values). However, it must be noted that the relations must be specified in (or converted to) the global reference frame, and that they stay constant during the whole transient calculation.

The user can store for each element (and each integration point), the values he wants (up to 10) in the ECR table. The ten locations are free.

### Examples:

The following example, taken from the standard benchmark "bm\_str\_meca\_lbr", treats the very simple case of springs in translation and rotation.

The 78 values are specified in the global reference frame, and the principal axes are parallel to the global ones. The translational stiffness is  $K_T = 1E3$ , and the rotational one is  $K_R = 4E6$ . The free material has the user-specified index 905.

The material data are as follows:

#### MATERIAUX

```
LIBRE  MECA  905  PARAM
      1e3
      0.0  1e3
      0.0  0.0  1e3
      0.0  0.0  0.0  4e6
      0.0  0.0  0.0  0.0  4e6
      0.0  0.0  0.0  0.0  0.0  4e6
     -1e3  0.0  0.0  0.0  0.0  0.0  1e3
      0.0 -1e3  0.0  0.0  0.0  0.0  0.0  1e3
      0.0  0.0 -1e3  0.0  0.0  0.0  0.0  0.0  1e3
      0.0  0.0  0.0 -4e6  0.0  0.0  0.0  0.0  0.0  4e6
      0.0  0.0  0.0  0.0 -4e6  0.0  0.0  0.0  0.0  0.0  4e6
      0.0  0.0  0.0  0.0  0.0 -4e6  0.0  0.0  0.0  0.0  0.0  4e6
      LECT  L_meca TERM
```

### Programming example relative to MMLIBR:

```
SUBROUTINE MMLIBR(NUM,TT,NBPAR,XMAT,X,DU,F,XMA,V,DFX,
&                ECR,SIG,DEPS,PI,DWINT,IEL,DTSTAB)
*
* -----
*
*          materiau libre mecanisme          m.lepareux 12-00
*
* -----
*
* entree :
*   num      = numero de repereage du materiau utilisateur
*   tt       = temps du calcul
*   nbpar    = nombre de parametres utilisateur
*   xmat(1:) = parametres du materiau
*   x(1:3,1:2) = coordonnees des 2 noeuds
*   du(1:12)  = déplacements des 2 noeuds
*   xma(1:12) = masses des 2 noeuds
*   v(1:12)   = vitesses des 2 noeuds
*   dfx(1:12) = déplacements cumules des 2 noeuds
*   sig(1:6)  = forces internes au debut du pas
*              en fait : sig = 0.5 * (f2 - f1)
```



```

*      deps(1:6) = deplacement relatif : deps = (u2 - u1)
*      iel       = numero de l'element
*      sorties :
*      f(1:12)   = forces internes appliquees aux 2 noeuds
*      ecr(:)    = variables internes (emplacements libres)
*      pi(1:6)   = forces internes a la fin du pas
*      dwint     = travail des forces internes
*      dtstab    = pas de stabilite
*
*      attention ! le materiau 905 est utilise par le benchmark
*                  "bm_str_meca_lbr.epx"
*
      IMPLICIT NONE
*
*---  variables globales :
      INTEGER, INTENT(IN)  :: NUM, IEL, NBPARG
      REAL(8), INTENT(IN)  :: TT, XMAT(*), X(3,2), DU(12), XMA(12),
&      V(12), DFX(12), SIG(6), DEPS(6)
      REAL(8), INTENT(OUT) :: F(12), PI(6), DWINT
      REAL(8), INTENT(INOUT) :: ECR(*), DTSTAB
*
*---  variables locales :
      INTEGER :: K, I, II
      REAL(8) :: R_K(12,12), RT(3), RR(3), DT(12)
*
*
      SELECT CASE (NUM)
      CASE(905)
        IF(NBPARG /= 78) THEN
          CALL ERRMSS('MMLIBR', 'IL FAUT 78 VALEURS')
          STOP ' MMLIBR'
        ENDIF
*
*--      construction de la matrice de raideur :
        II = 0
        DO K=1,12
          DO I=1,K
            II = II+1
            R_K(K,I) = XMAT(II)
            R_K(I,K) = R_K(K,I)
          END DO
        END DO
*
*--      calcul direct des forces internes :
        F(:) = 0D0
        DO K=1,12
          DO I=1,12
            F(K) = F(K) + R_K(K,I)*DFX(I)
          END DO
        END DO
*
*--      nouvelles forces internes (pour calculer wint)
        DO K=1,6
          PI(K) = 0.5D0 * (F(K+6) - F(K))
        END DO
*
*--      travail des forces internes (pendant le pas de temps)
        DWINT = 0D0
        DO K=1,6
          DWINT = DWINT + 0.5D0*(SIG(K)+PI(K))*DEPS(K)
        END DO
*
*--      ECR : variables internes (allongement)
        DO K=1,3
          RT(K) = DFX(K+6) - DFX(K)
          RR(K) = DFX(K+9) - DFX(K+3)
        END DO
        ECR(1) = SQRT(RT(1)*RT(1) + RT(2)*RT(2) + RT(3)*RT(3))
        ECR(2) = SQRT(RR(1)*RR(1) + RR(2)*RR(2) + RR(3)*RR(3))
*
*--      calcul du pas de stabilite :
        DTSTAB = 1000D0
        DO K=1,12
          DT(K) = SQRT(XMA(K)/R_K(K,K))
          DTSTAB = MIN(DTSTAB, 2*DT(K))
        END DO
*
      CASE DEFAULT
        CALL ERRMSS('MMLIBR', 'ROUTINE UTILISATEUR NON PROGRAMMEE')
        STOP ' "MMLIBR" ABSENT'
      END SELECT
*
      END

```

### 7.7.37 FREE MATERIAL OF TYPE BOUNDARY CONDITIONS

**Object:**

This directive introduces a user-defined constitutive behaviour of the boundary condition type (“CONDITION AUX LIMITES”).

**Syntax:**

```
"LIBR" "CLIM" num "PREF" pref < "PARA" a b c ... > /LECTURE/
```

"CLIM" num

Indicates that the free material of type “CONDITION AUX LIMITES” has the user-specified index **num**.

"PREF" pref

Reference pressure.

"PARA" ...

Key-word used to introduce a series of additional parameters.

LECTURE

List of the elements concerned.

**Comments:**

The number **num** enables several materials chosen by the user to be recognized. The only mandatory parameter is **pref**.

The user specifies his material's parameters after the keyword “PARAM”. When EUROPLEXUS finds the keyword “LECTURE”, it considers that the list of parameters is terminated, whatever the number of values that have been read.

However, the total number of parameters for this material may not exceed 100, including the single mandatory value **pref**.

If there are no additional parameters besides the mandatory one, the keyword “PARAM” may be omitted.

The parameters are used within the subroutine “CLIBRE” that must be written by the user, compiled and linked with the code libraries to produce a special code executable before launching the run.

The elements that accept the free material of type "CONDITION AUX LIMITES" are "CL1D" and "CLTU", which are respectively an element with one node and one dof and an element with one node and 7 dofs.

The main interest of this free material is to allow the user to specify boundary conditions applied to the 'fluid' degree of freedom of these elements, e.g. in order to model a special device mounted along a pipeline. In the case of "CLTU", only the 7-th dof is affected. The first 6 dofs concern the structure and are not affected.

The user can store for each element (and each integration point), the values he wants (up to 10) in the ECR table. For homogeneity with the other materials, the following data will be stored in the first two locations of the ECR table :

ECR(1) = DP : variation of pressure due to the device

ECR(2) = Density of the donor element

The eight other locations are free.

### Examples:

The following example, taken from the standard benchmark "bm\_cir\_conteneur\_eau", concerns the case of the brutal opening of a pressure container.

The container top is detached from the body and the opening (assumed circumferential) grows as the top moves away. The motion is parallel to the axis  $O_x$ . As a consequence of the detachment, the cross-section of the diaphragm across which the internal fluid passes is gradually increased, and the corresponding pressure drop is modified accordingly, as a function of the top distance.

The free material is identified by the index 906. Lines starting by a "!" are comments.

The material data are as follows:

### MATERIAUX

```

LIBRE CLIM 906 PREF 10E5
PARAM 1 25 22 0.1857
!-- ptfond ptcouv eldon diam
90e5 10e5 0.0 1.0
!-- pamon pext tau ksi idel'cik (sortie)
LECT esort TERM
!
! materiau libre 906 (ouverture circonferentielle) :
! param(1) = numero du premier point
! param(2) = numero du second point
! param(3) = numero de l'element donneur
! param(4) = diametre du tube
! param(5) = pression amont initiale
```

```
!           param(6) = pression externe
!           param(7) = constante de temps pour l'ouverture
!           param(8) = perte de charge en sortie (idel'cik)
!
```

### Programming example relative to CLIBRE:

```
SUBROUTINE CLIBRE(NUNU,PREF,PARAM,AIRE,RHO,PAMON,VN,T,ECR,DP)
*
* -----
*
*           materiau libre pour el. clld           m.lepareux 08-95
*
* -----
*
*   attention !
*
*   le materiau nunu = 906 est utilise par "bm_cir_conteneur_eau.epx"
*
*   entree (ne pas les modifier) :
*   nunu      : numero de reperage pour l'utilisateur
*   pref      : pression de reference (obligatoire)
*   param     : tableau des parametres du mat. libre
*   aire      : section de la tuyauterie
*   rho       : masse volumique amont
*   pamon     : pression amont
*   vn        : vitesse du fluide dans la tuyauterie
*   t         : temps
*
*   sortie :
*   dp        : variation de pression due a l'appareil
*   ecr(1)    : affecte a dp
*   ecr(2)    : affecte a rho amont
*   ecr(3:9)  : selon utilisateur
*
*
*   les ecr libres permettent la sortie graphique
*   des grandeurs qui leur sont affectees
*
*   IMPLICIT NONE
*
*-- variables globales :
INTEGER, INTENT(IN) :: NUNU
REAL(8), INTENT(IN) :: PREF,AIRE,RHO,PAMON,VN,T,PARAM(*)
REAL(8), INTENT(OUT) :: ECR(*),DP
*
REAL(8), PARAMETER :: ZERO=1D-6, RMIN=1D-3, RS2MIN=1.005D0
INTEGER, PARAMETER :: LON1=7
*
*-- variables locales :
INTEGER NP1,NP2,LON,NELDON,KAS
REAL*8 DIAM,PZERO,PAVAL,TAU,PEXT,VAL1(LON1),VAL2(LON1),DIST,SECT,
&      RAP,RS2,XKZ,PSEUIL,P0,XK,Q0,PP,ZMACH,
&      DPE,DPK,ROVK,DPMAX,ROVN,XKSI,CSON
LOGICAL OUVERT
!
!
SELECT CASE (NUNU)
CASE (906)
!-- diaphragme pour une ouverture progressive
NP1 = NINT(PARAM(1)) ! NUMERO DU PREMIER POINT
NP2 = NINT(PARAM(2)) ! NUMERO DU DEUXIEME POINT
NELDON = NINT(PARAM(3)) ! NUMERO DE L'ELEMENT DONNEUR
DIAM = PARAM(4) ! DIAMETRE DU TUBE
PZERO = PARAM(5) ! PRESSION AMONT INITIALE
PAVAL = PARAM(6) ! PRESSION EXTERNE
TAU = PARAM(7) ! CONSTANTE DE TEMPS POUR L'OUVERTURE
XKSI = PARAM(8) ! PERTE DE CHARGE EN SORTIE (IDEL'CIK)
!
OUVERT = .TRUE.
!
!   on va chercher les déplacements des 2 noeuds :
CALL QUIDNE (1,NP1,LON,VAL1)
CALL QUIDNE (1,NP2,LON,VAL2)
IF(LON > LON1) STOP ' CLIBRE : DIM INSUFFISANTES'
DIST = ABS( VAL2(1) + VAL1(1) )
SECT = 3.1416 * DIAM * DIST
RAP = SECT / AIRE
!
!-- le rapport des sections (rap) est limite a RMIN
!   (cas des petites ouvertures) :
IF( RAP .LT. RMIN ) THEN
RAP = RMIN
OUVERT = .FALSE.
ENDIF
RS2 = 1 / ( RAP*RAP )
!
!-- rs2 est limite a rs2min (cas des grandes ouvertures) :
IF(RS2 < RS2MIN) THEN
RS2 = RS2MIN
```

```

        RAP = SQRT(1/RS2MIN)
    ENDIF
!
!--      perte de charge (idel'cik) :
    XK = RS2 * XKSI
!
!-----  si tau .ne. 0 la pression aval chute progressivement :
!          (a condition que pamon > paval)
!
    PSEUIL = PZERO - PAVAL
    IF (TAU.GT.ZERO .AND. PSEUIL.GT.ZERO*PZERO) THEN
        PEXT = PAVAL + PSEUIL*EXP( -T / TAU)
    ELSE
        PEXT = PAVAL
    ENDIF
!
    DPMAX = PAMON - PREF
    IF (OUVERT) THEN
        DPE = PEXT - PREF
        DPK = 0.5*XK * RHO * VN*VN
    ELSE
        DPE = PAMON - PREF
        DPK = 0
    ENDIF
    DP = DPE + DPK
    IF (DP > DPMAX) DP = DPMAX
!
    ECR(1) = DP + PREF
    ECR(2) = RHO
    ECR(3) = RHO*VN      ! PRODUIT RHO*VN (DEBIT MASSIQUE UNITAIRE)
    ECR(4) = PEXT        ! PRESSION DE SORTIE ( PEXT OU PCRIT )
    ECR(5) = DIST        ! DISTANCE ENTRE LES FRAGMENTS
    ECR(6) = XK          ! COEF. DE PERTE DE CHARGE
!
!-----  coefficient de stabilite ( xk * rho * vn ) :
    ROVK = ABS(ECR(6)*ECR(3))
!
!--  on arrete le calcul quand pamon < pext :
    IF ( PAMON < PEXT )    CALL TILT
!
    CASE DEFAULT
        CALL ERRMSS('CLIBRE','ROUTINE UTILISATEUR NON PROGRAMMEE')
        STOP ' "CLIBRE" ABSENT'
    END SELECT
!
    RETURN
END

```

### 7.7.38 FREE PARTICLE MATERIAL

**Object:**

This directive allows users to define their own constitutive laws for the particle elements (BILLE).

**Syntax:**

```
"BILLE" $ "LIBR" num "RO" rho $  
    ... < "FONC" numfon >  
    ... < "PARA" a b c ... > /LECTURE/
```

**num**

Number of the material used for the particle elements.

**rho**

Density.

**numfon**

Number of the function used.

**"PARA" ...**

Introduces a series of complementary parameters.

**LECTURE**

List of the concerned elements.

**Comments:**

The number (num) allows to distinguish between several user-defined materials.

The rho parameter is mandatory.

The number (numfon) allows to identify the function used for the interaction.

The complementary parameters introduced by "PARA" may be as many as needed. EUROPLEXUS recognizes the end of the parameters when the "LECTURE" keyword is encountered.

The subroutine "MBLIBR", to be written by the user, computes the interaction forces between neighbouring particles of the "BILLE" element considered, starting from the quantities at the beginning of the step, which are known. Consult the following example for a list of the available variables.

The only element type accepting this material is "BILLE".

The user may store for each element the variables of his choice within the ECR table (up to 7 values). However, for uniformity with the other materials, it is advised to use the first two slots as follows:

Fluid :

```
ECR(1) = Pressure
ECR(2) = Density
```

Continuum structure:

ECR(1) = Pressure  
ECR(2) = Von Mises

**Example:**

Two new materials of the fluid type are defined: a material of type acoustic fluid, and the other depending upon the distance between two neighbouring particles.

The corresponding data will be, for example:

"BILL" "LIBR" 1 "RO" 1000 "PARA" 1000 /LECTURE/  
"BILL" "LIBR" 2 "RO" 800 "FONC" 1 /LECTURE/

### Programming example for routine MBLIBR:

```

SUBROUTINE MBLIBR(XMAT,DINI,DIST,A,B,C,NVOIS,NUMVOI,DVX,DVY,DVZ,
* ROCOUR,IEL,INOE,INOEV,IPFONC,TABFON,T,DT1,FORCE,SIG,ECR)
-----
MATERIAU "BILLE" "LIBRE"
-----
R.GALON 02/91
-----
ENTREE :
-----
XMAT(1)      = MASSE VOLUMIQUE INITIALE
XMAT(2)      = NUMERO DE REPERAGE DU MATERIAU UTILISATEUR
XMAT(3)      = NUMERO DE LA FONCTION ASSOCIEE
XMAT(4: )    = AUTRES PARAMETRES DU MATERIAU
DINI         = DIAMETRE INITIAL DE LA BILLE
DIST         = DISTANCE SEPARANT LES 2 BILLES EN INTERACTION
A            = COSINUS DIRECTEUR SUIVANT X DE LA LIAISON
B            = COSINUS DIRECTEUR SUIVANT Y DE LA LIAISON
C            = COSINUS DIRECTEUR SUIVANT Z DE LA LIAISON
NVOIS        = NOMBRE DE BILLES VOISINES DE LA BILLE TRAITEE
NUMVOI       = NUMVOI-IEME BILLE EN INTERACTION
DVX          = VITESSE RELATIVE DES 2 BILLES DE LA LIAISON
              SUIVANT X
DVY          = VITESSE RELATIVE DES 2 BILLES DE LA LIAISON
              SUIVANT Y
DVZ          = VITESSE RELATIVE DES 2 BILLES DE LA LIAISON
              SUIVANT Z
ROCOUR       = MASSE VOLUMIQUE ASSOCIEE A LA LIAISON
IEL          = NUMERO DE L ELEMENT TRAITE

```

```

C      INOE      =  NUMERO DU NOEUD ASSOCIE A L ELEMENT IEL
C      INOE      =  NUMERO DU NOEUD VOISIN DE LA BILLE
C      IPFONC     =  POINTE SUR LA TABLE DE FONCTION
C      TABFON     =  TABLE DE FONCTION ASSOCIEE AU MATERIAU
C      T          =  TEMPS DE CALCUL
C      DT1        =  INCREMENT DE TEMPS DE CALCUL
C
C
C      SORTIE :
C      -----
C      FORCE(1:3)  =  FORCES A APPLIQUER A LA BILLE TRAITEE
C      SIG(1:6)   =  CONTRAINTES A LA FIN DU PAS (FACULTATIF)
C      ECR(1:10)  =  EMPLACEMENTS LIBRES
C
C      REMARQUE : - DEUX BILLES SEPARÉES DE PLUS DE 1.3 * DINI SONT
C      -----      SUPPOSEES NE PAS POUVOIR INTERAGIR ENTRE ELLES.
C
C      - ON CUMULE TOUJOURS LES FORCES CAR ELLES PROVIENNENT
C      DE L INTERACTION DE TOUTES LES BILLES VOISINES DE LA
C      BILLE TRAITEE.
C
C      IMPLICIT REAL*8(A-H,O-Z)
C
C      DIMENSION XMAT(*),ECR(*),FORCE(*),TABFON(*),SIG(*),IPFONC(2,*)
C
C
C      NUM = XMAT(2)
C
C      IF(NUM.NE.1) GOTO 20
C
C      ----- CAS D UN MATERIAU DE TYPE FLUIDE
C      =====
C      RO = XMAT(1)
C      CSON = XMAT(4)
C      ----- POUR LA PREMIERE BILLE EN INTERACTION ON INITIALISE PAR EXEMPLE
C      LA MASSE VOLUMIQUE ET LA PRESSION MOYENNE DE L ELEMENT BILLE IEL
C      IF(NUMVOI.EQ.1)THEN
C          ECR(1)=0.
C          ECR(2)=0.
C          SIG(1)=0.
C          SIG(2)=0.
C          SIG(3)=0.
C
C      ENDIF
C      ----- DRO = VARIATION DE LA MASSE VOLUMIQUE
C      DRO = ROCOUR - RO
C      P = DRO * CSON * CSON
C      DVOL = (ROCOUR/RO) -1.DO
C      DP2 = DINI**3 / (DIST*(1.DO + DVOL))
C
C      ----- COEFICIENT DE PONDERATION POUR UN RESEAU HEXAGONAL DE BILLES
C      COEF = SQRT(2.DO)/4.DO
C
C      ----- COEFICIENT DE PONDERATION POUR UN RESEAU CUBIQUE DE BILLES
C      COEF = 1.DO
C      ----- FORCE DANS LA DIRECTION DE LA LIAISON APPLIQUEE A LA BILLE
C      FN = - DP2 * COEF * P
C      ----- ON PROJETTE LA FORCE DANS LE REPERE GLOBAL
C      FORCE(1) = FORCE(1) + A * FN
C      FORCE(2) = FORCE(2) + B * FN
C      FORCE(3) = FORCE(3) + C * FN
C      ----- CONTRAINTES DANS L ELEMENT (PRESSIONS)
C
C      SIG(1) = SIG(1) + P/NVOIS
C      SIG(2) = SIG(2) + P/NVOIS
C      SIG(3) = SIG(3) + P/NVOIS
C      SIG(4) = 0.
C      SIG(5) = 0.
C      SIG(6) = 0.
C      ----- MASSE VOLUMIQUE MOYENNE
C      ECR(1) = ECR(1) + ROCOUR/NVOIS
C      ----- PRESSION MOYENNE
C      ECR(2) = ECR(2) + P/NVOIS
C      RETURN
C
C
C      20  CONTINUE
C
C      ----- FORCE DEFINIE PAR UNE FONCTION
C      =====
C
C      REMARQUE : ON SUPPOSE ICI QUE LA FORCE AGISSANT SUR L ELEMENT BILLE
C      EST FONCTION UNIQUEMENT DE LA DISTANCE SEPARANT LES 2
C      BILLES EN INTERACTION (LA FONCTION EST DEFINIE PAR LA
C      DIRECTIVE "FONC".
C
C      IFONC = XMAT(3)
C
C      ----- FN EST LA FORCE CORRESPONDANT A UNE DISTANCE DIST SEPARANT LES
C      2 BILLES EN INTERACTION (ELLE EST APPLIQUEE A L ELEMENT IEL)
C
C      CALL FFONCT(IFONC,DIST,FN,IPFONC,TABFON)
C
C      ----- ON PROJETTE LA FORCE DANS LE REPERE GLOBAL
C
C      FORCE(1) = FORCE(1) + A * FN
C      FORCE(2) = FORCE(2) + B * FN
C      FORCE(3) = FORCE(3) + C * FN
C
C      ----- DISTANCE MOYENNE DANS ECR(1) PAR EXEMPLE OU TOUTE AUTRE VALEUR

```



```
C      QUE L ON DESIRE CONSERVER
C
C      ECR(1) = ECR(1) + DIST/NVOIS
C
C      RETURN
C      END
```

### 7.7.39 VON MISES (ISPRA IMPLEMENTATION)

#### Object:

This option enables to choose the Von Mises material with the implementation developed at Ispra. Elasto-plasticity is implemented via a radial return algorithm. Only isotropic hardening is activated to date. There is no dependency on temperature nor on strain rate.

#### Syntax:

```
"VM23" ![ "RO" rho "YOUN" young "NU" nu "ELAS" sige ...  
          <"FAIL" $[ VMIS ; PEPS ; PRES ; PEPR ]$ "LIMI" limit>  
          "TRAC" npts*(sig eps) ]!  
          /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

sige

Elastic limit.

FAIL

Optional keyword: introduces an element failure model, represented by a failure criterion and a by failure limit value. The available failure criteria are: **VMIS** for a criterion based upon Von Mises stress (isotropic criterion), **PEPS** for a criterion based upon the principal strain (see caveat below), **PRES** for a criterion based upon the hydrostatic stress, **PEPR** for a criterion based upon the principal strain if the hydrostatic stress is positive (traction): if the hydrostatic stress is negative (compression) there is no failure.

limit

Optional parameter, indicates the failure limit for the chosen criterion.

"TRAC"

This key-word announces the yield curve.

npts

Number of points (except the origin) defining the yield curve.

sig

Stress.

eps

Total strain (elastic + plastic).

#### LECTURE

List of the elements concerned.

#### Comments:

1/ - The young parameter defines Young's modulus during an elastic phase.

2/ - The points (sig, eps) may have any position; however, concerning the first point, there must be a compatibility between the coordinates, Young's modulus and the elastic limit.

3/ - The slope of the yield curve may not increase from one segment to the following one.

4/ - When using a failure criterion based upon the principal strains (PEPS or PEPR) be aware that the criterion is based upon the *cumulated* strains. These are usually a good approximation of the total strains for elements using a convected reference frame for the stresses and strains (such as e.g. plate, shell or bar elements). The approximation is likely to be very bad, instead, for continuum-like elements, at least when there are large rotations.

#### Outputs:

The components of the ECR table are as follows:

ECR(1): current hydrostatic pressure

ECR(2): current equivalent stress (Von Mises)

ECR(3): current equivalent plastic strain

ECR(4): current yield stress

ECR(5): sound speed

ECR(6): failure flag (0=virgin Gauss Point, 1=failed Gauss Point)

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

**7.7.40 VM1D MATERIAL****Object:**

This is the material to be used for the interface elements of type ED1D (see INT.80).

**Syntax:**

```
"VM1D"  "PT1D"  pt1d /LECTURE/
```

pt1d

Associated node index in the 1-D model.

**Comments:**

Note that when several ED1D elements are present in a coupled 1-D/multi-D calculation, then each ED1D element must have a separate VM1D material, because the material is used to carry the information of the associated 1-D node to each one ED1D element (pt1d).

**Outputs:**

The components of the ECR table are as follows :

ECR(1) : unused

ECR(2) : unused

ECR(3) : unused

ECR(4) : unused

ECR(5) : unused

ECR(6) : unused

### 7.7.41 DONE MATERIAL

**Object:**

This is a viscoplastic material model mostly used to describe the sensitivity of commonly used stainless steels (e.g. AISI 304 and 316) to the rate of loading. It uses the theory of viscoplasticity based on total strain and overstress. To date, it is limited to small strains.

**Syntax:**

```
"DONE"  "RO" rho  "YOUN" young  "NU" nu  "ELAS" sige ...  
...  "VIS1" vis1  "VIS2" vis2  "VIS3" vis3  ...  
...  "VIS4" vis1  "VIS5" vis2  "VIS6" vis3  ...  
...  "TRAC" npts*(sig eps) /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

sige

Elastic limit.

vis1,...,vis6

Viscous coefficients.

"TRAC"

This key-word announces the yield curve (static).

npts

Number of points (except the origin) defining the static yield curve.

sig

Stress.

eps

Total strain (elastic + plastic).

LECTURE

List of the elements concerned.

**Comments:**

1/ - The young parameter defines Young's modulus during an elastic phase.

2/ - The points (sig, eps) may have any position; however, concerning the first point, there must be a compatibility between the coordinates, Young's modulus and the elastic limit.

3/ - The slope of the static yield curve may not increase from one segment to the following one.

**Outputs:**

The components of the ECR table are as follows:

- ECR(1): current hydrostatic pressure
- ECR(2): current equivalent stress (Von Mises)
- ECR(3): current equivalent plastic strain
- ECR(4): current yield stress
- ECR(5): x-overstress
- ECR(6): y-overstress
- ECR(7): xy-overstress
- ECR(8): z-overstress
- ECR(9): previous time
- ECR(10): yz-overstress
- ECR(11): xz-overstress

Let P represent the point of intersection (in the equivalent stress - equivalent strain space) between the unloading path and the equilibrium stress-strain curve.

- ECR(12): total x-strain at point P
- ECR(13): total y-strain at point P
- ECR(14): total z-strain at point P
- ECR(15): total xy-strain at point P
- ECR(16): total yz-strain at point P
- ECR(17): total xz-strain at point P
- ECR(18): equivalent total strain at point P
- ECR(19): old equivalent total strain
- ECR(20): EPSC (equivalent strain parameter)

EPSC is defined by the cyclic hardening law. It corresponds to the distance between point P and the new origin in the strain direction.

ECR(21): current cumulative value of number of crossings of the unloading path with the equilibrium stress-strain diagram

ECR(22): new x-stress at point P

ECR(23): new y-stress at point P

ECR(24): new xy-stress at point P

ECR(25): new z-stress at point P

ECR(26): new yz-stress at point P

ECR(27): new xz-stress at point P

ECR(28): new equivalent stress at point P

ECR(29): old equivalent stress

ECR(30): old equilibrium equivalent stress

ECR(31): old (Young's modulus \* total strain)

ECR(32): sound speed

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

**7.7.42 VON MISES WITH VISCOPLASTIC REGULARIZATION****Object:**

This directive enables to choose an elastoplastic constitutive theory with Von Mises yield surface, associative flow rule, and isotropic hardening or softening, including a viscoplastic regularization. Elasto-plasticity is implemented via a radial return algorithm.

For more information about the theory, please refer to: J.C. Simo, J.G. Kennedy and S. Govindjee, "Non-Smooth Multisurface Plasticity and Viscoplasticity. Loading/Unloading Conditions and Numerical Algorithms", Int. J. Num. Meth. Eng., Vol 26, pp. 2161-2185 (1988).

**Syntax:**

```
"VMSF" "RO" rho "YOUN" young "NU" nu "ELAS" sige "ETA" eta  
... "TRAC" npts*(sig eps) /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

sige

Elastic limit.

eta

Viscoplastic parameter (relaxation time).

"TRAC"

This key-word announces the yield curve.

npts

Number of points (except the origin) defining the yield curve.

sig

Stress.

eps

Total strain (elastic + plastic).



**LECTURE**

List of the elements concerned.

**Comments:**

- 1/ - The young parameter defines Young's modulus during an elastic phase.
- 2/ - The points (sig,eps) may have any position; however, concerning the first point, there must be a compatibility between the coordinates, Young's modulus and the elastic limit.
- 3/ - The slope of the yield curve may become negative in the softening part of the curve.

**Outputs:**

The components of the ECR table are as follows:

- ECR(1): current hydrostatic pressure
- ECR(2): current equivalent stress (Von Mises)
- ECR(3): current equivalent plastic strain
- ECR(4): current yield stress
- ECR(5): x-stress before viscoplastic correction
- ECR(6): y-stress before viscoplastic correction
- ECR(7): xy-stress before viscoplastic correction
- ECR(8): z-stress before viscoplastic correction
- ECR(9): yz-stress before viscoplastic correction (3D only)
- ECR(10): xz-stress before viscoplastic correction (3D only)
- ECR(11): current time
- ECR(12): sound speed

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

### 7.7.43 DRUCKER PRAGER WITH VISCOPLASTIC REGULARIZATION

#### Object

This directive enables to choose an elastoplastic constitutive theory with Drucker Prager yield surface, associative or non-associative flow rule, including hardening or softening, and a viscoplastic regularization.

This material is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC.

The regularization technique is the same as the one implemented in the VMSF material, see: J.C. Simo, J.G. Kennedy and S. Govindjee, "Non-Smooth Multisurface Plasticity and Viscoplasticity. Loading/Unloading Conditions and Numerical Algorithms", Int. J. Num. Meth. Eng., Vol 26, pp. 2161-2185 (1988).

The model uses two parameters,  $\alpha$  and  $c$ , related to the angle and the cohesion parameters of the classical Drucker Prager model. These two parameters are not constant in general, but depend on the plastic strain. Hardening and/or softening are thus possible.

#### References

More information on the formulation of this material model may be found in reference [\[120\]](#).

#### Syntax

```
"DPSF"  "RO" rho  "YOUN" young  "NU" nu
        "ALF1" alf1 "C1" c1 "BETA" beta "ETA" eta
        <"FAIL" $[ PEPS ; PEPR ]$ "LIMI" limit >
        "TRAA" npta*(alfa epsp)
        "TRAC" npts*(c      epsp) /LECTURE/
```

**rho**

Density of the material.

**young**

Young's modulus.

**nu**

Poisson's ratio.

**alf1**

First value ("yield limit") of the TRAA curve for the  $\alpha$  parameter, see below.

**c1**

First value ("yield limit") of the TRAC curve for the c parameter, see below.

**beta**

Parameter indicating whether the model is associative or non associative. If the alfa parameter (given by the "TRAA" directive below) does not depend upon the plastic strain, and  $\beta = \alpha_1$ , then an associative rule is taken. Otherwise, the law is non associative. E.g.,  $\beta = 0$  corresponds to return along a cylinder.

**eta**

Viscoplastic parameter (relaxation time).

**FAIL**

Optional keyword: introduces an element failure model, represented by a failure criterion and a by failure limit value. The available failure criteria are: **PEPS** for a criterion based upon the principal strain (see caveat below), **PEPR** for a criterion based upon the principal strain if the hydrostatic stress is positive (traction): if the hydrostatic stress is negative (compression) there is no failure.

**limit**

Optional parameter, indicates the failure limit for the chosen criterion.

**"TRAA"**

This key-word announces the curve defining the variation of the alfa parameter with the plastic strain.

**npta**

Number of points defining the curve.

**alfa**

Value of the alfa parameter.

**epsp**

Corresponding value of the plastic strain.

**"TRAC"**

This key-word announces the curve curve defining the variation of the c parameter with the plastic strain.

**npts**

Number of points defining the curve.

**c**

Value of the c parameter.

**epsp**

Corresponding value of the plastic strain.

**LECTURE**

List of the elements concerned.

### Comments:

When using a failure criterion based upon the principal strains (PEPS or PEPR) be aware that the criterion is based upon the *cumulated* strains. These are usually a good approximation of the total strains for elements using a convected reference frame for the stresses and strains (such as e.g. plate, shell or bar elements). The approximation is likely to be very bad, instead, for continuum-like elements, at least when there are large rotations.

The parameter ETA can be effectively used to obtain a mesh size independence in case of static or quasi-static calculations. The parameter is very sensitive in case of dynamic simulations and must be set with care. It is recommended to set this parameter to 0 in case of fast dynamic simulations.

### Outputs:

The components of the ECR table are as follows:

- ECR(1): current hydrostatic pressure
- ECR(2): current equivalent stress (Von Mises)
- ECR(3): current equivalent plastic strain
- ECR(4): cohesion
- ECR(5): x-stress before viscoplastic correction
- ECR(6): y-stress before viscoplastic correction
- ECR(7): xy-stress before viscoplastic correction
- ECR(8): z-stress before viscoplastic correction
- ECR(9): yz-stress before viscoplastic correction (3D only)
- ECR(10): xz-stress before viscoplastic correction (3D only)
- ECR(11): current time
- ECR(12): alfa
- ECR(13): zone (sigma - tau plane)
- ECR(14): yield ( $f = \alpha * \sigma + \tau - \text{cohe}$ ),  $>0$  if plast
- ECR(15): failure flag (0=virgin Gauss Point, 1=failed Gauss Point).
- ECR(16): sound speed

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

**7.7.44 COMPOSITE MATERIAL (LINEAR ORTHOTROPIC) ISPRA IMPLEMENTATION****Object:**

The option is used to enter materials with a linear orthotropic behaviour into a coordinate system defined by the user. The model is suitable to represent e.g. composite materials.

**Syntax:**

```
"COMM"  "R0"  rho    "YG1"  yg1    "YG2"  yg2  "YG3"  yg3
          "NU12" nu12  "NU13" nu13  "NU23" nu23
          "G12"  g12   "G13"  g13   "G23"  g23
          /LECTURE/
```

rho

Density of the material.

yg1

Young's modulus along direction 1.

yg2

Young's modulus along direction 2.

yg3

young's modulus along direction 3.

nu12

Poisson's ration between direction 1 and 2.

nu 13

Poisson's ration between direction 1 and 3.

nu23

Poisson's ratio between direction 2 and 3.

g12

Shear modulus between direction 1 and 2.

g13

Shear modulus between direction 1 and 3.

g23

Shear modulus between direction 2 and 3.

#### LECTURE

List of the elements concerned.

#### Comments:

This option may be repeated as many times as necessary.

The associated orthotropy directions are to be specified via the `COMP ORTS` directive (see page C.97).

#### Outputs:

The different components of the ECR table are as follows:

ECR(1) : current hydrostatic pressure ( $1/3(SX+SY+ST)$ )

ECR(2) : current equivalent stress (von Mises)

ECR(3) : current equivalent plastic strain

ECR(4) : current yield stress

ECR(5): sound speed

ECR(6): angle alpha between lamina coordinate 1 and orthotropy direction 1

ECR(7): 10.

ECR(8): 10.

## 7.7.45 MODIFIED CAM-CLAY MATERIAL

### Object

The directive is used to enter materials with a modified Cam-clay behaviour. The model is suitable to represent e.g. cohesive soil materials.

Although the model includes some treatment of the water possibly present in soils, the use of this feature is strongly discouraged because the modeling appears somewhat inconsistent in that case: for example, water motion within the soil is not treated, water pressure is not taken into account to compute internal forces, and finally the calculation of masses seems inconsistent. To model a dry soil, just leave out the keyword ROW: then the code assumes  $\rho_w = 0$ , the value given for  $\rho$  is the density of the (dry) soil alone, and the value given for  $z_f$ , if any, is irrelevant.

### References

More information on the formulation of this material model may be found in reference [\[147\]](#).

### Syntax

```
CAMC  RO ro |[ NU nu ; G g ]|
      M m      LAM lam  K k      E e      <ROW row>
      KO kO    OCR ocr
      |[ ZF zf SLEV slev GRAV grav ; PRES pres ]|
      /LECT/
```

#### ro

Initial density  $\rho$  of the soil (including the water, if any: but see the comments above and below).

#### nu

Poisson's coefficient  $\nu$ . If this value is given, then the shear modulus  $G$  may not be given and the calculation is done with constant Poisson's coefficient ( $G$  will vary accordingly).

#### g

Shear modulus  $G$ . If this value is given, then  $\nu$  may not be given and the calculation is done with constant shear modulus ( $\nu$  will vary accordingly).

#### m

Critical state parameter  $M$ . Corresponds to the CLAY model's  $M$  parameter. For the physical meaning, see the Remarks below.

#### lam

Isotropic consolidation modulus ( $\lambda$ ).

**k**

Unloading-reloading modulus ( $\kappa$ ).

**e**

Initial void ratio  $e$ , defined as:  $e = V_{\text{Voids}}/V_{\text{Solid}}$ .

**row**

Water density  $\rho_w$ . Use 0.0 for dry soils, or just leave out this keyword since the default value is 0.0. Note that in this case the value for  $\rho$  given above indicates the density of the (dry) soil alone.

**k0**

Coefficient of earth pressure at rest ( $K_0$ ).

**ocr**

Overconsolidation ratio  $O_{cr}$ : ( $O_{cr} = 1$  for normal-consolidated soil).

**zf**

Upper level of water, i.e. water surface “vertical” coordinate ( $y$  in 2D,  $z$  in 3D). Used to compute the in-situ (initial) stress and hardening state. This quantity is unused, and thus any value may be given, e.g.  $z_f = 0$ , if the user has specified  $\rho_w = 0$  (dry soil).

**slev**

Upper level of soil, i.e. soil surface “vertical” coordinate ( $y$  in 2D,  $z$  in 3D). Used to compute the in-situ (initial) stress and hardening state.

**grav**

Acceleration of gravity along the “vertical” coordinate ( $y$  in 2D,  $z$  in 3D). Used to compute the in-situ (initial) stress and hardening state.

**pres**

Initial hydrostatic (uniform) pressure state. Note that here (but not for stresses SIG etc.) a positive value should be used to indicate an initial compression (negative stress).

**LECTURE**

List of the elements concerned.

## Comments

This option may be repeated as many times as necessary.

This material model seems unable to start from initial stress-free conditions, so that in-situ (initial) stresses should always be specified.

The initial in-situ conditions (stresses and some of the ECR components) for elements using this material are computed by using the parameters (**zf**, **slev**, **grav**) or **pres**. One and only



one of these two sets must be given. In the following discussion, the term “vertical” refers to the  $y$ -coordinate in 2D, to the  $z$ -coordinate in 3D calculations.

A) If **pres** ( $p$ ) is specified, then the initial state is uniform hydrostatic stress ( $-p$ ) all over the current CAMC material. This is typical, e.g., of simple one-element tests to check the behaviour of the constitutive law, or of simple laboratory experiments.

In this case, the code simply sets:

$$\sigma_1 = -p \quad , \quad \sigma_2 = -p \quad , \quad \sigma_3 = -p.$$

B) If (**zf**, **slev**, **grav**) are specified, then the initial conditions are computed as follows. The model assumes a horizontally stratified (homogeneous) soil, the lower part of which may contain water. The quantities  $z_f$  and  $s_{lev}$  are the vertical coordinates of the upper water and soil levels, respectively. Normally it should be  $s_{lev} > z_f$  so that the soil layer between  $z_f$  and  $s_{lev}$  is dry (no water) while the soil below that level is saturated by water.

For each element with the current CAMC material, the code computes the vertical coordinate of its centroid  $z_c$ . Then the vertical stress due to the soil weight (effective stress) is:

$$\sigma_v = -g(\rho - \rho_w)(s_{lev} - z_c),$$

where  $\rho$  is the density of the wet soil (soil plus water),  $\rho_w$  is the density of the water. Thus, the difference between the two is the density of the (dry) soil. The vertical stress may not be positive:

$$\sigma_v = \text{MIN}(\sigma_v, 0).$$

The horizontal stress is given by:

$$\sigma_h = K_0 \sigma_v,$$

where  $K_0$  is the **k0** parameter specified above. Then, the code sets:

$$\sigma_1 = \sigma_h \quad , \quad \sigma_2 = \sigma_h \quad , \quad \sigma_3 = \sigma_v.$$

In addition to soil (effective) stresses, the water pressure (hydrostatic) is also evaluated:

$$p_w = -g\rho_w(z_f - z_c).$$

The water pressure may not be positive:

$$p_w = \text{MIN}(p_w, 0).$$

This quantity is stored in ECR(7). Note, however, that the water pressure does *not* contribute to internal forces in the CAMC model: only the effective (soil) stresses are used.

Note also that if (**zf**, **slev**, **grav**) are specified one should also probably specify a “global” gravity term (equal to the value of  $g$  given above) by means e.g. of the **CHAR CONS GRAV** directive, in order to have (at least approximate) equilibrium in the initial configuration. In addition, suitable boundary conditions must also be prescribed along the envelope of the CAMC soil region.

## Outputs

The different components of the ECR table are as follows:

ECR(1) : current hydrostatic pressure  $\frac{1}{3}\text{tr}(\sigma)$

ECR(2) : square root of the second invariant of the deviatoric stress tensor  $J'_2$  (i.e. square root of Von Mises equivalent stress)

ECR(3) : current void ratio

ECR(4) : hardening parameter  $P_c$  (isotropic consolidation pressure)

ECR(5): sound speed

ECR(6): water overpressure ( $u$ )

ECR(7): initial water pressure ( $p_0$ ),  $p = p_0 + u$  ( $p$  is the total water pressure)

ECR(8): volumetric strain ( $\epsilon_V = \epsilon_x + \epsilon_y + \epsilon_z$ )

ECR(9): deviatoric strain

$$\epsilon_d = \sqrt{\frac{2}{3}[(\epsilon_x - \epsilon_y)^2 + (\epsilon_y - \epsilon_z)^2 + (\epsilon_x - \epsilon_z)^2] + (\gamma_{xy}^2 + \gamma_{yz}^2 + \gamma_{xz}^2)}$$

The components of the stress tensor are as follows:

SIG(1):  $\sigma_x$

SIG(2):  $\sigma_y$

SIG(3):  $\sigma_z$

SIG(4):  $\tau_{xy}$

SIG(5):  $\tau_{yz}$  (only 3D)

SIG(6):  $\tau_{xz}$  (only 3D)

## Remarks

Let  $M_1$  be the ratio between the second invariant of the stress tensor  $J_2$  and the first invariant of the stress tensor  $J_1$  at critical state (i.e. for stress points which lie on the failure surface). This is the quantity which is usually available from tests.

Let  $M_2$  be the ratio between the second invariant of the *deviatoric* stress tensor  $J'_2$  and the first invariant of the stress tensor  $J_1$  at critical state.

The  $M$  parameter defined above in the input syntax corresponds to  $M_1$ . However, note that in the model description of the CAMC material the quantity  $g(\theta)$  corresponds rather to  $M_2$ .

The following relation holds between the two quantities:  $M_2 = M_1/\sqrt{3}$ .

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements)

## 7.7.46 MODIFIED CAM-CLAY MATERIAL WITH VISCOPLASTIC REGULARIZATION

### Object

The option is used to enter materials with a modified Cam-clay behaviour. The model is suitable to represent e.g. (dry) soil materials. The main differences with respect to the CAMC material are that:

- the CLAY model uses a fully implicit backward algorithm for the trial stress point to return onto the yield surface (similar to the radial return algorithm);
- the model includes an optional viscoplastic regularization;
- no attempt is made to take into account the possible presence of water (this feature is dubious in the CAMC material anyway).

Like for the CAMC material, the user may choose between a calculation with constant shear modulus and one with constant Poisson's coefficient.

### References

More information on the formulation of this material model may be found in reference [\[123\]](#).

### Syntax

```
CLAY RO ro |[ NU nu ; G g ]|
      M m      LAM lam K k PO p0
      KO k0
      < BETA beta > < NUM num >
      |[ SLEV slev GRAV grav ; PRES pres ]|
      /LECT/
```

ro

Initial density  $\rho$  of the (dry) soil. Water content is not taken into account by this model.

nu

Poisson's coefficient  $\nu$ . If this value is given, then  $G$  may not be given and the calculation is done with constant Poisson's coefficient ( $G$  will vary accordingly).

g

Shear modulus  $G$ . If this value is given, then  $\nu$  may not be given and the calculation is done with constant shear modulus ( $\nu$  will vary accordingly).

m

Critical state parameter  $M$ . Corresponds to the CAMC model's  $M$  parameter. For the physical meaning, see the Remarks below.

lam

First loading slope ( $\lambda$ ). This is the slope of the normal consolidation line, divided by the reference volume  $V_\lambda$ . The normal consolidation line is defined in the plane  $[V, \ln(P)]$ , where  $V$  is the so-called specific volume ( $V = 1 + e$ ,  $e$  being the void ratio i.e. the volume of the voids divided by volume of the solid).  $V_\lambda$  is the specific volume at unit pressure.  $P$  is the pressure.

k

Unloading-reloading slope  $\kappa$ . This is the slope of the unloading-reloading line, divided by the reference volume  $V_\lambda$ . The unloading-reloading line is defined in the plane  $[V, \ln(P)]$ , where  $V$  is the so-called specific volume ( $V = 1 + e$ ,  $e$  being the void ratio i.e. the volume of the voids divided by volume of the solid).  $V_\lambda$  is the specific volume at unit pressure.  $P$  is the pressure.

p0

Initial value of the hardening parameter  $p_0$ .

k0

Coefficient of earth pressure at rest ( $K_0$ ).

beta

Relaxation modulus  $\beta$  for the viscoplastic regularization. If  $\beta = 0$ , then no regularization is performed. By default, the code assumes  $\beta = 0$ .

num

Index (integer)  $n$  used by the initialization routine INICLA in order to set some initial properties of the soil (initial stresses and initial hardening parameters). By default, the program assumes  $n = 0$ .

slev

Upper level of soil, i.e. soil surface “vertical” coordinate ( $y$  in 2D,  $z$  in 3D). Used to compute the in-situ (initial) stress and hardening state.

grav

Acceleration of gravity along the “vertical” coordinate ( $y$  in 2D,  $z$  in 3D). Used to compute the in-situ (initial) stress and hardening state.

pres

Initial hydrostatic (uniform) pressure state. Note that here (but not for stresses SIG etc.) a positive value should be used to indicate an initial compression (negative stress).

LECTURE

List of the elements concerned.

## Comments

This option may be repeated as many times as necessary.

The initial in-situ conditions (stresses and some of the ECR components) for elements using this material are computed by using the parameters (**slev**, **grav**) or **pres**. One and only one of these two sets must be given. In the following discussion, the term “vertical” refers to the  $y$ -coordinate in 2D, to the  $z$ -coordinate in 3D calculations.

A) If **pres** ( $p$ ) is specified, then the initial state is uniform hydrostatic stress ( $-p$ ) all over the current **CLAY** material. This is typical, e.g., of simple one-element tests to check the behaviour of the constitutive law, or of simple laboratory experiments.

In this case, the code simply sets:

$$\sigma_1 = -p \quad , \quad \sigma_2 = -p \quad , \quad \sigma_3 = -p.$$

B) If (**slev**, **grav**) are specified, then the initial conditions are computed as follows. The model assumes a horizontally stratified (homogeneous) soil in dry conditions, i.e. containing no water. The quantity  $s_{lev}$  is the vertical coordinate of the upper soil level.

For each element with the current **CLAY** material, the code computes the vertical coordinate of its centroid  $z_c$ . Then the vertical stress due to the soil weight (effective stress) is:

$$\sigma_v = -g\rho(s_{lev} - z_c),$$

where  $\rho$  is the density of the (dry) soil. The vertical stress may not be positive:

$$\sigma_v = \text{MIN}(\sigma_v, 0).$$

The horizontal stress is given by:

$$\sigma_h = K_0\sigma_v,$$

where  $K_0$  is the **k0** parameter specified above. Then, the code sets:

$$\sigma_1 = \sigma_h \quad , \quad \sigma_2 = \sigma_h \quad , \quad \sigma_3 = \sigma_v.$$

Note that if (**slev**, **grav**) are specified one should also probably specify a “global” gravity term (equal to the value of  $g$  given above) by means e.g. of the **CHAR CONS GRAV** directive, in order to have (at least approximate) equilibrium in the initial configuration. In addition, suitable boundary conditions must also be prescribed along the envelope of the **CLAY** soil region.

## Outputs

The different components of the ECR table are as follows:

ECR(1) : current hydrostatic pressure  $\frac{1}{3}(\sigma_x + \sigma_y + \sigma_z)$

ECR(2) : current bulk modulus

ECR(3) : second invariant of the deviatoric cumulated strain

ECR(4) : hardening parameter  $p_0$

ECR(5): sound speed

ECR(6): current value of the shear modulus  $G$

ECR(7): current value of the Poisson's coefficient  $\nu$

The components of the stress tensor are as follows:

SIG(1):  $\sigma_x$   
 SIG(2):  $\sigma_y$   
 SIG(3):  $\sigma_z$   
 SIG(4):  $\tau_{xy}$   
 SIG(5):  $\tau_{yz}$  (only 3D)  
 SIG(6):  $\tau_{xz}$  (only 3D)

### Remarks

Let  $M_1$  be the ratio between the second invariant of the stress tensor  $J_2$  and the first invariant of the stress tensor  $J_1$  at critical state (i.e. for stress points which lie on the failure surface). This is the quantity which is usually available from tests.

Let  $M_2$  be the ratio between the second invariant of the *deviatoric* stress tensor  $J'_2$  and the first invariant of the stress tensor  $J_1$  at critical state.

The  $M$  parameter defined above in the input syntax corresponds to  $M_1$ . However, note that in the model description of the CLAY material (*An Implementation of the Cam-Clay Elasto-Plastic Model Using a Backward Interpolation and Visco-Plastic Regularization*, Technical Note I.96.239) the quantity  $M$  corresponds rather to  $M_2$ .

The following relation holds between the two quantities:  $M_2 = M_1/\sqrt{3}$ .

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

**7.7.47 FUNE (SPECIALIZED CABLE MATERIAL)****Object:**

This model represents an elastoplastic cable, with no resistance in compression, and should be used in conjunction with special cable elements FUN2 (in 2D) and FUN3 (in 3D). The material is elasto-plastic in traction.

**Syntax:**

```
"FUNE"  "RO" rho  "YOUN" young  "NU" nu  "ELAS" sige "ERUP" erup ...  
...  "TRAC" npts*(sig eps) /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

sige

Elastic limit.

erup

Rupture strain.

"TRAC"

This key-word announces the yield curve (in traction).

npts

Number of points (except the origin) defining the yield curve.

sig

Stress.

eps

Total strain (elastic + plastic).

LECTURE

List of the elements concerned.

**Comments:**

- 1/ - The young parameter defines Young's modulus during an elastic phase.
- 2/ - The points (sig,eps) may have any position; however, concerning the first point, there must be a compatibility between the coordinates, Young's modulus and the elastic limit.
- 3/ - The slope of the yield curve may not increase from one segment to the following one.

**Outputs:**

The components of the ECR table are as follows:

- ECR(1): (free) (was total longitudinal strain of the cable element)
- ECR(2): (free) (was total lateral strain of the cable element)
- ECR(3): plastic longitudinal strain of the cable element
- ECR(4): current yield stress in traction (0 if broken)
- ECR(5): sound speed

Note that in order to post-process the total strains (which were formerly unappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).



### 7.7.48 JOHNSON-COOK MODEL

#### Object:

In the Johnson-Cook model Elasto-plasticity is implemented via a radial return algorithm. Only isotropic hardening is activated to date and strain-rate dependency is included in the model. However, no temperature effects are included in the present implementation.

#### References

The implementation of this material model is described in reference [167].

The Johnson-Cook constitutive relation is given by:

$$\sigma_{eq} = \left[ A_1 + A_2 (\varepsilon_{eq}^p)^{\lambda_2} \right] \left[ 1 + \lambda_1 \ln \left( \frac{\dot{\varepsilon}_{eq}^p}{\dot{\varepsilon}_{eq,ref}^p} \right) \right] (1 - \theta^m) \quad (2)$$

where:

- $A_1$  is COA1 (1st constant) in Europlexus
- $A_2$  is COA2 (2nd constant) in Europlexus
- $\lambda_1$  is CLB1 (3rd constant) in Europlexus
- $\lambda_2$  is CLB2 (hardening parameter) in Europlexus
- $\dot{\varepsilon}_{eq,ref}^p$  is SRRF (reference strain rate) in Europlexus
- $\theta$  is the homologous temperature  $\frac{T - T_{room}}{T_{melting} - T_{room}}$  (currently not implemented in Europlexus)
- $m$  is the homologous temperature exponent (currently not implemented in Europlexus)

The Johnson-Cook model is a simple empirical generalization of Ludwik's constitutive law (see VMLU on page C.253), represented by the first term of the above equation, trying to account for strain-rate effects (included in the second term of the equation) and for temperature effects (third and last term). The “reference” strain rate is the minimum plastic strain rate for which calibration of the model has been made.

In Johnson-Cook's model the Ludwik's law (first term) is multiplied by a function of the equivalent plastic strain rate. The form of this function is related to the often made experimental observation that the increase in flow stress is a logarithmic function of the strain rate.

The reference (or minimum) equivalent plastic strain rate  $\dot{\varepsilon}_{eq,ref}^p$  is the value of equivalent plastic strain rate under which the material behaves in a “static” (i.e., strain-rate independent) way. In practice, in the code, when the equivalent plastic strain rate is below this value, only the static part of the model is considered. The parameters equivalent plastic strain rate and  $\lambda_1$  are interconnected.

#### Syntax:

```

"VMJC"  "R0" rho  "YOUN" young  "NU" nu  "COA1" coa1
        "COA2" coa2 "CLB1" clb1  "CLB2" clb2 "SRRF" srrf
        <"FAIL" $[ "VMIS" "LIMI" limit ;
                  "DPLS" "LIMI" limit ;
                  "JOCO" "COD1" cod1 "COD2" cod2 "COD3" cod3 "COD4" cod4
                  ]$ >
... /LECTURE/

```

**rho**

Density of the material.

**young**

Young's modulus (elastic phase).

**nu**

Poisson's ratio.

**coa1**

1st constant ( $A_1$ ) in the Johnson-Cook model.

**coa2**

2nd constant ( $A_2$ ) in the Johnson-Cook model.

**clb1**

3rd constant ( $\lambda_1$ ) in the Johnson-Cook model.

**clb2**

Hardening coefficient ( $\lambda_2$ ) of the Johnson-Cook model.

**srrf**

Reference strain rate ( $\dot{\epsilon}_{eq,ref}^P$ ) of the Johnson-Cook model.

**FAIL**

Optional keyword: introduces an element failure model. The available failure criteria are: **VMIS** for a criterion based upon the equivalent Von-Mises stress, **DPLS** for a criterion based upon the equivalent plastic strain, **JOCO** for the so-called Johnson-Cook criterion based upon an equivalent plastic strain, depending on the strain rate and the triaxiality ratio. See comments below.

**limit**

Optional parameter, indicates the failure limit for the VMIS or PLAS criterion.

**cod1**

Optional parameter, 1st constant ( $D_1$ ) in the Johnson-Cook failure criterion.

**cod2**

Optional parameter, 2nd constant ( $D_2$ ) in the Johnson-Cook failure criterion.

cod3

Optional parameter, 3rd constant ( $D_3$ ) in the Johnson-Cook failure criterion.

cod4

Optional parameter, 4th constant ( $D_4$ ) in the Johnson-Cook failure criterion.

#### LECTURE

List of the elements concerned.

#### Comments:

The Johnson-Cook failure criterion is given by:

$$\varepsilon_p^f = [D_1 + D_2 \exp(D_3 \sigma^*)] \left[ 1 + D_4 \ln \left( \frac{\dot{\varepsilon}_{eq}^p}{\dot{\varepsilon}_{eq,ref}^p} \right) \right] \quad (3)$$

where:

- $D_1$  is COD1 (1st constant) in Europlexus
- $D_2$  is COD2 (2nd constant) in Europlexus
- $D_3$  is COD3 (3rd constant) in Europlexus
- $D_4$  is COD4 (4th constant) in Europlexus
- $\dot{\varepsilon}_{eq,ref}^p$  is SRRF (reference strain rate) in Europlexus
- $\sigma^* = \frac{p}{q}$  is the triaxiality ratio, where  $p$  is the hydrostatic pressure and  $q$  is the Von-Mises equivalent stress.

The damage parameter  $D$  triggers failure when it reaches 1. It is computed as:

$$D = \sum \frac{\Delta \varepsilon_p}{\varepsilon_p^f} \quad (4)$$

#### Outputs:

The components of the ECR table are as follows:

ECR(1): current hydrostatic pressure

ECR(2): current equivalent stress (Von-Mises)

ECR(3): current equivalent plastic strain

ECR(4): current yield stress

ECR(5): sound speed

ECR(6): equivalent strain rate (Von-Mises)

ECR(7): failure flag (0=virgin Gauss Point, 1=failed Gauss Point)

ECR(8): damage parameter for the Johnson-Cook failure criterion

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

### 7.7.49 LUDWIG-PRANDTL MODEL

#### Object:

This directive enables to choose the Ludwig-Prandtl model, a purely elasto-plastic model implemented at Ispra. Elasto- plasticity is implemented via a radial return algorithm. Only isotropic hardening is activated to date. There is no dependency on temperature but strain rate effects are included.

#### References

The implementation of this material model is described in reference [\[167\]](#).

#### Syntax:

```
"VMLP"  "RO" rho  "YOUN" young  "NU" nu  "COA1" coa1
        "COA2" coa2 "CLB1" clb1  "CLB2" clb2 "CLB3" clb3
        "CLB4" clb4
... /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

coa1

1st constant in the Ludwig-Prandtl model.

coa2

2nd constant in the Ludwig-Prandtl model.

clb1

3rd constant in the Ludwig-Prandtl model.

clb2

4th constant in the Ludwig-Prandtl model.

clb3

5th constant in the Ludwig-Prandtl model.

c1b4

6th constant in the Ludwig-Prandtl model.

#### LECTURE

List of the elements concerned.

#### Comments:

1/ - The young parameter defines Young's modulus during an elastic phase.

2/ - The points (sig,eps) may have any position; however, concerning the first point, there must be a compatibility between the coordinates, Young's modulus and the elastic limit.

3/ - The slope of the yield curve may not increase from one segment to the following one.

#### Outputs:

The components of the ECR table are as follows:

ECR(1): current hydrostatic pressure

ECR(2): current equivalent stress (Von Mises)

ECR(3): current equivalent plastic strain

ECR(4): current yield stress

ECR(5): sound speed

ECR(6): equivalent strain rate (Von Mises)

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

### 7.7.50 LUDWIK MODEL

**Object:**

This directive enables to choose the Ludwik model, a purely elasto-plastic model implemented at Ispra. Elasto-plasticity is implemented via a radial return algorithm. Only isotropic hardening is activated to date. There is no dependency on temperature nor on strain rate.

**References**

The implementation of this material model is described in reference [\[167\]](#).

**Syntax:**

```
"VMLU"  "RO" rho  "YOUN" young  "NU" nu  "ELAS" sige ...  
        "COA2" coa2 "COEN" coen  
... /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

sige

Elastic limit.

coa2

Plastic threshold value.

coen

Hardening coefficient.

LECTURE

List of the elements concerned.

**Comments:**

- 1/ - The young parameter defines Young's modulus during an elastic phase.
- 2/ - The points (sig,eps) may have any position; however, concerning the first point, there must be a compatibility between the coordinates, Young's modulus and the elastic limit.
- 3/ - The slope of the yield curve may not increase from one segment to the following one.

**Outputs:**

The components of the ECR table are as follows:

- ECR(1): current hydrostatic pressure
- ECR(2): current equivalent stress (Von Mises)
- ECR(3): current equivalent plastic strain
- ECR(4): current yield stress
- ECR(5): sound speed

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).



### 7.7.51 ZERILLI-ARMSTRONG MODEL

#### Object:

This directive enables to choose the Zerilli-Armstrong model with the implementation developed at Ispra. Elasto-plasticity is implemented via a radial return algorithm. Only isotropic hardening is activated to date and strain-rate dependency is included. However, no dependency on temperature exist in the present version of the model.

#### References

The implementation of this material model is described in reference [\[167\]](#).

#### Syntax:

```
"VMZA"  "RO" rho  "YOUN" young  "NU" nu  "COA1" coa1 ...  
        "COA2" coa2 "COA3" coa3 "COA4" coa4 "CLB1" clb1  
        "CLB2" clb2 "CLB3" clb3  
... /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

coa1

1st coefficient of the Zerilli-Armstrong model.

coa2

2nd coefficient of the Zerilli-Armstrong model.

coa3

3rd coefficient of the Zerilli-Armstrong model.

coa4

4th coefficient of the Zerilli-Armstrong model.

clb1

1st hardening coefficient of the Zerilli-Armstrong model.

c1b2

2nd hardening coefficient of the Zerilli-Armstrong model.

c1b3

3rd hardening coefficient of the Zerilli-Armstrong model.

#### LECTURE

List of the elements concerned.

#### Comments:

1/ - The young parameter defines Young's modulus during an elastic phase.

2/ - The points (sig,eps) may have any position; however, concerning the first point, there must be a compatibility between the coordinates, Young's modulus and the elastic limit.

3/ - The slope of the yield curve may not increase from one segment to the following one.

#### Outputs:

The components of the ECR table are as follows:

ECR(1): current hydrostatic pressure

ECR(2): current equivalent stress (Von Mises)

ECR(3): current equivalent plastic strain

ECR(4): current yield stress

ECR(5): sound speed

ECR(6): equivalent strain rate (Von Mises)

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

### 7.7.52 DRUCKER-PRAGER WITH HYDROSTATIC POST-FAILURE (JRC)

#### Object:

This directive enables to specify a Drucker-Prager material. The material behaves in a linear elastic way until failure is reached, and thereafter it behaves like a fluid (i.e. it resists only to compression). Failure occurs when the stress point in the  $J_1$ - $\sqrt{J'_2}$  space reaches the failure line (a straight line) of equation:

$$\sqrt{J'_2} = K - \alpha J_1$$

where  $J_1 = \sigma_x + \sigma_y + \sigma_z$  is the first invariant of the stress tensor and  $J'_2$  is the second invariant of the deviatoric stress tensor:

$$J'_2 = \frac{1}{3}(\sigma_x^2 + \sigma_y^2 + \sigma_z^2 - \sigma_x\sigma_y - \sigma_x\sigma_z - \sigma_y\sigma_z) + \tau_{xy}^2 + \tau_{xz}^2 + \tau_{yz}^2$$

The constant  $K$  is the intersection of the failure line with the vertical axis and represents the failure stress of the material in pure shear (e.g. in torsion): it is also called cohesion.

The constant  $\alpha$  is the slope of the failure line (tangent of the angle) and is also called the internal friction angle.

After failure is reached, the material behaves like a liquid: all tangential stresses are set to zero and the normal stresses are set to equal (hydrostatic) values if the material is under compression (negative volumetric strain), or to zero if the material is under traction (positive volumetric strain).

Due to its postulated after-failure behaviour, this material is not "erodable". That is, when failure is reached, even at all Gauss points of an element, the element is not removed from the calculation because it contributes to the solution with its post-failure (hydrostatic) behaviour. Of course, this only makes sense as long as the failed material remains confined (so that a hydrostatic pressure can build up in it).

#### References:

The material model is described in reference [13]. Note that although the material model had been originally denoted as a Mohr-Coulomb model, in reality it is a Drucker-Prager material. In fact, the yield surface corresponding to the expression given above (using  $J'_2$ ) is a cone with circular cross section (and not with hexagonal cross-section) in principle stress space.

#### Syntax:

```
"DRPR"  "RO" ro "YOUN" youn "NU" nu "COHE" cohe "FRIC" fric
/LECTURE/
```

ro

Density.

youn

Young's modulus.

nu

Poisson's ratio.

cohe

Failure stress  $K$  in pure shear, e.g. in torsion (cohesion).

fric

Slope  $\alpha$  of the failure line in the  $J_1$ - $\sqrt{J_2}$  diagram (internal friction angle).

/LECTURE/

Numbers of the elements concerned.

### Outputs:

The different components of the ECR table are as follows:

ECR(1): current  $J_1$  invariant ( $\sigma_1 + \sigma_2 + \sigma_3$ ).

ECR(2): current  $\sqrt{J_2}$  invariant.

ECR(3): failure flag (0=not failed, 1=failed).

ECR(4): sound speed

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

### 7.7.53 ALUMINIUM FOAM

#### Object:

This option enables to specify an aluminium foam material and follows the Deshpande-Fleck model as implemented at NTNU, Trondheim (N).

#### References:

More information on the formulation of this material model may be found in the following references:

1. V.S. Deshpande and N.A. Fleck, Isotropic models for metallic foams, J. Mech. Phys. Solids 48 (2000), pp. 1253–1283.
2. A. Reyes, O. S. Hopperstad, T. Berstad, A. G. Hansen, M. Langseth, Constitutive modeling of aluminum foam including fracture and statistical variation of density, European Journal of Mechanics – A/Solids, Vol 22, pp 815–835, 2003.

The stresses are calculated by using the following equation:

$$\sigma = \sigma_p + \gamma \frac{e}{e_D} + \alpha_2 \ln\left(\frac{1}{1 - (e/e_D)^\beta}\right)$$

The parameter  $e_D$  is taken from the recent foam density  $\rho_f$  and the density of the pure material  $\rho_{f0}$  by using this equation

$$e_D = 1 - \frac{\rho_f}{\rho_{f0}}$$

The parameter  $\alpha$  defines the shape of the yield surface and can be calculated by using the plastic Poission's ratio  $\nu_p$ :

$$\alpha^2 = \frac{9}{2} \frac{(1 - 2\nu_p)}{(1 + \nu_p)}$$

The parameter  $\gamma$  is the initial hardening factor by reaching the plastic regime. The parameters  $\alpha_2$  and  $\beta$  can be taken by a best fit of the experimental curve.

#### Syntax:

```
FOAM  RO_F ro_f  YOUN youn  NU  nu    SIGP sigp  RO_0 ro_0
      ALFA alfa  GAMM gamm  ALF2 alf2  BETA beta <DERF derf>
      <EF ef> <SF sf> <RNUM rnum> <WC wc>
      /LECTURE/
```

**ro\_f**

Initial density of the foam material, i.e. considering the voids.

**youn**

Young's modulus (initial).

**nu**

Poisson's coefficient (initial).

**sigp**

Yield stress.

**ro\_0**

Initial density of the material, not considering the voids (pure material,  $\rho_{f0}$ ).

**alfa**

Shape of the yield surface (see above).

**gamm**

Initial hardening factor by reaching the plastic regime.

**alf2**

Scale factor (material constant).

**beta**

Shape factor (material constant).

**derf**

Switch to choose the derivation of  $F_i$  in the model: 0 means numerical derivation, while 1 means normal derivation. The default is 1.

**efail**

Critical volumetric failure strain. A Gauss point fails if the volumetric strain exceeds **efail**. By default it is 0.0, meaning that the volumetric strain failure criterion is not active.

**sfail**

Critical failure stress. A Gauss point fails if the maximum principal stress exceeds **sfail** for a number of (consecutive) time steps greater than **rnum** (see next parameter). By default it is 0.0, meaning that the maximum principal stress failure criterion is not active.

**rnum**

Number of (consecutive) time steps with maximum principal stress exceeding **sfail** needed for a Gauss point to fail. By default it is  $\infty$ , meaning that the maximum principal stress failure criterion is not active.

**wc**

Critical failure energy (Cockcroft-Latham criterion). A Gauss point fails if the fracture energy exceeds **wc**. By default it is 0.0, meaning that the Cockcroft-Latham failure criterion is not active.

**/LECT/**

List of the concerned elements.

**Outputs:**

The components of the ECR table are as follows:

- ECR(1): Equivalent plastic strain ( $\epsilon_{eq}$ )
- ECR(2): Von Mises effective plastic strain ( $\epsilon_e$ )
- ECR(3): Volumetric strain ( $\epsilon_m$ )
- ECR(4): Equivalent stress ( $\sigma_{eq}$ )
- ECR(5): Von Mises effective stress ( $\sigma_e$ )
- ECR(6): Mean stress ( $\sigma_m$ )
- ECR(7): Isotropic hardening variable ( $R$ )
- ECR(8): Iteration counter
- ECR(9): ( $Y$ )
- ECR(10): sound speed
- ECR(11): first principal stress ( $ps1$ )
- ECR(12): second principal stress ( $ps2$ )
- ECR(13): third principal stress ( $ps3$ )
- ECR(14): counter of the consecutive number of steps where **ps1** > **sfail**
- ECR(15): Cockcroft-Latham damage accumulation ( $W$ ) when energy-based damage is activated and **ps1** > **sfail**
- ECR(16): “universal” damage parameter ( $D$ ). May be used in combination with AMR?
- ECR(17): Gauss point failure flag: 1 = virgin, 0 = failed.

### 7.7.54 GLRC: REINFORCED CONCRETE FOR SHELLS

#### Object:

This material is designed to model reinforced concrete shells, possibly with prestressing and steel liner. It consists in a resultant variables constitutive law, using both plasticity (double JOHANSEN's criterion with a kinematic softening) and damage (to take into account concrete cracking). For information about this model see references [928], [929].

#### New syntax for non-linear GLRC material (elastoplastic with or without damage):

```
"GLRC" < "DAMA" > < "SHEA" >
  "RO"    rho
  "H"     thickness
  "EB"    yconcrete "NUB" pconcrete
  "NLIT"  nlayer * ( |[ "NAPP" ("EA" ysteel < "FY" tsteel >
                        "OMX" ax      "OMY" ay
                        "RX"  rx      "RY" ry      ) ;
                    "PREC" ( "EA" ysteel < "FY" tsteel >
                        "OMX" ax      "OMY" ay
                        "RX"  rx      "RY" ry      ) ;
                    "LINR" ( "EA" ysteel < "FY" tsteel >
                        "OMLR" epliner "NULR" nuliner
                        "RLR"  rliner      ) ]| )
  < "OMT"  atrast      "EAT"  ytrast >
  < "BT1"  shear1      "BT2"  shear2 >

  < "BTD1" sheard1      "BTD2" sheard2 >
  < "TSD"  tsheard >

  < "FT"    tconcrete < "GAMM" gamma >
  "QP1"    qslope1    "QP2"    qslope2 >

  "C1N1" prgmemb1x "C1N2" prgmemb1y "C1N3" prgmemb1xy
  "C2N1" prgmemb2x "C2N2" prgmemb2y "C2N3" prgmemb2xy
  "C1M1" prgbend1x "C1M2" prgbend1y "C1M3" prgbend1xy
  "C2M1" prgbend2x "C2M2" prgbend2y "C2M3" prgbend2xy

  $[    "FC"    cconcrete      ;
  (    "MP1X" < "FONC" > plaslim1x
    "MP1Y" < "FONC" > plaslim1y
    "MP2X" < "FONC" > plaslim2x
    "MP2Y" < "FONC" > plaslim2y
    < "D1X"    "FONC"    dplaslim1x >
    < "D1Y"    "FONC"    dplaslim1y >
    < "D2X"    "FONC"    dplaslim2x >
    < "D2Y"    "FONC"    dplaslim2y >
    < "DD1X"   "FONC"    ddplaslim1x >
```



```

< "DD1Y"      "FONC"      ddplaslim1y >
< "DD2X"      "FONC"      ddplaslim2x >
< "DD2Y"      "FONC"      ddplaslim2y > ) ]$

< "PREX" nprecx  "PREY" nprecy  >

< "KRAY" kray    "MRAY" mray    >

/LECTURE/

```

**rho**

Density of the plate material (concrete and steel).

**thickness**

Thickness of the concrete (thickness of the plate).

**yconcrete**

Young's modulus of the concrete material.

**ypoisson**

Poisson's ratio of the concrete material.

**NAPP**

Keyword for the description of bending steel reinforcement (steel grid).

**PREC**

Keyword for the description of prestressing.

**LINR**

Keyword for the description of the steel liner.

**ysteel**

Young's modulus of the steel material.

**tsteel**

Yield stress of the steel. Used to calculate automatically the generalized Johansen criterion (when the **plaslim** functions are not specified).

**ax, ay**

Areas (per meter of plate) of the reinforcement layer in the x and y directions ( $m^2/m$ ).

**rx, ry**

Nondimensional position of the layer in the x and y directions ( $-1 \leq r \leq 1$ ).

**epliner**

Thickness of the liner.

**nuliner**

Poisson's ratio of the liner steel.

**rliner**

Nondimensional position of the liner ( $-1 \leq rliner \leq 1$ ).

**shear**

Coefficients of the elastic shear matrix (for elements that take into account the transverse shear like Q4GR or Q4GS):

$$\begin{bmatrix} T_x \\ T_y \end{bmatrix} = \begin{bmatrix} \text{shear1} & 0 \\ 0 & \text{shear2} \end{bmatrix} \begin{bmatrix} \gamma_x \\ \gamma_y \end{bmatrix}$$

When the shear coefficients are not specified, they are calculated, for Q4GR and Q4GS elements, using the following expression:

$$T = k \frac{h}{2} \left( \frac{E_b}{1 + \nu_b} + E_{aT} \omega_T \right) \gamma$$

with:

- $h$  : shell thickness
- $k$  : shear correcting coefficient usually set as 5/6 (Reissner theory)
- $E_b$  : Young's modulus of concrete (**yconcrete**)
- $\nu_b$  : Poisson's ratio of concrete (**ypoisson**)
- $E_{aT}$  : Young's modulus of transverse steel (**ytrast**)
- $\omega_T$  : Area of transverse steel (**atrast**)

If the keyword **SHEA** is specified then a nonlinear evolution of the shear force is taken into account ([942]). This nonlinear evolution can be compared to an elastoplastic constitutive law. Beyond a shear force defined by TSD, the shear force evolves according to a linear slope whose stiffness is defined through the keywords **BTD1** and **BTD2** and plate elements are then subjected to irreversible deformations.

When the values of the damaged shear coefficients **sheard1** and **sheard2** are not specified, they are calculated using the following expression:

$$\text{sheard1} = \text{sheard2} = k \frac{h}{2} \left( \frac{E_{aT}}{100} \omega_T \right)$$

**atrast**

Area (per square meter of plate) of the transverse reinforcement ( $m^2/m^2$ ). Used to calculate the elastic shear coefficients when **shear** are not specified.

**ytrast**

Young's modulus of the steel material for transverse reinforcement. Used for the computation of the elastic shear coefficients when **shear** are not specified. Default value is the standard **ysteel** value.

**tconcrete**

Tensile strength of concrete (tensile stress). Must be positive. Used to calculate the bending cracking moment. Used only for damage.

qslope1 qslope2

Slopes quotient for positive and negative bending. The quotient is supposed to be the slope of the (curvature,moment) graph after cracking over the slope before cracking. Used only for damage.

$$Q_p = \frac{p_{eac}}{p_{ebc}}$$

with:

- $Q_p$  : slope quotient **qslope** ( $0 < Q_p \leq 1$ )
- $p_{ebc}$  : slope before cracking (elastic concrete)
- $p_{eac}$  : slope after cracking (cracked concrete)

gamma

Damage computation parameter which characterizes the slope of the (curvature,moment) graph during cracking. **gamma** can be considered as the slope during cracking over the slope before cracking. If **gamma** > 0, the slope increases. If **gamma** < 0, the slope decreases and the stability is not warranted. In any case, we must have **gamma** < **qslope1** and **gamma** < **qslope2**. Default value is zero. Used only for damage.

$$\gamma = \frac{p_{edc}}{p_{ebc}}$$

with:

- $\gamma$  : **gamma**
- $p_{ebc}$  : slope before cracking (elastic concrete)
- $p_{edc}$  : slope during cracking

pragmemb, pragbend

Prager coefficients corresponding to the matrices linking the plastic strain and curvature to the back membrane force and the backmoment.

$$n = CN_1 \epsilon_1^p + CN_2 \epsilon_2^p$$

$$m = CM_1 \kappa_1^p + CM_2 \kappa_2^p$$

with:

$$\begin{aligned} \bullet \quad CN_1 &= \begin{bmatrix} \text{pragmemb1x} & 0 & 0 \\ 0 & \text{pragmemb1y} & 0 \\ 0 & 0 & \text{pragmemb1xy} \end{bmatrix} \\ \bullet \quad CN_2 &= \begin{bmatrix} \text{pragmemb2x} & 0 & 0 \\ 0 & \text{pragmemb2y} & 0 \\ 0 & 0 & \text{pragmemb2xy} \end{bmatrix} \\ \bullet \quad CM_1 &= \begin{bmatrix} \text{pragbend1x} & 0 & 0 \\ 0 & \text{pragbend1y} & 0 \\ 0 & 0 & \text{pragbend1xy} \end{bmatrix} \\ \bullet \quad CM_2 &= \begin{bmatrix} \text{pragbend2x} & 0 & 0 \\ 0 & \text{pragbend2y} & 0 \\ 0 & 0 & \text{pragbend2xy} \end{bmatrix} \end{aligned}$$

- $\epsilon_1^p$  and  $\kappa_1^p$  : plastic strain and curvature linked to the first criterion (`plaslim1`)
- $\epsilon_2^p$  and  $\kappa_2^p$  : plastic strain and curvature linked to the second criterion (`plaslim2`)

$$C = \frac{p_e p_p}{p_e - p_p}$$

with:

- $C$  : Prager coefficient
- $p_e$  : elastic slope (or slope after cracking, in case of bending)
- $p_p$  : plastic slope

#### `cconcrete`

Compressive strength of concrete. Used to calculate automatically the generalized Johansen criterion (when the `plaslim` functions are not specified).

#### `plaslim`

Functions used in the generalized Johansen criterion. They describe the "beam" plastic limit moment depending on the membrane force. When they are not specified, they are automatically calculated and interpolated.

#### `plaslim1x plaslim2x`

Positive and negative plastic limit moments for a perfect bending in the x-direction (referring to the orthotropic axes of the shell element). If the directive "FONC" is used, `plaslim1x` or `plaslim2x` are integers referring to a function number (function of  $N_x$ ). We should have usually `plaslim1x` > `plaslim2x`.

#### `plaslim1y plaslim2y`

Positive and negative plastic limit moments for a perfect bending in the y-direction (referring to the orthotropic axes of the shell element). If the directive "FONC" is used, `plaslim1y` or `plaslim2y` are integers referring to a function number (function of  $N_y$ ). We should have usually `plaslim1y` > `plaslim2y`.

#### `dplaslim1x dplaslim2x dplaslim1y dplaslim2y`

Function number of the first derivative of `plaslim1x`, `plaslim2x`, `plaslim1y` and `plaslim2y` plastic limit functions. They are used when the membrane plasticity is taken into account and when they cannot be computed directly from the `plaslim1x`, `plaslim2x`, `plaslim1y` and `plaslim2y` functions.

#### `ddplaslim1x ddplaslim2x ddplaslim1y ddplaslim2y`

Function number of the second derivative of `plaslim1x`, `plaslim2x`, `plaslim1y` and `plaslim2y` plastic limit functions. They are used when the membrane plasticity is taken into account and when they cannot be computed directly from the `plaslim1x`, `plaslim2x`, `plaslim1y`, `plaslim2y` or `dplaslim1x`, `dplaslim2x`, `dplaslim1y`, `dplaslim2y` functions.

#### `nprecx, nprecy`

Prestressing force in the x and y directions (should be negative since it is normally a compression force).

#### `kray, mray`

Rayleigh's stiffness and mass proportional damping coefficients, used only by finite elements of the following types: DKT3, T3GS, Q4GS. Default values: kray=0, mray=0. For information about Rayleigh's damping see reference [924].

## LECTURE

List of the elements concerned.

**Syntax for perforation analysis (always used with the new syntax) :**

```
"GLRC" < "DAMA" > "PERF" < "SHEA" >
  "RO"    rho
  "H"     thickness
  "EB"    yconcrete  "NUB"  pconcrete
  "NLIT"  nlayer * ( |["NAPP" ( "EA" ysteel < "FY" tsteel >
                           "FS" tsteelp
                           "OMX" ax      "OMY" ay
                           "RX"  rx      "RY"  ry      ) ;
                    "PREC" ( "EA" ysteel < "FY" tsteel >
                           "FS" tsteelp
                           "OMX" ax      "OMY" ay
                           "RX"  rx      "RY"  ry      ) ;
                    "LINR" ( "EA" ysteel < "FY" tsteel >
                           "FS" tsteelp
                           "OMLR" epliner "NULR" nuliner
                           "RLR"  rliner  ) || )
  "OMT"   atrast      < "EAT"   ytrast >   "FST"   tsteelp_t
< "BT1"   shear1      "BT2"   shear2 >

< "BTD1"  sheard1      "BTD2"  sheard2 >
< "TSD"   tsheard >

< "FT"    tconcrete   < "GAMM"  gamma   >
  "QP1"    qslope1     "QP2"    qslope2 >

  "FC"     cconcrete   "PHI"     friction < "NUFC"  eff_factor >
< "NPER"   nper >

  "C1N1"   pragememb1x "C1N2"   pragememb1y "C1N3"   pragememb1xy
  "C2N1"   pragememb2x "C2N2"   pragememb2y "C2N3"   pragememb2xy
  "C1M1"   pragbend1x  "C1M2"   pragbend1y  "C1M3"   pragbend1xy
  "C2M1"   pragbend2x  "C2M2"   pragbend2y  "C2M3"   pragbend2xy

<   "MP1X" < "FONC" >  plaslim1x
    "MP1Y" < "FONC" >  plaslim1y
    "MP2X" < "FONC" >  plaslim2x
    "MP2Y" < "FONC" >  plaslim2y
<   "D1X"   "FONC"    dplaslim1x >
<   "D1Y"   "FONC"    dplaslim1y >
<   "D2X"   "FONC"    dplaslim2x >
```

```

< "D2Y"      "FONC"      dplaslim2y >
< "DD1X"     "FONC"     ddplaslim1x >
< "DD1Y"     "FONC"     ddplaslim1y >
< "DD2X"     "FONC"     ddplaslim2x >
< "DD2Y"     "FONC"     ddplaslim2y >      >

< "PREX" nprecx  "PREY" nprecy >

< "KRAY" kray    "MRAY" mray    >

/LECTURE/

```

**cconcrete**

Compressive strength of concrete. Used to calculate automatically the generalized Johansen criterion (when the `plaslim` functions are not specified). Mandatory for perforation analysis.

**friction**

Friction angle of concrete (degrees). Mandatory for perforation analysis.

**tsteelp, tsteelp\_t**

Limit stress of steel (for each layer and for transverse reinforcement). Mandatory for perforation analysis.

**eff\_factor**

Effectiveness factor for concrete. When not specified, a default value is taken.

**nper**

Frequency of verification of the perforation criterion. Default value is 1 (every time step). For information about the perforation criterion see references [\[926\]](#), [\[928\]](#).

**Old syntax for the standard material (without damage):**

```

"GLRC" "OLD"
  "RO" rho "BN11" memb11      "BN12" memb12
        "BN22" memb22      "BN33" memb33
        "BM11" bend11      "BM12" bend12
        "BM22" bend22      "BM33" bend22
    < "BC11" coup11 > < "BC12" coup12 >
    < "BC22" coup22 > < "BC33" coup22 >
    < "BT1"  shear1 > < "BT2"  shear2 >
    < "C1N1" prgmemb1x "C1N2" prgmemb1y "C1N3" prgmemb1xy >
    < "C2N1" prgmemb2x "C2N2" prgmemb2y "C2N3" prgmemb2xy >
    "C1M1" prgbend1x "C1M2" prgbend1y "C1M3" prgbend1xy
    "C2M1" prgbend2x "C2M2" prgbend2y "C2M3" prgbend2xy
    "MP1X" < "FONC" > plaslim1x

```

```

      "MP1Y" < "FONC" > plaslim1y
      "MP2X" < "FONC" > plaslim2x
      "MP2Y" < "FONC" > plaslim2y
    < "D1X"      "FONC"      dplaslim1x >
    < "D1Y"      "FONC"      dplaslim1y >
    < "D2X"      "FONC"      dplaslim2x >
    < "D2Y"      "FONC"      dplaslim2y >
    < "DD1X"     "FONC"     ddplaslim1x >
    < "DD1Y"     "FONC"     ddplaslim1y >
    < "DD2X"     "FONC"     ddplaslim2x >
    < "DD2Y"     "FONC"     ddplaslim2y >
    /LECTURE/

```

rho

Density of the material.

memb, bend, coup

Coefficients of the elastic matrix:

$$\begin{bmatrix} N_{xx} \\ N_{yy} \\ N_{xy} \\ M_{xx} \\ M_{yy} \\ M_{xy} \end{bmatrix} = \begin{bmatrix} \text{memb11} & \text{memb12} & 0 & \text{coup11} & \text{coup12} & 0 \\ \text{memb12} & \text{memb22} & 0 & \text{coup12} & \text{coup22} & 0 \\ 0 & 0 & \text{memb33} & 0 & 0 & \text{coup33} \\ \text{coup11} & \text{coup12} & 0 & \text{bend11} & \text{bend12} & 0 \\ \text{coup12} & \text{coup22} & 0 & \text{bend12} & \text{bend22} & 0 \\ 0 & 0 & \text{coup33} & 0 & 0 & \text{bend33} \end{bmatrix} \begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ 2\epsilon_{xy} \\ \kappa_{xx} - \kappa_{xx}^p \\ \kappa_{yy} - \kappa_{yy}^p \\ 2(\kappa_{xy} - \kappa_{xy}^p) \end{bmatrix}$$

When the coupling coefficients are not specified, they take the zero value.

shear

Coefficients of the elastic shear matrix (for elements that take into account the transverse shear like Q4GR or Q4GS):

$$\begin{bmatrix} T_x \\ T_y \end{bmatrix} = \begin{bmatrix} \text{shear1} & 0 \\ 0 & \text{shear2} \end{bmatrix} \begin{bmatrix} \gamma_x \\ \gamma_y \end{bmatrix}$$

When the shear coefficients are not specified, they take the zero value. Classical assumptions in elasticity give the following expression:

$$T = h \frac{kE}{2(1 + \nu)} \gamma$$

with:

- $h$  : shell thickness
- $k$  : shear correcting coefficient usually set as 5/6 (Reissner theory)
- $E$  : Young's modulus
- $\nu$  : Poisson's ratio

pragmemb, pragmbend

Prager coefficients corresponding to the matrices linking the plastic strain and curvature to the back membrane force and the backmoment.

$$n = CN_1\epsilon_1^p + CN_2\epsilon_2^p$$

$$m = CM_1\kappa_1^p + CM_2\kappa_2^p$$

with:

$$\bullet \quad CN_1 = \begin{bmatrix} \text{pragmemb1x} & 0 & 0 \\ 0 & \text{pragmemb1y} & 0 \\ 0 & 0 & \text{pragmemb1xy} \end{bmatrix}$$

$$\bullet \quad CN_2 = \begin{bmatrix} \text{pragmemb2x} & 0 & 0 \\ 0 & \text{pragmemb2y} & 0 \\ 0 & 0 & \text{pragmemb2xy} \end{bmatrix}$$

$$\bullet \quad CM_1 = \begin{bmatrix} \text{pragbend1x} & 0 & 0 \\ 0 & \text{pragbend1y} & 0 \\ 0 & 0 & \text{pragbend1xy} \end{bmatrix}$$

$$\bullet \quad CM_2 = \begin{bmatrix} \text{pragbend2x} & 0 & 0 \\ 0 & \text{pragbend2y} & 0 \\ 0 & 0 & \text{pragbend2xy} \end{bmatrix}$$

- $\epsilon_1^p$  and  $\kappa_1^p$  : plastic strain and curvature linked to the first criterion (`plaslim1`)
- $\epsilon_2^p$  and  $\kappa_2^p$  : plastic strain and curvature linked to the second criterion (`plaslim2`)

The membrane Prager coefficients are not mandatory. If they are not specified by the user, the model takes into account only bending plasticity. Thus it has a non-normal plasticity flow if the plastic limits vary with the membrane force. This could lead to convergence problems. But if the membrane Prager coefficients are given, both membrane and bending plasticity are taken into account. The model is in fact regularized compared to the preceding one.

`plaslim1x plaslim2x`

Positive and negative plastic limit moments for a perfect bending in the x-direction (referring to the orthotropic axes of the shell element). If the directive "FONC" is used, `plaslim1x` or `plaslim2x` are integers referring to a function number (function of  $N_x$ ). We should have usually `plaslim1x` > `plaslim2x`.

`plaslim1y plaslim2y`

Positive and negative plastic limit moments for a perfect bending in the y-direction (referring to the orthotropic axes of the shell element). If the directive "FONC" is used, `plaslim1y` or `plaslim2y` are integers referring to a function number (function of  $N_y$ ). We should have usually `plaslim1y` > `plaslim2y`.

`dplaslim1x dplaslim2x dplaslim1y dplaslim2y`

Function number of the first derivative of `plaslim1x`, `plaslim2x`, `plaslim1y` and `plaslim2y` plastic limit functions. They are used when the membrane plasticity is taken into account and when they cannot be computed directly from the `plaslim1x`, `plaslim2x`, `plaslim1y` and `plaslim2y` functions.

`ddplaslim1x ddplaslim2x ddplaslim1y ddplaslim2y`



Function number of the second derivative of `plaslim1x`, `plaslim2x`, `plaslim1y` and `plaslim2y` plastic limit functions. They are used when the membrane plasticity is taken into account and when they cannot be computed directly from the `plaslim1x`, `plaslim2x`, `plaslim1y`, `plaslim2y` or `dplaslim1x`, `dplaslim2x`, `dplaslim1y`, `dplaslim2y` functions.

#### LECTURE

List of the elements concerned.

#### Old syntax for the material with damage (for cracking):

```
"GLRC" "OLD" "DAMA"
  "RO" rho "BN11" memb11      "BN12" memb12
        "BN22" memb22      "BN33" memb33
        "E"  young         "NU"  poisson
        "MF1" cracklim1    "MF2" cracklim2
        "QP1" qslope1     "QP2" qslope2
        "GAMM" gamma
        < "BT1" shear1 > < "BT2" shear2 >
        < "C1N1" pragemb1x "C1N2" pragemb1y "C1N3" pragemb1xy >
        < "C2N1" pragemb2x "C2N2" pragemb2y "C2N3" pragemb2xy >
        "C1M1" pragbend1x "C1M2" pragbend1y "C1M3" pragbend1xy
        "C2M1" pragbend2x "C2M2" pragbend2y "C2M3" pragbend2xy
        "MP1X" < "FONC" > plaslim1x
        "MP1Y" < "FONC" > plaslim1y
        "MP2X" < "FONC" > plaslim2x
        "MP2Y" < "FONC" > plaslim2y
        < "D1X"      "FONC"      dplaslim1x >
        < "D1Y"      "FONC"      dplaslim1y >
        < "D2X"      "FONC"      dplaslim2x >
        < "D2Y"      "FONC"      dplaslim2y >
        < "DD1X"     "FONC"      ddplaslim1x >
        < "DD1Y"     "FONC"      ddplaslim1y >
        < "DD2X"     "FONC"      ddplaslim2x >
        < "DD2Y"     "FONC"      ddplaslim2y >
      /LECTURE/
```

#### DAMA

Enable the option which allows to take in account the concrete cracking by damage.

`rho`, `shear`, `pragemb`, `pragbend`, `plaslim`, `dplaslim`, `ddplaslim`

Same parameters as those described for the standard GLRC material.

#### memb

Coefficients of the elastic matrix:

$$\begin{bmatrix} N_{xx} \\ N_{yy} \\ N_{xy} \end{bmatrix} = \begin{bmatrix} \text{memb11} & \text{memb12} & 0 \\ \text{memb12} & \text{memb22} & 0 \\ 0 & 0 & \text{memb33} \end{bmatrix} \begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ 2\epsilon_{xy} \end{bmatrix}$$

There is no elastic coupling between the bending and the membrane behaviour.

young, poisson

Homogenized elastic characteristics (Young's modulus and Poisson's ratio) for bending.

cracklim1 cracklim2

Positive and negative cracking limit moments.

qslope1 qslope2

Slopes quotient for positive and negative bending. The quotient is supposed to be the slope of the (curvature,moment) graph after cracking over the slope before cracking.

### Comments:

All the limit plastic moments must be defined carefully. When they are declared as functions (using "FONC"), the domain defined as `plaslim1-plaslim2 > 0` must be a close convex domain: note particularly that the program tries to find two intersections of `plaslim1` and `plaslim2`.

When the limit plastic functions are not defined as polynomial (e.g. when "LSQU" is not used), the program requires prolongation of the functions: it is necessary to compute the elastic predictor which can be located outside the close convex elastic domain.

The first and second derivative of the limit plastic functions can be surely computed from the original limit plastic functions (i.e. without using the functions associated with the "D1", "D2", "DD1" and "DD2" directives) when these limit plastic functions are polynomials (see "LSQU" 9.1 to use table functions as polynomials).

After (and never before) the definition of the material characteristics ("MATE" directive), the orthotropy characteristics of the elements are mandatory. The syntax is:

```
"COMP"  "ORTS"    vx vy vz    /LECTURE/
```

See the "ORTS" directive for more details.

### Outputs:

The components of the ECR table are as follows:

Plastic strain and curvature ( $\epsilon^p$  and  $\kappa^p$ ) in the orthotropic axes:

ECR(1):  $\epsilon_x^p$

ECR(2):  $\epsilon_y^p$

ECR(3):  $2 \times \epsilon_{xy}^p$

ECR(4):  $\kappa_x^p$

ECR(5):  $\kappa_y^p$

ECR(6):  $2 \times \kappa_{xy}^p$

Energy dissipated during plasticity:

ECR(7): plastic dissipation per Gauss point. The sum of ECR(7) on all Gauss points of the element gives the plastic dissipation in the element.

Damage parameters:

ECR(8):  $D_1/D_{1max}$  for positive bending

ECR(9):  $D_2/D_{2max}$  for negative bending

Energy dissipated during damage:

ECR(10): damage dissipation per Gauss point. The sum of ECR(10) on all Gauss points of the element gives the damage dissipation in the element.

Orthotropy characteristics:

- Components, in the global reference frame, of the vector whose projection on the local coordinate system of the 3D shell element indicates the orthotropy direction (data following the "ORTS" directive).

ECR(11):  $v_x$

ECR(12):  $v_y$

ECR(13):  $v_z$

- After the first time step, the orthotropy characteristics are:

ECR(11): angle defining the orthotropic axes referring to the local axes in the shell element plane.

ECR(12): 10.

ECR(13): 10.

Membrane force and moment minus back force and backmoment ( $N - n$  and  $M - m$ ) in the orthotropic axes:

ECR(14):  $N_x - n_x$

ECR(15):  $N_y - n_y$

ECR(16):  $N_{xy} - n_{xy}$

ECR(17):  $M_x - m_x$

ECR(18):  $M_y - m_y$

ECR(19):  $M_{xy} - m_{xy}$

Post-treatment parameters for the perforation criterion:

ECR(20): = 0 if the criterion is not reached

= 1 if the criterion is reached in bending mode

= 2 if the criterion is reached in shear mode

ECR(21): normalized value of the perforation criterion ( $> 0$  if the criterion is reached)

ECR(22):  $n_x$  (components of the vector which is

ECR(23):  $n_y$  normal to the failure plan,

ECR(24):  $n_z$  in the global reference frame)

### 7.7.55 HYPERELASTIC MATERIAL

#### Object:

This sub-directive enables materials with an hyperelastic behaviour to be used. Only two types of shell (Q4GS et DST3) and several solid elements (CUBE, TETR, etc.) can be used with this material. The following kinds of hyperelastic materials can be selected:

- Type 1: Mooney-Rivlin material, see also the new material MOON on page C.297.
- Type 2: Hart Smith material.
- Type 3: Ogden material (strongly *not* recommended, see below).
- Type 4: Ogden material (new formulation, still under development), see also the new OGDE material on page C.298.
- Type 5: Ogden-Storakers material (hyperelstic foam).

Note that a Blatz-Ko hyperelastic material model is also available, see the new BLKO material on page C.299.

For Type 1, the expression of the strain energy density corresponds to:

$$\begin{aligned} W = & c1*(I1-3) + c2*(I2-3) + c3*(I1-3)**2 + \\ & c4*(I1-3)*(I2-3) + c5*(I2-3)**2 + \\ & c6*(I1-3)**3 + c7*(I2-3)*(I1-3)**2 + \\ & c8*(I1-3)*(I2-3)**2 + c9*(I2-3)**3 + \\ & c10*(I1-3)**4 + c11*(I2-3)**2*(I1-3)**3 + \\ & c12*(I1-3)**2*(I2-3) + c13*(I1-3)*(I2-3) + \\ & K*(\text{Log}(I3))**2 \end{aligned}$$

For Type 2, the expression of the strain energy density corresponds to :

$$W = A \int C(I_1 - 3)^2 dI_1 + 3B \cdot \log(I_2) + K \cdot \log(I_3)^2 \quad (5)$$

Type 3, the Ogden material can be expressed with the following equation

$$W = \sum_{p=1}^N \frac{\mu_p}{\alpha_p} (\lambda_1^{\alpha_p} + \lambda_2^{\alpha_p} + \lambda_3^{\alpha_p} - 3) \quad (6)$$

with the principal stretch  $\lambda$ .

Type 4, the Ogden material (new formulation) can be expressed with the following equation

$$W = \sum_{p=1}^N \frac{\mu_p}{\alpha_p} (\lambda_1^{*\alpha_p} + \lambda_2^{*\alpha_p} + \lambda_3^{*\alpha_p} - 3) + K(J - 1 - \ln J) \quad (7)$$

with  $\lambda^* = \lambda J^{-\frac{1}{3}}$ .  $K$  is the bulk modulus,  $\mu_p$  and  $\alpha_p$  are the material parameters used for this expression.

Type 5, the Ogden-Storakers material can be expressed with the following equation

$$W = \sum_{p=1}^N \frac{\mu_p}{\alpha_p} (\lambda_1^{\alpha_p} + \lambda_2^{\alpha_p} + \lambda_3^{\alpha_p} - 3) + \sum_{p=1}^N \frac{\mu_p}{\alpha_p \beta_p} (J^{-\alpha_p \beta_p} - 1) \quad (8)$$

$\mu_p$ ,  $\alpha_p$  and  $\beta_p$  are the material parameters used for this expression.

The input parameters can be determined by the code if an experimental stress-strain curve is given, see the description below under Case **parameters identification**. A best-fit is done in this case in order to calculate them. The data must be provided in engineering strains from an 1-D experiment. The lateral deflection should not be limited by the experiment.

The Ogden formulation of Type 3 is not yet tested in detail. First tests show a shrinkage of the material under initially unloaded conditions. This is physically not possible. It is strongly recommended **not** to use this material type.

The material law uses total strains. These strains are sometimes not correct when large rotations occur.

### Syntax:

#### Case 1 : TYPE = 1.

```
"HYPE"
"TYPE"      1
"RO"        rho
"C01"        c1
.            .
.            .
"C014"       c14
"BULK"       K
/LECTURE/
```

rho

Density.

C01

First coefficient of the potential

C014

14st coefficient of the potential

K

Compressibility coefficient, if 0.0 incompressible material is considered

LECTURE

List of the concerned elements.

#### Case 2 : TYPE = 2.

```
"HYPE"
"TYPE"      2
"RO"        rho
"C01"        c1
```

```
"C02"      c2
"C03"      c3
"BULK"     K
/LECTURE/
```

rho

Density.

C01

First coefficient of the potential (=A)

C02

Second coefficient of the potential (=B)

C03

Third coefficient of the potential (=C)

K

Compressibility coefficient

LECTURE

List of the concerned elements.

**Case 3 : TYPE = 3.**

```
"HYPE"
"TYPE"      3
"R0"        rho
"C01"       c1
"C02"       c2
"C03"       c3
"C04"       c4
"C05"       c5
"C06"       c6
"C07"       c7
"C08"       c8
"C09"       c9
"C010"      c10
"C011"      c11
"C012"      c12
"BULK"      K
/LECTURE/
```

rho

Density

C01,C02,C03,C04

Alpha coefficients of the potential ( $\alpha_p$ )

C05,C06,C07,C08

Mu coefficients of the potential ( $\mu_p$ )

C09,C010,C011,C012

(1/D) coefficient of the potential (compressible contribution)

LECTURE

List of the concerned elements.

**Case 4 : TYPE = 4.**

```
"HYPE"
"TYPE"      4
"R0"        rho
"C01"        c1
"C02"        c2
"C03"        c3
"C04"        c4
"C05"        c5
"C06"        c6
"C07"        c7
"C08"        c8
"BULK"       K
/LECTURE/
```

rho

Density

C01,C02,C03,C04

Alpha coefficients of the potential ( $\alpha_p$ )

C05,C06,C07,C08

Mu coefficients of the potential ( $\mu_p$ )

LECTURE

List of the concerned elements.

**Case 5 : TYPE = 5.**

```
"HYPE"
"TYPE"      5
"R0"        rho
"C01"        c1
"C02"        c2
"C03"        c3
"C04"        c4
"C05"        c5
```

```

"C06"      c6
"C07"      c7
"C08"      c8
"C09"      c9
"C010"     c10
"C011"     c11
"C012"     c12
/LECTURE/

```

rho

Density

C01,C02,C03,C04

Alpha coefficients of the potential ( $\alpha_p$ )

C05,C06,C07,C08

Mu coefficients of the potential ( $\mu_p$ )

C09,C010,C011,C012

Beta coefficients of the potential ( $\beta_p$ )

LECTURE

List of the concerned elements.

### Case parameters identification : **TYPE = 1, 3, 4 or 5.**

This case is recognized by the presence of the PCAL keyword in the input data, as shown below.

```

"HYPE"
"TYPE"    [1|3|4|5]
<"BULK"   k>
<"NU"     nu>
"PCAL"    npar
"TRAC"    npts * (strain stress)

```

k

Compressibility coefficient (*not* used for Type 5).

nu

Poisson's ratio (*only* used for Type 5).

npar

Number of parameters that should be calculated (i.e. between 1 and 4).

npts

Number of (strain, stress) couples of values given.



The type and the number of elements is irrelevant since only the material is called. (However, note that at least one element must be defined in order to run the code.) Four different models are possible: for the Mooney-Rivlin material a two-parameter model is included (CO1 and CO2); for the Ogden material a six-parameter model is included neglecting the influence of the D parameter and for the Ogden New model a six parameter model is included.

Note that as soon as the code encounters the TRAC keyword in the above syntax it reads the traction curve, then performs the parameters calibration *and stops*. Therefore, any parameters given **after** the TRAC subdirective **are simply ignored**. This means that if values should be set for the optional keywords BULK or NU, they must be entered *before* and *not after* the TRAC subdirective, as indicated in the syntax above.

For this reason, the usual /LECT/ at the end of the material directive is *not* included in the syntax (since it would not be interpreted anyway).

### Range of validity

Note that the range of validity of the hyperelastic material models is as follows:

- Neo-Hookean:  $\epsilon < 30\%$  (1 parameter). This corresponds to a Mooney-Rivlin material with only the first parameter defined.
- Mooney-Rivlin:  $\epsilon < 100\%$  for 2 or 3 parameters,  $\epsilon < 200\%$  for 4 to 9 parameters.
- Ogden new:  $\epsilon < 700\%$  for third order.

### Outputs:

The components of the ECR table are as follows:

ECR(1): Pressure

ECR(2): Von Mises Stress

ECR(3): Normal transverse strain (shell elements) or tangential stiffness (solid elements)

ECR(4): Updated thickness (shell elements)

ECR(5): Initial thickness (shell elements) or initial volume (solid elements)

ECR(6): Energy potential

ECR(7): Maximum time step for the element

## 7.7.56 MINT: MATERIAL FOR INTERFACE ELEMENT

### Object:

This directive allows to choose the material applied to interface elements. Thus, it can only be used with interface elements INT4 (2D quadrilateral), INT6 (3D triangular prism) and INT8 (3D hexahedron). The combination of such elements and material MINT forms a cohesive zone model, suitable to solve problems like delamination and debonding.

Only TYPE 2 material is functional. Three damage laws could be chosen with material TYPE 2: exponential, linear or Cachan interface meso-model.

- To select the exponential law, parameters C01 to C07 are required.
- To select the linear law, parameters C01 to C08 are required.
- To select the Cachan interface meso-model law, parameters C01 to C09 (C08 is optional) are required.
- Parameters C010 to C012 are optional in any case.

### References:

For the Cachan interface damage meso-model:

- Allix O. and Ladevèze P., *Interlaminar interface modelling for the prediction of delamination*. Composite Structures 22, 1992.
- Lévêque D., *Analyse de la tenue au délaminage des composites stratifiés : identification d'un modèle d'interface interlaminaire*. PhD thesis, ENS Cachan LMT, 1998.

The implementation of material TYPE 2 is explained in [\[916\]](#).

### Syntax:

```
"MINT" "TYPE" 2
      "C01" co1 "C02" co2 "C03" co3
      "C04" co4 "C05" co5 "C06" co6
      "C07" co7 "C08" co8 "C09" co9
      "C010" co10 "C011" co11 "C012" co12
      /LECTURE/
```

co1

Young's modulus along direction 3.

co2

Shear modulus between direction 1 and 3.

co3

Shear modulus between direction 2 and 3.

co4

Critical release rate in mode 1.

co5

Critical release rate in mode 2.

co6

Critical release rate in mode 3.

co7

Power coefficient to couple the thermodynamic forces of the three modes.

co8

Thermodynamical force threshold for damage. Required for the linear damage law. Optional for the Cachan interface meso-model.

co9

Exponent  $n$  for the Cachan interface meso-model.

co10

Delay effect : parameter  $\tau$  (optional).

co11

Delay effect : parameter  $a$  (optional).

co12

Maximum damage (optional, default value = 1.0).

LECTURE

List of the concerned elements.

### Comments:

When damage reaches the maximum damage value `co12`, element stiffness becomes null. Erosion algorithm is activated with `EROS` keyword (see page A.30, Section 4.4).

### Outputs:

The components of the ECR table are as follows:

ecr(1): Damage

ecr(2): Equivalent thermodynamic force

ecr(3): Time at which damage reaches 1.0 for failed elements

**7.7.57 THE SL-ZA MODEL****Object:**

This directive enables to choose the SLZA model which is an extension of both STEINBERG LUND and ZERILLI ARMSTRONG models. This model uses an expression for the internal stress that comes from the ZA model and an expression of the effective stress that comes from SL model.

**Syntax:**

```
"SLZA" "RO" rho    "YOUN" young  "NU" nu    "SIGE" sige
      "YA" ya     "YMAX" ymax   "YP" yp    "ER" er
      "N"  n      "C1"  c1     "UK" uk    "CP" cp
      "TM" tm     "T0"  t0     "BETA" beta
      /LECTURE/
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

sige

Elastic limit at ambient temperature.

ya

Coefficient of the CEA SL-ZA model.

ymax

Coefficient of the CEA SL-ZA model.

yp

Coefficient of the CEA SL-ZA model.

er

Coefficient of the CEA SL-ZA model.

n

Coefficient of the CEA SL-ZA model.

c1

Coefficient of the CEA SL-ZA model.

uk

Coefficient of the CEA SL-ZA model.

cp

Heat capacity per unit mass of the solid.

tm

Melting temperature of the solid.

t0

Initial temperature of the solid.

beta

Taylor and Quiney coefficient.

LECTURE

List of the elements concerned.

### Comments:

The expression of the elastic limit is given by:

$$y_d = (y_a + (y_{\max} - y_a)((1 - \exp(-e_p/e_r))^n)) + y_p(1 - \sqrt{kt/2u_k} \log(c_1/\dot{\epsilon}))$$

where  $k$  is the Boltzmann constant and  $\dot{\epsilon}$  is the strain rate.

### Outputs:

The components of the ECR table are as follows:

ecr(1) : Hydrostatic pressure

ecr(2) : Von mises stress

ecr(3) : Equivalent plastic strain

ecr(4) = Increment of temperature

ecr(5) = Elastic limit

ecr(6) = Total strain at the last timestep

ecr(7) = Time of the last call of the element

ecr(8) = Equivalent strain rate

### 7.7.58 RTM composite material

#### Object:

This directive allows to chose a composite material made by a RTM process. The behavior is orthotropic and the 9 independant coefficients can defined by using abaques of 3 or 4 parameters. These parameters are the volumic fraction, the angle between warp and weft directions and the warp and weft ratio. The 4th parameter is the temperature which can be optionnal.

#### Syntax:

```
"CRTM"
  "RO" rho
  "NTEM" ntem "NVF" nvf "NANG" nang "NRCT" nrct
  "PTM" ptem "PVF" pvf "PANG" pang "PRCT" prct
  "E11" ne11
    PAR1 val-par1-1 PAR2 val-par2-1 PAR3 val-par3-1
    TABLE nval-par4
      nval-par4 *(E11 PAR4)
  PAR1 val-par1-1 PAR2 val-par2-1 PAR3 val-par3-2
  TABL nval-par4
    nval-par4*(E11 , PAR4)
... then loop on PARA3, then PAR2 and PARA1.

"E22" ne22
  -idem-

"E33" ne33
  -idem-

"G12" ng12
  -idem-

"G13" ng13
  -idem-

"G23" ng23
  -idem-

"NU12" nnu12
  -idem-

"NU13" nnu13
  -idem-

"NU23" nnu23
```

-idem-

/LECTURE/

rho

Density of the material.

ntem

Number of values of temperature

nvf

Number of values of volumic fraction

nang

Number of values of angle between warp and weft

nrct

Number of values of ratio between warp and weft

ptem

Number of the temperature parameter

pvf

Number of the volumic fraction parameter

pang

Number of the angle parameter

prct

Number of the ratio between warp and weft parameter

ne11

Number of the abaque for E11

val-par1-1

First value of the parameter 1

val-par2-1

First value of the parameter 2

val-par3-1

First value of the parameter 3

val-par3-2

Second value of the parameter 3

nval-par4

Number of values of parameter 4

#### LECTURE

List of the elements concerned.

#### Comments:

1/ - It is possible to suppress the temperature dependant. In this case, one can use 3 parameters (from 1 to 3).

2/ - By defining ptem, prct, pang and pvf, it is possible to declare that temperature is parameter 1, volumic fraction is parameter 2 and any combination the user likes. It permits to use as general as possible an abaque of 4 parameters.

3/ - The values of angle, volumic fraction and ration between warp and weft have to be define by using the directive RTMANG, RTMVF and RTMRCT (page C63). The temperature is defined as initial values (command INIT TETA page E80).

#### Outputs:

The components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : Von mises criterion

ECR(3) : modulus E11

ECR(4) : modulus E22

ECR(5) : modulus E33

ECR(6) : modulus G12

ECR(7) : modulus G13

ECR(8) : modulus G23

ECR(9) : Poisson coefficient NU12

ECR(10) : Poisson coefficient NU13

ECR(11) : Poisson coefficient NU23



### 7.7.59 TVMC (LOI ELASTOPLASTIQUE POUR COMPOSITES)

#### Object:

Ce materiau permet de modeliser le comportement elastoplastique endommageable de composites a fibres courtes.

C'est le cas par exemple des composites injectes de type thermoplastique charge de fibres (verre, carbone, ...) comme ULTEM 2100, ou encore des composites SMC-R de type polyester charge de fibres (verre, carbone).

Cette loi est utilisable pour les elements volumiques. Elle se decompose en trois etapes :

- homogeneisation micro-mecanique,
- endommagement,
- plasticite couplee a l'endommagement.

#### Syntax:

```
"TVMC"  "ROF" rhof  "ROM" rhom  "TAUX" taux  "EM"  em  "NUM"  num ...
...  "EF"  ef  "NUF" nuf  "R"  rap  "TVF" tvf  "TE"  te  ...
...  "PH"  ph  "NF"  nf  "Y1C" y1c  "Y2C" y2c  "CRIT" choix  ...
...  "NFD1" n1  "NFD2" n2  "NFR" n3  /LECTURE/
```

rhof

Masse volumique de la fibre.

rhom

Masse volumique de la resine (chargee ou non chargee).

taux

Taux de porosite.

em

Module d'Young de la matrice.

num

Coefficient de Poisson de la matrice.

ef

Module d'Young de la fibre.

nuf

Coefficient de Poisson de la fibre.

rap

Rapport de forme de la fibre (longueur sur diametre).

tvf

Taux volumique de fibres.

te

Orientation dans le plan de la fibre (inutilise ici).

ph

Orientation hors plan de la fibre (inutilise ici).

nf

Nombre d'orientations de fibres dans le plan.

y1c

Taux de restitution limite de la matrice en traction.

y2c

Taux de restitution limite de la matrice en cisaillement.

choix

Numero du critere definissant la forme de la surface de charge.

n1

Numero de la fonction definissant l'endommagement en traction-compression en X et Y  
.

n2

Numero de la fonction definissant l'endommagement en cisaillement.

n3

Numero de la fonction definissant la courbe de plasticite a ecrouissage isotrope.

LECTURE

List of the elements concerned.

### Comments:

Le parametre "CRIT" peut prendre l'une des 4 valeurs suivantes :

1 = VON MISES,

2 = TRESCA,

3 = TSAI-HILL (en  $\sigma_1$  et  $\sigma_4$ ),

4 = TSAI-HILL (en  $\sigma_1$ ,  $\sigma_2$  et  $\sigma_4$ ),

### Outputs:

The components of the ECR table are as follows:

- ECR(1): pression hydrostatique,
- ECR(2):  $Y1$  = taux de restitution d'énergie en traction,
- ECR(3):  $Y2$  = taux de restitution d'énergie en cisaillement,
- ECR(4):  $D1$  = endommagement en traction,
- ECR(5):  $D2$  = endommagement en cisaillement,
- ECR(6): deformation plastique cumulee,
- ECR(7) : limite elastique courante,
- ECR(8:10): inusites,
- ECR(11): vitesse du son locale (pour la stabilite).

### 7.7.60 HILL MATERIAL MODEL

#### Object:

This directive enables to choose the HILL model which is a model with isotropic plasticity associated with a HILL criterion. The elastic behaviour of the material can be orthotropic.

#### Syntax:

```
"HILL" "R0" rho    "YG1" yg1  "YG2" yg2  "YG3" yg3
          "G12" g12  "G13" g13  "G23" g23
          "NU12" nu12 "NU13" nu13 "NU23" nu23
          "XT1" xt1  "XT2" xt2  "XT3" xt3
          "RST1" rst1 "RST2" rst2 "RST3" rst3
"TRAC" npts*( sig eps ) /LECTURE/
```

/LECTURE/

rho

Density of the material.

yg1

Young's modulus - direction 1

yg2

Young's modulus - direction 2

yg3

Young's modulus - direction 3

g12

shear modulus - plane 12

g23

shear modulus - plane 23

g13

shear modulus - plane 13

nu12

shear modulus - plane 12

nu23

shear modulus - plane 23

nu13

shear modulus - plane 13

xt1

yield stress - direction 1

xt2

yield stress - direction 2

xt3

yield stress - direction 3

rst1

yield stress - plane 12

rst2

yield stress - plane 23

rst3

yield stress - plane 13

"TRAC"

This key-word introduces the yield curve.

npts

Number of points (except the origin) defining the yield curve.

sig

normalised stress.

eps

Equivalent plastic strain. Note that the first point must be always (1., 0.)

LECTURE

List of the elements concerned.

### **Outputs:**

The components of the ECR table are as follows:

ecr(1) : Hydrostatic pressure

ecr(2) : Von Mises stress

ecr(3) : Equivalent plastic strain

ecr(7) : New elastic limit

### 7.7.61 GLASS MATERIAL

#### Object:

This option enables to choose a material that considers the strain rate effect of glass. A linear elastic material is used up to the failure. The failure limit **PSAR** uses the area under the stress-time curve (equivalent constant stress).

#### Syntax:

```
"GLAS"  "RO" rho  "YOUN" young  "NU" nu  "CORR" corr
        "FAIL" $[ VMIS ; PEPS ; PRES ; PEPR; PSAR ]$ "LIMI" limit
```

**rho**

Density of the material.

**young**

Young's modulus.

**nu**

Poisson's ratio.

**corr**

Stress corrosion fraction. Default value is 16.

**FAIL**

Introduces an element failure model, represented by a failure criterion and a by failure limit value. The available failure criteria are: **VMIS** for a criterion based upon Von Mises stress (isotropic criterion), **PEPS** for a criterion based upon the principal strain (see caveat below), **PRES** for a criterion based upon the hydrostatic stress, **PEPR** for a criterion based upon the principal strain if the hydrostatic stress is positive (traction): if the hydrostatic stress is negative (compression) there is no failure. **PSAR** for a criterion based upon equivalent constant stress of the duration of 60 s.

**limit**

Indicates the failure limit for the chosen criterion.

#### Comments:

When using a failure criterion based upon the principal strains (**PEPS** or **PEPR**) be aware that the criterion is based upon the *cumulated* strains. These are usually a good approximation of the total strains for elements using a convected reference frame for the stresses and strains (such as e.g. plate, shell or bar elements). The approximation is likely to be very bad, instead, for continuum-like elements, at least when there are large rotations.

**Outputs:**

The components of the ECR table are as follows:

ECR(1): current hydrostatic pressure

ECR(2): current equivalent stress (Von Mises)

ECR(3) Area under the (principal stress to the power of CORR)-time curve, the stress is divided by 1.E6 to avoid too big numbers.

ECR(4): equivalent constant stress of the duration 60 s.

ECR(5): sound speed

### 7.7.62 BL3S: REINFORCED CONCRETE LAW FOR DEM

#### Object:

This material law prescribes properties of the reinforced concrete for structures modeled with the discrete element method (DEM) via ELDI elements. Usually, both steel and concrete phases are present. Nevertheless, they may be used separately, i.e. it is possible to use only one material phase, either concrete or steel.

This model was first developed in J.Rousseau's PhD thesis then reviewed and further developed in A.Masurel's PhD thesis, with EDF financial support and collaboration with 3S-R Laboratory (Grenoble). For theoretical description of the laws see [935], [948].

#### Syntax:

```
"BL3S" |[ "BETON"  "RO"    rho    "YOUN" youn  "NU"    nu
              "T"      tens   "CO"    cohe   "PHII"  phii
              "PHIC"  phic   "ADOU"  adou
              < "ALPH"  alpha  "BETA"  beta   "GAMM"  gamma >
              < "CNEL"  cnel   "CNPL"  cnpl   "YUNL"  yunl
              "XI1"   xi1    "XI2"   xi2 >
              < "ETA"   eta   >
              < "EPS1"  eps1   "EPS2"  eps2   "SIGC"  sigc  "DET2"  det2
              "AFIL"  afil >
              < "KRES"  kres   "KSKN"  kskn >
              < "ROLR"  rolr   "CDMR"  cdmr >

              < "BIMA"  "YOUN" youn  "NU"    nu    "TN"    tn
              "CN"    cn    "TE"    te    "TMAX"  tmax
              "UMAX"  dmax  "PHII"  phii  "PHIC"  phic >
              /LECTURE/ ;

              < "ACIER"  "RO"    rho    "YOUN" youn  "NU"    nu
              "T"      tens   "ECRO"  sigmr  "AMAX"  amax
              < "BIMA"  "YOUN" youn  "NU"    nu    "TN"    tn
              "CN"    cn    "TE"    te    "TMAX"  tmax
              "UMAX"  dmax  "PHII"  phii  "PHIC"  phic >
              >

              /LECTURE/

]|
```

#### Parameters for concrete (BETON):

rho

Density of the material



youn

Young's modulus

nu

Poisson's ratio

tens

Maximum tensile strength ( $T > 0$ ).

cohe

Cohesion

phii

Internal friction angle

phic

Contact friction angle

adou

Softening coefficient (ratio between elastic and softening slopes  $>0$ )

alpha

1st parameter for micro-macro relations  $K=f(E,nu,alpha,beta,gamma)$ . The default value is 3.9 (see [941]).

beta

2nd parameter for micro-macro relations  $K=f(E,nu,alpha,beta,gamma)$ . The default value is 3.03125 (see [941]).

gamma

3rd parameter for micro-macro relations  $K=f(E,nu,alpha,beta,gamma)$ . The default value is 4.8115 (see [941]).

cnel

Local elastic compression limit

cnpl

Local plastic compression limit

yunl

Young's modulus for compression unload

xi1

Softening in compression

xi2

Hardening in compression

eta

Reduced damping coefficient on concrete cohesive links if needed

eps1

First limit of the strain rate effect (under EPS1 the behavior of concrete is considered as quasi-static)

eps2

Second limit of the strain rate effect formula

sigc

Static compressive strength used to calculate the first delta exponent of the strain rate effect law (first range)

det2

Second exponent of the strain rate effect law (second range)

afil

Weighting coefficient for the strain rate filtering (default value: 0.)

kres

Coefficient of restitution for a normal nonelastic choc for granular medium (default value: 1.)

kskn

Ratio of tangential and normal contact stiffnesses for granular medium (default value: 0.)

rolr

Rolling stiffness for granular medium (default value: 0.)

afil

Rolling damping coefficient for granular medium (default value: 0.)

LECTURE

List of the elements concerned.

### Parameters for steel (ACIER):

rho

Density of the material

youn

Young's modulus

nu

Poisson's ratio

tens

Maximum elastic stress ( $T > 0$ ).

sigmr

Maximum stress for steel

amax

Maximum allongation (%)

LECTURE

List of the elements concerned.

**Parameters for steel-concrete interface (BIMA):**

youn

Young's modulus

nu

Poisson's ratio

tn

Maximum normal tensile strength (perpendicular to the steel bar)

cn

Maximum normal compression strength (perpendicular to the steel bar)

te

Elastic limit in the tangential direction

tmax

Maximum strenght in the tangential direction

dmax

Coefficient to define maximum tangential sliding ( $u_{max}=d_{max}*u_{glis}$ )

phii

Internal friction angle

phic

Contact friction angle

LECTURE

List of the elements concerned.

### Comments:

If only concrete is modeled through the discrete element formulation, the sequence open by BETON keyword should be used only. In this case, reinforcement is modeled by the beam finite element model and steel-concrete links are defined by ACBE link model.

If both the concrete and the reinforcement are modeled by discrete elements, their properties must be defined separately (keywords BETON and ACIER respectively), and it is necessary to define also a specific behavior for the steel-concrete interface. This can be done by using a sequence of parameters introduced by the BIMA option. This option should be used only once, either with BETON or ACIER definition. If the sequence BIMA is not specified, the steel-concrete interface behaves as a concrete without taking into account the main direction of the reinforcement.

Don't forget to use directive ARMA in CELDI to declare the steel discrete elements. ARMA calculates the main direction of the reinforcement needed to define normal and tangential forces for the BIMA links.

### Outputs:

In the discrete element calculation BL3S material is used for the links. However, for post-processing purpose the number of active links and the degree of damage are reported onto the discrete elements.

The components of the ECR table are as follows:

ECR(1): number of COHE-type links per element at  $t = t_0$

ECR(2): number of BIMA-type links per element at  $t = t_0$

ECR(3): number of COHE-type links per element at  $t \geq t_0$

ECR(4): number of BIMA-type links per element at  $t \geq t_0$

ECR(5): degree of damage of COHE-type links per element

ECR(6): degree of damage of BIMA-type links per element

ECR(7): diameter of the discrete element.

### 7.7.63 LAMINATED SECURITY GLASS MATERIAL

#### Object:

This option enables to choose a material that considers laminated security glass. A linear elastic material is used up to the failure. After the failure, the material can react to compression but not more to tension. This material is recommended with a sandwich structure, where the interlayer can be built up with a elastoplastic material.

#### Syntax:

```
"LSGL"  "RO" rho  "YOUN" young  "NU" nu  <"CORR" corr>
        <"FAIL" $[ VMIS ; PEPS ; PRES ; PEPR; PSAR; VMPR ]$ "LIMI" limit>
        <"CR2D"> <"NEIG"> <"REDU" redu>
```

rho

Density of the material.

young

Young's modulus.

nu

Poisson's ratio.

corr

Stress corrosion fraction. Default value is 16. This value is only used by the failure criterion PASR. See following reference: Beason, W. Lynn, Morgan, James R.: Glass failure prediction model. Journal of Structural Engineering, 110 (2), pp. 197-212, 1984.

FAIL

Introduces an element failure model, represented by a failure criterion and a by failure limit value. The available failure criteria are: VMIS for a criterion based upon Von Mises stress (isotropic criterion), PEPS for a criterion based upon the principal strain (see caveat below), PRES for a criterion based upon the hydrostatic stress, PEPR for a criterion based upon the principal strain if the hydrostatic stress is positive (traction): if the hydrostatic stress is negative (compression) there is no failure. PSAR for a criterion based upon equivalent constant stress of the duration of 60 s. VMPR for a criterion based upon Von Mises stress (isotropic criterion), if the hydrostatic stress is positive (traction): if the hydrostatic stress is negative (compression) there is no failure.

limit

Indicates the failure limit for the chosen criterion.

CR2D

Introduces two-dimensional cracks, which means that the direction of the principle stress or strain is used to introduce a first crack. This crack is implemented in such a way that only the stresses normal to the crack direction are set to 0 (in the case of tension). If the failure criterion is reached for the direction parallel to the crack, then the integration point fails in both directions.

#### NEIG

If this material is used for 3D calculations, the glass part of the model should mainly eroded after the erosion of the interlayer. By using the keyword **NEIG** erosion of an element of the **LSGL** material is only taken into account, if a neighbour element (e.g. interlayer of another **LSGL** element) is already eroded.

#### REDU

In case of hydrostatic tension, the stresses are set to 0. Using keyword **REDU** the decreasing of the stresses can be smoothed. The tension stresses are multiplied with the value **REDU**, which should be less than 1.0. Default value for **REDU** is 0.0.

#### Comments:

When using a failure criterion based upon the principal strains (**PEPS** or **PEPR**) be aware that the criterion is based upon the *cumulated* strains. These are usually a good approximation of the total strains for elements using a convected reference frame for the stresses and strains (such as e.g. plate, shell or bar elements). The approximation is likely to be very bad, instead, for continuum-like elements, at least when there are large rotations.

The material should only be used with shell elements. The third component of the stresses and strains is neglected in the calculation of the failure criterion.

#### Outputs:

The components of the ECR table are as follows:

ECR(1): current hydrostatic pressure

ECR(2): current equivalent stress (Von Mises)

ECR(3) Area under the (principal stress to the power of **CORR**)-time curve, the stress is divided by 1.E6 to avoid too big numbers.

ECR(4): equivalent constant stress of the duration 60 s.

ECR(5): sound speed

ECR(6): failure flag (0=virgin Gauss Point, 1=failed Gauss Point)

ECR(7): angle of failure

ECR(8): status of the spalling: 0 no failure of the g.p.; -1 g.p. under compression; +1 g.p. under tension.

Note that in order to post-process the total strains (which were formerly inappropriately stored in the ECR table for JRC materials) one has to use the EPST table related to the element (like for CEA elements).

### 7.7.64 SMAZ: MAZARS-LINEAR ELASTIC LAW WITH DAMAGE FOR SPHC ELEMENTS

#### Object:

Isotropic linear elastic with Mazars damage for SPHC elements.

#### References:

1- Jacky MAZARS, "Application de la mécanique de l'endommagement au comportement non linéaire et à la rupture du béton de structure", Thèse de doctorat, Université Pierre et Marie Curie - Paris 6, 1984.

#### Syntax:

```
"SMAZ" "RO"   rho  "YOUN" young  "NU"   nu   "EPSD" epsd
      "DCRI" dcri "A"    a      "B"    b
      "TAUC" tauc "CSTA" csta                      /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

epsd

Initial strain threshold.

dcri

Critical value of damage (=1 per default).

a

Parameter A of the tension law (asymptote of the curve stress-strain)

b

Parameter B of the tension law (shape of the curve stress-strain)

tauc

Characteristic time for delay-damage

csta

Parameter of the delay-damage (=1 per default)

**Outputs:**

The components of the ECR table are as follows:

ECR(1) : Pressure

ECR(2) : Von Mises criterion

ECR(3) : Equivalent strain

ECR(4) : Failure state (0: no failure, 1: failed)



**7.7.65 SLIN: LINEAR ELASTIC LAW WITH DAMAGE FOR SPHC ELEMENTS****Object:**

Isotropic linear elastic with damage for SPHC elements.

**Syntax:**

```
"SLIN" "RO" rho "YOUN" young "NU" nu "EPSD" epsd  
      "DCRI" dcri "EPSR" epsr  
      "TAUC" tauc "CSTA" csta /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's ratio.

epsd

Initial strain threshold.

dcri

Critical value of damage (=1 per default).

epsr

Maximum strain before failure.

tauc

Characteristic time for delay-damage

csta

Parameter of the delay-damage (=1 per default)

**Outputs:**

The components of the ECR table are as follows:

- ECR(1) : Pressure
- ECR(2) : Von Mises criterion
- ECR(3) : Equivalent strain
- ECR(4) : Failure state (0: no failure, 1: failed)

**7.7.66 JCLM****Object :**

This directive allows to describe the behaviour of an elasto-plastic material that may undergo some damage, according to the Lemaitre model. There is coupling between damage and plasticity, represented by the Johnson-Cook model. The damage evolution rate is a function of the triaxiality ratio of stresses and of the equivalent plastic strain rate. A failure criterion is implicitly contained within the model: rupture occurs when the damage exceeds a critical value. Two optional parameters allow to introduce a limitation of the damage rate (thanks to the delayed damage model) in order to avoid the mesh dependency.

**Syntax:**

```
"JCLM"  "RO" rho "YOUN" young "NU" nu
        "EPSD" epsd "S0" s0 "DC" dc
        <"CSTA" csta "TAUC" tauc "NOCO" noco>
        "COA1" coa1 "COA2" coa2
        "CLB1" clb1 "CLB2" clb2 "SRRF" srrf /LECTURE/
```

rho

Density.

young

Young's modulus.

nu

Poisson's coefficient.

epsd

Damage threshold (i.e. equivalent plastic strain, weighted by a function of stress triaxiality, within which damage vanishes).

s0

Parameter driving the damage evolution rate.

dc

Critical damage defining the rupture criterion.

csta

Parameter of the delayed damage model

tauc

Characteristic time of the delayed damage model.  $(1/\text{tauc})$  represents the maximum damage rate.

noco

Optional parameter indicating what to do when no convergence is reached in the material routine. The value 0 is the default and means that an error message is issued and the calculation is stopped. The value 1 indicates that the element (or more precisely, the element's current Gauss point) is made to fail (eroded).

**coa1**

1st constant in the Johnson-Cook model.

**coa2**

2nd constant in the Johnson-Cook model.

**clb1**

3rd constant in the Johnson-Cook model.

**clb2**

Hardening coefficient of the Johnson-Cook model.

**srrf**

Reference strain rate of the Johnson-Cook model.

**LECTURE**

List of concerned elements.

### Comments:

A detailed description of the damage model can be found in the report DMT/98-026A, available on request.

The implementation of the Johnson-Cook model is described in reference [\[167\]](#).

This material is currently restricted to SPHC elements.

### Outputs:

The components of the ECR table are as follows for **Continuum elements**:

ECR(1) : pressure

ECR(2) : Von Mises criterion

ECR(3) : equivalent plastic strain

ECR(4) : plasticity multiplier

ECR(5) : damage

ECR(7) : new elastic limit

When the "erosion" algorithm is activated (see page A.30, Section [4.4](#), keyword **FAIL**), an element is considered as failed if **damage**  $\geq$  **dc**.

### 7.7.67 VPJC

#### Object :

This directive allows to define a Von Mises elasto-thermo-viscoplastic material with non-linear isotropic hardening governed by a modified Johnson-Cook model with explicit elastic predictor and return mapping algorithm, a Voce saturation type of hardening and a Cockcroft-Latham failure criterion. See report [373] for full details. It can be used in 3D, 2D plane strain, 2D plane stress or 2D axisymmetric cases. This material model was developed at NTNU (Trondheim, Norway).

The original Johnson-Cook model was first introduced in: G. R. Johnson and W. H. Cook. A constitutive Model and Data for Metals Subjected to Large Strains, High Strain Rates and High Temperatures. Proceedings of the 7th International Symposium on Ballistics, Hague (1983), 541–547.

The so-called “modified” Johnson Cook material law, in which the strain-rate sensitivity term is adjusted so as to avoid non-physical softening, was introduced in: M. Ortiz and G. T. Camacho. Adaptive Lagrangian modelling of ballistic penetration metallic targets. Computer Methods in Applied Mechanics and Engineering 142 (1997), 269–301. See also: T. Børvik, O. S. Hopperstad, T. Berstad, M. Langseth. A computational model of viscoplasticity and ductile damage for impact and penetration. Eur. J. Mech. A/Solids 20 (2001), 685–712.

The Voce saturation type of hardening was proposed in: E. Voce. The relationship between stress and strain for homogeneous deformation. Journal of the Institute for Metals 74 (1948), 536–562.

The expression of the constitutive law is the following:

$$\sigma_y = [A + Q_1 (1 - e^{-C_1 p}) + Q_2 (1 - e^{-C_2 p})] (1 + \dot{p}^*)^C (1 - T^{*m}) \quad (9)$$

and is the product of three factors (from left to right): a strain hardening term (in square brackets), a strain-rate hardening term and a temperature softening term. The symbols indicate the following:

- $\sigma_y$  is the current yield stress of the material
- $A$  is the initial yield stress of the material, sometimes also indicated as  $\sigma_0$
- $p$  is the equivalent (or cumulated) plastic strain, i.e. the energy-conjugated variable to the equivalent stress
- $\dot{p}$  is the equivalent plastic strain rate
- $\dot{p}^*$  is the dimensionless strain rate  $\dot{p}^* = \frac{\dot{p}}{\dot{p}_0}$ , with  $\dot{p}_0$  the reference strain rate
- $T^*$  is the dimensionless temperature  $T^* = \frac{T - T_r}{T_m - T_r}$ , with  $T$  the absolute temperature,  $T_r$  the absolute room temperature and  $T_m$  the absolute melting temperature
- $Q_1$ ,  $C_1$ ,  $Q_2$  and  $C_2$  are material constants used in the first factor on the right-hand side of the material law (strain-hardening term)
- $C$  is a material constant, the exponent appearing in the second factor, which represents the strain-rate hardening

- $m$  is a material constant, the exponent appearing in the third factor, which represents the temperature softening

### Temperature softening

The last term of the above equation accounts for the thermal softening of the yield stress at elevated temperatures. However, the evolution of the temperature remains to be established. The heat transfer is modelled by assuming adiabatic conditions. This implies that there is no heat transfer into or out of the system during plastic straining. The plastic energy dissipation  $D_p$  per unit volume in the form of heat (Watt per cubic meter) is given by:

$$D_p = \chi \sigma_{eq} \dot{p} = \rho C_T \dot{T} \quad (10)$$

where:

- $\chi$  is the Taylor-Quinney coefficient, i.e. the fraction of plastic power that is converted to heat. The remaining fraction  $1 - \chi$  is assumed to remain in the material due to structural rearrangements
- $\sigma_{eq}$  is the equivalent stress
- $\dot{p}$  is the equivalent plastic strain rate
- $\rho$  is the material density
- $C_T$  is the material heat capacity
- $\dot{T}$  is the temperature rate due to adiabatic heating

From the above expression, the temperature rate  $\dot{T}$  is obtained:

$$\dot{T} = \frac{D_p}{\rho C_T} = \frac{\chi \sigma_{eq} \dot{p}}{\rho C_T} \quad (11)$$

and then this value is integrated in time at each Gauss point to obtain the current temperature at the point. The initial temperature is set to the room temperature  $T_r$  at each Gauss point. If during the calculation the temperature at a Gauss point reaches the melting temperature  $T_m$ , the Gauss point fails.

### Gauss point failure and element erosion

The Cockcroft-Latham fracture criterion based on plastic work per unit volume is assumed. See: M. G. Cockcroft and D. J. Latham. Ductility and the workability of metals. Journal of the Institute of Metals 96 (1968), 33–39.

Material failure takes place at a Gauss point when a damage parameter  $D$  reaches the damage threshold  $D_c$ . The  $D_c$  parameter should be set by the user (see **DC** keyword below) such that  $0 < D_c \leq 1$ . The value 1 should be used when not considering damage softening. The damage is computed according to the following expression:

$$D = \frac{W}{W_c} = \frac{1}{W_c} \int_0^p \langle \sigma_1 \rangle dp \quad (12)$$

where:

- $\sigma_1$  is the maximum principal stress at the Gauss point

- The expression  $\langle \sigma_1 \rangle$  is equivalent to the function  $\max(0, \sigma_1)$ , which implies that only positive values of the maximum principal stress  $\sigma_1$  (i.e. tensile stress) contribute to the damage evolution
- $W_c$  is the failure material parameter, which can be found by integrating the major principal stress in a uniaxial tension test during the entire equivalent plastic strain path until the plastic strain at failure  $p_f$ . In this case (uniaxial traction) the major principal stress is just the (longitudinal) stress

An element's Gauss point is considered as *failed* if  $D \geq D_c$ , i.e. if the damage reaches the chosen threshold. If the "erosion" algorithm is activated (see [GBA\\_0030](#), keyword **EROS**), an element is *eroded* as soon as a chosen fraction (see **ldam** parameter of the **EROS** keyword) of its Gauss points reach failure.

### Syntax:

```
"VPJC"  "RO" rho "YOUN" young "NU" nu "ELAS" elas
        <"TOL" tol "MXIT" mxit>
        "QR1" qr1 "CR1" cr1 "QR2" qr2 "CR2" cr2
        "PDOT" pdot "C" c
        "TQ" tq "CP" cp <"TR" tr> "TM" tm "M" m
        "DC" dc "WC" wc
        <"SOLU" solu> <"DEBU" debu> <"RESI" resi>
        /LECTURE/
```

**rho**

Density  $\rho$ . Typically in  $\text{kg/m}^3$ .

**young**

Young's modulus  $E$ . Typically in Pa.

**nu**

Poisson's coefficient  $\nu$ . Dimensionless.

**elas**

Initial yield stress (indicated as  $A$  above, or sometimes as  $\sigma_0$ ). Typically in Pa.

**tol**

Tolerance for Newton-Raphson internal iterations. Dimensionless. The default is  $10^{-5}$ .

**mxit**

Maximum number of Newton-Raphson internal iterations. The default is 50.

**qr1**

Material constant  $Q_1$ , asymptote of the first Voce hardening term. It has the dimension of a stress, typically in Pa.

**cr1**

Material constant  $C_1$ , hardening parameter of the first Voce hardening term. Dimensionless.

qr2

Material constant  $Q_2$ , asymptote of the second Voce hardening term. It has the dimension of a stress, typically in Pa.

cr2

Material constant  $C_2$ , hardening parameter of the second Voce hardening term. Dimensionless.

pdot

Reference strain rate  $\dot{p}_0$  for the calculation of  $\dot{p}^*$ . Typically in  $\text{s}^{-1}$ .

c

Material constant  $C$ , hardening parameter (exponent) of the viscous term. Dimensionless. By setting  $C = 0$  one can model a quasi-static test, in which the visco-plasticity effect is not included.

tq

Taylor-Quinney coefficient  $\chi$ . Dimensionless.

cp

Specific heat capacity of the solid material  $C_T$ . Typically in  $\text{J}/(\text{kg}\cdot\text{K})$ .

tr

Room temperature  $T_r$  in K for the calculation of  $T^*$ . The default is 293 K. This is also taken as the initial temperature of the material.

tm

Melting temperature  $T_m$  in K for the calculation of  $T^*$ .

m

Material constant  $m$ , hardening parameter of the temperature term. Dimensionless. By using the special value  $m = 0$  the temperature softening effect is excluded from the model, i.e. the code assumes  $T^{*m} = 0$ , and therefore the temperature hardening term becomes  $(1 - T^{*m}) = (1 - 0) = 1$ . Note also that in this case the temperature is not updated, so that it remains to the room value  $T_r$ .

dc

Upper limit  $D_c$  of the damage  $D$  when softening occurs. Dimensionless. Material failure takes place at a Gauss point when the damage parameter  $D$  reaches  $D_c$ . The  $D_c$  parameter should be set by the user such that  $0 < D_c \leq 1$ . The value 1 should be used when not considering damage softening.

wc

Failure parameter  $W_c$  of the Cockcroft-Latham failure criterion. It has the dimension of work per unit volume, i.e.  $[\text{J}/\text{m}^3]$ , i.e. of a stress, typically expressed in Pa. By setting  $W_c$  to a very large value the failure of the material (and the consequent element erosion, if specified by the user) can be excluded from the model.

#### **solu**

Solution algorithm. By default (or by specifying **SOLU 1**) a cutting plane algorithm is adopted, which requires internal Newton-Raphson iterations (up to a maximum number prescribed via **MXIT**). The cutting plane algorithm was originally developed for rate-independent plasticity and should be used with some care for rate-dependent plasticity models. This is due to the fact that the plastic strain rate  $\dot{p}$  actually increases during the iterative update scheme and reaches the correct value of  $\dot{p}$  only at the final iteration. The result is that the return to the dynamic yield condition  $F = 0$  occurs at strain rates that are too low. Optionally, by specifying **SOLU 2**, one may choose a radial return solution algorithm. The radial return method also requires internal (Newton-Raphson) iterations and is a special case of the (implicit) backward Euler return map algorithm developed for the von Mises yield criterion with the associated flow rule. In this case, the return to the yield surface from the elastic trial state is radial to the yield surface in the deviatoric (stress) plane, which significantly simplifies the algorithm and makes the algorithm exceptionally stable and accurate. Note, however, that the radial return solution algorithm cannot be used with plane stress or uniaxial stress states (but can be used in 3D, 2D axisymmetric and 2D plane strain cases).

#### **debu**

Debugging option. By default (or by specifying **DEBU 0**) no debugging is activated. By specifying **DEBU 1**, whenever the maximum number of iterations **MXIT** is exceeded, before stopping the complete set of input arguments to the routine is written (to machine's precision) on the listing and on a binary file **\_VPJC.dat**. This allows to debug the routine by reading back the data and feeding them to the material routine under debugging control. Note that activating this option will slightly slow down the execution since the complete set of input data to the routine must be stored each time the material routine is called.

#### **resi**

Optional keyword to decide what to do when **MXIT** is reached without convergence. By default (or by specifying **RESI 0**) the code simply stops, with an error message (and stores the faulty state if **DEBU 1** has been set). By specifying **RESI 1**, whenever the maximum number of iterations **MXIT** is exceeded, the code assumes that convergence has been reached anyway and the calculation continues.

#### **/LECT/**

List of the elements concerned.

#### **Comments:**

All parameters are mandatory except **TOL**, **MXIT** and **TR**, which by default have the values  $10^{-5}$ , 50 and 293.0 K, respectively.

The various parameters can be grouped in the following classes:

- Elastic constants and density (**RO**, **YOUN** and **NU**).



- Yield stress and strain hardening (ELAS, QR1, CR1, QR2 and CR2).
- Strain rate hardening (PDOT and C).
- Damage evolution (DC and WC).
- Adiabatic heating and temperature softening (CP, TQ, TM, TR and M).
- Convergence of internal iterative Newton-Raphson procedure (TOL and MXIT).

Orientatively, some values of the parameters for typical materials could be as follows

- Docol 600 DL medium-strength steel

```
VPJC RO 7850 YOUN 210.0E9 NU 0.33 ELAS 370.0E6
      QR1 236.4E6 CR1 39.3 QR2 408.1E6 CR2 4.5
      PDOT 5.E-4 C 0.001 TQ 0.9 CP 452
      TM 1800.0 M 1.0 DC 1.0 WC 473.0E6
```

- S355 structural steel

```
VPJC RO 7850 YOUN 210.0E9 NU 0.33 ELAS 333.1E6
      QR1 236.3E6 CR1 16.5 QR2 416.5E6 CR2 1.2
      PDOT 5.E-4 C 0.011 TQ 0.9 CP 452
      TM 1800.0 M 0.94 DC 1.0 WC 848.0E6
```

- X65 offshore pipeline steel

```
VPJC RO 7850 YOUN 208.0E9 NU 0.33 ELAS 299.0E6
      QR1 160.0E6 CR1 25.0 QR2 400.0E6 CR2 0.25
      PDOT 5.E-4 C 0.01 TQ 0.9 CP 452
      TM 1993.0 M 1.0 DC 1.0 WC 1595.0E6
```

- X65 steel

```
VPJC RO 7800.0 YOUN 2.08E11 NU 0.30 ELAS 465.5E6 mxit 20
      QR1 147.0E6 CR1 10.62 QR2 665.9E6 CR2 0.50
      PDOT 8.06E-4 C 0.0104 TQ 0.9 CP 452.0
      TM 1800.0 M 1.0 DC 1.0 WC 1562.0E6
      RESI 1
```

- Weldom 500E steel

```
VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 605E6 mxit 20
      QR1 139.0E6 CR1 10.26 QR2 709.0E6 CR2 0.48
      PDOT 5.E-4 C 0.0166 TQ 0.9 CP 452.0
      TM 1800.0 M 1.0 DC 1.0 WC 1516.0E6
      RESI 1
```

- Aluminium alloy 1050-H14

```
VPJC RO 2700 YOUN 70.0E9 NU 0.3 ELAS 80.0E6
      QR1 49.3E6 CR1 1457.1 QR2 5.2E6 CR2 121.5
      PDOT 5.E-4 C 1.4E-2 TQ 0.9 CP 910.0
      TM 893.0 M 1.0 DC 1.0 WC 54.0E6
```

- Aluminium alloy 6016-T4

VPJC RO 2700 YOUN 70.0E9 NU 0.3 ELAS 137.0E6  
 QR1 19.1E6 CR1 592 QR2 170.0E6 CR2 11.4  
 PDOT 5.E-4 C 1.0E-3 TQ 0.9 CP 910.0  
 TM 893.0 M 1.0 DC 1.0 WC 140.0E6

- Aluminium alloy 6070-O

VPJC RO 2700 YOUN 70.0E9 NU 0.3 ELAS 38.8E6  
 QR1 79.5E6 CR1 56.9 QR2 88.2E6 CR2 4.0  
 PDOT 5.E-4 C 1.25E-2 TQ 0.9 CP 910.0  
 TM 893.0 M 1.0 DC 1.0 WC 179.0E6

- Aluminium alloy 6070-T4

VPJC RO 2700 YOUN 70.0E9 NU 0.3 ELAS 172.7E6  
 QR1 35.6E6 CR1 80.6 QR2 247.7E6 CR2 6.5  
 PDOT 5.E-4 C 1.25E-2 TQ 0.9 CP 910.0  
 TM 893.0 M 1.0 DC 1.0 WC 244.0E6

- Aluminium alloy 6070-T6

VPJC RO 2700 YOUN 70.0E9 NU 0.3 ELAS 350.0E6  
 QR1 30.1E6 CR1 185.9 QR2 72.8E6 CR2 7.7  
 PDOT 5.E-4 C 1.25E-2 TQ 0.9 CP 910.0  
 TM 893.0 M 1.0 DC 1.0 WC 130.0E6

- Aluminium alloy 6070-T7

VPJC RO 2700 YOUN 70.0E9 NU 0.3 ELAS 292.5E6  
 QR1 55.3E6 CR1 317.2 QR2 31.1E6 CR2 10.0  
 PDOT 5.E-4 C 1.25E-2 TQ 0.9 CP 910.0  
 TM 893.0 M 1.0 DC 1.0 WC 170.0E6

The material parameters are taken from the literature. See:

- J. K. Holmen, O.S. Hopperstad, T. Børvik. *Low velocity impact on multi-layered dual-phase steel plates*. International Journal of Impact Engineering **78** (2015), 161–177.
- J. K. Holmen, J. Johnsen, O.S. Hopperstad, T. Børvik. *Influence of fragmentation on the capacity of aluminium alloy plates subjected to ballistic impact*. European Journal of Mechanics A/Solids **55** (2016), 221–233.
- V. Aune, E. Fagerholt, K.O. Hauge, M. Langseth, T. Børvik. *Experimental study on the response of aluminium and steel plates subjected to airblast loading*. International Journal of Impact Engineering **90** (2016), 106–121.
- O.-G. Lademo, O. Engler, S. Keller, T. Berstad, K.O. Pedersen, O.S. Hopperstad. *Identification and validation of constitutive model and fracture criterion for AlMgSi alloy with application to sheet forming*. Materials and Design **30** (2009), 3005–3019.
- T. Børvik, S. Dey, A.H. Clausen. *Perforation resistance of five different high-strength steel plates subjected to small-arms projectiles*. International Journal of Impact Engineering **36** (2009), 948–964.

- M. Kristoffersen, F. Casadei, T. Børvik, M. Langseth, O.S. Hopperstad. *Impact against empty and water-filled X65 steel pipes – Experiments and simulations*. International Journal of Impact Engineering **71** (2014), 73–88.

### Outputs:

The components of the ECR table are as follows (the name of the variable in the material routine is also given, whenever applicable):

- ECR(1) : SIGMAH. Hydrostatic pressure ( $\frac{1}{3}\sigma_{kk}$ )
- ECR(2) : PHI. Von Mises equivalent stress ( $\sigma_{eq}$ )
- ECR(3) : P. Equivalent plastic strain ( $p$ )
- ECR(4) : PHITRIAL. Elastic trial equivalent (von Mises) stress
- ECR(5) : F. Yield function (which should be close to 0.0)
- ECR(6) : R. Total hardening of the material
- ECR(7) : DDLAMBDA. Change of the incremental plastic multiplier (from one time step to another)
- ECR(8) : DLAMBDA. Incremental plastic multiplier
- ECR(9) : NRITER. Number of iterations to obtain convergence
- ECR(10) : DLAMBDA / DT. Rate of plastic multiplier increment in time
- ECR(11) : D. Damage ( $D$ ), i.e. fraction of voids with respect to the gross cross-sectional area
- ECR(12) : Failure indicator: 1.0 = Virgin Gauss Point, 0.0 = Failed Gauss Point
- ECR(13) : T. Absolute temperature ( $T$ )
- ECR(14) : WE. Cockcroft-Latham damage accumulation ( $W$ )
- ECR(15) : Sound speed
- ECR(16) : First principal stress ( $\sigma_1$ )
- ECR(17) : Second principal stress ( $\sigma_2$ )
- ECR(18) : Third principal stress ( $\sigma_3$ )
- ECR(19) : RESNOR. Residual of the yield function, used to check convergence of the loop internal to the routine.
- ECR(20) : Stress triaxiality
- ECR(21) : Lode parameter

### 7.7.68 RIGI (Rigid Material)

**Object :**

This directive allows to define a rigid material to be associated with a rigid body. The geometrical characteristics of a rigid body are defined by using the COMP RIGI directive, see Page C.99B.

**Syntax:**

```
"RIGI"  "RO" rho  
        /LECTURE/
```

rho

Density  $\rho$ . Typically in kg/m<sup>3</sup>.

/LECT/

List of the elements concerned.

**Comments:**

All elements listed in the /LECT/ directive *must* belong to a rigid body declared in the COMP RIGI directive as described on Page C.99B.

The values of the density  $\rho$  is ignored by the code if the total mass, the center of gravity or the inertia tensor of the rigid body are prescribed by the user, see Page C.99B (RIGI directive) for details. However, even in this case a value for  $\rho$  *must* be specified in the present RIGI material for input completeness.

ECR(1) : empty at the moment.

### 7.7.69 DCMS (Damage in Coarsely Meshed Shells)

#### Object :

This directive may be used to model the onset of Damage, up to failure, in Coarsely Meshed metallic Shell (DCMS) structures. The DCMS material can only be used with shell elements, namely with elements subjected to plane stress conditions ( $\sigma_z = 0$ ).

For the formulation of this material see the following references:

- Storheim M., Alsos H.S, Hopperstad O.S, Amdahl J. *A damage-based failure model for coarsely meshed shell structures*. International Journal of Impact Engineering 83 (2015) 59–75.
- Alsos H.S, Hopperstad O.S, Törnqvist R., Amdahl J. *Analytical and numerical analysis of sheet metal instability using a stress based criterion*. International Journal of Solids and Structures 45 (2008) 2042–2055.

#### Syntax:

```
"DCMS"  "RO" rho "YOUN" young "NU" nu "ELAS" elas
        "K" k "N" n "EPSY" epsy "GF" gf
        "IMES" imes "IDAM" idam
        /LECTURE/
```

rho

Density  $\rho$ . Typically in  $\text{kg/m}^3$ .

young

Young's modulus  $E$ . Typically in Pa.

nu

Poisson's coefficient  $\nu$ . Dimensionless.

elas

Initial yield stress. Typically in Pa.

k

Power-law hardening coefficient.

n

Power-law hardening exponent.

n

Yield plateau strain.

gf

Fracture energy.

imes

Mesh scaling: 0 means no mesh scaling, 1 means mesh scaling.

idam

Damage coupling: 0 means no damage coupling, 1 means damage coupling.

/LECT/

List of the elements concerned.

### Comments:

Blabla ...

### Outputs:

The components of the ECR table are as follows (the name of the variable in the material routine is also given, whenever applicable):

ECR(1) : SIGH. Hydrostatic pressure.

ECR(2) : PHI. von Mises equivalent stress  $\Phi$ .

ECR(3) : EPSP. Equivalent plastic strain  $\epsilon_p$ .

ECR(4) : DAM. Damage  $D$ . The damage is limited to 0.95, that is  $D = \min(0.95, 1 - ((P_u - \epsilon_p)/(P_u - P_c)))$ .

ECR(5) : TRIAX. Triaxiality  $\tau$ .

ECR(6) : YF. Yield function  $Y_f$ .

ECR(7) : ITER. Number of iterations for plasticity  $N$ .

ECR(8) : ALFA. Alfa ratio  $\alpha = s_2/s_1$  where  $s_1$  is the maximum principal stress and  $s_2$  the minimum principal stress.

ECR(9) : BETA. Beta coefficient  $\beta = (2\alpha - 1)/(2 - \alpha)$ .

ECR(10) : THICK. Element thickness  $t$ .

ECR(11) : SQRT(SAREA). Equivalent element length  $L_e = \sqrt{A}$ .

ECR(12) : THICK/SQRT(SAREA). Thickness/length ratio  $t/L_e$ .

ECR(13) : HSV(10). Integration point has reached BWH (Bressan, Williams, Hill) instability (0=no, 1=yes).

ECR(14) : PC. Plastic strain when BWH instability is reached  $P_c$ .

ECR(15) : PU. Plastic strain at element failure  $P_u$ .

ECR(16) : SIG1. First (maximum) principal stress  $s_1$  of the plane stress state.

ECR(17) : SIG2. Second (minimum) principal stress  $s_2$  of the plane stress state.

### 7.7.70 MOONEY-RIVLIN MATERIAL

#### Object:

This sub-directive defines a hyperelastic material of the Mooney-Rivlin type. An *incompressible* Mooney-Rivlin hyperelastic material is described by:

$$W = C_1(\bar{I}_1 - 3) + C_2(\bar{I}_2 - 3) \quad (13)$$

where  $W$  is the strain energy density function,  $C_1$  and  $C_2$  are empirically determined material constants and:

$$\bar{I}_1 = J^{-2/3} I_1 \quad I_1 = \lambda_1^2 + \lambda_2^2 + \lambda_3^2 \quad (14)$$

$$\bar{I}_2 = J^{-4/3} I_2 \quad I_2 = \lambda_1^2 \lambda_2^2 + \lambda_2^2 \lambda_3^2 + \lambda_3^2 \lambda_1^2 \quad (15)$$

$$I_3 = J^2 = \lambda_1^2 \lambda_2^2 \lambda_3^2 \quad (16)$$

Here  $I_1$  and  $I_2$  are the first and second invariants of the unimodular component of the left Cauchy-Green deformation tensor and:

$$J = \det \underline{F} = \lambda_1 \lambda_2 \lambda_3 \quad (17)$$

with  $\underline{F}$  the deformation gradient. For an incompressible material  $J = 1$ .

For a *compressible* Mooney-Rivlin material eq. (13) becomes:

$$W = C_1(\bar{I}_1 - 3) + C_2(\bar{I}_2 - 3) + K(\ln I_3)^2 \quad (18)$$

with  $K$  the bulk modulus and  $I_3$  is the third invariant, given by eq. (16).

The material parameters  $C_1$  and  $C_2$  can be determined by EPX itself by a best fit procedure if a 1-D experimental stress-strain curve is available (see **Parameters Calibration mode** below).

The range of validity of this material model is as follows:

- 1 parameter ( $C_1$ ) :  $\epsilon < 30\%$  (Neo-Hookean).
- 2 parameters ( $C_1$  and  $C_2$ ) :  $\epsilon < 100\%$ .

#### Syntax:

Two input syntaxes are available. The first one is for the normal use of the material model, while the second one (introduced by the special keyword PCAL, for Parameters CALibration) is used to identify the material parameters.

```
"MOON" $ "RO" rho      <"BULK" k> "C1" c1 "C2" c2 <"INIS" inis>
          <"GINF" ginf> <"G1" g1> <"TAU1" tau1> <"G2" g2"> <"TAU2" tau2>
                      <"G3" g3> <"TAU3" tau3> <"G4" g4"> <"TAU4" tau4>
                      <"G5" g5> <"TAU5" tau5> <"G6" g6"> <"TAU6" tau6>
                                                    /LECT/ ;
"PCAL" npar <"BULK" k> "TRAC" npts * (strain stress) $
```

**Normal mode****rho**

Density.

**k**Compressibility coefficient. If omitted, the code takes  $K = 0$  and an incompressible material is modelled.**c1**First coefficient of the potential  $C_1$ .**c2**Second coefficient of the potential  $C_2$ . If  $C_2 = 0$ , the model becomes a Neo-Hookean material.**inis**

Initial stiffness (used to compute the sound speed in the material). If omitted, the code estimates it.

**ginf** $g_\infty$ . Optional viscosity-related parameter.**g1 ... g6** $g_1 \dots g_6$ . Optional viscosity-related parameters.**tau1 ... tau6** $\tau_1 \dots \tau_6$ . Optional viscosity-related parameters.**/LECT/**

List of the concerned elements.

**Parameters Calibration mode****PCAL npar**Special keyword that activates the Parameters CALibration mode. If present, the **PCAL** keyword must immediately follow the **MOON** keyword. The **npar** value indicates the number of parameters that should be computed (which *must be* 2 for this material model).**k**Compressibility coefficient. If omitted, the code takes  $K = 0$  and an incompressible material is modelled.**TRAC**Introduces the definition of the experimental traction curve. The number of points is **npts** and then exactly **npts** couples of values must be specified, which are interpreted as stress-strain pairs.



This mode is activated by the presence of the **PCAL** keyword immediately following the **MOON** keyword in the input data, as mentioned above. A best fit is performed in order to calculate the parameters. The traction curve data must be provided in engineering terms from a purely 1-D experiment (that is, lateral strains should *not* be restrained in the experiment.)

In this mode, the type and the number of elements is irrelevant since the material routine is called directly from the material reading procedure. Then the code computes the best fit and stops immediately. (However, note that at least one element must be defined in order to keep EPX happy.)

For this reason, the usual **/LECT/** at the end of the material directive is *not* included in this second syntax (since it would not be interpreted anyway.)

### Outputs:

The components of the ECR table are as follows:

ECR(1): Pressure.

ECR(2): Von Mises Stress.

ECR(3): Normal transverse strain (shell elements) or tangential stiffness (solid elements).

ECR(4): Updated thickness (shell elements).

ECR(5): Initial thickness (shell elements) or initial volume (solid elements).

ECR(6): Energy potential.

ECR(7): Maximum time step for the element.

ECR(8-35): Unused.

### 7.7.71 OGDEN MATERIAL

#### Object:

This sub-directive defines a hyperelastic material of the Ogden type. The expression of the strain energy density is one of the following expressions:

$$\text{Type 1} \quad W = \sum_{p=1}^N \frac{\mu_p}{\alpha_p} (\lambda_1^{*\alpha_p} + \lambda_2^{*\alpha_p} + \lambda_3^{*\alpha_p} - 3) + K(J - 1 - \ln J) \quad (19)$$

$$\text{Type 2} \quad W = \sum_{p=1}^N \frac{2\mu_p}{\alpha_p^2} (\lambda_1^{*\alpha_p} + \lambda_2^{*\alpha_p} + \lambda_3^{*\alpha_p} - 3) + \frac{1}{2}K(J - 1)^2 \quad (20)$$

where  $\lambda^* = \lambda J^{-\frac{1}{3}}$ ,  $K$  is the bulk modulus,  $\mu_p$  and  $\alpha_p$  are material parameters. The present implementation can go up to 4 terms ( $N = 4$ ) plus the volumetric one if  $K \neq 0$  in the expression of the potential  $W$ . At least the first term ( $\alpha_1, \mu_1$ ) must be defined. The first form eq. (19) is the classical one, the second form eq. (20) is the one found in some codes, e.g. Abaqus.

The material parameters  $\alpha_p$  and  $\mu_p$  can be determined by EPX itself by a best fit procedure if a 1-D experimental stress-strain curve is available (see **Parameters Calibration mode** below).

Note that the range of validity of this material model is :

- $\epsilon < 700\%$  for third order.

#### Syntax:

Two input syntaxes are available. The first one is for the normal use of the material model, while the second one (introduced by the special keyword PCAL, for Parameters CALibration) is used to identify the material parameters.

#### Syntax

```
"OGDE" $ "RO" rho      <"BULK" k>
                        "AL1" al1 <"AL2" al2> <"AL3" al3> <"AL4" al4>
                        "MU1" al1 <"MU2" al2> <"MU3" al3> <"MU4" al4>
                        <"INIS" inis> <"TYPE" type>
<"GINF" ginf> <"G1" g1> <"TAU1" tau1> <"G2" g2> <"TAU2" tau2>
                <"G3" g3> <"TAU3" tau3> <"G4" g4> <"TAU4" tau4>
                <"G5" g5> <"TAU5" tau5> <"G6" g6> <"TAU6" tau6>
                                /LECT/ ;

"PCAL" npar "TRAC" npts * (strain stress)      $
                % <"BULK" k> % not implemented
```

#### Normal mode

rho

Density.

k

Compressibility coefficient. If omitted, the code takes  $K = 0$  and an incompressible material is modelled.

**a11,a12,a13,a14**

Alpha coefficients of the potential ( $\alpha_p$ ). At least **a11 must** be specified.

**mu1,mu2,mu3,mu4**

Mu coefficients of the potential ( $\mu_p$ ). At least **mu1 must** be specified.

**inis**

Initial stiffness (used to compute the sound speed in the material). If omitted, the code estimates it.

**type**

Type of the formulation, i.e. either 1 for formula (19) or 2 for formula (20). If omitted, the code uses type 1, i.e. the classical formula.

**ginf**

$g_\infty$ . Optional viscosity-related parameter.

**g1 ... g6**

$g_1 \dots g_6$ . Optional viscosity-related parameters.

**tau1 ... tau6**

$\tau_1 \dots \tau_6$ . Optional viscosity-related parameters.

**/LECT/**

List of the concerned elements.

## Parameters Calibration mode

**PCAL npar**

Special keyword that activates the Parameters CALibration mode. If present, the **PCAL** keyword must immediately follow the **OGDE** keyword. The **npar** value indicates the number of parameters that should be computed (which *must be* between 1 and 4 for this material model).

**TRAC**

Introduces the definition of the experimental traction curve. The number of points is **npts** and then exactly **npts** couples of values must be specified, which are interpreted as stress-strain pairs.

This mode is activated by the presence of the **PCAL** keyword immediately following the **OGDE** keyword in the input data, as mentioned above. A best fit is performed in order to calculate the parameters. The traction curve data must be provided in engineering terms from a purely 1-D experiment (that is, lateral strains should *not* be restrained in the experiment.)

In this mode, the type and the number of elements is irrelevant since the material routine is called directly from the material reading procedure. Then the code computes the best fit and

stops immediately. (However, note that at least one element must be defined in order to keep EPX happy.)

For this reason, the usual /LECT/ at the end of the material directive is *not* included in this second syntax (since it would not be interpreted anyway.)

### Outputs:

The components of the ECR table are as follows:

ecr(1) = Pressure  
 ecr(2) = Von misses stress  
 ecr(3) = Max principal Cauchy stress  
 ecr(4) = W (internal energy density)  
 ecr(5) = Von Mises natural strain  
 ecr(6) = Max principal natural strain  
 ecr(7) = Von Mises engineering strain  
 ecr(8) = Max principal engineering strain  
 ecr(9) = Max von mises stress historical  
 ecr(10) = Historical Max principal Cauchy stress  
 ecr(11) = Historical Max von Mises natural strain  
 ecr(12) = Historical Max principal natural strain  
 ecr(13) = Historical Max von Mises engineering strain  
 ecr(14) = Historical Max principal engineering strain  
 ecr(15) = Historical Max Pressure (traction)  
 ecr(16) = Historical Min Pressure (compression)  
 ecr(17) = CSON0 (sound speed estimated by new method)  
 ecr(18) = CSON1 (sound speed estimated like for material HYPE TYPE 4) (the maximum between CSON0 and CSON1 is retained as speed of sound)  
 ecr(19) = Initial Volume (needed for the calculation of the internal energy)  
 ecr(20) = PI1. Pressure resulting from the Prony series (only when viscosity is enabled in the materuial model)  
 ecr(21:23) = LAMB(1:3) Principal stretches of the element  
 ecr(24:29) = Sich (1:6) (only when viscosity is enabled in the materuial model)  
 ecr(30:35) = Si1 (1:6) (only when viscosity is enabled in the materuial model)

### 7.7.72 BLATZ-KO MATERIAL

#### Object:

This sub-directive defines a hyperelastic material of the Blatz-Ko type. This material is still under development.

$$W = \frac{\mu\alpha}{2} \left[ (I_1 - 3) + \beta(I_3^{-1/\beta} - 1) \right] + \frac{\mu(1-\alpha)}{2} \left[ \left( \frac{I_2}{I_3} - 3 \right) + \beta(I_3^{1/\beta} - 1) \right] \quad (21)$$

where  $W$  is the strain energy density function,  $\alpha[0 \leq \alpha \leq 1]$  a material constant,  $\beta = \frac{1-2\nu}{\nu}$  and being  $\mu$  &  $\nu$  the shear and the Poisson modulus respectively in small strains (in large strains it does not have physical sense).

$$I_1 = \lambda_1^2 + \lambda_2^2 + \lambda_3^2 \quad (22)$$

$$I_2 = \lambda_1^2\lambda_2^2 + \lambda_2^2\lambda_3^2 + \lambda_3^2\lambda_1^2 \quad (23)$$

$$I_3 = J^2 = (\det \underline{F})^2 = \lambda_1^2\lambda_2^2\lambda_3^2 \quad (24)$$

Here  $I_1$ ,  $I_2$  and  $I_3$  are the first, second and third invariants of the unimodular component of the left Cauchy-Green deformation tensor,  $J$  the Jacobian with  $\underline{F}$  the deformation gradient. For an incompressible material  $J = 1$ .

Note that for *incompressibility* ( $J = I_3 = 1$ ) the Blatz-Ko material have an similar expression as the *Mooney-Rivlin*.

$$W = \frac{\mu\alpha}{2}(I_1 - 3) + \frac{\mu(1-\alpha)}{2}(I_2 - 3) \quad (25)$$

The material parameters  $C_1$  and  $C_2$  can be determined by EPX itself by a best fit procedure if a 1-D experimental stress-strain curve is available (see **Parameters Calibration mode** below).

The range of validity of this material model is as follows:

- 1 parameter ( $C_1$ ) :  $\epsilon < xx\%$  (Neo-Hookean).
- 2 parameters ( $C_1$  and  $C_2$ ) :  $\epsilon < xx\%$ .

#### Syntax:

Two input syntaxes are available. The first one is for the normal use of the material model, while the second one (introduced by the special keyword PCAL, for Parameters CALibration) is used to identify the material parameters.

```
"BLK0" $ "RO" rho    <"ALPH" alpha> "NU" nu "MU" mu <"INIS" inis>
          <"GINF" ginf> <"G1" g1> <"TAU1" tau1> <"G2" g2> <"TAU2" tau2>
          <"G3" g3> <"TAU3" tau3> <"G4" g4> <"TAU4" tau4>
          <"G5" g5> <"TAU5" tau5> <"G6" g6> <"TAU6" tau6>
                                     /LECT/ ;
"PCAL" npar <"nu" nu> "TRAC" npts * (strain stress) $
```

**Normal mode****rho**

Density.

**alpha** $[0 \leq \alpha \leq 1]$  Material constant**nu**For small strains, Poisson modulus ( $\nu$ ). For large strains does not have physical sense.**mu**For small strains, Shear modulus ( $\mu$ ). For large strains does not have physical sense.**inis**

Initial stiffness (used to compute the sound speed in the material). If omitted, the code estimates it.

**ginf** $g_\infty$ . Optional viscosity-related parameter.**g1 ... g6** $g_1 \dots g_6$ . Optional viscosity-related parameters.**tau1 ... tau6** $\tau_1 \dots \tau_6$ . Optional viscosity-related parameters.**/LECT/**

List of the concerned elements.

**Parameters Calibration mode****PCAL npar**Special keyword that activates the Parameters CALibration mode. If present, the **PCAL** keyword must immediately follow the **MOON** keyword. The **npar** value indicates the number of parameters that should be computed (which *must be* 2 for this material model).**k**Compressibility coefficient. If omitted, the code takes  $K = 0$  and an incompressible material is modelled.**TRAC**Introduces the definition of the experimental traction curve. The number of points is **npts** and then exactly **npts** couples of values must be specified, which are interpreted as stress-strain pairs.

This mode is activated by the presence of the **PCAL** keyword immediately following the **BLK0** keyword in the input data, as mentioned above. A best fit is performed in order to calculate the parameters. The traction curve data must be provided in engineering terms from a purely 1-D experiment (that is, lateral strains should *not* be restrained in the experiment.)

In this mode, the type and the number of elements is irrelevant since the material routine is called directly from the material reading procedure. Then the code computes the best fit and stops immediately. (However, note that at least one element must be defined in order to keep EPX happy.)

For this reason, the usual **/LECT/** at the end of the material directive is *not* included in this second syntax (since it would not be interpreted anyway.)

### Outputs:

The components of the ECR table are as follows:

ECR(1): Pressure.

ECR(2): Von Mises Stress.

ECR(3): Normal transverse strain (shell elements) or tangential stiffness (solid elements).

ECR(4): Updated thickness (shell elements).

ECR(5): Initial thickness (shell elements) or initial volume (solid elements).

ECR(6): Energy potential.

ECR(7): Maximum time step for the element.

ECR(8-35): Unused.

## 7.8 FLUID MATERIALS

### Object:

The following directives describe fluid materials for continuum elements.

Here are the different material types:

number	name	ref	law of behaviour
34	ADCR	7.8.19	homogeneous mixture with 3 components (1 liquid + 2 gases)
53	ADCJ	7.8.25	hypothetical core disruptive accident with law of type JWL for the bubble
57	BILL	7.8.26	specialised equation of state for the particle elements
59	BUBB	7.8.38	Balloon model for air blast simulations
68	CDEM	7.8.39	Discret Equation Method for Combustion
51	CHOC	7.8.22	Shock waves, Rankine-Hugoniot equation
110	DEMS	7.8.40	Discret Equation Method for Two Phase Stiffened Gases
22	EAU	7.8.9	two-phase water (liquid + vapour)
49	EXVL	7.8.20	hydrogen explosion Van Leer
27	FLFA	7.8.15	rigid tube bundles (homogeneous acoustic model)
86	FLMP	7.8.35	Fluid multi-phase
7	FLUI	7.8.2	isothermal fluid ( $c = \text{cte}$ )
36	FLUT	7.8.30	fluid, to be specified by the user
73	GAZD	7.8.41	Detonation in gas Mixture
9	GAZP	7.8.4	perfect gas
118	GGAS	7.8.1	generic ideal gas material
52	GPDI	7.8.23	diffusive perfect gas Van Leer
48	GVDW	7.8.28	Van Der Waals gas
40	GZPV	7.8.24	perfect gas for Van Leer
28	HELI	7.8.10	helium
50	JWL	7.8.21	explosion (Jones-Wilkins-Lee model)
66	JWLS	7.8.29	Explosion (Jones-Wilkins-Lee for solids)
23	LIQU	7.8.14	incompressible (or quasi-) fluid
82	MCFF	7.8.34	multicomponent fluid material (far-field)
81	MCGP	7.8.33	multicomponent fluid material (perfect gas)
33	MHOM	7.8.16	pipe bundle (homogeneous asymptotic model)
25	MULT	7.8.13	multiple materials (coupled monodim.)
10	NAH2	7.8.7	sodium-water reaction (1 liquid and 1 gas)
56	PARO	7.8.11	friction and heat exchange for pipeline walls
39	PUFF	7.8.17	equation of state of type "PUFF"
54	RSEA	7.9.13	sodium-water reaction (1 liquid and 2 gases)
103	SG2P	7.8.36	Multicomponent Stiffened Gases - Conservative formulation
104	SGMP	7.8.37	Multicomponent Stiffened Gases models
24	SOUR	7.8.6	imposed time-dependent internal pressure
102	STIF	7.8.5	Stiffened Gas
101	TAIT	7.8.3	Tait Equation of State

### Comments:



These materials are detailed in the following pages.

All pressures given as parameters are absolute pressures that must account for the external pressure. If one wants to avoid an unwanted transient expansion, it is necessary to specify the reference pressure "PREF", which must be the same for all fluid materials in a calculation.

For example, for a reservoir filled with gas at the relative pressure of 10 MPa ( $P_{int} - P_{ext} = 10 \text{ MPa}$ ), it is necessary to specify an internal pressure of 10.1 MPa if the atmospheric pressure is 0.1 MPa. Then, two cases are possible:

1) The reservoir is initially in equilibrium:

The calculation aims at simulating the response of the reservoir to an overpressure which appears later on (shock, explosion, imposed velocity ...). The reference pressure must then be:  $p_{ref} = 10.1 \text{ MPa}$ , so that the reservoir remains initially in equilibrium.

2) The reservoir is not initially in equilibrium:

The calculation aims at simulating the response of the reservoir to an an internal pressure which appears abruptly. The reference pressure must then be:  $p_{ref} = 0.1 \text{ MPa}$ , so that the final status be correct.

### 7.8.1 GENERIC IDEAL GAS

**Object:**

Perfect gas ( $P = \rho(\gamma - 1)E_{\text{internal}}$ )

This option enables materials with a ideal gas behaviour to be used. It is an interface to convert the input to the appropriate material (GAZP 7.8.4, FLUT 7.8.30) for the elements used.

**Syntax:**

```
"GGAS" ![ "R0" rho "GAMMA" gamma ["PINI" pini | "EINI" eini]  
... < "PREF" pref > ]! /LECTURE/
```

**rho**

Initial density.

**gamma**

Ratio  $c_P/c_V$  (supposed constant).

**pini**

Initial pressure.

**pref**

Reference pressure. Note that, by default, it is assumed **pref** = **pini** for CEA elements (GAZP) and **pref**=0 for JRC elements (FLUT).

/LECTURE/

List of the elements concerned.

**Outputs:**

The output variables are according to the material in which the generic material is converted.

## 7.8.2 FLUID

### Object:

This option enables a fluid (liquid-like) behaviour for continuum elements to be input. The fluid (isothermal) can be perfect (no viscosity) or viscous.

The expression used to compute the absolute pressure  $p$  in the fluid is:

$$p = p_{\text{ini}} + (\rho - \rho_{\text{ini}})c^2$$

where  $p_{\text{ini}}$  is the fluid pressure in the initial state,  $\rho$  is the current density,  $\rho_{\text{ini}}$  is the initial density and  $c$  is the sound speed, which is considered constant.

By default the fluid is considered “free” (i.e. fluid alone, keyword **LIBR**). However, it is also possible to take into account the volume occupied by some fixed internal structures (which are not meshed) by specifying the optional keyword **POREUX**. Such a “porosity” may be specified either in 2D or in 3D, but only for the elements of type **CAR1**, **CUBE** and **PRIS**.

### Syntax:

For a "free" fluid (no internal structures) :

-----

```
"FLUID"  < "LIBR" >  "RO" rho  "C" c  <"PINI" pini>  ...
... <"PREF" pref >  <"PMIN" pmin >  <"VISC" mu >  ...
... /LECTURE/
```

For a "porous" fluid (with internal structures) :

-----

```
"FLUID"  "PORE"  "RO" rho  "C" c  <"PINI" pini>  ...
... <"PREF" pref >  <"PMIN" pmin >  <"VISC" mu >  ...
... "PORO" alpha  < "SMOU" sur >  < "BETA" beta >  ...
... $ "KPER" kp ; "KPX" kpx  "KPY" kpy  < "KPZ" kpz > $ ...
... /LECTURE/
```

### "LIBR"

The fluid is “free”, i.e. without internal structures. This is the default option.

### "PORE"

The fluid is “porous”, i.e. it occupies just one part of the meshed volume, the rest being occupied by some fixed internal structures.

### rho

Initial density  $\rho_{\text{ini}}$  of the fluid.

### c

Sound speed  $c$  in the fluid, considered constant.

**pini**

Absolute initial pressure  $p_{\text{ini}}$  in the fluid. By default,  $p_{\text{ini}} = 0$ .

**pref**

Absolute reference pressure  $p_{\text{ref}}$  in the fluid. By default,  $p_{\text{ref}} = p_{\text{ini}}$  (even when  $p_{\text{ini}} = 0$ ).

**pmin**

Absolute minimum pressure  $p_{\text{min}}$  in the fluid. By default,  $p_{\text{min}} = 0$ . Obviously, it must be  $p_{\text{min}} \leq p_{\text{ini}}$ . The minimum density  $\rho_{\text{min}}$  results then from the expression:

$$\rho_{\text{min}} = \rho_{\text{ini}} + \frac{p_{\text{min}} - p_{\text{ini}}}{c^2}$$

**mu**

Dynamic viscosity coefficient  $\mu$  (2D or 3D).

**alpha**

Value of the porosity: ratio of the volume occupied by the fluid with respect to the total volume.

**sur**

Relative wet surface (value 1 by default).

**kp**

Head loss coefficient by unit length, assumed isotropic.

**kpx, kpy, kpz**

Head loss coefficient by unit length in the direction  $Ox$  (respectively  $Oy, Oz$ ).

**beta**

Reduced damping coefficient for high frequencies. It is zero by default, and should always be very small.

**/LECTURE/**

List of the elements concerned.

### Comments:

The parameters **R0** and **C** are compulsory.

### Role of **PREF**:

When the reference pressure is different from the initial one, the fluid is not in equilibrium at the beginning. This is the case e.g. when a membrane is breaking at  $t = 0$ , releasing a compressed fluid. For further detail, see page C.300.

In various problems, studies relate to acoustic effects; since it is supposed that a fluid in equilibrium evolves under the effects of loading (motion of a piston, shock,...), in this case it must be:  $p_{\text{ref}} = p_{\text{ini}}$ .

If **PREF** is omitted, EUROPLEXUS considers that the fluid is in equilibrium and  $p_{\text{ref}} = p_{\text{ini}}$  (even when  $p_{\text{ini}} = 0$ ).

For a given minimum pressure  $p_{\text{min}}$ , the fluid pressure is always greater than or equal to that value, even if the density is decreasing. This is a very simple way to model cavitation. The default value of  $p_{\text{min}}$  is  $p_{\text{min}} = 0$ .

### Viscosity:

In the presence of viscosity, the tensor of stresses in the fluid has the following form:

$$\sigma(i, j) = -P \delta(i, j) + 2\mu \dot{\epsilon}(i, j)$$

with:

$P$  : pressure

$\delta(i, j)$  : Kronecker's symbol

$\dot{\epsilon}(i, j)$  : strain rate (derived from  $\epsilon(i, j)$ )

For water at 20 degrees Celsius:  $\mu = 0.001$  SI units (Kg/(m\*s)).

### Porous fluid:

If the fluid is porous, the parameter **PORO** is mandatory. In this case an equivalent fluid is used by the code for the calculations, which occupies the entire volume of the element. However, the used variables (pressure, velocity, etc.) are those of the **REAL** fluid, so as to obtain directly the physical state of the fluid in the presence of internal structures.

If these internal structures generate a head loss, the parameter **kp** allows to model it in case this loss is isotropic. Otherwise, the parameters **kpx**, **kpy**, **kpz** allow to distinguish between the three directions in the global reference.

The former coefficients are given per unit length. For example, if the head loss is  $\Delta P = 0.25$  bar over a length of  $L = 2$  m, for a fluid of density  $\rho = 1000 \text{ kgm}^{-3}$  with a velocity  $V = 5 \text{ ms}^{-1}$ , the coefficient will be  $K_p = 1$  according to the formula:

$$\Delta P = \frac{1}{2} K_p L \rho V^2$$

The parameter **SMOU** (relative wet surface) is obsolete and may be omitted. It is only kept for compatibility with old input files.

**Correlation between bulk modulus and sound speed:**

This material model can be compared to that of a fluid with constant bulk modulus (e.g. the FLUT material with NUM 9) as follows. For the latter, the absolute pressure is given by:

$$p = p_{\text{ini}} + B\eta$$

where  $B$  is the bulk modulus (assumed constant and usually expressed in Pa) and  $\eta$  is the relative volume variation:

$$\eta = -\epsilon_V = \frac{V - V_{\text{ini}}}{V_{\text{ini}}} = 1 - \frac{\rho_{\text{ini}}}{\rho}$$

From these expressions one obtains:

$$p = p_{\text{ini}} + (\rho - \rho_{\text{ini}}) \frac{B}{\rho}$$

By comparing this with the pressure expression of the FLUI material, one sees that:

$$c = \sqrt{\frac{B}{\rho}}$$

Therefore, strictly speaking the two models are different because in one the sound speed is assumed constant (so that the bulk modulus varies with the density) while in the other the bulk modulus is assumed constant (so that the sound speed varies with the density). However, by assuming that the density varies only slightly from the initial value  $\rho_{\text{ini}}$ , one obtains the following relation between  $c$  and  $B$ :

$$c \approx \sqrt{\frac{B}{\rho_{\text{ini}}}}$$

**Outputs:**

The components of the ECR table are as follows:

ECR(1): absolute pressure

ECR(2): density

**Reference:**

CEA report to appear.

### 7.8.3 TAIT EoS

#### Object:

This option enables a barotropic fluid (liquid-like) behaviour for continuum elements to be input. The Tait Equation of State (EoS) can be used to model liquids and is frequently used to model water in underwater-explosion simulations. For water, classical constants are:  $\gamma = 7.15$  and  $b = 331 MPa$ .

The expression used to compute the absolute pressure  $P$  in the fluid is:

$$P = B \left( \left( \frac{\rho}{\rho_{\text{ref}}} \right)^{\gamma} - 1 \right)$$

where  $P$  is the fluid pressure,  $\rho$  is the current density and  $\rho_{\text{ref}}$  is the reference density.

#### Syntax:

```
"TAIT"      "R0" rho    "PINI" pini <"PREF" pref >  <"PMIN" pmin >
...         "GAMM" gamma  "B" b
...         /LECTURE/
```

**rho**

Reference density  $\rho_{\text{ref}}$  of the fluid.

**pini**

Absolute initial pressure  $p_{\text{ini}}$  in the fluid.

**pref**

Absolute reference pressure  $p_{\text{ref}}$  in the fluid. By default,  $p_{\text{ref}} = p_{\text{ini}}$ .

**pmin**

Absolute minimum pressure  $p_{\text{min}}$  in the fluid. By default,  $p_{\text{min}} = 0$ . Obviously, it must be  $p_{\text{min}} \leq p_{\text{ini}}$ . The minimum density  $\rho_{\text{min}}$  results then from the expression:

$$\rho_{\text{min}} = \rho_{\text{ref}} \left( \frac{p_{\text{min}}}{B} + 1 \right)^{\frac{1}{\gamma}}$$

**gamma**

first coefficient of the TAIT EoS.

**b**

second coefficient of the TAIT EoS.

**/LECTURE/**

List of the elements concerned.

**Comments:**

For the TAIT EoS, the initial density  $\rho_{\text{ini}}$  is computed from the expression:

$$\rho_{\text{ini}} = \rho_{\text{ref}} \left( \frac{p_{\text{ini}}}{B} + 1 \right)^{\frac{1}{\gamma}}$$

with:

$p_{\text{ini}}$  : initial pressure,  $\rho_{\text{ref}}$  : reference density

The expression used for the sound speed is:

$$C = \sqrt{\frac{\gamma(P + B)}{\rho}}$$

**Role of PREF:**

When the reference pressure is different from the initial one, the fluid is not in equilibrium at the beginning. This is the case e.g. when a membrane is breaking at  $t = 0$ , releasing a compressed fluid. For further detail, see page C.300.

In various problems, studies relate to acoustic effects; since it is supposed that a fluid in equilibrium evolves under the effects of loading (motion of a piston, shock,...), in this case it must be:  $p_{\text{ref}} = p_{\text{ini}}$ .

If PREF is omitted, EUROPLEXUS considers that the fluid is in equilibrium and  $p_{\text{ref}} = p_{\text{ini}}$ .

For a given minimum pressure  $p_{\text{min}}$ , the fluid pressure is always greater than or equal to that value, even if the density is decreasing. This is a very simple way to model cavitation. The default value of  $p_{\text{min}}$  is  $p_{\text{min}} = 0$ .

**Outputs:**

The components of the ECR table are as follows:

ECR(1): absolute pressure

ECR(2): current density

ECR(3): sound speed



### 7.8.4 PERFECT GAS

#### Object:

Euler : perfect gas ( $P = \rho(\gamma - 1)E_{\text{internal}}$ );

Lagrange: adiabatic perfect gas ( $P = k\rho^\gamma$ ).

In a 1-D case, the frictions against the walls can be taken into account, since the dissipated energy will heat up the gas (modification of the internal energy). To this end the user has to add a PARO material, which must be associated with GAZP by means of the MULT material (see pages C.370 and C.380).

#### Syntax:

```
"GAZP" ![ "R0" rho "GAMMA" gamma "PINI" pini <"VISC" mu > ...
          ... < "CV" cv > < "PREF" pref > ]! /LECTURE/
```

rho

Initial density.

gamma

Ratio  $c_P/c_V$  (supposed constant).

pini

Initial pressure.

mu

Dynamic viscosity of the gas (for 2-D and 3-D).

cv

Specific heat at constant volume  $c_V$  (used to compute the temperature).

pref

Reference pressure. Note that, by default, it is assumed **pref** = **pini**.

/LECTURE/

List of the elements concerned.

#### Comments:

The reference pressure **pref** enables the initial state to be defined. If **pref** = **pini**, the gas is in equilibrium just before the computation starts; it will be perturbed by an external action, by the motion of a piston, for instance. If **pref** = 0, the problem consist in a computation with initial stresses determined by **pini**. This is the case when a membrane which was separating two gases at different states disappears at the initial instant.

If **cv** is omitted, the temperature is not computed. If it is present, the temperature is expressed in degrees Celsius.

**Outputs:**

The different components of the ECR table are as follows:

ECR(1): pressure

ECR(2): density

ECR(3): velocity of sound

ECR(4): maximum pressure ever experienced

ECR(5): minimum pressure ever experienced

ECR(6): dynamic pressure: ( $P_{\text{dyn}} = \frac{1}{2}\rho v^2$ )

ECR(7): temperature (if  $c_V$  is not zero) in degrees Celsius

ECR(8): total specific energy ( $E = h + \frac{1}{2}v^2$ )

### 7.8.5 STIFFENED GAS

#### Object:

This Equation of State can be used for both liquids and gases. It takes the following form:

$$P = (\gamma - 1)\rho(e - q) - \gamma P_{inf}$$

Where  $e$  is the internal energy per unit mass,  $\rho$  the density.  $\gamma$  is an empirical constant for liquids.  $P_{inf}$  is a constant representing the molecular attraction between molecules (liquid) and  $q$  is an additional constant. This expression is identical to the ideal gas EoS when  $P_{inf}$  and  $q$  is zero.

#### Syntax:

```
"STIF" "RO" rho "PINI" pini < "PMIN" pmin > < "PREF" pref >
... "GAMMA" gamma "PI" pinf "Q" q
... /LECTURE/
```

rho

Initial density.

pini

Initial initial pressure.

pref

Reference pressure. Note that, by default, it is assumed `pref = pini`.

pmin

Absolute minimum pressure `pmin` in the fluid. Note that, by default, it is assumed `pmin = 0`.

gamma

Ratio  $c_P/c_V$  (supposed constant) for gases and an empirical constant for liquid.

pinf

Constant parameter for liquid to take into account molecular attraction between molecules.

q

Internal energy of the fluid at a given reference state (most time one take `q = 0`).

/LECTURE/

List of the elements concerned.

#### Comments:

The reference pressure `pref` enables the initial state to be defined. If `pref = pini`, the gas is in equilibrium just before the computation starts; it will be perturbed by an external action, by the motion of a piston, for instance. If `pref = 0`, the problem consist in a computation with initial stresses determined by `pini`. This is the case when a membrane which was seperating two gases at different states disappears at the initial instant.

The expression used for the sound speed is:

$$C = \sqrt{\frac{\gamma(P + P_{inf})}{\rho}}$$

**Outputs:**

The different components of the ECR table are as follows:

ECR(1): pressure

ECR(2): density

ECR(3): sound speed

### 7.8.6 SOURCE

#### Object :

This instruction enables a time dependent pressure to be imposed, inside an element.

For fluids modelled in ALE, this material allows to create a source of mass flow, if it is used in conjunction with an imposed velocity directive. However, this source is limited to the case of a liquid (case "FLUI"), of a perfect gas ("GAZP"), of a two-phase mixture of water ("EAU") or of a liquid-gas mixture ("ADCR").

#### Syntax:

```
"SOUR"    $ "FLUI" ... ; "GAZP" ... ; "EAU" ... ; "ADCR" ... $
          ... < "FONC" nufo < "FACT" coef > > /LECTURE/
```

FLUI, GAZP, EAU, ADCR

This keyword indicates the source material data which are strictly identical to those of the material with the same name.

For "FLUI" see 7.8.2 page C.305, for "GAZP" see 7.8.4 page C.310, for "EAU" see 7.8.9 page C.350, for "ADCR" see 7.8.19 page C.430,

**nufo**

Number of the 'FONCTION' allowing to define the pressure as a function of time

**coef**

Multiplying factor for the pressures given by the preceding function. By default coef=1.

**LECTURE**

List of the concerned elements.

#### Comments:

The "FONCTION" directive is described in 9.1 (page E.15).

The initial pressure, must correspond to the origin of the curve.

The element deforms only under the action of forces due to the imposed pressure (no stiffness). The temperature is assumed constant. In the case of water, for example, if the pressure increases one can pass from a liquid phase to a vapor phase, but always at the same temperature.

#### Outputs:

The different components of the ECR table are the same as the components for the material with the same name.

### 7.8.7 SODIUM-WATER REACTION ("NAH2")

#### Object:

Explosion caused by water injection into liquid sodium.

In 2D or 3D, one element or more (contiguous) elements may be affected by the chemical reaction. In 1D, just one "TUBE" or "TUYA" element must be used with this material.

In 1D or 3D there are two options for the water mass flow rate:

- 1) imposed curve as a function of time (keyword "DEBIT")
- 2) calculation of a water-filled pipeline meshed by elements of type TUBE or TUYA and coupling with the sodium mesh (keyword "DCOUP").

#### Syntax:

```
"NAH2"  "RO" rho  "C" c  "PINI" pinit  "PV" pv  "FACT" facteur
... < "PREF" pref > < "PMIN" pmin > < "CMIN" cmin > < "CH2" ch2 >
...$[ "DEBIT"  npt*(temps , debit) ;
      "DCOUP"  1      tini    qini ]$/LECTURE/
```

rho

Initial density (pure sodium).

c

Sound speed in the sodium.

pinit

Initial pressure.

pref

Reference pressure (see page C.300).

pv

Value of the product P\*V for the unit mass of hydrogen at the initial temperature.

facteur

Number of gas moles formed starting from one mole of injected water.

ch2

Initial mass fraction for the hydrogen. By default, ch2 = 0 and as a maximum ch2 = 1.

**cmin**

Minimum mass fraction of the hydrogen in the pure sodium (1.E-8 by default).

**pmin**

Minimum pressure (zero by default).

**"DEBIT"**

Keyword that announces the introduction of the curve of total mass flow rate of water injected in the sodium for the whole set of elements concerned.

**npt**

Number of points defining the mass flow rate curve for the water.

**(temps, débit)**

Coordinates of the points (in axisymmetric, divide the mass flow rate by  $2\pi$ , as the calculation refers to one radian).

**"DCOUP"**

The water mass flow rate at the outlet of a pipe is computed (1D) by directive "IMPE" "NAH2" (page C.610).

**tini**

Initial time.

**qini**

Initial mass flow rate.

**/LECTURE/**

List of the elements affected by the injection.

### **Comments:**

For the dimensioning, an injection curve requires a space similar to that of a traction curve. The user will therefore have to specify "TRAC" n1 n2, with n1 the maximum number of curves to be entered (traction and injection), and n2 the maximum number of points.

In the elements affected by the reaction, it is assumed that the reaction is instantaneous, that the mixture of reacting components is always homogeneous and that the reaction is isothermal.

The "facteur" parameter allows to account for the vaporised 'soude'. If there is none, then facteur = 0.5.

If the volume fraction of hydrogen (printed as ECR(4)) is above 1, the program treats the mixture like pure hydrogen.

In a Lagrangian calculation (default option) the "NAH2" material is attached only to the elements where the reaction takes place. The "FLUI" material is used for the other elements.

In an Eulerian calculation (option "EULER" page A.30), the material "NAH2" is affected to ALL fluid elements. The mass flow rate curve for the water is only affected to the elements where the reaction occurs, for the others one must write:

"DEBIT" 0 /LECTURE/

### Outputs:

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : density

ECR(3) : hydrogen concentration

ECR(4) : hydrogen volume occupation ratio

ECR(5) : total water mass flow rate for the set of elements

ECR(6) : water mass injected in each element

ECR(7) : sound speed in the Na+H2O mixture

ECR(8) : hydrogen mass per unit volume (in Eulerian)

ECR(10): phase indicator (1= saturated in H2; else 0)



### 7.8.8 SODIUM-WATER REACTION ("RSEA")

#### Object:

Explosion caused by water injection into liquid sodium.

It is possible to repeat this material as many times as needed, provided the injections occur in different elements.

There are two possibilities concerning the water mass flow rate:

- 1) a time function is invoked (keyword "NUFO")
- 2) compute a water-filled pipeline meshed with TUBE or TUYA elements and coupled with the sodium mesh (keyword DCOU).

#### Syntax:

```
"RSEA"  "PTOT" ptot  "PNA" pna      "RONA" rhona  "CSNA" csona
        "PBU" pbu   "ROBU" rhobu  "NBU" nbu    "GBU" gbu
        < "PARG" parg  "ROAR" rhoar  "GAR" gar >
        < "PSAT" psat  "ROSA" rhovap >
        < "XBU" xbu > < "XAR" xar > < "PREF" pref >
        < "CMIN" cmin > < "BETA" beta >
        < "VINA" vina > < "VIBU" vibu > < "VIAR" viar >
        < $[ "NUFO" nufo  "COEF" coef ; "DCOU" 1 ]$ >
        < "FACT" facteur >
        /LECTURE/
```

#### ptot

Total pressure of the mixture.

#### pna

Zero pressure of sodium, defining the equation of state of the liquid, by means of the following rhona and csona parameters.

#### rhona

Zero density of sodium.

#### csona

Sound speed in the sodium.

#### pbu

Zero pressure of the hydrogen, defining the equation of state of the gas, by means of the following rhobu and nbu parameters.

rhobu

Zero density of the hydrogen.

nbu

Polytropic coefficient of the transformation followed by the hydrogen. For an isothermal:  $n_{bu} = 1$ .

gbu

Ratio  $C_p/C_v$  for the hydrogen.

parg

Zero pressure of the argon, defining the equation of state of this gas, by means of the following rhoar and gar parameters.

rhoar

Zero density for the argon.

gar

Ratio  $C_p/C_v$  for the argon.

psat

Saturation pressure of the sodium vapor. A priori this is very low, and allows to treat correctly the possible cavitation phenomena.

rhovap

Density of the sodium vapor.

xbu

Initial mass fraction of the hydrogen. It allows to account for the presence of gas in the considered domain.

xar

Initial mass fraction of the argon. This refers to the subdomains that are filled with argon initially, such as the cover gas region or the zones located behind membranes.

pref

Reference pressure (see page C.300).

cmin

Maximum mass fraction of gas in the sodium in order to consider it as pure (1.E-8 by default).

vina

Dynamic viscosity of sodium.

vibu

Dynamic viscosity of the gas in the bubble.

**viar**

Dynamic viscosity of argon.

**beta**

Reduced damping coefficient for high frequencies. It is zero by default, and should always be very small (  $< 0.05$  ).

**nufo**

Number of the adimensional function defined via the "FONCTION" directive of EUROPLEXUS. This function allows to describe the variation of injected water mass flow rate in time.

**coef**

Multiplicative factor por the precediing function. This allows to account for the units and the number of ruptured pipes.

**DCOU**

This keyword specifies that a coupled water flow calculation is desired. It is always followed by an integer value.

**facteur**

Number of gas moles formed by one mole of injected water.

**/LECTURE/**

List of the concerned elements.

### Comments:

In the elements affected by the reaction, it is assumed that the reaction is instantaneous, and that the mixture of reacting materials is always homogeneous.

The "facteur" parameter allows to account for the vapor 'soude'. If there is none, then  $\text{facteur} = 0.5$ .

The mesh will be subdivided in as many zones as necessary, and for each of these an RSEA material will be defined, by possibly varying the initial concentrations and the total pressures, but the other parameters must be identical, so as to have exactly the same constitutive laws for the different components of each zone. Then, starting from the given concentration and the total pressure  $p_{\text{tot}}$ , EUROPLEXUS will compute de density of the mixture. EUROPLEXUS will also recompute the gas concentrations in order to account for the sodium vapor, if  $p_{\text{sat}}$  is not zero.

The elements where the reaction occurs will be distinguished by one of two possible options: imposed injection or injection coupled with the calculation of water mass flow rate (DCOU).

If  $p_{\text{sat}}$  and  $\text{rhovap}$  are absent, or if one only of these values is given, it is the default value which is used:  $p_{\text{sat}}$  is taken equal to one thousandth of  $p_{\text{tot}}$ . The value of  $\text{rhovap}$  is then proportional to  $p_{\text{sat}}$ , and corresponds to a monoatomic vapor at a temperature close to  $300^{\circ}\text{C}$ .

**Outputs:**

The components of the ECR table are as follows:

- ECR(1) : absolute pressure,
- ECR(2) : density of the two-phase mixture,
- ECR(3) : sound speed in the mixture,
- ECR(4) : void fraction,
- ECR(5) : argon mass fraction,
- ECR(6) : hydrogen mass fraction,
- ECR(13) : water mass flow rate (dm/dt)
- ECR(14) : mass of water injected since the beginning.

### 7.8.9 WATER

#### Object :

This directive allows to treat water and its vapour as an homogeneous mixture. It is also possible to treat a water vapor explosion when energy is released within liquid water.

#### Syntax :

```
"EAU"    $[ "EQUI"                                ;
           "META"  "NBUL" nbul  "ALFN" alfn ]$    ...

...  "PINI"  pini  |[ "TINI" tini ; "TITR" x ]| ...
...  < "PREF" pref > < "BETA"  beta > ...
...  < "VISL" mul   "VISV"  muv   >   ...
...  < "BETD" betd > ...
```

For a direct injection:

```
$ < "ENMA" enma    "FONC" numf    ...
... < "XCOR" xcor > < "MODE" mode ; "COEF" coef > > $
... < "DPROP" dpropag  "ORIGINE"  /LECTURE/ > $
```

For an injection of corium particles:

```
...$ < "DIAM" diam    "CECH"  hh    ...
...   "TCOR" tcor    "VCOR" vcor    > $
```

One ends this directive by:

```
.../LECTURE/
```

#### EQUI

The mixture will be in equilibrium (same pressure and same temperature for the liquid and vapour phases).

#### META

The mixture will be metastable (same pressure but different temperatures for the liquid and vapour phases).

#### nbul

Number of vapour bubbles by unit volume (of the order of 1.E9).

#### alfn

Minimal void fraction for the nucleation of vapor bubbles (of the order of 1.E-4).

#### pini

Initial pressure of the mixture.

**tini**

Initial temperature of the equilibrium mixture (in degrees Celsius).

**x**

Initial mass title of the vapor, between 0 and 1. (eau=0, vapeur=1).

**pref**

Reference pressure (for its meaning see page C.300). By default, it is equal to the initial pressure. All reference pressures must be equal.

**beta**

Reduced damping coefficient for high frequencies. It is zero by default, and should always be very small (  $< 0.05$  ).

**mul**

Dynamic viscosity of the water. Recall: below 1 bar, at  $25^{\circ}C$ ,  $mul = 9.E-4$  Poiseuille or Pascal \* second. This value drops rapidly as the temperature increases. By default  $mul = 0$ .

**muv**

Dynamic viscosity of the water vapor. Recall: below 1 bar, at  $100^{\circ}C$ ,  $muv = 1.3E-5$  Poiseuille. This value increases with the temperature. By default,  $muv = 0$ .

**betd**

If equal to 1, the beta coefficient multiply the trace of the stress Matrix. If not equal to 1, the beta coefficient multiplied the the first value of the stress Matrix. By default,  $betd = 0$ .

### **For a direct injection of energy in the water:**

**enma**

Specific power injected in the water. Multiplicative coefficient (dimensional) of the following function.

**numf**

Number of the non-dimensional function defined by the "FONCTION" directive of EUROPLEXUS. This function allows to vary the injected power in time.

**xcor**

Ratio between the corium mass and the water mass contained within the element. The specific power (enma) applies only to the present corium. By default,  $xcor = 1$ .

**mode**

Choice of the injection mode: (0, 1, 2 or 3). The meaning is explained in the comments below. By default,  $mode = 0$ .

coef

Ratio between the limit density and the initial density. This allows to limit the injection. By default, coef = 0. Is only relevant for mode = 0 or 1.

dpropag

Propagation velocity of the energy injection signal for the elements of the considered domain. By default, the injection occurs simultaneously in all such elements.

ORIGINE

This keyword announces the reading of the element in which the injection is initiated. The injection then propagates with the speed dpropag in all directions.

### For an injection by means of corium particles:

diam

Diameter of the particles.

hh

Heat exchange coefficient between a corium particle and the liquid water.

tcor

Initial temperature of the corium particles.

vcor

Volume fraction occupied by the corium.

LECTURE

List of the concerned elements.

### Comments :

Do not forget to create the tables of physical properties of the water by means of directive "TEAU" or "TH2O" (page C.74).

The "TH2O" table is only working with equilibrated water.

If the mixture is single-phase, one should give the pressure and the temperature, but the title is irrelevant. If the mixture is two-phase, one should give the pressure and the title: EUROPLEXUS then computes the temperature.

The damping coefficient beta allows to damp out high-frequency oscillations caused by the discretisation. By default beta is zero. However, it is advised to use beta between 0.1% et 5%. One should be aware of the inevitable attenuations of lower frequencies, especially if the mesh is coarse. In fact, the eigenfrequencies of the structure are not very different from the frequencies associated with the finite elements.

If the viscosity has to be considered, the two parameters mul and muv must be related and given together.

**Energy injection:**

The "MODE" parameter allows to account for the evolution of fluid close to the injection zone, in a less brutal fashion compared with directive "COEF". Its meaning is as follows:

- mode = 0 : the injected energy is independent from the fluid mass and nature,
- mode = 1 : the injected energy is proportional to the mass of water, but independent from its nature of liquid or vapor,
- mode = 2 : the injected energy is proportional to the mass of liquid water,
- mode = 3 : the injected energy is proportional to the volume of liquid water.

Modes 2 and 3 should not be used if the pressure exceeds the critical value ( $P_{crit} = 221$  bar).

The "COEF" directive allows to limit the quantity of injected energy: if the density during the computation becomes lower than the limit density, then the injection is stopped. This directive is brutal and not advisable. It is preferable to use the "MODE" directive.

One may obtain the energy quantity released in a certain region of the mesh by means of keyword "WINJ" in directive "REGION" (page G.100).

By assigning a propagation velocity associated with an origin element allows to avoid a brutal and instantaneous injection over an extensive domain, which is unrealistic: this option is recommended in case of a steam explosion calculation.

**Initial pressure and temperature:**

Initial pressure and temperature can be taken into account with INIT MEDL VCVI ECRO option. The input MED file must contain fluid velocity and internal variables fields. The pressure must be the first component and the temperature must be the fifth of the internal variables field.

**Outputs:**

The components of the ECR table are as follows:

- ECR(1) : absolute pressure
- ECR(2) : density of the mixture
- ECR(3) : sound speed
- ECR(4) : mass title of the vapor (vapor mass/total mass)
- ECR(5) : temperature of the mixture for equilibrated water, liquid temperature for meta-stable
- ECR(6) : enthalpy of the mixture
- ECR(7) : temperature of the mixture for equilibrated water, vapor temperature for meta-stable
- ECR(9) : void ratio or volume ratio of the vapor (vapor volume/total volume)



**For meta-stable water:**

ECR(21) : vapor relative density (vapor mass/total volume)

ECR(23) : index : 0 = equilibrium ; 1 = meta-stable

ECR(24) : specific enthalpy of the liquid water

**In case of direct energy injection:**

ECR(8) : power injected in the element

ECR(14) : corium mass within the element

**In case of energy injection by particles:**

ECR(19) : initial volume of corium within the element

ECR(20) : mean temperature of a corium particle

**For a "BREC" element:**

ECR(25) : Pipeline rupture area

ECR(26) : Mass flow

ECR(27) : Total ejected mass

### 7.8.10 HELIUM

**Object :**

This directive allows to treat helium and its liquid as an homogeneous mixture.

**Syntax :**

```
"HELI"    "PINI"  pini    |[ "TINI" tini ; "TITR" x ]| ...  
... < "PREF" pref > < "BETA" beta >      ...  
... < "VISL" mul   "VISV" muv   >      ...  
... /LECTURE/
```

**pini**

Initial pressure of the mixture.

**tini**

Initial temperature of the equilibrium mixture (in degrees Kelvin).

**x**

Initial mass title of the vapour, between 0 and 1. (liquid=0, vapour=1).

**pref**

Reference pressure (for its meaning see page C.300). By default, it is equal to the initial pressure. All reference pressures must be equal.

**beta**

Reduced damping coefficient for high frequencies. It is zero by default, and should always be very small ( < 0.05 ).

**mul**

Dynamic viscosity of the liquid. By default mul = 0.

**muv**

Dynamic viscosity of the vapour. By default, muv = 0.

**Comments :**

Do not forget to create the tables of physical properties of helium by means of directive "THEL" (page C.75).

If the mixture is single-phase, one should give the pressure and the temperature, but the title is irrelevant. If the mixture is two-phase, one should give the pressure and the title: EUROPLEXUS then computes the temperature.

The damping coefficient  $\beta$  allows to damp out high-frequency oscillations caused by the discretisation. By default  $\beta$  is zero. However, it is advised to use  $\beta$  between 0.1% et 5%. One should be aware of the inevitable attenuations of lower frequencies, especially if the mesh is coarse. In fact, the eigenfrequencies of the structure are not very different from the frequencies associated with the finite elements.

If the viscosity has to be considered, the two parameters  $\mu_l$  and  $\mu_v$  must be related and given together.

**Outputs:**

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : density of the mixture

ECR(3) : sound speed

ECR(4) : mass title of the vapor (vapor mass/total mass)

ECR(5) : temperature of the mixture

ECR(6) : specific enthalpy of the mixture

### 7.8.11 1D WALL

#### Object:

This directive allows, in association with the **MULT** material, to account for the effects of pipe walls and cavity walls, for elements of type **TUBE**, **TUYA** or **CAVI**.

#### Syntax:

```
"PARO"  $[  "RUGO"  rug      "VISC" mu                ;
            "RUGO"  rug      "VISL" mul      "VISV" muv    ;
            "TPAR"  teta     "COND" cof    < "SURF" su >    ;
            "PSIL"  kl
            "FONC"  numf < "SURF" su    >                ]$

< "COEF" nbr >  /LECTURE/
```

#### rug

Absolute rugosity of the pipe (warning, this parameter has the dimension of a length).

#### mu

Dynamic viscosity (single-phase fluid).

#### mul

Liquid dynamic viscosity (two-phase fluid).

#### muv

Vapor dynamic viscosity (two-phase fluid).

#### teta

Wall temperature.

#### cof

Conductance.

#### su

Heat exchange surface.

#### kl

Head loss coefficient per unit length (1/m).

#### nbr

Multiplicative coefficient in the case of an assembly of identical pipes (nbr = 1 by default).

**numf**

Number of the **FUNCTION** allowing to define the conductance as a function of the flow velocity of the fluid.

**LECTURE**

List of the elements concerned.

**Comments:**

The **MULT** material is used (page C.380) to associate the wall defined by **PARO** with the corresponding internal fluid.

If the internal fluid is of type **NAH2**, **EAU** or **RSEA**, it is mandatory to specify the viscosities of both phases, **mul** for the liquid and **muv** for the vapor. In all other cases, the fluid is supposed to be single-phase, and only one viscosity **mu** will be required.

In the case of heat exchange with the wall, the keyword **SURF** is mandatory for the **CAVI** elements. For the elements of type **TUBE** and **TUYA** it may be omitted, and in this case **EUROPLEXUS** will compute the exchange surface starting from the geometrical characteristics of the elements.

The **k1** coefficient (keyword **PSIL**) allows to compute the head loss in the following way:

$$DP = k1 * long * 0.5 * rho * V ** 2$$

with:

**DP** : head loss,

**long** : length of the pipe,

**rho** : fluid density,

**V** : mean velocity,

Hence:  $k1 = \psi / Dh$ , where **Dh** is the hydraulic diameter.

**Outputs:**

The parameters associated with the wall material not included in **ECR** are placed after the **ECR** for the fluid (see the fluid documentation).

The components of the **ECR** table are as follows:

**ECR(1)** : wall temperature

**ECR(2)** : conductance

**ECR(4)** : Reynolds number

**ECR(5)** :  $\psi$  (of the formula  $K = \psi * L / Dh$ )

### 7.8.12 PIPE BREAK PARAMETERS

**Object:**

This directive allows, in association with the **MULT** material and elements of type **BREC** to enter parameters for the computation of the fluid effects after the pipe break (outlet pressure, critical mass flow rate...).

**Syntax:**

```
"BREC" $[  
    "DCRI" idcri ;  
    "PIMP" ipimp < "CONT" cont "EPAI" epai >  
]$
```

**idcri**

Mode for the critical mass flow rate computation (see comment below).

**ipimp**

Mode for the imposed pressure model (see comment below).

**cont**

Jet contraction ratio at critical section (default value: 0.84) (see comment below).

**epai**

Pipe thickness of the broken pipe (see comment below)

**Comments:**

The **MULT** material is used (page C.380) to associate the pipe break parameters defined by **BREC** with the corresponding internal fluid. Only the water material **EAU** is currently allowed for the fluid.

The keyword **DCRI** is used to impose a critical mass flow rate after the pipe break. Critical mass flow rate is computed using additional **CL** elements placed on both ends of the **BREC** element. Two modes are allowed:

- Mode 0 : the break area is the one defined by additional **CL** elements. It is independent of the pipe geometry.
- Mode 1 : break area is computed inside the **BREC** element, according to the pipe geometry, taking into account the 'real' position of both parts of the broken pipe (as initially defined in the mesh).

The keyword **PIMP** is used to impose a pressure drop at the break with the imposed pressure being the pressure at saturated conditions, modified in order to take into account the break area [933]. This option is supposed to give a better representation of pressure wave generation at the break, focusing on the very first instant after break opening. Two modes are allowed:

- Mode 0 : no mass flow rate limitation is imposed at the break. In this case, the mass flow rate doesn't converge towards the critical mass flow rate.
- Mode 1 : the mass flow rate at the break is limited by the critical mass flow rate, calculated using Moody's model.

Two optional keywords **CONT** and **EPAI** can be used in conjunction with the keyword **PIMP**:

- **cont**: ratio between the critical area and break area (assumption of a jet contraction after the break)
- **epai**: pipe thickness at the break. This option is used to compute the jet impact generated by the flow onto the opposite broken pipe. If the keyword **CONT** is not given, the jet impact force is not calculated. The jet impact is evaluated with a simplified approach considering a liquid jet.

For keyword **DCRI**, additional **CL** elements are needed for the mass flow rate to be calculated. If no **CL** elements are used, no critical mass flow rate is computed and no flow limitation is imposed. For keyword **PIMP**, **BREC** element can be used with or without **CL** elements, with no impact on the calculation. If used, before the pipe break, the **CL** elements are automatically deactivated (see page D.590 for the definition of the break time).

### 7.8.13 MULTIPLE MATERIALS

**Object:**

This directive allows to assign several materials to the same element. For example, it is the case of a pipeline element, where one has to specify both the material for the internal fluid and the material of the wall plus, if necessary, a material describing the friction.

**Syntax:**

```
"MULT"  n1  n2  < n3 > /LECTURE/
```

**n1**

Number assigned to the first material.

**n2**

Number assigned to the second material.

**n3**

Number assigned to the third material.

**LECTURE**

List of the elements concerned.

**Comments:**

The materials concerned must be defined previously, and are referenced by their law index (see LOI, page C.100). This is either the number explicitly given by the LOI keyword, or the material definition order in the input file.

In the case of "TUBE" elements, n1 will be the index of the fluid material and n2 the index of the friction material.

In the case of "TUYA" elements, the fluid must be referenced first, the wall second, and the friction third, when present. For example, if one has defined the materials in the following order: the wall material first (1), then the fluid material (2), and finally the friction material (3), the "MULT" directive must be coded as follows:

```
"MULT"  2  1  3  /LECTURE/
```

**Outputs:**



The stresses and the hardening parameters will be those of the component materials. For example, for an element of pipeline, the printed stresses will be those of the associated beam (no printout of the stresses for the internal fluid), and the ECR(i) will give first the quantities related to the fluid material, then those related to the wall material.

### 7.8.14 LIQUID

**Object:**

This option enables the processing of an incompressible or quasi-incompressible fluid. The implicit algorithm of "LIAISONS" is used.

**Syntax:**

```
"LIQU"  "RO"  rho < "C"  c > < "PINI" pini > ...  
... < "PREF" pref > < "VISC" visc > < "RUGO" rugo >  
/LECTURE/
```

rho

Density.

c

Velocity of sound (only for a compressible fluid).

pini

Initial pressure.

pref

Reference pressure (see page C.300).

visc

Twice the dynamic viscosity ( $2\mu$ ).

rugos

Absolute rugosity.

LECTURE

List of the elements concerned.

**Comments:**

This material only makes sense if the option "NAVIER" has been required for the definition of the problem.

If the material is incompressible, c is useless. On the contrary, c is necessary if the fluid is compressible, even at a low level. In this case c is read and then the value of  $(1/c^2)$  is stored in the material property.

**Warning:**

It is essential to invert the connection matrix at each step. Do not forget to add the option "FREQ" 1 when using the instruction "LIAISON" (see page D.20).

**Outputs:**

The components of the ECR table are as follows:

ECR(1): absolute pressure of the element due to the viscous terms

ECR(2): density

ECR(3): additive term to the diagonal of  $B_L$ , the connections matrix.

ECR(4): additive term to the right-hand side of the connections system

ECR(5): multiplicative term of the pressure

ECR(6): friction coefficient (see M1FROT)

ECR(7): Reynolds number

ECR(8:10): unused

### 7.8.15 TUBE BUNDLES

#### Object:

Replaces a heterogeneous medium composed by a bundle of tubes submerged in a fluid, by an equivalent homogeneous isotropic medium in the acoustic sense. The densities and sound speeds will be different along the three directions in space.

In the case of helicoidal coaxial bundles (2D axisymmetric or 3D), the axis of the bundle must be along the Oz direction, the helices have all the same axial step and are regularly spaced.

In the case of a straight bundle, (2D or 3D), the bundle axis is Oz, and a side of the base 'motif' must be parallel to Ox or Oy.

There are two options for the definition of the three densities and of the three sound speeds:

- a) The values are computed by EUROPLEXUS as a function of the geometrical data (plane waves propagation).
- b) The values have another origin, and are prescribed.

The possible combinations between options and geometries of the bundle are given in the following table:

Geometry	Value of "TYPE"		
Option	2D	3D	
	DPLA	AXIS	TRID
Triangular step	yes	no	yes
Rectangular step	yes	yes	yes
Imposed anisometry:			
Frame Ox,Oy,Oz	no	no	yes
Frame Or,Ot,Oz	no	yes	yes

\* These values are automatically affected to "TYPE" by EUROPLEXUS.

#### Syntax:

```
"FLFA" "RO" rho "C" c < "PINI" pini > < "PREF" pref >
< "PMIN" pmin > ...

$[ "DIAM" d $[ "PRAD" pr "PAXI" pa ]$ < "VISC" mu > < "COEF" coef>
$[ "PTRI" pt "BASE" ba ]$ ;
```

```

"ROX" rox "ROY" roy "ROZ" roz "CX" cx ...
... "CY" cy "CZ" cz "TYPE" type "TAUX" taux ;

"ROR" ror "ROT" rot "ROZ" roz "CR" cr ...
... "CT" ct "CZ" cz "TYPE" type "TAUX" taux ]$

... /LECTURE/

```

rho

Density.

c

Sound speed.

pini

Initial pressure.

pref

Reference pressure (see meaning on page C3.300).

pmin

Minimum pressure (see meaning on page C3.305).

mu

Fluid dynamic viscosity.

coef

Multiplicative factor for the friction (= 1 by default).

d

External diameter of the tubes.

pr

Radial step of the tubes.

pa

Axial step of the tubes.

pt

Equilateral triangular step.

ba

Direction of the triangle base (ba=1 or ba=2).

rox, roy, roz

Densities along directions Ox, Oy and Oz.

**cx, cy, cz**

Sound speeds along directions Ox, Oy and Oz.

**ror, rot, roz**

Densities along directions Or, Ot and Oz.

**cr, ct, cz**

Sound speeds along directions Or, Ot and Oz.

**type**

Type of bundle: 1 = helicoidal, 2 = straight.

**taux**

Volume fraction of the fluid ( $0 < \text{taux} < 1$ ).

**LECTURE**

List of the concerned elements.

### Comments:

The calculation may be done in Lagrangian or in Eulerian.

To conserv the fluid mass, an apparent fluid mass is used (printed in ECR(2)), corresponding to that of a fictitious liquid that would occupy the whole volume.

The modelisation chosen for the bundle implies anisotropy effects on inertia and compressibility. For each principal direction  $i$  of the bundle, the two parameters  $\rho_{oi}$  and  $\rho_{oi} \cdot c_i^2$  must be defined. Hence the nodal masses are quite different from those computed from the apparent density.

However, for an Eulerian calculation, the fluxes involve the mass effectively transferred from an element to the other, i.e. the code uses the density of the free fluid and the total cross section of the passage. Consequently, the computed (and printed) velocity is that of a fictitious free fluid placed at the bundle entry (entry speed). In order to estimate the mean velocity of the fluid within the tubes, it is necessary to multiply the computed velocity by the ratio between the cross-sections.

If one has to impose an absorbing boundary condition at the bundle border using elements CL2D, CL3D ou CL3T, he must care that the product  $\rho \cdot c$  be the one corresponding to the considered direction; since the CLxD take as  $\rho$  the value of the neighbouring element, i.e. the apparent density of the bundle, the sound of speed must be accordingly corrected within directive "IMPE ABSO".

The presence of keyword "VISC" followed by the value of the fluid viscosity triggers the calculation of the head losses in the bundle. The formulation given by I.E.Idel'cik is adopted

(Mémento des pertes de charge, Eyrolles, Paris, 1978) for the two principal directions of the bundle in the plane orthogonal to the tubes. The Blasius formula is used in the direction parallel to the tubes (the Reynolds number is then computed by means of the hydraulic diameter).

**Outputs:**

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : apparent density

Then if VISC is present:

ECR(3) : half-coefficient of friction along direction Ox or Or

ECR(4) : half-coefficient of friction along direction Oy or Ot

ECR(5) : half-coefficient of friction along direction Oz

### 7.8.16 HOMOGENEISATION OF TUBE BUNDLES

#### Object:

Replaces a heterogeneous medium composed by a tube bundle surrounded by fluid, by a homogenised medium (asymptotic homogenisation method).

#### Syntax :

```
"MHOM"  "RO"  rho  "C"  c  "EPSILON"  epsilon  "YSTAR"  ystar ...
...      "COEF"  coef  "MTUBE"  mtube  "KTUBE"  ktube...
...      "NBTUBE"  nbtube*( "BXX"  bxx  "BXY"  bxy  "BYX"  byx  "BYX"  byx ...
...      "CXX"  cxx  "CXY"  cxy  "CYX"  cyx)  /LECTURE/
```

**rho**

Fluid density.

**c**

Sound celerity in the fluid.

**epsilon**

Size of the elementary cell.

**ystar**

Fluid surface in an increased elementary cell.

**coef**

Ratio  $y/y_{\text{star}}$  ( $y$ : total surface of the increased elementary cell).

**mtube**

Mass of a tube.

**ktube**

Stiffness of a tube.

**nbtube**

Number of tubes per elementary cell.

**bxx**

Value of  $\beta_{xx}$  obtained by a homogeneisation pre-treatment.

**bxy**



Value of  $\beta$ -xy obtained by a homogeneisation pre-treatment.

byy

Value of  $\beta$ -yy obtained by a homogeneisation pre-treatment.

cxx

Value of  $\eta$ -xx obtained by a homogeneisation pre-treatment.

cxy

Value of  $\eta$ -xy obtained by a homogeneisation pre-treatment.

cyy

Value of  $\eta$ -yy obtained by a homogeneisation pre-treatment.

LECTURE

List of the elements concerned.

## 7.8.17 PUFF

**Object:**

Equation of state of type PUFF, allowing to treat very fast mechanical phenomena for which the material behaviour is hydrodynamic.

**Syntax:**

```
"PUFF"  "RO"  rho  "MK"  mk                      ...
... < "PINI"  pini  "PREF" pref >                ...
... "GAMMA"  gamma  "D"  d  "S"  s                ...
... "GAMZ"  gamz  "CV"  cv                      ...
... $[ "ES"  es                      "SIMPLE"  ;
      "GAM1" gam1  "N"  n  "EV"  ev  "COMPLEXE" ]$ ...
...                                           /LECTURE/
```

rho

Initial density.

mk

Compressibility modulus.

pini

Value of the initial pressure.

pref

Value of the reference pressure.

gamma

Constant of perfect gases.

d

Parameter of the PUFF material.

s

Parameter of the PUFF material.

gamz

Gruneisen coefficient for the initial density.

cv

Specific heat at constant volume.

gam1

Parameter of the PUFF material (equals zero for SIMPLEs materials).

n

Parameter of the PUFF material. Example:  $n=0.5$  for materials "SIMPLE";  $n=1.67$  for ceramic oxydes.

es

Sublimation enthalpy.

ev

energy at the beginning of vaporisation.

SIMPLE

SIMPLE material (in this case:  $\text{gam1} = 0$ ,  $n = 0.5$ ).

COMPLEXE

PUFF material of type COMPLEXE.

/LECTURE/

List of the concerned elements.

### Comments:

The energy will be under form of a field of initial temperatures (see directive INITIAL, page E.50).

Since the phenomenon is very rapid, it is supposed to be adiabatic.

The meaning of PINI and PREF is the same as for the FLUIDE material (page C3.305).

### Outputs:

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : density

ECR(3) : temperature

### 7.8.18 MIEG (Mie-Grüneisen)

#### Object:

Equation of state of the Mie-Grüneisen type, allowing to treat very fast mechanical phenomena for which the material behaviour is hydrodynamic.

The Mie-Grüneisen equation of state is a relation between the pressure and the volume of a solid at a given temperature. It is used to determine the pressure in a shock-compressed solid. The most commonly used form of the equation (first-order) is:

$$p = \frac{\rho_0 C_0^2 \chi}{(1 - s\chi)^2} \left( 1 - \frac{\Gamma_0}{2} \chi \right) + \Gamma_0 E \quad (26)$$

with:

$$\chi = 1 - \frac{\rho_0}{\rho} = 1 - \frac{V}{V_0} \quad (27)$$

where  $p$  is the absolute pressure,  $\rho_0$  the initial density,  $\rho$  the current density,  $V$  the current volume,  $V_0$  the initial volume,  $E$  the internal energy *per unit volume*,  $s$  and  $\Gamma_0$  are (dimensionless) parameters.

Note that  $E$  is *not* the more commonly used specific internal energy (or internal energy per unit mass)  $i$ , but the internal energy per unit volume, which has the dimensions of a pressure ( $\text{J/m}^3 = \text{N/m}^2 = \text{Pa}$ ). The relation between  $E$  and  $i$  is:

$$E = \rho i \quad (28)$$

In the current implementation the specific internal energy (per unit mass)  $i$  is assumed to be a function only of the temperature  $T$  and therefore  $i$  typically stays constant during the computation (since usually  $T$  does not vary). Also the speed of sound is assumed to be constant and independent of the density ( $C = C_0$ ).

#### Syntax:

```
"MIEG" "RO" ro "C0" c0 "S" s "GAM0" gam0 "ETA" eta "PREF" pref
      "PINI" pini
/LECT/
```

ro

Initial density  $\rho_0$ .

c0

Initial speed of sound  $C_0$  (assumed as constant).

s

$s$  parameter (dimension-less).

gam0

$\Gamma_0$  parameter (dimension-less).

eta

$\eta$  parameter (for the viscosity). This has still to be implemented!

pref

Reference pressure  $p^{\text{ref}}$ .

pini

Initial (absolute) pressure  $p_0$ . The code will automatically evaluate the initial internal energy per unit mass  $i_0$  (and the initial internal energy per unit volume  $E_0 = \rho_0 i_0$ ) such that from the EoS (26) it results  $p(0) = p_0$ . Since in the initial conditions  $\chi_0 = 0$ , from (26) it follows that  $E_0 = \rho_0 i_0 = p_0/\Gamma_0$  and  $i_0 = p_0/(\rho_0 \Gamma_0)$ .

/LECT/

List of the concerned elements.

#### Comments:

All parameters are mandatory, including the reference pressure  $p^{\text{ref}}$ .

#### Outputs:

The components of the ECR table are as follows:

ECR(1) : absolute pressure  $p$ .

ECR(2) : density  $\rho$ .

ECR(3) : specific internal energy (per unit mass)  $i = E/\rho$ .

**7.8.19 ADCR****Object :**

Modelling of the behaviour of a liquid-gas mixture. The mixture, which has three components, is assumed to be homogeneous and includes two phases.

By this model it is possible to analyse the consequences of an explosion in a liquid contained within a tank in the presence of a cover gas.

The calculation is A.L.E. only, and for elements CAR1 and TRIA (2D), or CUBE, PRIS and TETR (3D), or TUBE, TUYA, CAVI and BIFU (1D).

**Syntax :**

```
"ADCR"    "ROLI" roli    <"PLIQ"  pliq  > "CSON"  cson
... <"PSAT" psat    "ROSA" rosat > <"BETA"  beta  >
...  "ROBU" robu    "PBU"  pbu    "GBU"  gbu    "NBU"  nbu
...  "ROGA" roga    "PGAZ" pgaz    "GAMG" gamg
...  "PTOT" ptot    "CBU"  cbu    "CGAZ" cgaz
... <"PREF" pref > <"VIBU" mubu > <"VILI" muli  >
... <"CMIN" cmin > <"ENER" mene > /LECTURE/
```

**roli, pliq**

Initial density and pressure of the liquid assumed alone, defining the state equation for the liquid. If "PLIQ" is absent, the pressure considered is that of the cover gas.

**cson**

Sound speed in the liquid.

**rosat, psat**

Density and pressure of the saturated liquid vapor at the temperature of the liquid, defining the equation of state for the vapor. These parameters are coupled: if one is present, the other must also appear. If they are omitted, they are supposed to be zero.

**beta**

Reduced damping coefficient for high frequencies. It is zero by default, and should always be very small ( < 0.05 ).

**robu, pbu**

Initial density and pressure of the bubble gas, supposed alone, defining the equation of state for the gas.

**gbu**

Ratio  $C_p/C_v$  for the bubble gas.

nbu

Polytropic coefficient of the bubble gas.

roga, pgaz

Initial density and pressure of the cover gas, assumed alone, defining the equation of state of the gas.

gamg

Ratio  $C_p/C_v$  of gas.

pref

Reference pressure. By default, one takes that of liquid.

cmin

Maximum mass concentration of gas within the liquid, so that it is considered to be pure (1.E-8 by default).

mene

For CCFV only. Way in which the internal energy of each single component is computed. If 0, it is a fraction of the total internal energy (the mass fraction is used); if 1, it is computed using the EOS. For the liquid, the energy of the vapour is omitted. (0 by default; see also the description of WSUM for more details).

mul i

Dynamic viscosity of liquid.

mubu

Dynamic viscosity of the gas in the bubble.

ptot

Total pressure (see also the comments below).

cbu

Mass concentration of the gas in the bubble (0 or 1).

cgaz

Initial mass concentration of the cover gas (0 or 1).

/LECTURE/

List of the elements concerned.

**Remarks :**

This material has been initially developed to treat the behaviour of fast reactors in the case of a containment accident (ADC in French), or of a hypothetical core disruptive accident (HCDA in English), but the adopted method lends itself to a more general use.

For this reason, some aliases are introduced. For the liquid, ROLI, PLIQ, CSON, VILI, are identical to RONA, PNA, CNA, VISO (sodium). For the cover gas, ROGA, PGAZ, GAMG, CAR are identical to ROAR, PARG, GAR and CAR (argon). For the bubble generated by the explosion, no aliases are available.

### Comments :

The mesh will be subdivided into three zones (bubble, liquid, cover gas) and the ADCR material will be listed three times by varying each time the initial concentrations of the bubble and of the cover gas (0 or 1 depending on the case), but the other parameters must be identical, in order to have the same constitutive laws for the components of each zone. Then, starting from these concentrations and from the total pressure  $p_{tot}$ , EUROPLEXUS will compute the density of the mixture. EUROPLEXUS will also recompute the concentrations in the gases in order to account for the liquid vapor, if  $p_{sat}$  is not zero.

It is possible, however, to use more than 3 zones, if for example the bubble or the liquid are subdivided into concentric zones. In such cases the total pressure  $p_{tot}$ , varying from zone to zone, will define the initial state. EUROPLEXUS will automatically compute the mixture density using the constitutive law of each component. Of course, also in this case the other parameters such as  $robu$ ,  $pbu$ ,  $roli$ ,  $roga$ , ... will have to be identical.

The gas generated by the explosion is assumed perfect and the state variables are linked by the following equation:

$$P_B = P_{B0} \left( \frac{\rho_B}{\rho_{B0}} \right)^{n_B}$$

Here  $P_{B0}$  and  $\rho_{B0}$  are the initial values and  $n_B$  is the exponent. This exponent may be an arbitrary real number. The value  $n_B = 1$  corresponds to an isothermal transformation while  $n_B = \gamma_B$  to an adiabatic transformation.

The keyword **PBU**, **ROBU**, **NBU** and **GBU** allow to specify the parameters  $P_{B0}$ ,  $\rho_{B0}$ ,  $n_B$  and  $\gamma_B$  in the material data.

The behaviour of the cover gas is of the adiabatic perfect gas type and follows the equation:

$$P_A = P_{A0} \left( \frac{\rho_A}{\rho_{A0}} \right)^{\gamma_A}$$

Here again,  $P_{A0}$  and  $\rho_{A0}$  are the initial values of the gas pressure and mass density, and  $\gamma_A$  the ratio of specific heats.

The keywords **PGAZ**, **ROGA** and **GAMG** allow to specify the parameters  $P_{A0}$ ,  $\rho_{A0}$  and  $\gamma_A$  in the material data.

The liquid is assumed compressible and isothermal. The state equation is the same as for the FLUI material (page C.305). It is of the form:

$$P_L = P_{L0} + (\rho_L - \rho_{L0})c_L^2$$



Here  $P_{L0}$  and  $\rho_{L0}$  are the initial values, and  $c_L$  the sound speed, assumed constant.

The liquid pressure is not allowed to decrease below a minimal value corresponding to the saturation pressure of the vapor  $P_{sat}$ . The vapor (if it exists) then has a density  $\rho_{sat}$ .

The parameters  $P_{L0}$ ,  $\rho_{L0}$ ,  $c_L$ ,  $P_{sat}$  and  $\rho_{sat}$  are defined by means of the keywords **ROLI**, **PLIQ**, **CSO**, **PSAT** and **ROSA**, respectively.

Within an element, the mixture is assumed homogeneous and each component is followed by the evolution of its mass concentration.

Initially, the composition is determined by parameters **CBU** and **CGAZ**, which impose the initial concentrations for the bubble and the cover gas,  $x_B$  and  $x_A$ . The concentration of the liquid may be computed as the difference, since:

$$x_A + x_B + x_L = 1$$

During the transient calculation, these concentrations evolve as a function of the mass flow across the elements, which determines the the mass density of the mixture and the proportions of each component.

It is then necessary to compute the corresponding pressure of the mixture by respecting the state equations. This is done by successive iterations.

It is also possible to assign dynamic viscosities to the liquid and to the gas (parameters **VILI** and **VIBU**).

Finally, the parameter **CMIN** allows to define the threshold of concentration below which one (or two) component(s) is (are) considered to be absent.

One may obtain the work of pressure forces in a region of the mesh by means of the "PDV" keyword of the "REGION" directive (page G.100).

Specific energies related to the liquid, the bubble and the cover gas can be computed and plotted in the post-treatment section (see page ED.110 for syntax).

## Outputs :

The components of the ECR table are as follows:

- ECR(1) : absolute pressure
- ECR(2) : density of the two-phase mixture
- ECR(3) : sound speed in the mixture
- ECR(4) : mass concentration of the cover gas
- ECR(5) : mass concentration of the bubble gas
- ECR(6) : cover gas mass per unit volume
- ECR(7) : bubble gas mass per unit volume
- ECR(8) : cover gas mass increment

ECR(9) : bubble gas mass increment

ECR(10) : volume fraction occupied by the gases

## 7.8.20 EXVL—VAN LEER HYDROGEN DETONATION

## Object:

Modeling of hydrogen detonation by Van Leer formulation.

## Syntax:

```

"EXVL"  <"ROIN" roin>  "TINI" tini    "PINI" pini
...     <"PREF" pref>   "TSEU" tseu
...     "ALPH" alph     <"GAMM" gamm >
...     <"CVME" cvme>   "CONS" cons
...     "MOO2" moo2     "GAO2" gao2     "COO2" coo2
...     "MON2" mon2     "GAN2" gan2     "CON2" con2
...     "MOH2" moh2     "GAH2" gah2     "COH2" coh2
...     "MOH"  moh      "GAH"  gah      "COH"  coh
...     "MOOH" mooh     "GAOH" gaoh     "COOH" cooh
...     "MEAU" meau     "GEAU" geau     "CEAU" ceau
...     "DEBX" debx     "DEBY" deby     <"DEBZ" debz >
...     "CPVA"          /LECTURE/

```

roin

Initial density of the mixture.

tini

Initial temperature of the mixture.

pini

Initial pressure of the mixture.

pref

Reference pressure.

tseu

Temperature at which the reaction is started.

alph

Parameter for the delay time of the reaction.

gamm

Ratio  $C_p/C_v$  of the mixture.

cvme

	Mean $C_v$ of the mixture.
cons	
	Constant $R$ of perfect gases.
moo2	
	Oxygen molar mass.
gao2	
	Ratio $C_p/C_v$ of the oxygen.
coo2	
	Volume concentration of the oxygen.
mon2	
	Molar mass of nitrogen.
gan2	
	Ratio $C_p/C_v$ of nitrogen.
con2	
	Volume concentration of nitrogen.
moh2	
	Molar mass of hydrogen.
gah2	
	Ratio $C_p/C_v$ of hydrogen.
coh2	
	Volume concentration of hydrogen.
moh	
	Molar mass of H.
gah	
	Ratio $C_p/C_v$ of H.
coh	
	Volume concentration of H.
mooh	
	Molar mass of OH.
gaoh	
	Ratio $C_p/C_v$ of OH.

cooh

Volume concentration of OH.

meau

Molar mass of water.

geau

Ratio  $C_p/C_v$  of water.

ceau

Volume concentration of water.

debx

Horizontal mass flow rate in x (density \* velocity = mass flow rate)

deby

Vertical mass flow rate in y (density \* velocity = mass flow rate)

debz

Mass flow rate in z (density \* velocity = mass flow rate)

cpva

Keyword indicating that the  $C_p$  and  $C_v$  of the different components vary with temperature.

/LECTURE/

List of the concerned elements.

### Comments:

This material may be used in EULERIAN with the option DPLA only.

When one uses the keyword "CPVA" the  $C_p$  and  $C_v$  of the various components are functions of the temperature (interpolation of the JANAF tables by a fourth-degree function). Generally during a detonation it is necessary to reduce the stability step at a value of the order of 0.1 microseconds.

### Outputs:

The different components of the ECR table are:

ECR(1) : pressure  
ECR(2) : density  
ECR(3) : sound speed  
ECR(4) : mass flow rate along x  
ECR(5) : mass flow rate along y  
ECR(6) : mass flow rate along z (in 3D)  
ECR(18) : Mach number ( $u/c$ )  
ECR(19) : temperature  
ECR(20) : entropy  
ECR(35) : concentration in O2  
ECR(36) : concentration in N2  
ECR(37) : concentration in H2  
ECR(38) : concentration in H  
ECR(39) : concentration in OH  
ECR(40) : concentration in H2O

### 7.8.21 JW—JONES-WILKINS-LEE LAW

#### Object:

State equation of JW (Jones-Wilkins-Lee) type, allowing to treat explosive phenomena.

#### Syntax:

```
"JWL"  "RO" rho  <"ROS" ros>  "EINT" eint      ...
...    "A" a    "B" b    "R1" r1  "R2" r2  "OMEG" omeg  ...
...    <"BETA" beta>  <"PREF" pref>      ...
...    /LECTURE/
```

#### rho

Initial density  $\rho_0$ .

#### ros

Optional density  $\rho_{\text{sol}}$  of the explosive solid phase, before detonation. **If  $\rho_{\text{sol}}$  is absent**, the code assumes  $\rho_{\text{sol}} = \rho_0$ , which means that at the beginning of the simulation the detonation (assumed instantaneous for this material) has just occurred and the detonation products still occupy exactly the volume previously occupied by the solid explosive (the density has not yet changed). **If  $\rho_{\text{sol}}$  is present** (with  $\rho_{\text{sol}} > \rho_0$ ), then the simulation starts from a time successive to the detonation and it is assumed that the detonation products occupy a volume larger than the original volume of the solid explosive before detonation and have a uniform state (uniform bubble of detonation products), i.e. the same (initial) pressure, density and temperature (specific energy) everywhere in the detonation bubble.

#### eint

Initial specific internal energy  $e_{\text{int},0}$  (internal energy per unit mass).

#### a, b, r1, r2, omeg

Coefficients  $A$ ,  $B$ ,  $R_1$ ,  $R_2$ ,  $\omega$  of the state equation of JW (see below).

#### beta

Reduced damping coefficient  $\beta$  for high frequencies. It is zero by default, and should always be very small ( $\beta < 0.05$ ).

#### pref

Reference pressure  $p_{\text{ref}}$ . By default it is  $p_{\text{ref}} = p_0$ , that is, the reference pressure is taken equal to the initial pressure  $p_0$  which results from the JW equation by using the initial state values ( $\rho_0$ ,  $e_{\text{int},0}$  and possibly  $\rho_{\text{sol}}$ ) chosen by the user.

/LECTURE/

List of the elements concerned.

#### Comments:

The JWL equation of state gives the current value of pressure  $p$  according to the expression:

$$p = A \left( 1 - \frac{\omega}{R_1 V} \right) e^{-R_1 V} + B \left( 1 - \frac{\omega}{R_2 V} \right) e^{-R_2 V} + \omega \rho e_{\text{int}} \quad (29)$$

with:

$\rho$  : current density,

$e_{\text{int}}$  : current internal energy per unit mass,

$V$  : the current ratio  $\rho_{\text{sol}}/\rho$  where  $\rho_{\text{sol}}$  is the density of the solid explosive before detonation (a constant).

The coefficients  $A$ ,  $B$ ,  $R_1$ ,  $R_2$  and  $\omega$  are the equation of state (EoS) parameters, together with  $\rho_{\text{sol}}$  which represents the density of the solid explosive before detonation.  $R_1$ ,  $R_2$  and  $\omega$  are dimensionless parameters, but  $A$  and  $B$  have the dimensions of a pressure.

Once exhausted the solid explosive, and after the expansion of the resulting combustion gases, the above formula tends asymptotically towards the perfect gas law:

$$p = \omega \rho e_{\text{int}}$$

Therefore, the parameter  $\omega$  is related to the ratio  $\gamma$  between the specific heats  $C_p$  and  $C_v$  of the gas by the equation:

$$\omega = \gamma - 1 = \frac{C_p}{C_v} - 1$$

Often in the literature one does not find directly the value of  $e_{\text{int},0}$  to be given in input, but the internal energy *per unit volume*  $E_0$  of the explosive, which also has the dimension of a pressure, like  $A$  and  $B$ . In order to find  $e_{\text{int},0}$  it is sufficient to divide  $E_0$  by  $\rho_{\text{sol}}$ :

$$e_{\text{int},0} = E_0 / \rho_{\text{sol}}$$

As an example, here are the values for TNT extracted from a publication of the Lawrence Livermore Laboratory by E. Lee, M. Finger, W. Collins (1973):

$$\begin{array}{lll} A = 3.738 & B = 0.03747 & E_0 = 0.0600 \\ R_1 = 4.15 & R_2 = 0.90 & \omega = 0.35 \end{array}$$

The above listed pressures ( $A$ ,  $B$ ,  $E_0$ ) are expressed in megabars (Mbar, with  $1 \text{ Mbar} = 10^{11} \text{ Pa}$ ), and the density of the solid TNT is  $\rho_{\text{sol}} = 1630 \text{ kg/m}^3$ .

Note that the initial pressure  $p_0$  in the material is *not* directly specified by the user, but it results from the equation of state (29) by using the initial state values chosen by the user ( $\rho_0$ ,  $e_{\text{int},0}$  and possibly  $\rho_{\text{sol}}$ ):

$$p_0 = A \left( 1 - \frac{\omega}{R_1 V_0} \right) e^{-R_1 V_0} + B \left( 1 - \frac{\omega}{R_2 V_0} \right) e^{-R_2 V_0} + \omega \rho_0 e_{\text{int},0}$$

with  $V_0 = \rho_{\text{sol}}/\rho_0$  (that is,  $V_0 = 1$  if  $\rho_{\text{sol}}$  has been omitted).

When doing an ALE calculation, it is necessary that all the fluids contained in the same domain (which may therefore mix up with one another) have the same constitutive law. However, the initial status may be different at different locations. For example, if the explosion takes place in air, one may use the JWL (or JWLS) model both for the explosive and for the air itself. For the latter, use the limiting perfect gas behaviour of the preceding equation by specifying the corresponding density  $\rho_0$ , internal energy  $e_{\text{int},0}$  and the density of the solid  $\rho_{\text{sol}}$ . In this case, one assumes that the  $\gamma$  values are the same for the air and for the combustion gases.



### Typical use:

Two typical uses of the JWL model can be made, to simulate the detonation of an explosive charge.

In the **first case**, the simulation starts at the end of the detonation (chemical reaction with phase change) of the (typically solid) explosive charge and computes the expansion of the detonation products and their interaction with the surrounding medium (typically the atmosphere, which has a perfect-gas-like behaviour). The solid detonation process is not modelled, and is assumed to be instantaneous.

The initial conditions consist of a bubble of (just formed) gaseous detonation products which (still) occupies exactly the same volume as the solid charge, and thus has the same density  $\rho_{\text{sol}}$  as the solid explosive (e.g., 1630 kg/m<sup>3</sup> for TNT). The initial velocity of the gas in the bubble is assumed to be zero, and the initial conditions (pressure, temperature, sound speed) are those resulting from the JWL EoS, and are uniform over the whole bubble. In this case, the user specifies  $\rho_0$  (which is equal to  $\rho_{\text{sol}}$ , for what has been stated above) but not  $\rho_{\text{sol}}$ . The code then automatically sets  $\rho_{\text{sol}} = \rho_0$ . The constant material parameters  $A$ ,  $B$ ,  $R_1$ ,  $R_2$ ,  $\omega$  are chosen (from the literature) depending on the particular explosive material. The initial specific energy  $e_{\text{int},0}$  is specified as  $e_{\text{int},0} = E_0/\rho_{\text{sol}}$  where again  $E_0$  is taken from the literature for the specific explosive.

This type of simulation may require a very fine mesh to represent the initial bubble of detonation products and therefore it can be too expensive in some cases.

In the **second case**, the simulation starts some time after the end of the detonation (chemical reaction with phase change) of the explosive charge. The initial situation is that of a bubble of detonation products which has already expanded to a certain degree and so has a density  $\rho_0$  which is less than that of the solid explosive ( $\rho_0 < \rho_{\text{sol}}$ ). This typically allows to use a (much) coarser mesh than in the previous case and may substantially reduce the cost of the simulation.

Like in the previous case, the initial velocity of the gas in the bubble is assumed to be zero, and the initial conditions (pressure, temperature, sound speed) are those resulting from the JWL EoS, and are uniform over the whole bubble. In this case, the user specifies both  $\rho_0$  (which is less than  $\rho_{\text{sol}}$ ) and  $\rho_{\text{sol}}$  (which is used in the EoS in order to compute  $V$ ). The value of  $\rho_0$  can be computed as  $\rho_0 = \rho_{\text{bub},0} = (V_{\text{sol}}/V_{\text{bub},0})\rho_{\text{sol}}$  where  $V_{\text{sol}}$  is the volume of the solid charge (before detonation) and  $V_{\text{bub},0}$  is the volume of the bubble (after detonation) in the assumed initial configuration.

The constant material parameters  $A$ ,  $B$ ,  $R_1$ ,  $R_2$ ,  $\omega$  are chosen (from the literature) depending on the particular explosive material, like previously. However, the initial specific energy  $e_{\text{int},0}$  is *not* simply equal to  $e_{\text{int},0} = E_0/\rho_{\text{sol}}$  like in the previous case, because the detonation products have already expanded by a certain amount. If the pressure (assumed uniform) of the bubble in the initial configuration is known, or can be estimated, then the value of  $e_{\text{int},0}$  can be obtained from the EoS. In any case, if the bubble is not much larger than the solid charge then the exponential terms prevail by far in the EoS so that the perfect gas term ( $\omega\rho_0 e_{\text{int},0}$ ) containing  $e_{\text{int},0}$  is probably negligible and an exact determination of  $e_{\text{int},0}$  is not very important.

In **both typical scenarios** outlined above, another JWL material model (besides the one representing the detonation products) would typically be used in order to model the surrounding air, at least if the explosion takes place in the atmosphere. The interest of this choice, over the use of a different material such as e.g. GAZP in order to model the air, is that since all fluids are modelled by the same EoS (although in different initial conditions) an ALE or Eulerian description can be assumed which models the intemixing of detonation products with the air without needing a specific multi-component material model.

For this second JWL material, the user specifies both  $\rho_0$ , which is the standard density of the atmosphere and thus much less than  $\rho_{\text{sol}}$ , and  $\rho_{\text{sol}}$  (which is used in the EoS in order to compute

$V$ ). The constant material parameters  $A$ ,  $B$ ,  $R_1$ ,  $R_2$ ,  $\omega$  should be taken equal to those of the “first” JWL material (the one used for the detonation products), in order to allow material mix-up. In any case, the contribution of the exponential terms in the EoS for the air material is negligible due to the very low density compared with the solid. Finally, since the desired initial pressure  $p_0$  (typically the atmospheric pressure here, i.e 1 bar) is known, the value of  $e_{\text{int},0}$  to be given in input such that an initial pressure  $p_0$  will be computed by the code can be obtained from the EoS as:

$$e_{\text{int},0} \approx \frac{p_0}{\omega \rho_0} \quad \text{for } \rho_0 \ll \rho_{\text{sol}} \quad (30)$$

### Outputs:

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : density

ECR(3) : sound speed

**7.8.22 CHOC—RANKINE-HUGONIOT SHOCK****Object**

Equation of state derived from the Rankine-Hugoniot equations, allowing to treat phenomena of shock wave propagations. It is based upon a linear relationship that links the wave propagation velocity to the velocity of fluid particles (empirical relation).

**Syntax:**

```
"CHOC"  "RO"  rho  "A"  a  "B"  b
... < "PMIN" pmin > /LECTURE/
```

rho

Initial density.

a,

Coefficients of the law.

pmin

Cavitation pressure (by default, pmin = 0).

List of the elements concerned.

**Comments**

Let  $c$  be the wave propagation speed and  $v_p$  the velocity of the particles. These two velocities are related by the following linear relationship:

$$c = a + b * v_p$$

The  $a$  quantity is the wave propagation speed at rest.

For details on this material, see the corresponding theoretical report.

**Outputs:**

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : density

ECR(3) : sound speed

**7.8.23 GPDI—DIFFUSIVE VAN LEER PERFECT GAS****Object:**

Diffusive Van Leer perfect gas.

**Syntax:**

```
"GPDI"    "ROIN" roin    "TINI" tini    "PINI" pini
...  <"PREF" pref>    "GAMA" gama    "CLAM" clam
...    "PLAM" plam    "CMU" cmu    "PMU" pmu
...    "CKAP" ckap    "PKAP" pkap    <"IGRA" igra>
...  <"TMUR" tmur>    <"QMUR" qmur>    <"ICLI" icli>
...  <"ECLI" ecli>    "DEBX" debx    <"DEBY" deby>
...                                     /LECTURE/
```

roin

Initial density of the mixture.

tini

Initial temperature of the mixture.

pini

Initial pressure of the mixture.

pref

Reference pressure.

gama

Ratio  $C_p/C_v$  of the gas.

clam

Constant  $\lambda$  (Lamé coefficient).

plam

Slope of  $\lambda$ .

cmu

Constant  $\mu$  (Lamé coefficient).

pmu

Slope of  $\mu$ .

ckap

Constant kappa (conductivity)

pkap

Slope of kappa.

igra

Choice of the gradient.

tmur

Wall temperature.

qmur

Wall flux.

icli

Boundary layer choice (0 or 1).

ecli

Thickness coefficient of the boundary layer.

debx

Horizontal mass flow rate in x (density \* velocity = mass flow rate)

deby

Vertical mass flow rate in y (density \* velocity = mass flow rate)

/LECTURE/

List of the concerned elements.

### Comments:

This material may be used in EULERIAN only with the DPLA option.

If icli is equal 1, the boundary layer is taken into account.

If igrad is equal 0, the gradient is computed at the beginning of step n, if igrad is 1 the gradient estimation is done at the step  $n + 1/2$ . When igrad is 2, the gradient is computed at the end of the step.

### Outputs:

The various components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : mass flow rate in x

ECR(5) : mass flow rate in y

ECR(18) : Mach number ( $u/c$ )

ECR(19) : temperature

ECR(20) : entropy

### 7.8.24 VAN LEER PERFECT GAS

#### Object:

Euler: perfect gas ( $P = \rho * (\gamma - 1) * E_{\text{interne}}$ ).

#### Syntax:

```
"GZPV"  "RO"  rho  "GAMMA" gamma  "PINI"  pini  <"PREF" pref > ...
        "DEBX" debx "DEBY" deby  <"DEBZ" debz>    /LECTURE/
```

**rho**

Initial density.

**gamma**

Ratio  $C_p/C_v$  (supposed constant).

**pini**

Initial pressure.

**pref**

Reference pressure.

**debx**

Horizontal mass flow rate in x (density \* velocity = mass flow rate).

**deby**

Vertical mass flow rate in y (density \* velocity = mass flow rate).

**debz**

Mass flow rate in z (density \* velocity = mass flow rate).

**/LECTURE/**

List of the concerned elements.

#### Comments:

Material usable in EULERIAN or ALE, with the option DPLA or TRIDI.

The reference pressure *pref* allow to define the initial state. if *pref* = *pini*, the gas is initially in equilibrium and will be perturbed by an external action, e.g. the motion of a piston. If *pref* = 0, the problem reduces to an initial stress problem whose value corresponds to *pini*. It is the case of a membrane separating two gases in different states, which disappears at the initial calculation time.

**Outputs:**

The different components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : mass flow rate in x

ECR(5) : mass flow rate in y

ECR(6) : mass flow rate in z (in 3D)

ECR(18) : Mach number ( $u/c$ )

ECR(19) : temperature

ECR(20) : entropy



### 7.8.25 ADCJ

#### Object :

Modeling of the behaviour of a liquid-gas mixture. The mixture, which has 3 components, is assumed homogeneous and has 2 phases.

By this model it is possible to analyse the consequences of an explosion in a liquid contained within a tank in the presence of a cover gas.

This material is similar to material ADCR (page C.430) but the behaviour of the gas generated by the explosion is of type JWL instead of being a perfect gas (see page C.440).

The calculation is A.L.E. for elements CAR1 and TRIA (2D), or CUBE, PRIS and TETR (3D).

#### Syntax :

```
"ADCJ"  "ROLI" roli  <"PLIQ" pliq >  "CSON" cson
...  <"PSAT" psat   "ROSA" rosat >
...  "ROBU" robu    "A"    a        "B"    b
...  "R1"  r1       "R2"  r2       "OMEG" omega
...  "EIBU" eibu    <"PARA" param > <"ROBZ" robz >
...  "ROGA" roga    "PGAZ" pgaz    "GAMG" gamg
...  "PTOT" ptot    "CBU"  cbu      "CGAZ" cgaz
...  <"PREF" pref > <"VILI" muli  >
...  <"CMIN" cmin >  /LECTURE/
```

roli, pli

Initial density and pressure of liquid, assumed alone, defining the equation of state for the liquid. If "PLIQ" is absent, the assumed pressure is that of the cover gas.

cson

Sound speed in the liquid.

rosat, psat

Density and pressure of saturated liquid vapor, at the temperature of the liquid, defining the equation of state for the vapor. These parameters are coupled: if one is present, the other must also be given. If they are missing, they are assumed to be zero.

robu

Initial density of the bubble gas, assumed alone.

a, b ,r1 ,r2, omeg

Coefficients of the JWL state equation for the bubble (see C.440).

eibu

Initial specific internal energy of the bubble gas.

param

Number of integrations to solve the adiabaticity condition for a JWL gas. By default, param = 100, suggested value for a good precision level.

robz

Density of the bubble gas, defining the equation of state for the gas. By default, robz is equal to robu.

roga, pgaz

Initial density and pressure of the cover gas, assumed alone, defining the state equation of the gas.

gamg

Ratio  $C_p/C_v$  of the cover gas.

pref

Reference pressure. By default, one takes that of liquid.

cmin

Maximum mass concentration of gas in the liquid so that it is considered to be pure (1.E-8 by default).

mul i

Dynamic viscosity of liquid.

ptot

Total pressure (see also the comments). In the case of the JWL gas, the pressure is re-computed strating from the energy.

cbu

Initial mass concentration of the bubble gas (0 or 1).

cgaz

Initial mass concentration of the cover gas (0 or 1).

/LECTURE/

List of the concerned elements.

### Remarks :

This material has been initially developed to treat the behaviour of fast reactors in the case of a containment accident (ADC in French), or of a hypothetical core disruptive accident (HCDA in English), but the adopted method lends itself to a more general use.

For this reason, some aliases are introduced. For the liquid, ROLI, PLIQ, CSON, VILI, are identical to RONA, PNA, CNA, VISO (sodium). For the cover gas, ROGA, PGAZ, GAMG, CAR are identical to ROAR, PARG, GAR and CAR (argon).

**Comments :**

The mesh will be subdivided into three zones (bubble, liquid, cover gas) and the ADCJ material will be listed three times by varying each time the initial concentrations of the bubble and of the cover gas (0 or 1 depending on the case), but the other parameters must be identical, in order to have the same constitutive laws for the components of each zone. Then, starting from these concentrations and from the total pressure  $p_{tot}$ , EUROPLEXUS will compute the density of the mixture. EUROPLEXUS will also recompute the concentrations in the gases in order to account for the liquid vapor, if  $p_{sat}$  is not zero.

It is possible, however, to use more than 3 zones, if for example the bubble or the liquid are subdivided into concentric zones. In such cases the total pressure  $p_{tot}$ , varying from zone to zone, will define the initial state. EUROPLEXUS will automatically compute the mixture density using the constitutive law of each component. Of course, also in this case the other parameters such as  $robu$ ,  $pbu$ ,  $rona$ ,  $roar$ , ... will have to be identical.

The gas generated by the explosion is assumed to follow a JWL behaviour. On this subject, see page C.440. The state variables are linked by the following equation:

$$P_B = A \left( 1 - \frac{\Omega\eta}{R_1} \right) e^{-\frac{R_1}{\eta}} + B \left( 1 - \frac{\Omega\eta}{R_2} \right) e^{-\frac{R_2}{\eta}} + \Omega\rho_B E_B$$

The variable  $\eta$  is the reduced mass density ( $\eta = \frac{\rho_B}{\rho_{B0}}$ ), and  $E_B$  the specific internal energy.

The other parameters ( $A$ ,  $B$ ,  $R_1$ ,  $R_2$ ,  $\Omega$ ) are characteristic constants of the gas under consideration. It is remarked that if  $A = B = 0$  and  $\Omega = \gamma_B - 1$ , the equation of state of a perfect gas is recovered:

$$P_B = (\gamma_B - 1)\rho_B E_B$$

The keywords **ROBU**, **EIBU**, **A**, **B**, **R1**, **R2** and **OMEG** allow to specify, respectively, the initial density  $\rho_{B0}$ , the initial internal energy  $E_{B0}$ , and the 5 constants of the JWL law.

The state variables  $P_B$ ,  $\rho_B$  and  $E_B$  evolve during the calculation but remain linked by the state equation. In addition, it is assumed that the transformation followed by the bubble gas is adiabatic, which allows to eliminate one of these state variables, e.g.  $E_B$ . One then recovers an equation of the form:

$$P_B = f(\rho_B)$$

The other parameters are identical to those of the ADCR material (see page C.430).

One may obtain the work of pressure forces in a region of the mesh by means of the "PDV" keyword of the "REGION" directive (page G.100).

For more informations see reference [599].

**Outputs :**

The components of the ECR table are as follows:

- ECR(1) : absolute pressure
- ECR(2) : density of the two-phase mixture
- ECR(3) : sound speed in the mixture
- ECR(4) : mass concentration of the cover gas
- ECR(5) : mass concentration of the bubble gas
- ECR(6) : cover gas mass per unit volume
- ECR(7) : bubble gas mass per unit volume
- ECR(8) : mass increment of the cover gas
- ECR(9) : mass increment of the bubble gas
- ECR(10) : volume occupation rate of the gases
- ECR(11) : partial pressure of the cover gas
- ECR(12) : partial pressure of the bubble gas if  $\text{ecr}(5) > \text{cmin}$ , else initial pressure
- ECR(13) : density of the cover gas
- ECR(14) : density of the bubble gas if  $\text{ecr}(5) > \text{cmin}$ , else initial density

### 7.8.26 FLUID PARTICLE

**Object:**

This directive allows to specify a fluid behaviour for the particle elements (BILLE). The fluid is isothermal and perfect.

**Syntax:**

```
"BILLE" "FLUIDE" "R0" rho "C" c <"PINI" pini>
      <"PREF" pref > <"PMIN" pmin >
      <"VISC" mu > /LECTURE/
```

rho

Density.

c

Sound speed (constant).

pini

Value of the initial pressure.

pref

Value of the reference pressure.

pmin

Value of the minimum pressure (by default, pmin = 0).

mu

Dynamic viscosity.

/LECTURE/

List of the concerned elements.

**Comments:**

The first two parameters are mandatory. If the initial pressure is given, EUROPLEXUS will take it into account for the calculation of the absolute pressure (ECR(1)).

**Rôle of PREF:**

When the reference pressure is different from the initial pressure, the fluid is initially not in equilibrium. It is the case of a membrane which breaks at  $t = 0$ , thus releasing a compressed fluid.

In numerous problems the focus is on acoustic effects, and one assumes that a fluid initially in equilibrium evolves under the effects of a loading; in such cases one will take  $p_{ref} = p_{ini}$ .

If "PREF" is omitted, EUROPLEXUS considers that the fluid is in equilibrium and that  $p_{ref} = p_{ini}$  (even in the case that  $p_{ini} = 0$ ).

The pressure in the fluid will always remain greater than or equal to the minimum pressure  $p_{min}$ , even though the density diminishes. This is a very simplified way of treating cavitation. The default value of  $p_{min}$  is  $p_{min} = 0$ .

If "PINI" is omitted, the initial pressure is null.

### **Outputs:**

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : density

**7.8.27 PRGL—POROUS JELLY****Object :**

This directive allows to specify a behaviour simulating porous jelly (used in the bird impact studies onto turbine blades) for particle elements and SPH. For the description of this material, please consult the thesis of Antoine Letellier.

**Syntax:**

```

"PRGL"  "R01" rho1    "R02" rho2    "GAMM" gamma          ...
...     "CSN1" csn1   "CSN2" csn2   "CVT1" cvt1    "CVT2" cvt2    ...
...     "PROP" prop   "PINI" pini   "PREF" pref     "PMIN" pmin    ...
...     <"VISC" mu >                                     ...

...                                           /LECTURE/

```

rho1

Density of the fluid.

rho2

Density of the gas.

gamma

Perfect gas constant for the gas.

csn1

Sound speed for the fluid.

csn2

Sound speed for the gas.

cvt1

Coefficient expressing a linear relationship between the shock speed and the impact velocity for the fluid (see comments below).

cvt2

Idem for the gas.

prop

Fluid proportion in the mixture (in volume).

pini

Value of the initial pressure.

pref

Value of the reference pressure.

pmin

Value of the cavitation pressure.

mu

Dynamic viscosity.

/LECTURE/

List of the concerned elements

### Comments:

The coefficients cvt1 and cvt2 are used in the following equation:

$$V_{\text{choc}} = V_{\text{son}} + \text{cvt} * V_{\text{impact}}$$

### Outputs:

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : density



## 7.8.28 VAN DER WAALS GAS

### Object :

This directive allows to specify a real gas material with a Van Der Waals behaviour.

In the 1-D case, the friction onto the walls may be accounted for, and the dissipated energy heats up the gas (modification of the internal energy). To this end, one must add a "PARO" material and associate it with "GVDW" by means of a "MULT" material (pages C.370 and C.380).

### Syntax :

```
"GVDW"  "RO"  rho  "PINI"  pini  < "PREF"  pref >  ...  
...  "RM"  rm  "CV"  Cv  ...  
...  $[ "A"  a  "B"  b  ; "PCRI"  pc  "TCRI"  tc ]$ ...  
...  /LECTURE/
```

rho

Initial density.

pini

Initial pressure.

pref

Reference pressure.

rm

Constant of perfect gases R divided by the molecular mass M (assumed constant).

Cv

Specific heat at constant volume.

a

Constant a of the gas (assumed constant).

b

Constant b of the gas (assumed constant).

pc

Pressure at the critical point.

tc

Temperature at the critical point.

/LECTURE/

List of the concerned elements.

### Comments :

The state equation of a Van der Waals gas has the form:

$$P = \frac{r_m T}{(\frac{1}{\rho} - b)} - a\rho^2$$

or, equivalently:

$$P = \frac{r_m(e + a\rho)}{C_v(\frac{1}{\rho} - b)} - a\rho^2$$

The first form is obtained by the state variables  $P$ ,  $\rho$  and  $T$  (respectively pressure, density and temperature), while the second uses  $P$ ,  $\rho$  and  $e$ , i.e. the temperature is replaced by the specific internal energy.

Some theoretical complements are given in the report [628], which lists also the values of parameters  $a$  and  $b$  for some fluids.

These coefficients  $a$  and  $b$  are related to the critical pressure  $P_c$  and to the critical temperature  $T_c$  by the following expressions :

$$P_c = \frac{a}{27b^2} \quad T_c = \frac{8a}{27br_m}$$

The parameter  $r_m$  is the ratio between the perfect gas constant ( $R=8.31441 \text{ JK}^{-1}\text{mol}^{-1}$ ) and the molar mass  $M$  of the gas. Pay attention to units: for example, for air it is  $M = 0.029 \text{ Kg}$ .

The reference pressure  $\text{pref}$  allows to define the initial state. If  $\text{pref} = \text{pini}$ , the gas is initially in equilibrium and will be perturbed by an external action, e.g. motion of a piston. If  $\text{pref} = 0$ , the problem reduces to a calculation with initial stresses, defined by  $\text{pini}$ . It is the case of a membrane, that initially separates two gases in different conditions, which suddenly breaks at the initial time.

### Outputs :

The various components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : temperature

ECR(5) : dynamic pressure:  $(\frac{1}{2}\rho V^2)$

ECR(6) : total energy  $(e + \frac{1}{2}V^2)$ .

## 7.8.29 JWLS

**Object:**

State equation of JWL (Jones-Wilkins-Lee) type, similar to the JWL material on page C.440, but allowing also to account for the initial propagation of detonation (at an assumed constant speed) in the solid charge. The user may specify the initiation point (trigger) of the detonation process in the (initially) solid charge.

The propagation of detonation (chemical reaction with phase change) in the initially solid charge can be spherical (by default) or planar.

Optionally, a simplified modelling of the afterburning effect can be added, i.e. the release of energy by additional chemical reactions (combustion) of the detonation products during their expansion after the detonation proper. (The afterburning effect cannot presently be taken into account by the JWL material model of page C.440.)

**Syntax:**

```
"JWLS"  "RO" rho  <"ROS" ros>  "EINT" eint  <"PINI" pini>    ...
...    "A" a  "B" b  "R1" r1  "R2" r2  "OMEG" omeg          ...
...    <"BETA" beta> <"PREF" pref>                          ...
...    <"D" d  "XDET" xdet  "YDET" ydet  <"ZDET" zdet>>      ...
...    <"PCJ" pcj>                                           ...
...    <"VXD" vxd  "VYD" vyd  <"VZD" vzd>>                  ...
...    <"EAFT" eaft  "CONF" conf  "CHAR" char  "TSTA" tsta    ...
...    "TEND" tend>                                          ...
...    /LECTURE/
```

rho

Initial density  $\rho_0$ .

ros

Optional density  $\rho_{\text{sol}}$  of the explosive solid phase, before detonation. **If  $\rho_{\text{sol}}$  is absent**, the code assumes  $\rho_{\text{sol}} = \rho_0$ . Then, two sub-cases may occur. In the first sub-case, which is the most typical use of JWLS, the explosive is still solid and the progressive detonation of the solid charge has to be modelled. This is recognized by the presence of D (the detonation speed) and of XDET, YDET, ZDET (the position of the trigger). In the second sub-case, the solid explosive has just (instantaneously) detonated (so it still occupies the same volume as the solid, the density has not changed yet) and the JWLS material behaves just like JWL. This is recognized by the absence of D, XDET, YDET, ZDET in the input. **If  $\rho_{\text{sol}}$  is present** (with  $\rho_{\text{sol}} > \rho_0$ ), then the simulation starts from a time successive to the detonation and it is assumed that the detonation products occupy a volume larger than the original volume of the solid explosive before detonation and have a uniform state (uniform bubble of detonation products), i.e. the same (initial) pressure, density and temperature (specific energy) everywhere in the detonation bubble.

eint

Initial specific internal energy  $e_{\text{int},0}$  (internal energy per unit mass).

pini

Initial (absolute) pressure  $p_0$  in the solid (undetonated) material. This quantity *must* be specified if  $d$  (detonation speed in the solid) is also specified, because the initial pressure in the solid cannot be obtained from the JWL EoS, which is only valid for the (gaseous) detonation products. If  $d$  is not specified (so that we start from an already detonated explosive) then  $p_0$  *must not* be specified, since in this case the initial pressure will be computed from the JWL equation by using the initial state values ( $\rho_0$ ,  $e_{\text{int},0}$  and possibly  $\rho_{\text{sol}}$ ) chosen by the user.

**a, b, r1, r2, omeg**

Coefficients  $A$ ,  $B$ ,  $R_1$ ,  $R_2$ ,  $\omega$  of the state equation of JWL (see below).

**beta**

Reduced damping coefficient  $\beta$  for high frequencies. It is zero by default, and should always be very small ( $\beta < 0.05$ ).

**pref**

Reference pressure  $p_{\text{ref}}$ . By default it is  $p_{\text{ref}} = p_0$ , that is, the reference pressure is taken equal to the initial pressure  $p_0$  which results from the JWL equation by using the initial state values ( $\rho_0$ ,  $e_{\text{int},0}$  and possibly  $\rho_{\text{sol}}$ ) chosen by the user.

**d**

Velocity  $d$  (assumed constant) of the detonation wave (chemical reaction with phase change) in the solid charge. The wave shape is assumed to be spherical by default, but a planar wave can be chosen in alternative by specifying `vxd ...` (see below).

**xdet, ydet, zdet**

Coordinates of the detonation trigger point in the solid charge. In 2D `zdet` is ignored.

**pcj**

Chapman-Jouguet pressure. Note that this parameter may be specified but it is *unused in the present model*.

**vxd, vyd, vzd**

Components of a vector (not necessarily unitary, but non-zero) defining the direction of propagation of the detonation wave (chemical reaction with phase change) in the solid charge. In this case (i.e. if any of these three quantities is specified) the detonation wave in the solid is assumed to be planar rather than spherical (which is the default). In 2D `vzd` is ignored.

**eaft**

Afterburning effect: additional energy for the afterburning (per kg).

**conf**

Afterburning effect: volume of the confinement.

**char**

Afterburning effect: total mass of the charge.

**tsta**

Afterburning effect: start time when afterburning energy should be added.

tend

Afterburning effect: end time until which afterburning energy should be added.

/LECTURE/

List of the concerned elements.

### Comments:

The JWL state equation gives the value of the pressure according to the following formula (see also page C.440):

$$p = A \left( 1 - \frac{\omega}{R_1 V} \right) e^{-R_1 V} + B \left( 1 - \frac{\omega}{R_2 V} \right) e^{-R_2 V} + \omega \rho e_{\text{int}} \quad (31)$$

with:

$\rho$  : current density,

$e_{\text{int}}$  : current internal energy per unit mass,

$V$  : the current ratio  $\rho_{\text{sol}}/\rho$  where  $\rho_{\text{sol}}$  is the density of the solid explosive before detonation (a constant).

If  $\rho_{\text{sol}}$  is omitted, the propagation of the detonation wave (chemical reaction) in the solid explosive charge is modelled. In this case the code assumes  $\rho_{\text{sol}} = \rho_0$  (i.e., the initial density  $\rho_0$  to be specified in input is that of the solid) and it is mandatory to specify the detonation speed  $d$  and the position of the ignition point (trigger)  $x_{\text{det}}$ ,  $y_{\text{det}}$ , (and  $z_{\text{det}}$  in 3D), in order to start up the reaction.

The coordinates  $x_{\text{det}}$ ,  $y_{\text{det}}$ , (and  $z_{\text{det}}$  in 3D) define the position of the ignition point (trigger) in the initially solid charge where the reaction begins. From this point the reaction wave progresses with constant velocity  $d$ , and when it reaches the center of each finite element the reaction is (instantaneously) activated in that element, whose material passes from solid to gas.

Like for the JWL material, it is possible to use this constitutive law to approximate the behaviour of a perfect gas, such as the atmospheric air for example. To this end it is necessary to specify  $\rho_{\text{sol}}$  and to choose the initial values  $\rho_0$  and  $e_{\text{int},0}$  for the gas such that the correct initial pressure (usually the atmospheric pressure) will result from the EoS. Note that in this case  $p_0$  is not used by the model and should be omitted and it is also useless to specify  $d$  and the coordinates of the ignition point.

Like for the other fluid models, the reference pressure  $p_{\text{ref}}$  allows to account for an external pressure.

### Typical use:

The typical use of the JWLS model to simulate the detonation of an (initially solid) explosive charge is described hereafter. The advantage over the use of JWL is the possibility of modelling the advancing detonation (chemical reaction) front in the solid charge, which changes progressively phase, and (optionally) also to add afterburning effects.

The simulation starts at the beginning of the detonation (chemical reaction with phase change) of the (typically solid) explosive charge, at the instant that the detonation is triggered

at a user-defined point in the solid charge. The code should first compute the advancement of the chemical reaction in the solid and then (like for the JWL material) the expansion of the detonation products and their interaction with the surrounding medium (typically the atmosphere, which has a perfect-gas-like behaviour). **The solid detonation process is modelled**, albeit very primitively, and is not assumed to be instantaneous, unlike for JWL.

The initial conditions consist of a mass of solid explosive material of density  $\rho_0 = \rho_{\text{sol}}$  (e.g., 1630 kg/m<sup>3</sup> for TNT). The user specifies  $\rho_0$  (which is equal to  $\rho_{\text{sol}}$ , for what has been stated above) but not  $\rho_{\text{sol}}$ . The code then automatically sets  $\rho_{\text{sol}} = \rho_0$ . The constant material parameters  $A, B, R_1, R_2, \omega$  are chosen (from the literature) depending on the particular explosive material. The initial specific energy  $e_{\text{int},0}$  is specified as  $e_{\text{int},0} = E_0/\rho_{\text{sol}}$  where again  $E_0$  is taken from the literature for the specific explosive.

Then, the detonation speed  $d$  and the position of the trigger  $x_{\text{det}}, y_{\text{det}}$  (and  $z_{\text{det}}$  in 3D) are also specified, in order to activate the progressive detonation of the solid. A spherical wave is assumed by default, but alternatively the user can choose a planar wave by specifying also  $v_{\text{xd}}, v_{\text{yd}}$  (and  $v_{\text{zd}}$  in 3D). The afterburning model may optionally be activated by specifying its optional parameters.

Note that in this case the value of the initial pressure  $p_0$  *must* be specified. This is the initial pressure in the solid phase (usually equal to the atmospheric pressure), which cannot be computed by the code (it does *not* result from the JWL EoS, because the equation applies only to the detonation products i.e. the gaseous phase).

This type of simulation may require a very fine mesh to represent the initial bubble of detonation products and therefore it can be very expensive in some cases. However, a price has to be paid for the added sophistication with respect to JWL.

Then, **another JWLS material model** (besides the one representing the solid charge, gradually converted into detonation products) would typically be used in order to model the surrounding air, at least if the explosion takes place in the atmosphere. The interest of this choice, over the use of a different material such as e.g. GAZP in order to model the air is that, since all fluids are modelled by the same EoS (although in different initial conditions), an ALE or Eulerian description can be assumed which models the intemixing of detonation products with the air without needing a specific multi-component material model.

For this second JWLS material, the user specifies both  $\rho_0$ , which is the standard density of the atmosphere and thus much less than  $\rho_{\text{sol}}$ , and  $\rho_{\text{sol}}$  (which is used in the EoS in order to compute  $V$ ). The constant material parameters  $A, B, R_1, R_2, \omega$  should be taken equal to those of the “first” JWLS material (the one used for the solid charge and detonation products), in order to allow material mix-up. In any case, the contribution of the exponential terms in the EoS for the air material is negligible due to the very low density compared with the solid. Since the pressure  $p_0$  (typically the atmospheric pressure here, i.e 1 bar) is known, the value of  $e_{\text{int},0}$  to be given in input such that an initial pressure  $p_0$  will be computed by the code can be obtained from the EoS as:

$$e_{\text{int},0} \approx \frac{p_0}{\omega \rho_0} \quad \text{for } \rho_0 \ll \rho_{\text{sol}} \quad (32)$$

Finally, note that the initial pressure  $p_0$  *must not* be specified in this case, since it will be computed by the code using the EoS.

## Outputs:

The components of the ECR table are as follows:

ECR(1) : absolute pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : burnt (detonated) fraction: either 0 (for undetonated solid) or 1 (for gas, i.e. detonation products).

### 7.8.30 USER-DEFINED FLUID (FLUT)

#### Object:

This material is used with specialised elements of type "FLU1", "FLU3", "FL23", "FL24", "FL34", "FL35", "FL36" and "FL38". The subroutine describing the fluid's equation of state has to be written by the user (an example is given below).

#### Syntax:

```
"FLUT"  "RO" ro  "EINT" eint
        $[ "GAMM" gamm ; R r C0 c0 C1 c1 C2 c2 ]$
        < "CL" cl  "CQ" cq  "PB" pb  "PMIN" pmin
          "AHGF" ahgf  "ITER" iter  "ALF0" alf0
          "BET0" bet0  "KINT" kint  "NUM" num
          "VXFF" vxff  "VYFF" vyff  "VZFF" vzff
          "CONV" conv  "PREF" pref  "RREF" rref
          "RMIN" rmin  "RMAX" rmax
          "IMIN" imin  "IMAX" imax  "RANG" rang
          "GENE" gene  "GENM" genm          >
        < additional_data_for_JWLS_material  >
        /LECTURE/
```

ro

Initial mass density.

eint

Specific internal energy.

gamm

Constant value of the ( $\gamma = c_p/c_v$ ) ratio; alternatively (but only for a perfect gas, i.e. NUM=1), the next four parameters may be given which are used to describe the dependence of  $c_p$ , and hence  $c_v$ ,  $\gamma$  and the internal energy of the perfect gas upon temperature. To this end, we use the following relations:  $c_p(T) = c_0 + c_1T + c_2T^2$  where  $T$  is the temperature in K,  $c_p$  and  $c_0$  are typically expressed in J/kgK,  $c_1$  in J/kg(K2) and  $c_2$  in J/kg(K3). Then, the specific heat at constant volume  $c_v$  is given in J/kgK by the expression  $c_v(T) = c_p(T) - R$ , where  $R$  is the specific perfect gas constant of the gas considered, also expressed in J/kgK. Note that  $R = R'/w$  where  $R'$  is the universal constant of perfect gases expressed in J/kmolK (which has the standard value of 8314.3) and  $w$  is the molar weight of the gas expressed in kmol/kg. Alternativley, one could express  $c_p$  and  $c_0$  in J/kmolK,  $c_1$  in J/kmol(K2) and  $c_2$  in J/kmol(K3). In this case,  $c_v$  would result in J/kmolK from the expression  $c_v(T) = c_p(T) - R'$ , where  $R'$  is the universal constant of perfect gases defined above.

cl



Coefficient of linear artificial viscosity. Range is 0.0 to 0.8, default is 0.0.

cq

Coefficient of quadratic artificial viscosity. Range is 0.0 to 4.0, default is 0.0.

pb

Constant used in sound speed evaluation. For material 9 this is the reference pressure. For material 10 this is the index of the FONC used to describe the imposed pressure as a function of time.

pmin

Cut-off pressure. Default is 0.0.

ahgf

Anti-hourglass coefficient. Suggested value is 0.01, default is 0.0.

iter

There are two possibilities. If a number greater than or equal to 1 is given, then this quantity represents the (fixed) number of iterations for the calculation of tilde pressure and tilde specific energy. If, instead, one gives a (real) value smaller than 1 (for example `ITER 1.E-6`), then this quantity is interpreted as the tolerance for convergence of the tilde pressure and tilde specific energy, and the maximum number of iterations for convergence is in that case limited to 100. Finally, note that if one does not specify this quantity (or specifies 0), then by default just one iteration is performed.

alf0

Parameter used to compute the donor element weighting factor in connection with mass and energy transport across element boundaries. When `alf0=0` mass and energy fluxes are centered; when `alf0=1` the fluxes are full donor.

bet0

Parameter used to compute the donor element weighting factor in connection with momentum transport across element boundaries. When `bet0=0` momentum flux is centered; when `bet0=1` the flux is full donor.

kint

In the 2D case: represents the integration type for the forces due to momentum transport (FL24 element only) (-1: no momentum transport at all, to be used only for debugging purposes; 0: use centroid values, is the default; 1: use values at integration points, i.e. a  $2 \times 2$  Gauss rule; 2: use a  $3 \times 3$  Gauss rule, which is exact in axisymmetric cases). Note that the default value (`kint = 0`) is known to destroy symmetry even in 2D plane cases, so use at least `kint = 1` if a perfectly symmetric solution is desired (e.g. in academic validation tests). In the 3D case: represents the integration type for nodal mass distribution (0: element mass/number of nodes (default); 1: Gauss integration).

num

Number of the material in the user's subroutine FLUTIL (see below). By default, `num=1`.

vxff

X-Component of the far-field velocity (this and the following two components are only used in the case that the FLUT material is used to specify a far-field status via a CLxx element); by default it is 0.0.

**vyff**

Y-Component of the far-field velocity; by default it is 0.0.

**vzff**

Z-Component of the far-field velocity; by default it is 0.0.

**conv**

Units conversion factor for the pressure; it is used only in empirical or semi-empirical user's laws; by default it is 1.0.

**pref**

Reference pressure (see Note below), by default it is 0.0.

**rref**

Reference mass density to be used in the material law; by default, it is assumed  $rref=ro$ , i.e. the reference density is assumed equal to the initial density.

**rmin**

The minimum value of the density that can be accepted; by default it is 0.

**rmax**

The maximum value of the density that can be accepted; by default it is set equal to the 'grand' quantity (usually of the order of 1.E12, but the actual number depends on the computer).

**imin**

The minimum value of the specific internal energy that can be accepted; by default it is 0.

**imax**

The maximum value of the specific internal energy that can be accepted; by default it is set equal to the 'grand' quantity (usually of the order of 1.E12, but the actual number depends on the computer).

**rang**

An integer defining the type of check that should be performed. When 0 is used no range check is performed (this is the default); when 1 is specified, a warning message is issued each time the validity ranges are violated but, in order to reduce the number of messages, the current value of the range is updated each time a message is issued. The last messages will therefore show the absolute minimum or maximum out-of-range values that have been used in the calculation; when 2 is specified, a warning message is issued each time the validity ranges are violated, so there may be a lot of messages in practical cases; finally, when 3 is used an error message is given the first time ranges are violated, and the calculation is stopped immediately.

**gene**

Associate energy generation to this material: gene is the number of the function (see "FONC" directive) used to describe the variation in time of the specific power (i.e., per unit mass) that will be generated in this FLUT material.

**genm**

Associate mass generation to this material: genm is the number of the function (see "FONC" directive) used to describe the variation in time of the specific mass (i.e., per unit mass) that will be generated in this FLUT material.

**Additional data for JWLS material**

For the special case of a Jones-Wilkins-Lee (JWLS) material (NUM = 11), the user may specify a set of additional parameters that are detailed below. This material is suitable for the modelling of explosives and is similar to CEA's JWL/JWLS materials, but adapted for use with JRC's specialized fluid elements.

The JWL state equation gives the value of the pressure according to the following formula (see also page C.440) :

$$P = A\left(1 - \frac{\omega}{R_1 V}\right) e^{-R_1 V} + B\left(1 - \frac{\omega}{R_2 V}\right) e^{-R_2 V} + \omega \rho e_{int}$$

with :

$\rho$  : current density,

$e_{int}$  : current internal energy per unit mass,

$V$  : the ratio  $\frac{\rho_{sol}}{\rho}$  where  $\rho_{sol}$  is the density of the explosive (specific volume).

The release of the chemical energy can be controlled by the burn mass fraction. The burn mass fraction smears the detonation front over a certain number of time steps.

$$P = P_{EOS} \min(1, F_1)$$

with :

$$F_1 = \begin{cases} (t - t_1)d/(B_s \cdot l_e) & \text{for } t > t_1 \\ 0 & \text{for } t \leq t_1 \end{cases}$$

with :

$t_1$  : ignition time of the current element (calculated with the detonation velocity  $d$ ),

$B_s$  : controls the width of the burn wave

$l_e$  : average element length

**Syntax:**

```

< <"ROS" ros>  "A" a  "B" b
    "R1" r1 "R2" r2
    <"D" d  "XDET" xdet "YDET" ydet <"ZDET" zdet>>
    <"TDET" tdet> "PINI" pini <"BMF" bmf> >

```

**ros**

Density of the explosive in solid state (before detonation). If omitted, it is assumed  $ROS = RO$  i.e. the explosion starts from the solid state, whose density coincides with the initial density of the material ( $RO$  parameter given in the standard FLUT properties). In this case, it is mandatory to specify also the detonation speed  $D$  and the ignition point  $XDET$ ,  $YDET$  and  $ZDET$  from which the detonation starts.

**a**

Coefficient  $A$  of the JWL equation of state. It has the dimensions of a pressure.

**b**

Coefficient  $B$  of the JWL equation of state. It has the dimensions of a pressure.

**r1**

Coefficient  $R_1$  of the JWL equation of state. It has no dimensions.

**r2**

Coefficient  $R_2$  of the JWL equation of state. It has no dimensions.

**d**

Detonation speed. The detonation is supposed to start at point  $XDET$ ,  $YDET$  and  $ZDET$  and to propagate through the (solid) charge according to a spherical wave traveling with constant speed given by  $d$ .

**xdet, ydet, <zdet>**

Ignition point. If **zdet** is omitted, it is assumed equal to 0.

**tdet**

Starting time for the detonation. By default it is 0.

**pini**

Initial pressure (absolute) of the undetonated material.

**bmf**

Parameter for the burn mass fraction  $B_S$ . If the parameter equals 0 or is not defined, burn mass fraction is not activated. The suggested value is 2.5.

The parameter  $\omega$  appearing in the JWL equation of state is not given explicitly since  $\omega = \gamma - 1$  where  $\gamma = C_p/C_v$  is the ratio of specific heats that is provided by means of the **gamm** parameter.

Like for CEA's materials JWJ and JWLS it is possible to use this constitutive law to approximate the behaviour of a perfect gas. It is sufficient to give ROS and the initial values RO and EINT for the gas to find the corresponding pressure by the state equation. In this case PINI is not used by the model. In this case, it is useless to specify D, the coordinates of the ignition point and the starting time of the detonation.

## Comments

The meaning of the reference pressure *pref* is as follows. The pressure value *p* resulting from the material equation of state is considered as the absolute pressure. Its value is stored in ECR(1), see below. However, in order to compute the internal forces, the "effective" pressure value ( $p' = p - pref$ ) is used. This is useful when e.g. modeling a pressure vessel filled in by fluids and surrounded by the atmosphere in the outside part. In this case, if the material law for the inside fluid is given in absolute terms, we may proceed in two ways in order to take into account the outside pressure. The first possibility is to ignore *pref* (thus, *pref*=0.0) and explicitly impose an outside pressure of 1 atmosphere on the external surface of the model, either by the directive "CHAR" or by using specialized CLxx elements. However, this is likely to be too expensive for just representing a constant external pressure. The second possibility is to define *pref*=1 atmosphere for the internal fluids, so that only the pressure exceeding this value will be used in order to compute the forces.

If a material of this type is assigned to far-field boundary condition elements only, then all the above optional parameters are ignored if specified, except NUM, VXFF, VYFF and VZFF. On the other hand, VXFF, VYFF and VZFF are only used when the material is assigned to a far-field (CLxx) element.

The optional directives GENE and GENM allow to associate energy or mass generation, respectively, to a FLUT material. During the transient calculation, the generation "follows" this material, rather than being tied to a spatial zone (elements) like in the INJE injection model (see INJE directive). Another difference is that in the GENE model the given time function (FONC) represents the specific generated energy per unit time, and not the total energy per unit time like in the INJE model.

It should be noted that the behaviour of the GENE/GENM generation model will be different, depending on whether the associated FLUT material is a single-component, or part of a multi-phase multi-component (FLMP) material model.

In the first case, the generation will actually remain confined to the elements that are assigned the present material in the initial configuration, which is probably NOT what the user would expect (unless of course the generation zone boundary is Lagrangian, but in that case the generation model would coincide with the INJE model). This 'wrong' behaviour is due to the way the program deals with fluid transport across boundaries in the single-component model: even when fluid is transported each element retains its (initial) material index, so the generation information may not be transported to neighbours.

On the other hand, when the FLMP multiphase multicomponent model is used, then full tracking of each single component is performed, and therefore the generation model works as expected (generation information is transported to neighbour elements).

Mass generation requires some conventions as to what are the conditions of the injected mass, since this mass will also introduce corresponding energy, in the form of internal energy and possibly of momentum. Here it is assumed for simplicity that the mass generation occurs in the initial conditions of the corresponding material, and that the injection process does not perturbate the velocity field (i.e., the mass enters the element already at the current velocity).

### Outputs:

The components of the ECR table are as follows:

ECR(1): current element fluid absolute pressure  $p$

ECR(2): current element mass density

ECR(3): current element sound speed

ECR(4): current element specific internal energy

ECR(5): current element bulk modulus. Attention! Until recently this quantity was incorrectly indicated as bulk modulus in the code. In reality, this is the derivative of pressure with respect to density,  $dp/dr$ , which is related to the true bulk modulus  $B$  by the relation:  $B = (dp/dr) * r$ . Note that the quantity is only used (and updated at each time step) if the user selects the anti-hourglass model, i.e. specifies a non-zero value for the AHGF coefficient.

ECR(6): current element pseudoviscous pressure

ECR(7): current element minimum pressure flag (0 if  $p > p_{min}$ , 1 if  $p = p_{min}$ )

ECR(8): maximum pressure ever experienced by this element

ECR(9): minimum pressure ever experienced by this element

ECR(10): fraction of detonated material (for JWLS only) (0 or 1)

The components of the SIG table are as follows:

SIG(1): current element fluid relative pressure ( $p - p_{ref}$ )

The following global results can be accessed via TPLLOT when materials having either energy or mass generation are present in the calculation:

GENE: Total energy generated in materials with generation

GENM: Total mass generated in materials with generation

### Comparing FE and FV solutions.

Frequently it is desirable to compare solutions obtained with Finite Element (FE) models to equivalent ones obtained with Finite Volumes (FV), typically for the case of perfect gases.

In the case of MCGP material used for Finite Volumes (see page C.550), the equivalence of initial conditions is not completely straightforward.

A procedure for converting between the two formulations is detailed on Page C.550.

### Example of user-defined material subroutine.

The name of the subroutine is FLUTIL. The variables received in input and returned in output are explained in the comments.

```

C FLUTIL      SOUPLEX   ISPRA      89/04/19    21:02:26
      SUBROUTINE FLUTIL(NUM,rcur,rref,GAM,UP,PB,PMIN,conver,
        >                pres,SOUND,XKP)
C-----
C                                     F.CASADEI,J.P.HALLEUX 11-87
C USER'S EQUATION(S) OF STATE FOR FLUID MATERIAL(S) OF TYPE "FLUT"
C UPDATES FLUID PRESSURE, SOUND SPEED AND MINIMUM PRESSURE FLAG
C-----
c Input:
c =====
C   NUM      = NUMBER OF CURRENT "FLUT" MATERIAL
C   rcur      = CURRENT ELEMENT MASS DENSITY
C   rref      = reference MATERIAL MASS DENSITY
C   GAM       = num=1-8: CP/CV, RATIO OF MATERIAL SPECIFIC HEATS
c               num=9 : Bulk modulus (same units as pressure)
C   UP        = CURRENT ELEMENT SPECIFIC INTERNAL ENERGY
C   PB        = num=1-8: CONSTANT USED IN SOUND SPEED EVALUATION
c               num=9 : Reference pressure
c               num=10: Index of FONC defining the pressure p(t)
C   PMIN      = CUT-OFF PRESSURE
c   conver    = units conversion factor for the pressure
c Output:
c =====
C   pres      = NEW      ELEMENT FLUID PRESSURE: pres=pres(rcur,UP)
C   SOUND      = NEW      ELEMENT SOUND SPEED
C   XKP        = NEW      ELEMENT MINIMUM PRESSURE FLAG (0 IF P>PMIN,
C                                     1 IF P=PMIN)
C
      include 'R8AHOZ.INC'
c
      include 'ALLO.INC'
      include 'CONSTA.INC'
      include 'TEMPX.INC'
C
c=====
C EQUATIONS OF STATE
C
      GOTO(1,2,3,4,5,6,7,8,9,10),NUM
      CALL ERRMSS('FLUTIL',' FLUT MATERIAL NUMBER OUT OF RANGE')
      STOP
C
C 1/ PERFECT GAS -----
C
      1 pres=(GAM-1.D0)*rcur*UP
      GO TO 100
C
C 2/ IT8 LOW DENSITY EXPLOSIVE CHARGE -----
C   (pinc=[DYNES/CM2])

```

C

```

2 v=rref/rcur
  T1=1.7039D+11*(1.0D0-1.0D0/(90.0D0*V))*EXP(-9.0D0*V)
  T2=1.1595D+10*(1.0D0-1.0D0/(24.0D0*V))*EXP(-2.4D0*V)
  t3=-1.02884d9/v
  t4=0.1D0*rcur*up
  pinc=t1+t2+t3
  pres=conver*pinc+t4
  GO TO 100

```

C

C 3/ APRICOT-4 AND ITS LIQUID WATER -----

C (pinc=[DYNES/CM2])

C

```

3 v=rref/rcur
  if(rcur.eq.rref) then
    t1=0.0d0
  ELSE
    am=1.0D0-V
    a2=am*am
    a3=2.086D0*am
    a3m=a3-1.0D0
    a4=a3m*a3m
    a5=0.8293D0*a2
    a6=2.796D0*am
    a7=sqrt(a4+a5)
    om=(a3-1.0D0+a7)/a6
    a8=(0.1483D0+2.086D0*om-1.398D0*om*om)*om
    a9=1.0D0-0.14D0*am/v
    t1=1.0D12*a9*a8
  END IF
  t2=0.28d0*rcur*up/v
  pinc=t1
  pres=conver*pinc+t2
  GO TO 100

```

C

C 4/ CONT PROBLEM EXPLOSIVE BUBBLE -----

C (pinc=[Pa])

C

```

4 VV=rcur/rref
  IF(VV.GT.0.0D0) THEN
    pinc=1.0D7*VV**GAM
  ELSE
    pinc=0.0D0
  ENDIF
  pres=conver*pinc
  GO TO 100

```

C

C 5/ CONT PROBLEM LIQUID SODIUM AT 773 K AND ABOUT 10 MPA -----

C (pinc=[Pa])

C

```

5 VV=rcur/rref

```



```

      AMU=VV-1.0D0
      T1=4.440D3*AMU
      T2=4.328D9*AMU*ABS(AMU)
      T3=1.218D0*rcur*UP*(1.D0+AMU)
      pinc=t1+t2
      pres=conver*pinc+t3
      GO TO 100
c
C 6/ APRICOT-4 EXPLOSIVE GAS PRODUCTS -----
C   (pinc=[DYNES/CM2])
c
      6 v=rref/rcur
      t1=6.70695d12*(1.0d0-0.25d0/(4.660599d0*v))*exp(-4.660599d0*v)
      t2=9.26460d10*(1.0d0-0.25d0/(0.991617d0*v))*exp(-0.991617d0*v)
      t3=0.25d0*rcur*up/v
      pinc=t1+t2
      pres=conver*pinc+t3
      go to 100
c
C 7/ WTO LOW-DENSITY EXPLOSIVE CHARGE -----
C   (pinc=[DYNES/CM2])
c
      7 v=rref/rcur
      t1=1.7039D11*(1.d0-1.d0/(90.0d0*v))*exp(-9.0d0*v)
      t2=1.1595D10*(1.d0-1.d0/(24.0d0*v))*exp(-2.4d0*v)
      t3=0.1d0*rcur*up
      pinc=t1+t2
      pres=conver*pinc+t3
      GO TO 100
c
C 8/ WTO LIQUID WATER -----
C   (pinc=[DYNES/CM2])
c
      8 vv=rcur/rref
      if(vv.ge.1.d0) then
        t1=1.2222D11*vv*vv+5.1562D10-(1.2222D11+5.1562D10)*vv**1.28D0
      else
        t1=-1.2222D11*vv*vv-8.5937D10+(1.2222D11+8.5937D10)*vv**1.28D0
      endif
      pinc=t1
      pres=conver*pinc
      GO TO 100
c
C 9/ Liquid with bulk response -----
c
      9 v=rref/rcur
      eta=1.0d0-v
      p0=pb
      bulk=gam
      pres=p0+bulk*eta
      go to 100

```

```
c
C 10/ Imposed pressure -----
c
  10 ifonc=pb
    tcur=t
    call ffonct(ifonc,tcur,valfon,a(n91),a(n92))
    pres=valfon
    go to 100
C
C=====
C PRESSURE CUT-OFF TEST
C
  100 IF(pres.LE.PMIN) THEN
    pres=PMIN
    XKP=1.D0
  ELSE
    XKP=0.D0
  ENDIF
C
C=====
C SOUND SPEED
C
    goto(101,101,101,101,101,101,101,101,102,103),num
c
c materials 1-8
c
  101 SOUND=sqrt(GAM*(pres+PB)/rcur)
    go to 999
c
c material 9
c
  102 sound=sqrt(bulk/rcur)
    go to 999
c
c material 10
c
  103 sound=zero
    go to 999
C
  999 RETURN
    END
```

### 7.8.31 MATERIAL FOR MINERAL OIL PYROLISIS

#### Object

This material is used to simulate mineral oil pyrolysis phenomena subsequent to electrical arcs in oil-filled electrical apparatuses (e.g. transformers).

The material is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC.

Rather than a new material, it is an extension of the user-defined FLUT type material (see Page C3.520). The extension is introduced by the keyword "PYRO", as explained in the syntax below.

#### References

More information on the formulation of this material model may be found in references [106, 160].

#### Syntax

```
"FLUT"  FLUT_material_data (see P. C.520)
  < "PYRO" "NC" nc "EACT" eact "RGAS" rgas "TREF" tref
      "ROIL" roil "KOIL" koil "TOIL" toil "PINI" pini
      "TINI" tini "QTAB" qtab "DHRO" dhro
      "NGAS" ngas
      ngas * ( "GAS" 'nomgas' "STEC" stec "PMOL" pmol
                "DHR" dhr "CP" cp )
      "CARB" 'nomcar' "STEC" stec "PMOL" pmol
                "DHR" dhr "CP" cp
  >

/LECTURE/
```

**nc**

Number of carbon atoms in oil molecule.

**eact**

Activation energy.

**rgas**

Gas constant.

**tref**

Reference temperature.

roil

Liquid oil density.

koil

Liquid oil thermal conductivity.

toil

Liquid oil temperature.

pini

Initial pressure of the bubble.

tini

Initial temperature of the bubble.

qtab

Index of "FONCTION" for electrical power. This function will describe the power as a function of time.

dhro

Liquid oil enthalpy of formation.

ngas

Number of gas products.

nomgas

Name of gas product (8 chars max.).

stec

Stechiometric coefficient for this product in the pyrolysis reaction.

pmol

Molar weight of the product.

dhr

Molar enthalpy of formation of the product.

cp

Heat capacity at constant pressure of the product.

nomcar

Name of the carbon product.

**Comments:**

This model is still under development and testing and should therefore be used with great care.

Do not forget to dimension adequately, see keyword "PYRO", page A.70. Currently there may be up to 4 distinct pyrolysis bubbles in a calculation.

### 7.8.32 ADVECTION-DIFFUSION FLUID (ADFM)

**Object:**

This material is used with specialised elements of type "ADC8" or "ADQ4", for advection-diffusion calculations.

**Syntax:**

```
"ADFM" ![ "RO" rho  "EXPA" expa  "TREF" tref  
          < "COND" cond  "CAPA" capa  "VISC" visc  
            "LAPI" lapi                                     > ]!  
/LECTURE/
```

rho

Initial mass density.

expa

Volumetric expansion coefficient.

tref

Reference temperature at which volumetric expansion is null.

cond

Thermal conductivity. By default, is f(T).

capa

Thermal capacity. By default is f(T).

visc

Dynamic viscosity. By default is f(T).

lapi

Lapudus viscosity. Default is 0.0.

**Comments:**

If omitted, thermal conductivity, capacity and viscosity are assumed to be temperature dependent. The user must insuch cases supply routines that return temperature dependent values.

**7.8.33 MULTICOMPONENT FLUID MATERIAL (MCGP)****Object:**

This material is used with specialised elements of type MCxx for multicomponent fluid flows. A mixture of calorically perfect gases is assumed, i.e. the internal energy is a function of the temperature only. This function may be different for each component and has a generic polynomial form.

The contribution of the CESI team (formerly at ENEL, Milano) to the development of this material model in collaboration with JRC is gratefully acknowledged.

**Syntax:**

```
"MCGP"  "NCOM" ncom  "R" r
          ncom * ( "COMP" 'nomcomp'
                   "PM" pm
                   "CV1" cv1 "CV2" cv2 "CV3" cv3 )
          /LECTURE/
```

**ncom**

Number of components in this material.

**r**

Universal constant of gases ( $R$ ), must be expressed in J/(kmol K).

**nomcomp**

Name of the component (max 8 characters) in quotes.

**pm**

Molecular weight  $w$  of the component, in kg/kmol.

**cv1/2/3**

Coefficients of the expression of the specific heat at constant volume ( $c_v$ ), which must be expressed in J/kmolK, as a function of absolute temperature  $T$ :  $c_v(T) = c_{v1} + c_{v2}T + c_{v3}T^2$ . The specific internal energy  $i$  results from  $i(T) = c_v T = c_{v1}T + c_{v2}T^2 + c_{v3}T^3$ . Note that  $i$  must be expressed in J/kmol.  $T$  is the temperature in K. Consequently,  $c_{v1}$  must be in J/kmolK,  $c_{v2}$  must be in J/kmol(K2) and  $c_{v3}$  must be in J/kmol(K3).

**Comments:**

Only one material of type multicomponent fluid is allowed in a model. This is not a restriction, since the number of components is arbitrary and component mass fractions can be locally zero.

The polytropic exponent  $\gamma = c_p/c_v = (R + c_v)/c_v$ , is determined by the code. Therefore, note that the units of the various  $c_v$  coefficients in the above expression for  $i$  should be consistent with the units used for  $R$ . For example,  $c_{v1}$  should be expressed in J/(kmol K).

Also the density is determined by the code.

### Comparing FE and FV solutions.

Frequently it is desirable to compare solutions obtained with Finite Element (FE) models to equivalent ones obtained with Finite Volumes (FV), typically for the case of perfect gases. In the case of FLUT material used for Finite Elements (see page C.520), the equivalence of initial conditions is not completely straightforward. A procedure for converting between the two formulations is detailed hereafter.

The two fluid solvers have completely different ways to carry out the numerical discretization of the same governing equations (Euler equations); each of them has its specific formulation, its own set of variables and its own parameters, whose value has to be assigned in the input data.

More in detail, in the FE model the perfect gas state equation used to close the system of Euler equations has the form:

$$p = (\gamma - 1)\rho i,$$

where  $p$  is the pressure (Pa),  $\rho$  is the density (Kg/m<sup>3</sup>),  $i$  is the internal energy per unit mass (J/Kg) and  $\gamma$  (-) is the ratio between the constant pressure and constant volume specific heats  $c_p$  (J/kmolK) and  $c_v$  (J/kmolK). The user must in this case provide in input the values of  $\gamma$ ,  $\rho$  and  $i$  (see MATE FLUT on Page C.520).

In the FV model the same state equation takes the form:

$$p = RT \frac{\rho}{w},$$

where  $R$  is the universal constant of gases (J/kmolK), which has the standard value of 8314.3,  $T$  is the absolute temperature (K) and  $w$  is the molar weight (kg/kmol). Note that the state equation in the FV model could be more complex, taking into account a more general mixture of Joule gases. We consider here a single-component perfect gas for simplicity. The user must in this case provide in input the values of  $R$ ,  $c_v$  and  $w$  for the material (see MATE MCGP above), plus the initial values of  $p$  and  $T$  at each node (and thus at each finite volume), via the directive INIT MCOM (see Page E.150).

Switching from FV to FE, an equivalent input can be obtained readily from the identities:

$$\gamma = \frac{R}{c_v} + 1 \quad i = \frac{c_v}{w} T \quad \rho = \frac{wp}{RT}$$



The inverse path is not so straightforward. The switch from FE to FV is not univocally determined unless the molar weight  $w$  is known. Indeed the physics of the problem only depends on the internal energy  $i$  (see above), which is proportional to the ratio  $T/w$  by means of the value:

$$c_v = \frac{R}{\gamma - 1}$$

Then it is possible to choose any couple  $T$  and  $w$  so as to have the appropriate  $i$ , but values of temperature would in general *not* be correct during a calculation.

As an example, consider the following set of initial conditions, which have been chosen without actual physical relevance and have been rounded in order to easily check the equivalence of the several parameters values in the FE and FV representation.

Assume we want to simulate two perfect gases, one initially at high pressure and the other initially at low pressure. Let  $\gamma = 1.5$  and  $\rho = 2 \text{ kg/m}^3$  for both gases. If the HP-gas has an initial pressure of  $p_H = 5.E5 \text{ Pa}$ , then we get from the equation of state in FE form:  $i_H = p_H/(\gamma - 1)\rho = 5.E5 \text{ J/kg}$ . Similarly, for the LP-gas at, say,  $p_L = 1.E5 \text{ Pa}$  we obtain  $i_L = p_L/(\gamma - 1)\rho = 1.E5 \text{ J/kg}$ . These values completely define the FE material data.

To get an equivalent FV description, we must provide the constant of perfect gases, which in standard units is about  $R = 1.E4 \text{ J/kmolK}$ , and we must choose a molar weight, say  $w = 20 \text{ kg/kmol}$  for both gases. Then we obtain the specific heat at constant volume (same for both gases) from the relation  $c_v = R/(\gamma - 1) = 2.E4 \text{ J/kmolK}$ .

Assuming for both gases the same initial density  $\rho = 2 \text{ kg/m}^3$  as in the FE case, we may compute the temperature from the relation  $T = wp/\rho R$ . For the H-P gas ( $p_H = 5.E5 \text{ Pa}$ ) this gives  $T = 500 \text{ K}$ , while for the L-P gas ( $p_L = 1.E5 \text{ Pa}$ ) this gives  $T = 100 \text{ K}$ .

### 7.8.34 MULTICOMPONENT FAR-FIELD FLUID MATERIAL (MCFF)

#### Object:

This material is used with specialised elements of type CL22, CL3I or CL3Q to specify far-field conditions of multicomponent flows. It specifies the constant physical state:  $\rho(1)$ , ...  $\rho(\text{ncom})$ ,  $\rho u$ ,  $\rho v$ ,  $\rho w$ ,  $\rho E$ , assumed outside the discretized fluid domain.

Unlike material MCGP, which must be unique in a single calculation, an arbitrary number of MCFF materials is allowed. However, note that the declaration of the MCGP material MUST precede the declaration of the MCFF(s) in the input data set.

The contribution of the CESI team (formerly at ENEL, Milano) to the development of this material model in collaboration with JRC is gratefully acknowledged.

#### Syntax:

```
"MCFF" "BDF0" bdfo "TEMP" temp "PRES" pres "VEL1" vel1
      "VEL2" vel2 "VEL3" vel3
      ncom * ( "COMP" 'nomcomp' "MFRA" mfra )
      /LECTURE/
```

**bdfo**

Option for boundary flux: 1 = Roe, 2 = Van Leer, 3 = Steger-Warming. Recall that the flux type in the bulk fluid is chosen (independently from the boundary flux type) by directive OPTI MC NUFL.

**temp**

Temperature of the far-field state.

**pres**

Pressure of the far-field state.

**vel1**

X-velocity of the far-field state.

**vel2**

Y-velocity of the far-field state.

**vel3**

Z-velocity of the far-field state.

**ncom**

Number of components (must be the same as for the MCGP material).

**nomcomp**

Name of the component (max 8 characters) in quotes, must be spelled exactly as in the definition of the MCGP material.

**mfra**

Mass fraction of the component.

**Comments:**

The key-words **TEMP ... VEL3** must precede the declaration of the mass fractions **COMP ... MFRA** in the input data set.

### 7.8.35 MULTIPHASE MULTICOMPONENT FLUID MATERIAL (FLMP)

**Object:**

This material is used with FLxx elements to describe multi-phase multicomponent fluid flows. In this formulation, more than one fluid material (liquids, perfect gases) may be present at the same time inside a generic finite element. The material is treated as a homogeneous mixture of the component fluids.

The velocity field is unique (i.e., all components have the same velocity). The pressure is defined at the element level as follows: if one or more liquids are present, they are subjected to the same pressure. If more than one gas is present, the gases in the mixture follow Dalton's law: the sum of the partial pressures of the gas components equals the pressure of the mixture.

This 'material' (type FLMP) is composed of several FLUT materials (see page C3.520).

Note: this material is still under development and testing. It has to be used with great care.

**Syntax:**

```
"FLMP" "NLIQ" nliq "NGAS" ngas
      nliq * ( "FLUT" liquid_material_description )
      ngas * ( "FLUT" gas_material_description      )
```

nliq

Total number of liquid materials in the mixture.

ngas

Total number of gas materials in the mixture.

**Comments:**

The liquid materials must precede the gas materials.

The program considers the first nliq FLUT-material descriptions encountered in the input file after the "FLMP" directive as descriptions of the liquid materials, and the successive ngas FLUT-material descriptions as descriptions of the gas materials.

The elements to which each FLUT material is associated are specified as usual via a /LECTURE/ directive at the end of each FLUT-material description. As a consequence, each element containing a FLMP mixture will effectively contain only one component (with a 100 per cent mass fraction) at the initial time. Because of this, it is not possible to effectively have more than one material in any element in the initial conditions. However, during the transient analysis the materials will mix up because of transport between adjacent elements, thus forming the mixture.

**Outputs:**

The components of the ECR table are as follows:

Positions 1-9 are equivalent to those of the FLUT material:

ecr(1): current element pressure of the fluid mixture

ecr(2): current AVERAGE density of the fluid mixture

ecr(3): current sound speed of the fluid mixture

ecr(4): current AVERAGE specific internal energy of the fluid mixture

ecr(5): current AVERAGE bulk modulus of the fluid mixture

ecr(6): current pseudoviscous pressure of the fluid mixture

ecr(7): current minimum pressure flag of the fluid mixture

ecr(8): maximum pressure ever experienced by the element

ecr(9): minimum pressure ever experienced by the element

ecr(10): number of effective components in the element

Then, for each component icom of the mixture:

`iad=nfixmp+(icom-1)*necrmp (see FLUTMP.INC)`

ecr(iad+1): current relative mass fraction of the component

ecr(iad+2): current density of the component

ecr(iad+3): current specific internal energy of the component

ecr(iad+4): current partial pressure of the component

ecr(iad+5): mass fraction of the component at the end of Lagrangian phase

ecr(iad+6): specific internal energy of the component at the end of Lag. ph.

There may be at most 4 different components in a FLMP material, at present. Thus, there is place for up to  $(10 + 6 \times 4) = 34$  components of ECR at each Gauss point of an element with a FLMP material.

**7.8.36 SG2P—Multicomponent Stiffened Gases - Fully Conservative Formulatoin****Object:**

Modeling of multicomponent flows involving extended Stiffened Gas Equation of State (EoS). Each component of the mixture is described by an Stiffened Gas EoS:

$$P = (\gamma - 1)\rho(e - q) - \gamma P_{inf}$$

Where  $e$  is the internal energy per unit mass,  $\rho$  the density.  $\gamma$  is an empirical constant for liquids.  $P_{inf}$  is a constant representing the molecular attraction between molecules (liquid) and  $q$  is an additional constant. This expression is identical to the ideal gas EoS when  $P_{inf}$  and  $q$  is zero. The governing equations are the Euler Equations in conservative form and a transport equation for the volume fraction (For more details see also [894]).

**Syntax:**

```
"SG2P"  "PINI"  pini  "PREF" pref  "PMIN" pmin
...     "NESP"  nesp

...

...     "COMP1"
...     "ROI"   roi
...     "GAMM" gamm  "PI" pinf  "Q"  q  "ALPH" alpha
...     "CP"   cp   "CV"  cv   "QPRI" qprim

...

...     "COMPnesp"
...     "ROI"   roi
...     "GAMM" gamm  "PI" pinf  "Q"  q  "ALPH" alpha
...     "CP"   cp   "CV"  cv   "QPRI" qprim

... /LECTURE/
```

**pini**

Initial pressure of the mixture.

**pref**

Reference pressure.

**pmin**

Absolute minimum pressure **pmin** in the fluid.

**nesp**

Number of components involved in the mixture

**"COMP1", ..., "COMPnesp"**

Keywords which state that we will describe the properties of the components 1, ..., nesp.

**roi**

initial density of the component

**gamma**

Ratio  $c_P/c_V$  (supposed constant) for gases and an empirical constant for liquid.

**pinf**

Constant parameter for liquid to take into account molecular attraction between molecules.

**q**

Internal energy of the fluid at a given reference state (most time one take  $q = 0$ ).

**alpha**

Initial volume fraction of the component

**cp**

Constant pressure specific heat

**cv**

Constant volume specific heat

**qprim**

parameter used in the determination of the entropy and free Gibbs energy of each component in case of phase change (Under Development !).

**/LECTURE/**

List of the elements concerned.

**Comments:**

The reference pressure `pref` enables the initial state to be defined. If `pref = pini`, the gas is in equilibrium just before the computation starts; it will be perturbed by an external action, by the motion of a piston, for instance. If `pref = 0`, the problem consist in a computation with initial stresses determined by `pini`. This is the case when a membrane which was separating two gases at different states disappears at the initial instant.

The sum of all initial volume fractions of the different components should be equal to 1 .

### Outputs:

Then the different components of the ECR table are:

ECR(1) : pressure

ECR(2) : density of the mixture

ECR(3) : sound speed of the mixture

ECR(4) : free

ECR(4 + iesp) : mass fraction of the iesp-th component

### Example:

We consider a mixture of three stiffened gases in a box The initial configuration consists to a 1 m × 1 m square box filled with a gas at rest at atmospheric pressure containing two concentric bubbles which are at rest as well, with different pressure and physical characteristics. The radius of the bubbles are 0.1 m and 0.25 m, respectively. The three different gases are denoted 1 (zone SS1) in the box, 2 in the middle bubble (ZONE SPH1) and 3 in the smaller one (ZONE SPH2) and are modeled as perfect gases ( $p_{inf} = 0$  and  $q = 0$  in EoSs). For this example, more details can be found in [894].

The initial conditions are given:

$$\left\{ \begin{array}{lll} \gamma = 1.4, & \rho = 1.376363 \text{ kg.m}^{-3}, & p = 10^5 \text{ Pa} & \text{for gas 1 in the box} \\ \gamma = 1.667, & \rho = 0.192 \text{ kg.m}^{-3}, & p = 10^6 \text{ Pa} & \text{for gas 2 in the middle bubble} \\ \gamma = 1.249, & \rho = 3.1538 \text{ kg.m}^{-3}, & p = 5 \times 10^5 \text{ Pa} & \text{for gas 3 in the smaller bubble} \end{array} \right.$$

```
*
*                               GAS IN THE BOX
*                               -----
*
*    SG2P
*
*    PINI 1.E5 PMIN 1e-3 PREF 1E5 NESP 3
*
*    COMP1
*    COMP1
*    ROI  1.376363
*    PI   0
```



```
GAMM 1.4
ALPH 1
Q    0
CP   0
CV   0
QPRI 0

*   COMP2
    COMP2
    ROI  0.192
    PI   0.
    GAMM 1.667
    ALPH 0
    Q    0
    CP   0
    CV   0
    QPRI 0

*   COMP3
    COMP3
    ROI  0.192
    PI   0.
    GAMM 1.249
    ALPH 0
    Q    0
    CP   0
    CV   0
    QPRI 0

*
                                LECT SS1 TERM

*
*   GAS IN THE SMALLER BUBBLE
*   -----
*
*   SG2P

*   PINI 5E5 PMIN 1E-3 PREF 1E5 NESP 3

*   COMP1
    COMP1
    ROI  1
    PI   0.
    GAMM 1.4
    ALPH 0
    Q    0
    CP   0
    CV   0
    Qpri 0

*   COMP2
    COMP2
```

```
ROI 0.192
PI 0.
GAMM 1.667
ALPH 0
Q 0
CP 0
CV 0
Qpri 0

* COMP3
COMP3
ROI 3.1538
PI 0.
GAMM 1.249
ALPH 1
Q 0
CP 0
CV 0
QPRI 0

LECT SPH2 TERM

*
* GAS IN THE MIDDLE BUBBLE
* -----
*
SG2P
*
PINI 10E5 PMIN 1E-3 PREF 1E5 NESP 3

* COMP1
COMP1
ROI 0.181875
PI 0.
GAMM 1.4
ALPH 0
Q 0
CP 0
CV 0
Qpri 0

* COMP2
COMP2
ROI 0.192
PI 0.
GAMM 1.667
ALPH 1
Q 0
CP 0
CV 0
Qpri 0

* COMP3
```

COMP3  
ROI 0.192  
PI 0.  
GAMM 1.249  
ALPH 0  
Q 0  
CP 0  
CV 0  
QPRI 0

\*

LECT SPH1 TERM

### 7.8.37 SGMP—Multicomponent Stiffened Gases models

#### Object:

Modeling of multicomponent flows involving extended Stiffened Gas Equation of State (EoS). Each component of the mixture is describe by an Stiffened Gas EoS:

$$P = (\gamma - 1)\rho(e - q) - \gamma P_{inf}$$

Where  $e$  is the internal energy per unit mass,  $\rho$  the density.  $\gamma$  is an empirical constant for liquids.  $P_{inf}$  is a constant representing the molecular attraction between molecules (liquid) and  $q$  is an additional constant. This expression is identical to the ideal gas EoS when  $P_{inf}$  and  $q$  is zero. The governing equations are the Euler Equations in conservative form and a transport equation for the volume fraction (For more details see also [894, 850, 897]).

#### Syntax:

```
"SGMP"  "PINI"  pini  "PREF" pref  "PMIN" pmin
...     "NESP"  nesp  <"MODE" imod>

...

...     "COMP1"
...     "ROI"   roi
...     "GAMM"  gamm  "PI"  pinf  "Q"  q  "ALPH" alpha
...     "CP"    cp  "CV"   cv  "QPRI" qprim

...

...     "COMPnesp"
...     "ROI"   roi
...     "GAMM"  gamm  "PI"  pinf  "Q"  q  "ALPH" alpha
...     "CP"    cp  "CV"   cv  "QPRI" qprim

... /LECTURE/
```

**pini**

Initial pressure of the mixture.

**pref**

Reference pressure.

**pmin**

Absolute minimum pressure **pmin** in the fluid.

**nesp**

Number of components involved in the mixture

**imod**

Type of model used

1 multicomponents ALE extension of the two-phase flow model of Allaire et al. designed for interface problems (default value).

2 Multicomponents ALE extension of the two-phase flow model of Kapila et al.

**"COMP1", ..., "COMPnesp"**

Keywords which state that we will describe the properties of the components  $1, \dots, nesp$ .

**roi**

initial density of the component

**gamma**

Ratio  $c_P/c_V$  (supposed constant) for gases and an empirical constant for liquid.

**pinf**

Constant parameter for liquid to take into account molecular attraction between molecules.

**q**

Internal energy of the fluid at a given reference state (most time one take  $q = 0$ ).

**alpha**

Initial volume fraction of the component

**cp**

Constant pressure specific heat

**cv**

Constant volume specific heat

**qprim**

parameter used in the determination of the entropy and free Gibbs energy of each component in case of phase change (Under Development !).

/LECTURE/

List of the elements concerned.

### Comments:

Theses models are **Only available** with the HLLC Solver (Option FCONV 6 which is the default value)

The reference pressure `pref` enables the initial state to be defined. If `pref = pini`, the gas is in equilibrium just before the computation starts; it will be perturbed by an external action, by the motion of a piston, for instance. If `pref = 0`, the problem consist in a computation with initial stresses determined by `pini`. This is the case when a membrane which was seperating two gases at different states disappears at the initial instant.

The sum of all initial volume fractions of the differents components should be equal to 1 .

Each phase is governed by its own EOS allowing the determination of the speed of sound of each phase:  $c_k^2 = (p/\rho_k^2 - \partial_{\rho_k} e_k) (\partial_p e_k)^{-1}$ .

For model 1 the mixture sound speed is some kind of average of the phase speed of sound (often call frozen sound speed [850]) and is designed for interface problems. For model 2 the mixture sound speed  $\check{c}$  obeys the Wood formula [850]:

$$\frac{1}{\rho \check{c}^2} = \frac{\alpha_1}{\rho_1 c_1^2} + \frac{\alpha_2}{\rho_2 c_2^2} \quad (33)$$

### Outputs:

Then the different components of the ECR table are:

ECR(1) : pressure

ECR(2) : density of the mixture

ECR(3) : sound speed of the mixture

ECR(3 + `iesp`) : mass fraction of the `iesp`-th component

ECR(3 + `nesp` + `iesp`) : volume fraction of the `iesp`-th component

ECR(3 + 2\*`nesp` + `iesp`) : temperature of the `iesp`-th component (for phase change : Under Development)

ECR(3 + 3\*`nesp` + `iesp`) : free Gibbs energy of the `iesp`-th component (for phase change : Under Development)

### Example:

We consider a mixture of three stiffened gases in a box The initial configuration consists to a 1 m × 1 m square box filled with a gas at rest at atmospheric pressure containing two concentric bubbles which are at rest as well, with different pressure and physical characteristics. The radius of the bubbles are 0.1 m and 0.25 m, respectively. The three different gases are denoted 1 (zone SS1) in the box, 2 in the middle

bubble (ZONE SPH1) and 3 in the smaller one (ZONE SPH2) and are modeled as perfect gases ( $pinf = 0$  and  $q = 0$  in EoSs). For this example, more details can be found in [894].

The initial conditions are given:

$$\begin{cases} \gamma = 1.4, & \rho = 1.376363 \text{ kg.m}^{-3}, & p = 10^5 \text{ Pa} & \text{for gas 1 in the box} \\ \gamma = 1.667, & \rho = 0.192 \text{ kg.m}^{-3}, & p = 10^6 \text{ Pa} & \text{for gas 2 in the middle bubble} \\ \gamma = 1.249, & \rho = 3.1538 \text{ kg.m}^{-3}, & p = 5 \times 10^5 \text{ Pa} & \text{for gas 3 in the smaller bubble} \end{cases}$$

```

*
*                               GAS IN THE BOX
*                               -----
*
*   SGMP
*
*       PINI 1.E5 PMIN 1e-3 PREF 1E5 NESP 3 MODEL 1
*
*   COMP1
*   COMP1
*   ROI  1.376363
*   PI   0
*   GAMM 1.4
*   ALPH 1
*   Q    0
*   CP   0
*   CV   0
*   QPRI 0
*
*   COMP2
*   COMP2
*   ROI  0.192
*   PI   0.
*   GAMM 1.667
*   ALPH 0
*   Q    0
*   CP   0
*   CV   0
*   QPRI 0
*
*   COMP3
*   COMP3
*   ROI  0.192
*   PI   0.
*   GAMM 1.249
*   ALPH 0
*   Q    0
*   CP   0
*   CV   0
*   QPRI 0
*
*
```

## LECT SS1 TERM

```
*
*
*      GAS IN THE SMALLER BUBBLE
*      -----
*
*      SGMP
*
*      PINI 5E5 PMIN 1E-3 PREF 1E5 NESP 3 MODEL 1
*
*      COMP1
*      COMP1
*      ROI  1
*      PI   0.
*      GAMM 1.4
*      ALPH 0
*      Q    0
*      CP   0
*      CV   0
*      Qpri 0
*
*      COMP2
*      COMP2
*      ROI  0.192
*      PI   0.
*      GAMM 1.667
*      ALPH 0
*      Q    0
*      CP   0
*      CV   0
*      Qpri 0
*
*      COMP3
*      COMP3
*      ROI  3.1538
*      PI   0.
*      GAMM 1.249
*      ALPH 1
*      Q    0
*      CP   0
*      CV   0
*      QPRI 0
```

## LECT SPH2 TERM

```
*
*
*      GAS IN THE MIDDLE BUBBLE
*      -----
*
*      SGMP
*
*      PINI 10E5 PMIN 1E-3 PREF 1E5 NESP 3 MODEL 1
*
*      COMP1
```



COMP1  
ROI 0.181875  
PI 0.  
GAMM 1.4  
ALPH 0  
Q 0  
CP 0  
CV 0  
Qpri 0

\* COMP2  
COMP2  
ROI 0.192  
PI 0.  
GAMM 1.667  
ALPH 1  
Q 0  
CP 0  
CV 0  
Qpri 0

\* COMP3  
COMP3  
ROI 0.192  
PI 0.  
GAMM 1.249  
ALPH 0  
Q 0  
CP 0  
CV 0  
QPRI 0

\*

LECT SPH1 TERM

### 7.8.38 BUBBLE MODEL

#### Object :

This directive simulates an explosion in air. It allows to load a structure without having to model the explosive as an initially solid material which changes phase to gas. Instead of the solid explosive a compressed bubble is used. The overpressure of this bubble is automatically calculated depending on the mass of the charge and the volume of the bubble.

The model can be used with either FE or VFCC models of the fluid domain. The BUBB material is automatically mapped to either GAZP or FLUT material, as appropriate. GAZP is used if the bubble and its neighbours are discretized by either VFCC (CUVF etc.) or CEA's fluid FE (CUBE etc.), while FLUT is used if the bubble and its neighbors are discretized by JRC's fluid FE (FL38 etc.).

#### Syntax:

```
"BUBB" "MASS" m /LECTURE/
```

m

Mass of the explosive in kilograms.

```
/LECT/
```

Elements concerned (bubble volume).

#### Comments :

The mass of the explosive is always the mass of the bubble as defined by the concerned elements. This has to be taken into account in simulations with symmetry conditions or in a case of a hemispherical explosion. At a border with symmetry, also the charge has to be cut. At a border of a hemispherical model, the charge must not be cut.

The BUBB material adopts the same material parameters as the fluid elements (or finite volumes) neighboring the bubble zone. Therefore the material for the neighbors has to be defined *before* the BUBB material itself.

#### Outputs:

The different components of the ECR table are the components of the GAZP or FLUT material to which the BUBB material is mapped. This depends on the elements used to discretize the fluid bubble: GAZP is used by either VFCC or CEA's fluid FE, while FLUT is used by JRC's fluid FE.

### 7.8.39 CDEM—Discrete Equation Method for Combustion

#### Object:

Extension of the Reactive Discrete Equation Method (RDEM method) proposed by Le Métayer *et al.* (see: O. Le Métayer, J. Massoni, R. Saurel. Modelling evaporation fronts with reactive Riemann solvers. *Journal of Computational Physics* **205**: 567–610, 2005) to the combustion case [844, 845, 862, 864, 892, 893], with the purpose of propagating a combustion front.

The combustion is supposed irreversible and complete; then at each point we can have either the burnt gas (or **burnt “phase”**) or the unburnt gas (or **unburnt “phase”**), interacting with each other only via the chemical reaction occurring at the combustion front. Then in the exact solution the fraction volume of each phase is either 0 or 1, presenting a discontinuity at the combustion front. But for numerical reasons both phases are present in each cell at any moment, with each phase occupying a certain volume fraction of the cell (of course the two volume fractions sum up to 1.0).

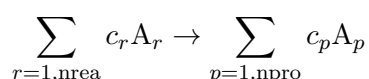
Each of the two phases has its own density, pressure, velocity and temperature, independent of the values for the other phase (the only coupling being the chemical reaction at the combustion front). Global values of pressure, density, temperature and velocity for the mixture of the two phases (i.e. for the cell) are computed as weighted averages of the values of the two phases, the weights being the respective volume fractions in the cell. This way of computing global values is consistent with the fact that in the exact solution only one phase is present at each point.

Note that this approach is radically different from the one used in other “multi-material” models available in the code. For example, in the multi-phase multi-component (non-reactive) fluid material model FLMP which can be used with fluid Finite Elements (not VFCCs), see [GBC\\_0570](#), in case of a mixture of only gases each component is assumed to occupy the whole volume of the cell, so that the global pressure and density are the sums (not the weighted averages) of the values for each component, and moreover all components are supposed to have the same temperature (thermal equilibrium) and the same velocity.

In the CDEM model each phase is a mixture of different gases, i.e. of different chemical species. If the concentration of species is exactly stoichiometric, then the unburnt phase contains only the reactants and the inerts (if any), while the burnt phase contains only the reaction products and the inerts (if any). However, the model is fully general and even non-stoichiometric concentrations can be considered, so that in general each phase may contain all the involved species (reactants, reaction products and inerts, if any).

Note that in general it is necessary to take into account the presence of inerts. Although they do not participate in the chemical reaction, their mass fraction can be important (consider e.g. nitrogen in the case of air) so that their heat capacity gives an important contribution to the global heat balance of the mixture.

The combustion is governed by an **irreversible exothermic** chemical reaction. Usually (in chemistry) this is written as follows, i.e. with reactants only on the left hand side and with reaction products only on the right hand side:



where **nrea** is the number of chemical species on the left-hand side of the chemical reaction (i.e., the number of reactants), while **npro** is the number of chemical species on the right hand side of the chemical reaction (i.e. the number of reaction products). Any inerts, which can be present in the mixture, are usually not explicitly included in the expression of the chemical reaction. Thus, the  $c_r$  and  $c_p$  coefficients in the above expression are strictly positive.

In the CDEM model this expression is re-arranged as follows, by bringing all terms to the left hand side and by letting also the inerts (if any) appear:

$$\sum_{i=1, \text{nesp}} c_i A_i \rightarrow 0$$

where **nesp** is the total number of chemical species present (i.e. the reactants, the reaction products and the inerts, if any): **nesp** = **nrea** + **npro** + **nine**. The coefficient  $c_i$  is thus strictly positive for reactants, strictly negative for reaction products and zero for inerts, if any.

The governing equations are the Euler Equations and a transport equation for the volume fraction [864, 892, 893]. The main conserved variables for each phase are the mass densities of the components, the momentum and the total energy (sensible + chemical + kinetic) per unit volume.

### Syntax:

```
"CDEM"  "TINI" tini    "PINI" pini  <"PREF" pref>
...     "KSI0" ksi0    "K0" k0
...     "TMAX" tmax    "R" rgas
...     "NESP" nesp     "ORDP" ordp   "NLHS" nlhs
...
...     "COMP1"
...     "MMOL" mmol    "H0" h0      "CREA" crea
...     "CV0" cv0    "CV1" cv1 ... "CVordp" cvordp
...     "YMAS" ymas
...
...     . . .
...
...     "COMPnesp"
...     "MMOL" mmol    "H0" h0      "CREA" crea
...     "CV0" cv0    "CV1" cv1 ... "CVordp" cvordp
...     "YMAS" ymas
...
...     <"KOF" kof>
...     <"UCDS" ucds>
...     <"DIRE" dire>
...     <"TO " temp0>
...     <"H " hcoef>
...     <"GX " gx>
...     <"GY " gy>
...     <"GZ " gz>
...     <"CFLA" 1 "SL " sl  "PSL " psl "TSL " tsl
...           "DL " dl  "LE " le "PU " pu
...           "XIG " xig "YIG " yig "ZIG " zig>
...     <"CFLA" 2 "SL " sl  "PSL " psl "TSL " tsl
...           "DL " dl  "LE " le "A " acoef "B " bcoef
...           "R0 " r0  "XIG " xig "YIG " yig "ZIG " zig>
/LECT/
```

First, some **properties of the mixture** as a whole (unburnt plus burnt phases) are defined. The code accepts any consistent units set (under the user's responsibility). The SI units are given below in brackets for each quantity  $[\cdot]$  just as a reference.

**tini**

Initial temperature of the mixture [K].

**pini**

Initial pressure of the mixture [Pa].

**pref**

Reference pressure [Pa].

**ksi0**

Initial volume fraction  $[-]$  of the burnt phase  $\Xi_{0b}$  i.e. volume of the burnt phase over total volume of the mixture (sum of burnt plus unburnt phases). Typical initial values would be 0.0 for an initially completely unburnt zone and 1.0 for an initially completely burnt zone (trigger). However, *for numerical reasons* it is advised to specify slightly different (non-round) values, e.g. 0.001 and 0.999, respectively. Note that the sum of  $\Xi_{0b}$  over the different initial zones is not necessarily 1.0. It is the sum  $(\Xi_{0b} + \Xi_{0u})$ , where  $\Xi_{0u}$  is the initial volume fraction of the unburnt phase (not explicitly defined and used in this model), which must be 1.0, of course. Note that the initial volume fractions of the burnt and unburnt phases  $\Xi_{0b}$  and  $\Xi_{0u}$  are totally unrelated with the initial mass fractions  $Y_{i,u}$  of components in the unburnt state (see **y<sub>mas</sub>** below).

**k0**

Part of the fundamental flame speed **k** transported with the flow [m/s]. The fundamental flame speed is given by  $\mathbf{k} = \mathbf{k0} + \mathbf{k0f}$ , where **k0f** is optionally defined below (else it is 0). Note that, in alternative to assigning it by giving **k0** and **k0f**, the fundamental flame speed **k** can be automatically computed by the code. To this end, activate the **CFLA 1** optional keyword (see below). In this case a value for **k0** must still be specified, but it is simply ignored by the code.

**tmax**

Maximum value of the temperature for the computation of the specific heats [K]. For temperatures  $T > T_{\max}$  the code takes  $C_V(T) = C_V(T_{\max})$ .

**rgas**

Gas constant  $R$ , equal to 8.3144621 in SI units [J/(mol·K)].

**nesp**

Total number of species  $[-]$  involved in the mixture: reactants, reaction products and inerts, if any.

**nlhs**

Number of species  $[-]$  in the left hand side of the chemical reaction, i.e. number of reactants. If there are any inerts, these are **not** counted here.

**ordp**

Temperature polynomial degree  $[-]$  for the constant volume specific heat computation.

Then, we specify the **properties of each species** (i.e. of each component) of the mixture. Note that the species must be listed in the order in which they appear in the chemical reaction equation.

### "COMP1", ..., "COMPnsp"

Keywords which state that we will describe the properties of the species 1, ..., nsp. Note, however, that the numbers 1, ..., nsp are only a visual indication for the user: the code interprets only the first four letters of each keyword (COMP in this case), so these digits (1, 2, etc.) are ignored. It is the User's responsibility to list the components in the correct order, as specified above.

**mmol**

Molar mass [kg/mol].

**h0**

Formation enthalpy [J/kg] at  $T = 0$  K.

**crea**

Coefficient  $[-]$  in the chemical reaction (positive if the species is a reactant).

**cv0, ..., cvordp**

Coefficient of  $T^0, \dots, T^{\text{ordp}}$  for the computation of the constant volume specific heat. The expression used is  $C_V(T) = C_{V0} + C_{V1}T + C_{V2}T^2 \dots$  so that  $C_{V0}$  is in J/(Kg·K),  $C_{V1}$  is in J/(Kg·K<sup>2</sup>), etc.

**ymas**

Initial mass fraction  $Y_{i,u}$  of the current component  $[-]$  before the combustion occurs, i.e. in the **unburnt** state. This is the mass of the current component divided by the total mass of the unburnt phase in the cell. Note that the possible presence also of some burnt phase in the same cell is not considered for the evaluation of  $Y_{i,u}$ , so that the sum of all  $Y_{i,u}$  for  $i$  ranging from 1 to the number of components of the burnt phase must equal 1.0. Note that here we **must** always specify the mass fraction of the **unburnt** phase, even in the case that the mesh portion associated with the currently described CDEM material is in reality initially occupied by the burnt phase! Note also that the initial mass fractions  $Y_{i,u}$  of components in the unburnt state are totally unrelated with the initial volume fractions of the burnt and unburnt phases  $\Xi_{0b}$  and  $\Xi_{0u}$  (see **ksi0** above).

Next, one can define some **optional additional parameters** for the combustion.

**k0f**

Part of the fundamental flame speed [m/s] fixed in space (default value 0). The fundamental flame speed is given by  $k = k0 + k0f$ , where  $k0$  has been defined above in the general properties of the mixture. Note that, in alternative to assigning it by giving  $k0$  and  $k0f$ , the fundamental flame speed  $k$  can be automatically computed by the code. To this end, activate the **CFLA 1** optional keyword (see below). In this case a value for  $k0$  must still be specified, but it is simply ignored by the code.

**ucds**

Order of reconstruction  $[-]$  for the volume fraction.

- 0 The same limited reconstruction as in [845] (default value).
- 1 Limited reconstruction combined with the Upwind Downwind Controlled Splitting (see [862, 864, 892, 893]). At the moment, this is the recommended method (but note that it is not the one used by default!).
- 2 Anti-diffusive reconstruction combined with the Upwind Downwind Controlled Splitting (see [862, 864, 892, 893]).
- 3 Anti-diffusive reconstruction with tanh-correction, combined with the Upwind Downwind Controlled Splitting (in progress).

**dire**

Indicates how to compute the fundamental flame speed in multi-dimensional computations  $[-]$ .

- 0 The fundamental flame speed is equal to  $((k_0 + k_{0f}) (\vec{n} \cdot \vec{n}_f))$ , where  $\vec{n}$  and  $\vec{n}_f$  are the normal to the flame surface and the normal to the interface of the finite volume cell (default value).
- 1 The fundamental flame speed is equal to  $(k_0 + k_{0f})$ . This should be used only for debugging purposes.

**temp0, hcoef**

Loss coefficients  $([K], [J/(kg \cdot K)])$ , respectively) for the total energy  $Q$  exchanged with the surrounding environment, which can behave like a heat sink (default value is zero). The loss of energy  $Q$  per unit time and per unit volume of the gas is given by:

- For the unburnt phase,  $Q_U = \alpha_u \text{hcoef}(T_u - \text{temp0})$
- For the burnt phase,  $Q_B = \alpha_b \text{hcoef}(T_b - \text{temp0})$

In these formulas,  $\alpha_u$  and  $\alpha_b$  are the *current* volume fractions of the unburnt and burnt phases, respectively (while  $\Xi_{0b}$  and  $\Xi_{0u}$  used above are the *initial* values thereof), and  $T_u$ ,  $T_b$  the corresponding temperatures.

**gx, gy, gz**

Components  $(\vec{g} = (gx, gy, gz))$   $[m/s^2]$  of the gravity acceleration for the computation of buoyancy force. Default values are zero. The buoyancy force is computed according to the following approximate formulae.

- $\alpha_u \vec{g}(\rho_u - \rho_u) = 0$  (buoyancy force per unit volume acting on the unburnt phase)
- $\alpha_b \vec{g}(\rho_b - \rho_u)$  (buoyancy force per unit volume acting on the burnt phase)

In these formulas,  $\alpha_u$  and  $\alpha_b$  are the *current* volume fractions of the unburnt and burnt phases, respectively (while  $\Xi_{0b}$  and  $\Xi_{0u}$  used above are the *initial* values thereof), and  $\rho_1, \rho_2$  the corresponding mass densities. The buoyancy force contribution is typically irrelevant in detonation problems, but can become important in slow deflagration problems.

Finally, by setting **CFLA** 1 and by specifying the following additional parameters, one may optionally require that the code itself computes the flame speed, as an alternative to specifying **k0** and **k0f** in the above set of input data. Note that the value of **k0** must be specified anyway, but will be simply ignored in this case.

sl, psl, tsl, dl, pu, le, xig, yig, zig

**Parameters of the combustion model 1** ("CFLA" 1, see [890, 891]) to compute the flame speed. Formulas here implemented are the ones presented in [891]. The expression for the fundamental flame speed is given by formula (7)

$$K_0(x, y, z) = \text{sl } \Theta_{TH} \Theta_{TURB} \Theta_{WRIN}.$$

$\Theta_{TH}$  given by formula (8),

$$\Theta_{TH} = \left( \frac{P}{\text{psl}} \right)^{-0.5} \left( \frac{T}{\text{tsl}} \right)^{2.2},$$

$P$  and  $T$  being the local pressure and temperature of the unburnt mixture.

$\Theta_{TURB}$  given by formula (15),

$$\Theta_{TURB} = 1 + 1.334 \overbrace{\gamma}^{0.6} \cdot \text{pu} \cdot \left( \frac{u'}{\text{sl}} \right)^{0.55} \left( \frac{L_t}{\text{dl}} \right)^{0.15} (\text{le})^{-0.3}$$

$$L_t = 0.2 \overbrace{\Delta}^{\text{mesh size}}$$

$$u' = L_t \|S_{ij}\|, \quad S_{ij} = \frac{1}{2} \left[ \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right],$$

$\vec{u}$  being the local velocity of the unburnt mixture.  $\Theta_{WRIN}$  is given by formula (16),

$$\Theta_{WRIN} = \max(1, R)^{1/3}, \quad R = \sqrt{(x - \text{xig})^2 + (y - \text{yig})^2 + (z - \text{zig})^2},$$

$R$  being expressed in meters.

The parameters involved in these formulas are the following ones.

- sl - fundamental laminar flame velocity (m/s in SI)
- psl - pressure at which sl is specified (Pa in SI)
- tsl - temperature at which sl is specified (K in SI)
- dl - laminar flame thickness (m in SI)
- le - effective Lewis number
- pu - user parameter
- xig - x-coordinate of the ignition point (m in SI)
- yig - y-coordinate of the ignition point (m in SI)
- zig - z-coordinate of the ignition point (m in SI)

As in the previous case, by setting CFLA 2 and by specifying the following additional parameters, one requires that the code itself computes the flame speed, as an alternative to specifying  $k_0$  and  $k_0f$  in the above set of input data. Note that the value of  $k_0$  must be specified anyway, but will be simply ignored in this case.

sl, psl, tsl, dl, le, pu, xig, yig, zig, r0

**Parameters of the combustion model 2** ("CFLA" 2). The expression for the fundamental flame speed is given by

$$K_0(x, y, z) = \text{sl } \Theta_{TH} \Theta_{TURB}.$$



$\Theta_{TH}$  given by

$$\Theta_{TH} = \left( \frac{P}{\text{psl}} \right)^{-0.5} \left( \frac{T}{\text{tsl}} \right)^{2.2},$$

$P$  and  $T$  being the local pressure and temperature of the unburnt mixture.

$\Theta_{TURB}$  given by

$$\Theta_{TURB} = 1 + 1.334 \overbrace{\gamma}^{0.6} \left( \frac{u'}{\text{sl}} \right)^{0.55} \left( \frac{L_t}{\text{dl}} \right)^{0.15} (\text{le})^{-0.3}$$

$$L_t = \begin{cases} \frac{1}{5} \cdot \Delta & \text{if } R \leq r_0 \\ 1 \cdot \Delta & \text{if } R > r_0 \text{ and } \|\vec{\nabla}_S \vec{u}\|_2 \leq g_{co} \\ 1 \cdot \Delta + \frac{|\vec{u} \cdot \vec{n}|}{\|\vec{\nabla}_S \vec{u}\|_2} & \text{if } R > r_0 \text{ and } \|\vec{\nabla}_S \vec{u}\|_2 > g_{co} \end{cases}$$

$$R = \sqrt{(x - \text{xig})^2 + (y - \text{yig})^2 + (z - \text{zig})^2}.$$

The cut-off gradient is given by

$$g_{co} = \frac{\text{pu} \cdot \text{sl}}{\Delta}.$$

$$\vec{\nabla}_S \vec{u} = \vec{\nabla} \vec{u} - \vec{n} \left( \vec{n} \cdot \vec{\nabla} \vec{u} \right)$$

$$u' = L_t \|S_{ij}\|, \quad S_{ij} = \frac{1}{2} \left[ \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right],$$

$\vec{u}$  being the local velocity of the unburnt mixture.

The parameters involved in these formulas are the following ones.

- sl - fundamental laminar flame velocity (m/s in SI)
- psl - pressure at which sl is specified (Pa in SI)
- tsl - temperature at which sl is specified (K in SI)
- dl - laminar flame thickness (m in SI)
- le - effective Lewis number
- pu - user parameter for the cut-off gradient
- xig - x-coordinate of the ignition point (m in SI)
- yig - y-coordinate of the ignition point (m in SI)
- zig - z-coordinate of the ignition point (m in SI)
- r0 - distance at which the turbulence is fully developed.

/LECT/

List of the elements (Cell-Centred Finite Volumes) to which the current material is associated.

**Comments:**

For the first species (COMP1), the coefficient of the chemical reaction `crea` should be equal to 1.

The sum of all initial mass fractions `ymas` should be equal to 1.

### Outputs:

The different components of the ECR table are as follows. First, some values for the **whole mixture**:

ECR(1) : pressure  $p$  of the mixture, i.e. weighted average of the pressures of the unburnt and burnt phases  $p_u$  and  $p_b$ :  $p = \alpha_u p_u + \alpha_b p_b$ . The weighting coefficients  $\alpha_u$  and  $\alpha_b$  are the *current* volume fractions of the unburnt and burnt phases.

ECR(2) : density  $\rho$  of the mixture, i.e. weighted average of the densities of the unburnt and burnt phases  $\rho_u$  and  $\rho_b$ :  $\rho = \alpha_u \rho_u + \alpha_b \rho_b$ . The weighting coefficients are the *current* volume fractions of the unburnt and burnt phases.

ECR(3) : maximum between the sound speed in the unburnt phase and the sound speed in the burnt phase:  $c = \max(c_u, c_b)$ .

Then, some data for the **unburnt phase**:

ECR(4) : current volume fraction of the unburnt phase  $\alpha_u$ . Note that, since the numerical schemes adopted in the CDEM model are LED (Local Extremum Diminishing), the value of  $\alpha_u$  cannot increase above the maximum value initially specified in the whole domain and cannot decrease below the minimum value initially specified in the whole domain

ECR(5) : density of the unburnt phase  $\rho_u$ .

ECR(6) : velocity along x of the unburnt phase  $v_{x,u}$ .

ECR(7) : velocity along y of the unburnt phase  $v_{y,u}$ .

ECR(8) : velocity along z of the unburnt phase  $v_{z,u}$  (3D only, i.e. this data is missing in 2D cases).

ECR(6 + idim) : pressure of the unburnt phase  $p_u$ . Here `idim` is the space dimension (2 or 3).

Then, the same data but for the **burnt phase**:

ECR(7 + idim) : current volume fraction of the burnt phase  $\alpha_b$ . Note that, since the numerical schemes adopted in the CDEM model are LED (Local Extremum Diminishing), the value of  $\alpha_b$  cannot increase above the maximum value initially specified in the whole domain and cannot decrease below the minimum value initially specified in the whole domain.

ECR(8 + idim) : density of the burnt phase  $\rho_b$ .

ECR(9 + idim) : velocity along x of the burnt phase  $v_{x,b}$ .

ECR(10 + idim) : velocity along y of the burnt phase  $v_{y,b}$ .

ECR(11 + idim) : velocity along z of the burnt phase  $v_{z,b}$  (3D only, i.e. this data is missing in 2D cases).

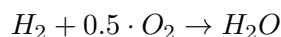
ECR(9 + (2 idim)) : pressure of the burnt phase  $p_b$ .

Finally, some **additional data**:

```
iespmax = min(nesp, 20 - (9 + (2 idim)))
ECR(9 + iesp + (2 idim)) : mass fraction of the iesp-th reagent species after the
combustion occurs (iesp = 1, ..., nlhs; iesp ≤ iespmax).
ECR(9 + iesp + (2 idim)) : mass fraction of the iesp-th product species before the
combustion occurs (iesp = nlhs + 1, ..., nesp - 1; iesp ≤ iespmax).
...
nn = min(20, (9 + (2 idim) + nesp)) = 9 + (2 idim) + iespmax
ECR(nn) : difference between the mass fractions of the first species in the unburnt
phase and in the burnt phase.
ECR(21) : burnt surface per unit volume. When multiplied by the cell's volume,
this is the total area of the burning surface over all faces of the current finite volume.
ECR(22) : fundamental flame speed (including k0f), or computed by the code in
the case CFLA 1/2.
ECR(23) : absolute temperature of the mixture  $T$  [K]. This is computed internally
as the weighted average of the temperatures of the unburnt and burnt phases  $T_u$  and
 $T_b$ :  $T = \alpha_u T_u + \alpha_b T_b$ , where the weighting coefficients  $\alpha_u$  and  $\alpha_b$  are the current
volume fractions of the unburnt and burnt phases.
ECR(24) : integral turbulent scale  $L_t$  computed by the code in the case CFLA 1/2;
0 otherwise
ECR(25) : rms turbulent velocity  $u'$  computed by the code in the case CFLA 1/2; 0
otherwise
```

### Example:

Consider for example the combustion of a mixture of hydrogen, oxygen, water vapor and nitrogen. The chemical reaction is usually written without the inerts, as follows



This can be re-written, by bringing all terms to the left hand side and by letting appear also the inerts (nitrogen in this case)

$$1 \cdot H_2 + 0.5 \cdot O_2 - 1 \cdot H_2O + 0 \cdot N_2 \rightarrow 0$$

Here COMP1, ..., COMP4 represent H<sub>2</sub>, O<sub>2</sub>, H<sub>2</sub>O and N<sub>2</sub>, respectively (in this order!). Therefore, nesp=4 and nlhs=2. The constant volume specific heat coefficients are obtained via a fourth order regression of JANAF tables (ordp=4).

Assume that the fluid mesh is initially subdivided into two regions: a region called "burnt" containing (mostly) the burnt gases and a region called "unburnt" containing (mostly) the unburnt (fresh) gases.

One way of providing the input data for this problem is to specify two CDEM materials, one for each region, as follows.

```
CDEM ! This CDEM material is used to represent the burnt phase
PINI 10.E5 PREF 1.e5 TINI 2000.0
KSI0 0.999
```

```
KO 45.2
TMAX 6000.  R      8.31441
NESP 4
NLHS 2
ORDP 4
COMP1 ! H2
  MMOL 2.01594E-3 HO -4.195E6  CREA 1.
  CV0 9834.91866 CV1 0.54273926 CV2 0.000862203836
  CV3 -2.37281455E-07 CV4 1.84701105E-11
  YMAS 0.1
COMP2 ! O2
  MMOL 31.9988E-3 HO -2.634E5  CREA 0.5
  CV0 575.012333 CV1 0.350522002 CV2 -0.000128294865
  CV3 2.33636971E-08 CV4 -1.53304905E-12
  YMAS 0.2
COMP3 ! H2O
  MMOL 18.01534E-3 HO -1.395D7 CREA -1.0
  CV0 1155.95625 CV1 0.768331151 CV2 -5.73129958E-05
  CV3 -1.82753232E-08 CV4 2.44485692E-12
  YMAS 0.3
COMP4 ! N2
  MMOL 28.0134E-3 HO -2.953D5 CREA 0.0
  CV0 652.940766 CV1 0.288239099 CV2 -7.80442298E-05
  CV3 8.78233606E-09 CV4 -3.05514485E-13
  YMAS 0.4
LECT burnt TERM ! "burnt" is the name of the mesh zone
                  ! initially containing the burnt phase

CDEM ! This CDEM material is used to represent the unburnt phase
PINI 1.E5 PREF 1.e5  TINI 300.0
KSIO 0.001
KO 45.2
TMAX 6000.  R      8.31441
NESP 4
NLHS 2
ORDP 4
COMP1 ! H2
  MMOL 2.01594E-3 HO -4.195E6  CREA 1.
  CV0 9834.91866 CV1 0.54273926 CV2 0.000862203836
  CV3 -2.37281455E-07 CV4 1.84701105E-11
  YMAS 0.1
COMP2 ! O2
  MMOL 31.9988E-3 HO -2.634E5  CREA 0.5
  CV0 575.012333 CV1 0.350522002 CV2 -0.000128294865
  CV3 2.33636971E-08 CV4 -1.53304905E-12
  YMAS 0.2
COMP3 ! H2O
  MMOL 18.01534E-3 HO -1.395D7 CREA -1.0
  CV0 1155.95625 CV1 0.768331151 CV2 -5.73129958E-05
  CV3 -1.82753232E-08 CV4 2.44485692E-12
  YMAS 0.3
```

```

COMP4 ! N2
MMOL 28.0134E-3 H0 -2.953D5 CREA 0.0
CV0 652.940766 CV1 0.288239099 CV2 -7.80442298E-05
CV3 8.78233606E-09 CV4 -3.05514485E-13
YMAS 0.4
LECT unburnt TERM ! "unburnt" is the name of the mesh zone
                  ! initially containing the unburnt phase

```

Note from the above example that the component parameters (COMP<sub>n</sub>) for the two materials must be identical, except the mass fractions (but be warned that the code does not check this!).

Alternatively (and perhaps more intuitively) one can define a *single CDEM material for the whole fluid mesh*, and then specify the initial conditions zone by zone by means of the INIT VFCC directive (for CDEM material), see [GBE.0066](#). In the present example, this would correspond to the following input:

```

CDEM ! This CDEM material is used to represent the whole gas
      ! (i.e. both the burnt and the unburnt zones)
PINI 10.E5 ! This value will be overridden in the INIT directive
PREF 1.e5
TINI 2000.0 ! This value will be overridden in the INIT directive
KSI0 0.999 ! This value will be overridden in the INIT directive
K0 45.2 ! This value will be overridden in the INIT directive
TMAX 6000.
R 8.31441
NESP 4
NLHS 2
ORDP 4
COMP1 ! H2
MMOL 2.01594E-3 H0 -4.195E6 CREA 1.
CV0 9834.91866 CV1 0.54273926 CV2 0.000862203836
CV3 -2.37281455E-07 CV4 1.84701105E-11
YMAS 0.1 ! This value will be overridden in the INIT directive
COMP2 ! O2
MMOL 31.9988E-3 H0 -2.634E5 CREA 0.5
CV0 575.012333 CV1 0.350522002 CV2 -0.000128294865
CV3 2.33636971E-08 CV4 -1.53304905E-12
YMAS 0.2 ! This value will be overridden in the INIT directive
COMP3 ! H2O
MMOL 18.01534E-3 H0 -1.395D7 CREA -1.0
CV0 1155.95625 CV1 0.768331151 CV2 -5.73129958E-05
CV3 -1.82753232E-08 CV4 2.44485692E-12
YMAS 0.3 ! This value will be overridden in the INIT directive
COMP4 ! N2
MMOL 28.0134E-3 H0 -2.953D5 CREA 0.0
CV0 652.940766 CV1 0.288239099 CV2 -7.80442298E-05
CV3 8.78233606E-09 CV4 -3.05514485E-13
YMAS 0.4 ! This value will be overridden in the INIT directive
LECT fluid TERM ! "fluid" is the name of the mesh zone
                ! initially containing both the burnt and
                ! the unburnt phases

```

. . .

```

INIT ! Now we fine-tune the initial conditions
      ! for the CDEM material
VFCC ! Initial conditions for the burnt phase
      VITX 0.0
      VITY 0.0
      VITZ 0.0
      PINI 10.E5
      TINI 2000.0
      KSIO 0.999
      K0 45.2
      Y1 0.1
      Y2 0.2
      Y3 0.3
      Y4 0.4
LECT burnt TERM ! "burnt" is the name of the mesh zone
                  ! initially containing (mostly) the burnt phase
VFCC ! Initial conditions for the unburnt phase
      VITX 0.0
      VITY 0.0
      VITZ 0.0
      PINI 1.E5
      TINI 300.0
      KSIO 0.001
      K0 45.2
      Y1 0.1
      Y2 0.2
      Y3 0.3
      Y4 0.4
LECT unburnt TERM ! "unburnt" is the name of the mesh zone
                   ! initially containing (mostly) the unburnt phase

```

In case of more complex initial distributions (e.g. varying element-by-element) of the properties, the VFCC ... /LECT/ sub-directive of the INIT directive can be repeated as many times as necessary in order to set the initial conditions of each zone (or element), see [GBE.0066](#).

### Example of contents of the ECR table

We continue the above practical example with **4 species and 2 reactants** (i.e. 2 components on the left hand side of the chemical reaction).

In **2D space** (idim = 2), we have the following components of the ECR table:

- $\text{iespmax} = \min(\text{nesp}, 20 - (9 + (2 \text{ idim}))) = \min(4, 20 - 13) = 4$
- $\text{nn} = \min(20, (9 + (2 \text{ idim}) + \text{nesp})) = 9 + (2 \text{ idim}) + \text{iespmax} = 17$
- $\text{ECR}(9 + 1 + (2 \text{ idim})) = \text{ECR}(14)$  : mass fraction of the  $\text{H}_2$  after the combustion occurs.

- ECR(9 + 2 + (2 idim)) = ECR(15) : mass fraction of the O<sub>2</sub> after the combustion occurs.
- ECR(9 + 3 + (2 idim)) = ECR(16) : mass fraction of the H<sub>2</sub>O before the combustion occurs.
- ECR(nn) = ECR(17) : difference between the mass fraction of the H<sub>2</sub> in the unburnt phase and in the burnt phase.
- ECR(21) : burnt surface per unit volume.
- ECR(22) : fundamental flame speed (including `k0f`).
- ECR(23) : absolute temperature of the mixture  $T$  [K]. This is computed internally as the weighted average of the temperatures of the unburnt and burnt phases  $T_u$  and  $T_b$ :  $T = \alpha_u T_u + \alpha_b T_b$ , where the weighting coefficients  $\alpha_u$  and  $\alpha_b$  are the *current* volume fractions of the unburnt and burnt phases.
- ECR(24) : Unused.
- ECR(25) : Unused.

In **3D space** (idim = 3), we have the following components of the ECR table:

- iespmax = min(nesp, 20 - (9 + (2 idim))) = min(4, 20 - 15) = 4
- nn = min(20, (9 + (2 idim) + nesp)) = 9 + (2 idim) + iespmax = 19
- ECR(9 + 1 + (2 idim)) = ECR(16) : mass fraction of the H<sub>2</sub> after the combustion occurs.
- ECR(9 + 2 + (2 idim)) = ECR(17) : mass fraction of the O<sub>2</sub> after the combustion occurs.
- ECR(9 + 3 + (2 idim)) = ECR(18) : mass fraction of the H<sub>2</sub>O before the combustion occurs.
- ECR(nn) = ECR(19) : difference between the mass fractions of the H<sub>2</sub> in the unburnt phase and in the burnt phase.
- ECR(21) : burnt surface per unit volume.
- ECR(22) : fundamental flame speed (including `k0f`).
- ECR(23) : absolute temperature of the mixture  $T$  [K]. This is computed internally as the weighted average of the temperatures of the unburnt and burnt phases  $T_u$  and  $T_b$ :  $T = \alpha_u T_u + \alpha_b T_b$ , where the weighting coefficients  $\alpha_u$  and  $\alpha_b$  are the *current* volume fractions of the unburnt and burnt phases.
- ECR(24) : Unused.
- ECR(25) : Unused.

Note that one can easily determine the mass fraction in both the burnt and the unburnt phases, as explained in [844]. Indeed, via the knowledge of ( $Y_{H_2,u} - Y_{H_2,b}$ ), one can determine the other variations by using the molar mass (`mmol`) and the reaction coefficient (`crea`):

$$\frac{Y_{H_2,u} - Y_{H_2,b}}{2.01594 \cdot 10^{-3} \cdot 1} = \frac{Y_{O_2,u} - Y_{O_2,b}}{31.9988 \cdot 10^{-3} \cdot 0.5} = \frac{Y_{H_2O,u} - Y_{H_2O,b}}{18.01534 \cdot 10^{-3} \cdot (-1)}$$

Note also that the absolute temperature  $T_u, T_b$  [K] of each phase (unburnt or burnt) can be determined from the expression:

$$T = \frac{p}{R\rho}$$

where  $p$  is the absolute pressure [Pa],  $\rho$  is the density [kg/m<sup>3</sup>] and  $R$  [J/(kg·K)] is given by:

$$R = \frac{R_m}{\sum \frac{Y_i}{W_i}}$$

Here  $R_m = 8.314$  [J/(mol·K)] is the (molar) constant of gases,  $W_i$  is the molar mass [kg/mol] of the  $i$ -th gas and  $Y_i$  is the mass fraction [–] of the  $i$ -th gas.



### 7.8.40 DEMS—Discrete Equation Method for Two Phase Stiffened Gases

#### Object:

Modeling of two phase flows involving stiffened gases via the Discrete Equation method of Abgrall and Saurel 2003 (see also [855, 862]). The governing equations are the Euler Equations in conservative form (the main conserved variables for each phase are the mass densities of the components, the momentum, the total energy (sensible + chemical + kinetic) per unit volume) and a transport equation for the volume fraction.

#### Syntax:

```
"DEMS"  "PINI"  pini  <"PREF" pref>
...      "ALP1"  alp1  "ROI1" roi1 "ROI2" roi2
...      "NESP"
...
...      "COMP1"
...      "GAMM"  gamm  "CP"  cp  "PI"  pi
...      "YMA1"  yma1
...      "YMA2"  yma2
...      "MMOL"  mmol   "H0"  h0

...
...      "COMPnesp"
...      "GAMM"  gamm  "CP"  cp  "PI"  pi
...      "YMA1"  yma1
...      "YMA2"  yma2
...      "MMOL"  mmol   "H0"  h0
...

...      <"RELA" rela> <'RHOM' rhom> <'EPSM' epsm>
...      <"UCDS" ucds>
...      <"ERE1" ere1>
...      <"ERE2" ere2>
...      <"ADCR" adcr>
...      <"MODE" nmod>
...      <"FONC" nfon>
```

pini

Initial pressure of the mixture.

pref

Reference pressure.

alp1

Initial volume fraction of the phase 1

roi1

Mass density of the phase 1

roi2

Mass density of the phase 2

nesp

Number of species involved in each phase

**"COMP1", ..., "COMPnesp"**

Keywords which state that we will describe the properties of the species 1,...,nesp.

gamm

Specific heat ratio

cp

Constant pressure specific heat

pi

Molecular attraction effect parameter

yma1

Mass fraction in the phase 1

yma2

Mass fraction in the phase 2

mmol

Molar mass.

h0

Formation enthalpy at  $T = 0$  K.

ucds

High order reconstruction for the volume fraction.

0 The same limited reconstruction as in [\[845\]](#) (default value).

- 1 Limited reconstruction combined with the Upwind Downwind Controlled Splitting (see [862]).
  - 2 Anti-diffusive reconstruction combined with the Upwind Downwind Controlled Splitting (see [862]).
  - 3 Anti-diffusive reconstruction with tanh-correction, combined with the Upwind Downwind Controlled Splitting (in progress).
- rela**
- 1 Relaxation of the pressure and of the velocity (the pressure and the velocity are the same on the two phases).
  - 21 As 1, with isentropic transformation in the phase 1.
  - 22 As 1, with isentropic transformation in the phase 2.
  - 23 As 1, with isentropic transformation in the phase which presents less mass than the other (suggested option if one wants to relax).
  - 0 No relaxation of the pressure and of the velocity (default value).
- rhom, epsm** When rela is equal to 21–23, we inject mass in the phase in which the isentropic transformation occurs to have rhom as threshold value for the minimum density, providing that the injected mass is lower than epsm times the mass of the other phase. Default values: 0 and 1.0D-6.
- ere1, ere2** Reference energies for the phase 1 and 2 (0 is their default value).
- adcr**
- 1 DEMS model is used for HCDA (hypothetical core disruptive accident, ADC in French) computations (see below).
  - 0 DEMS model is not used for HCDA computations (default value).
- nmod** In case of HCDA computations (adcr equal to 1) it is possible to inject energy into the bubble. A negative value allows removal of energy (injection of a negative energy).
- 0 No energy injection (default value).
  - $\pm 1$  Homogeneous energy injection into the bubble (we specify dE/dt)
  - $\pm 2$  Homogeneous energy injection into the bubble (we specify dE/dVol)
- nfon** In the case imod > 0, number of the function defined by the directive FONC in which we specify the power (W) as function of time (s).

### Comments:

The sum of all initial mass fractions should be equal to 1 in each phase.

### Outputs:

We define  $n_{\text{esp}_{\text{ecr}}} = \min(n_{\text{esp}}, 7 - n_{\text{dim}})$ , which represents the number of species that are represented in the ECR table. Then the different components of the ECR table are:

ECR(1) : pressure  
 ECR(2) : density  
 ECR(3) : maximum of the sound speed in the two phases  
 ECR(4) : velocity along x  
 ECR(5) : velocity along y  
 ECR(6) : velocity along z (if existing)  
 ECR(4 + ndim) : volume fraction of the phase 1  
 ECR(5 + ndim) : density of the phase 1  
 ECR(6 + ndim) : velocity along x of the phase 1  
 ECR(7 + ndim) : velocity along y of the phase 1  
 ECR(8 + ndim) : velocity along z of the phase 1 (if existing)  
 ECR(6 + (2 \* ndim)) : pressure of the phase 1  
 ECR(7 + (2 \* ndim)) : mass fraction of the first component of the phase 1  
 ECR(6 + (2 \* ndim) + nesp<sub>ecr</sub>) : mass fraction of the nesp<sub>ecr</sub>-th component of the phase 1  
 ECR(7 + (2 \* ndim) + nesp<sub>ecr</sub>) : volume fraction of the phase 2  
 ECR(8 + (2 \* ndim) + nesp<sub>ecr</sub>) : density of the phase 2  
 ECR(9 + (2 \* ndim) + nesp<sub>ecr</sub>) : velocity along x of the phase 2  
 ECR(10 + (2 \* ndim) + nesp<sub>ecr</sub>) : velocity along y of the phase 2  
 ECR(11 + (2 \* ndim) + nesp<sub>ecr</sub>) : velocity along z of the phase 1 (if existing)  
 ECR(9 + (3 \* ndim) + nesp<sub>ecr</sub>) : pressure of the phase 2  
 ECR(10 + (3 \* ndim) + nesp<sub>ecr</sub>) : mass fraction of the first component of the phase 2  
 ECR(9 + (2 \* ndim) + (2 \* nesp<sub>ecr</sub>)) : mass fraction of the nesp<sub>ecr</sub>-th component of the phase 2

### Example.

We consider a mixture of two stiffened gases in a shock tube (two zones, Z\_HP and Z\_BP). The zone Z\_HP is almost totally occupied by the phase 1 (volume fraction of the phase 1 equal to 0.9999) and it only contains the first stiffened gas. The zone Z\_BP is almost totally occupied by the phase 2 (volume fraction of the phase 1 equal to 0.0001) and it only contains the second stiffened gas.

```

*                               ZONE 1
*                               -----
*
*    DEMS
*
*    PINI 9.12E3 PREF 1e0
*    ALP1 0.9999 ROI1 1.271 ROI2 0.99
*
*    NESP 2
*

```

```

COMP ! AIR. H0 is false, but not used for the moment
  GAMM 1.4 CP 1010. PI 0.0
  YMA1 1.0    ! YMA1 = mass fraction in the phase 1
  YMA2 0.0    ! YMA2 = mass fraction in the phase 2
  MMOL 28.8E-3 H0 0.0

*
*

COMP ! STIFFENED GAS.
  GAMM 7.0 CP 7990 PI 3.0E3
  YMA1 0.00
  YMA2 1.00
  MMOL 18E-3 H0 0.0

*

                                LECT Z_HP TERM

*
*                                ZONE 2
*                                -----
DEMS

*
PINI 1.0E0 PEF 1e0
ALP1 0.0001 ROI1 1.271  ROI2 0.99

*
*

NESP 2

*
COMP ! AIR. H0 is false, but not used for the moment
  GAMM 1.4 CP 1010. PI 0.0
  YMA1 1.0
  YMA2 0.0
  MMOL 28.8E-3 H0 0.0

* *

COMP ! STIFFENED GAS.
  GAMM 7.0 CP 7990 PI 3.0E3
  YMA1 0.00
  YMA2 1.00
  MMOL 18E-3 H0 0.0

*
*

                                LECT Z_BP TERM

```

### Example HCDA computations ( $\text{adcr} = 1$ ).

We consider a mixture involving 3 stiffened gases. Phase 1 is liquid and is completely occupied by sodium. Phase 2 is gaseous and can contain the bubble or the argon.

```
DEMS
*
      PINI 1e5 PREF 0.0
      ALP1 1.0E-8      ROI1 856.      ROI2 1.0
*
      NESP 3
```

```

*
COMP ! NA
  GAMM 2.0 CP 1230 PI 2.4E9
  YMA1 1.0 ! YMA1 = mass fraction in the phase 1
  YMA2 0.0 ! YMA2 = mass fraction in the phase 2
  MMOL 23.0E-3 HO 0.0
*
COMP ! BULLE
  GAMM 1.4 CP 1000 PI 0.0
  YMA1 0.0
  YMA2 1.0
  MMOL 20E-3 HO 0.0
*
COMP ! AR
  GAMM 1.67 CP 523 PI 0.0
  YMA1 0.0
  YMA2 0.0
  MMOL 40.0E-3 HO 0.0
*
UCDS 1
RELA 1
*
ERE1 = (PINI + (gamma PINF)) / (ROI1 * (gamma - 1)) =
      = ((1E5 + (2 * 2.4E9)) / (856 )) = 0.560759345794E+07
*
ERE21 = (PINI + (gamma PINF)) / (ROI1 * (gamma - 1)) =
      = ((1E5 + 0.0) / (1.0 * 0.4)) = 2.5E5
*
*
ERE1 0.560759345794E+07
ERE2 2.5E5
ADCR 1
MODE 1
FONC 3
*
LECT FLUD TERM
*
...
*
*
FONC 1 TABLE 3
      0.      1.0e0
      1e-5    1.0e0
      1e+4    1.0e0
*
      2 TABLE 3
      0.      0.
      1e-5    0.
      1e+4    0.
*
*

```

\* FONC 3 is the power injected in the bubble as function of time  
\*

3 TABLE 77

0.000000E+00 -.200000E+06

0.100000E-01 -.202020E+06

0.200000E-01 -.204082E+06

...

0.760000E+00 -.833333E+06

## 7.8.41 GAZD—Detonation in gas Mixture

**Object:**

The governing equations are the Euler Equations for ideal gases Mixtures with temperature-dependent heat capacities. It means that the vibrational degrees of freedom of poly-atomic molecules are taken into account via the dependence of specific heat capacities with respect to the temperature.

The experimental Temperature dependence of specific heat capacities can be found in the JANAF tables (NIST-JANAF Thermochemical tables - Fourth ed., Journal of Physical and Chemical Reference Data, Chase M.W.,1998).

In general it is necessary to take into account the presence of inerts. Although they do not participate in the chemical reaction, their mass fraction can be important (consider e.g. nitrogen in the case of air) so that their heat capacity gives an important contribution to the global heat balance of the mixture.

The combustion is governed by an **irreversible exothermic** chemical reaction. Usually this is written as follows, i.e. with reactants only on the left hand side and with reaction products only on the right hand side:

$$\sum_{r=1, \text{nrea}} c_r A_r \rightarrow \sum_{p=1, \text{npro}} c_p A_p$$

where **nrea** is the number of chemical species on the left-hand side of the chemical reaction (i.e., the number of reactants), while **npro** is the number of chemical species on the right hand side of the chemical reaction (i.e. the number of reaction products). Any inerts, which can be present in the mixture, are usually not explicitly included in the expression of the chemical reaction. Thus, the  $c_r$  and  $c_p$  coefficients in the above expression are strictly positive.

In the GAZD model this expression is re-arranged as follows, by bringing all terms to the left hand side and by letting also the inerts (if any) appear:

$$\sum_{i=1, \text{nesp}} c_i A_i \rightarrow 0$$

where **nesp** is the total number of chemical species present (i.e. the reactants, the reaction products and the inerts, if any): **nesp** = **nrea** + **npro** + **nine**. The coefficient  $c_i$  is thus strictly positive for reactants, strictly negative for reaction products and zero for inerts, if any.

The governing equations are the Euler Equations and the conserved variables are the mass density, the momentum, the total energy of the mixture and the mass fraction of the (**nesp** - 1) species defined.

**Syntax:**

```
"GAZD"  "TINI" tini    "PINI" pini  <"PREF" pref>

...  "A" a  "B" b  "EA" Ea  "R" rgas
...  "TC" Tcomb  "ALPH" alpha
...  "TMAX" tmax  "NESP" nesp    "ORDP" ordp    "NLHS" nlhs
...
...  "COMP1"
...  "MMOL" mmol  "H0" h0    "CREA" crea
...  "CV0" cv0  "CV1" cv1 ... "CVordp" cvordp
```



```

... "YMAS" ymas
...
... . . .
...
... "COMPnesp"
... "MMOL" mmol "H0" h0 "CREA" crea
... "CV0" cv0 "CV1" cv1 ... "CVordp" cvordp
... "YMAS" ymas

/LECT/

```

As generally the coefficients of the Arrhenius Law are given for SI units, it is safer to use SI units for all dataset variables.

The SI units are given below in brackets for each quantity [·] just as a reference.

**tini**

Initial temperature of the mixture [K].

**pini**

Initial pressure of the mixture [Pa].

**pref**

Reference pressure [Pa].

**a**

Pre-exponential factor. The units of this factor will vary depending on the order of the reaction.

**b**

Second factor that makes explicit the temperature dependence of the pre-exponential factor of the classical Arrhenius law.

**Ea**

activation energy [J/mol]

**rgas**

Gas constant  $R$ , equal to 8.3144621 in SI units [J/(mol·K)].

**tcomb**

Temperature at which the reaction is started [K].

**alpha**

Not used for instance.

**tmax**

Maximum value of the temperature for the computation of the specific heats [K]. For temperatures  $T > T_{\max}$  the code takes  $C_V(T) = C_V(T_{\max})$ .

**nesp**

Total number of species involved in the mixture: reactants, reaction products and inerts, if any.

**ordp**

Temperature polynomial degree for the constant volume specific heat computation.

**nlhs**

Number of species in the left hand side of the chemical reaction, i.e. number of reactants. If there are any inerts, these are **not** counted here.

Then, we specify the **properties of each species** (i.e. of each component) of the mixture. Note that the species must be listed in the order in which they appear in the chemical reaction equation.

**"COMP1", ..., "COMPnesp"**

Keywords which state that we will describe the properties of the species 1, ..., nesp. Note, however, that the numbers 1, ..., nesp are only a visual indication for the user: the code interprets only the first four letters of each keyword (**COMP** in this case), so these digits (1, 2, etc.) are ignored. It is the User's responsibility to list the components in the correct order, as specified above.

**mmol**

Molar mass [kg/mol].

**h0**

Formation enthalpy [J/kg] at  $T = 0$  K.

**crea**

Coefficient [-] in the chemical reaction (positive if the species is a reactant).

**cv0, ..., cvordp**

Coefficient of  $T^0, \dots, T^{\text{ordp}}$  for the computation of the constant volume specific heat. The expression used is  $C_V(T) = C_{V0} + C_{V1}T + C_{V2}T^2 \dots$  so that  $C_{V0}$  is in J/(Kg·K),  $C_{V1}$  is in J/(Kg·K<sup>2</sup>), etc.

**ymas**

Initial mass fraction  $Y_{i,u}$  of the current component before the detonation occurs, i.e. in the **unburnt** state. This is the mass of the current component divided by the total mass of the unburnt phase in the cell. The sum of all  $Y_{i,u}$  for  $i$  ranging from 1 to the number of components of the burnt phase must equal 1.0.

**/LECT/**

List of the elements (Cell-Centred Finite Volumes) to which the current material is associated.

**Comments:**

The sum of all initial mass fractions  $y_{\text{mas}}$  should be equal to 1.

The modified Arrhenius law has the form :

$$k = aT^b \exp\left(\frac{-E_a}{RT}\right)$$

For the Hydrogen detonation in Air we can take the following constants:

- $a = 1.2 \cdot 10^{14}$  in SI units
- $b = -0.91$
- $E_a = 69.1 \text{ kJ}$
- R Gas constant equal to 8.3144621 in SI units
- T temperature in Kelvin

### Outputs:

The different components of the ECR table are as follows:

ECR(1) : pressure  $p$  of the mixture.

ECR(2) : density  $\rho$  of the mixture.

ECR(3) : the sound speed of the mixture.

ECR(4) : temperature  $T$  of the mixture.

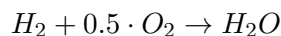
ECR(4 + iesp) : mass fraction of the iesp-th species

ECR(4 + nesp + 1) : free

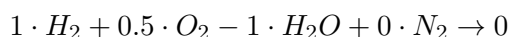
ECR(4 + nesp + 2) : specific internal energy

**Example:**

Consider for example the combustion of a mixture of hydrogen, oxygen, water vapour and nitrogen. The chemical reaction is usually written without the inerts, as follows



This can be re-written, by bringing all terms to the left hand side and by letting appear also the inerts (nitrogen in this case)



Here COMP1, ..., COMP4 represent H<sub>2</sub>, O<sub>2</sub>, H<sub>2</sub>O and N<sub>2</sub>, respectively (in this order!). Therefore, nesp=4 and nlhs=2. The constant volume specific heat coefficients are obtained via a fourth order regression of JANAF tables (ordp=4).

Assume that the fluid mesh is initially subdivided into two regions: a region called "ZONE1" to initiate the detonation and a region called "ZONE2" containing the unburnt gases where detonation propagates.

One way of providing the input data for this problem is to specify two GAZD materials, one for each region, as follows.

```

*                ZONE No1 (to initiate the detonation)
*                =====
GAZD
*      a) initial conditions
*      -----
*      PINI 2.020D6 PREF 1e5 TINI 2000
*
*      b) Arrhenius law
*      -----
*      a 1.2E14 b -0.91 Ea 69.1e3 R 8.314
*      tc 350
*      alpha 1E-3 !not used
*
*      c) gas material properties
*      -----
*      TMAX 6000 NESP 4 ORDP 4 NLHS 2
*
*      COMP1 (H2)
*      COMP1
*      MMOL 2.01594E-3 H0 -4.195E6 CREA 1.
*      CV0 10310
*      CV1 0.819
*      CV2 0.00042075
*      CV3 -1.284E-7
*      CV4 1.01497E-11
*      YMAS 0.03
*
*      COMP2 (O2)
*      COMP2
*      MMOL 31.9988E-3 H0 -2.634E5 CREA 0.5

```

```
CV0 649.56
CV1 0.2727
CV2 -0.9984E-4
CV3 1.8181E-8
CV4 -0.7782E-12
YMAS 0.21
*
* COMP3 (H2O)
COMP3
MMOL 18.01534E-3 H0 -1.395D7 CREA -1.0
CV0 1153.7
CV1 0.548
CV2 -4.087E-5
CV3 -1.303E-8
CV4 -1.743E-12
YMAS 0.02
*
* COMP4 (N2)
COMP4
MMOL 28.0134E-3 H0 -2.953D5 CREA 0.0
CV0 741.97
CV1 0.225
CV2 -6.097E-5
CV3 6.8611E-9
CV4 -2.386E-13
YMAS 0.74
*
LECT ZONE1 TERM
*
* ZONE No2
* =====
GAZD
*
* a) Initial conditions
* -----
*
* PINI 1.028D5 PREF 1e5 TINI 299
*
* b) Arrhenius law
* -----
*
* a 1.2E14 b -0.91 Ea 69.1e3 R 8.314
* tc 800 ! temperature
* alpha 1E-3 ! not used
*
* c) Gas material properties
* -----
*
* TMAX 6000 NESP 4 ORDP 4 NLHS 2
*
* COMP1 (H2)
```

```
COMP1
MMOL 2.01594E-3 H0 -4.195E6  CREA 1.
CV0 10310
CV1 0.819
CV2 0.00042075
CV3 -1.284E-7
CV4 1.01497E-11
YMAS 0.03
*
* COMP2 (O2)
COMP2
MMOL 31.9988E-3 H0 -2.634E5  CREA 0.5
CV0 649.56
CV1 0.2727
CV2 -0.9984E-4
CV3 1.8181E-8
CV4 -0.7782E-12
YMAS 0.21
*
* COMP3 (H2O)
COMP3
MMOL 18.01534E-3 H0 -1.395D7  CREA -1.0
CV0 1153.7
CV1 0.548
CV2 -4.087E-5
CV3 -1.303E-8
CV4 -1.743E-12
YMAS 0.02
*
* COMP4 (N2)
COMP4
MMOL 28.0134E-3 H0 -2.953D5  CREA 0.0
CV0 741.97
CV1 0.225
CV2 -6.097E-5
CV3 6.8611E-9
CV4 -2.386E-13
YMAS 0.74
```

LECT ZONE2 TERM

Note from the above example that the component parameters (COMP<sub>n</sub>) for the two materials must be identical, except the mass fractions (but be warned that the code does not check this!).

### 7.8.42 HMEM—Homogeneous two-phase multi-miscible-component fluid

#### Object:

This material aims at modeling a two-phase flow, where each phase is composed of multiple miscible species. The main characteristics of the model are (for more details, see the **Comments** paragraph):

- the mixture is at **kinematic equilibrium** (both phases get the same velocity),
- the mixture is at **thermal equilibrium** (both phases get the same temperature),
- the mixture is at **mechanical equilibrium** (both phases get the same pressure),
- within each phase, the potential multiple species follow the **Dalton's law**,
- the governing equation of state for each species is the **stiffened gas** law (however, the model cannot deal with miscible species which are not gases),
- the sound velocity of the mixture is **Wood's sound velocity**,
- a **heat source term** can be defined in three different ways: constant in time, time dependent, dependent on the total volume of one species.

#### Syntax:

```
"HMEM"  "PTOT"  ptot  <"PREF" pref>  "TEMP"  temp
...  "NCP1"  ncomp1  "NCP2"  ncomp2  "ALPH"  alpha

...  "COMP1"
...  "ZPRE"  zpre  "GAMM"  gamma  "QEOS"  q
...  "PINF"  pinf  "CVOL"  cv      "QSRC"  qsrc

...  "COMPncp1"
...  "ZPRE"  zpre  "GAMM"  gamma  "QEOS"  q
...  "PINF"  pinf  "CVOL"  cv      "QSRC"  qsrc

...  "COMPm"
...  "ZPRE"  zpre  "GAMM"  gamma  "QEOS"  q
...  "PINF"  pinf  "CVOL"  cv      "QSRC"  qsrc

...  "COMPncp"
...  "ZPRE"  zpre  "GAMM"  gamma  "QEOS"  q
...  "PINF"  pinf  "CVOL"  cv      "QSRC"  qsrc

...  <"ADCR"  adcr>
...  <"MODE"  nmod>
...  <"FONC"  nfon>
```

**ptot**

Initial total pressure of the mixture.

**pref**

Reference pressure.

**temp**

Initial total temperature of the mixture.

**ncomp1**

Number of species in phase 1.

**ncomp2**

Number of species in phase 2.

**alpha**

Volume fraction of phase 1.

"COMP1", ..., "COMPncomp1"

Keywords implying the description of the properties for the species 1, ..., ncomp1 in phase 1.

"COMPm", ..., "COMPncomp"

Keywords implying the description of the properties for the species ncomp1+1, ..., ncomp1+ncomp2 in phase 2.

**zpre**

Pressure fraction of the species within its phase.

**gamma**

Specific heat ratio of the species.

**q**

$q$  constant in the stiffened gas EOS.

**pinf**

$p^\infty$  constant in the stiffened gas EOS.

**cv**

Heat capacity at constant volume of the species.

**qsrc**

Parameter related to an energy source term. If  $qsrc \neq 0$  and

$nmod = 0$  then constant source term, with value **qsrc**,

$nmod \neq 0$  then time- or volume-dependent source term is applied with value provided by **nfon**.



**adcr** 1 HMEM model is used for HCDA (hypothetical core disruptive accident) computations (see below),  
 0 HMEM model is not used for HCDA computations (default value).

**nmod** Non-constant energy injection parameter.

0 No energy or constant energy injection (default value),  
 $\pm 1$  Homogeneous energy injection into the species with **qsrc**  $\neq 0$  (we specify  $dE/dt$ ),  
 $\pm 2$  Homogeneous energy injection into the species with **qsrc**  $\neq 0$  (we specify  $dE/dVol$ ).

**nfon** When **nmod**  $\neq 0$ , number of the function defined by the directive FONC in which we specify the power (W) as a function of time (s).

### Comments:

First, let us comment some values of different parameters:

- The total number of species cannot currently be more than 10.
- In each phase, the sum of all pressure fractions must equal 1.
- In each phase, one cannot have multiple species when one species gets **pinf**  $\neq 0$ .

Then, let us provide some theoretical information about the HMEM mixture model. The thermodynamical derivation of a three component model with two miscible phases can be found in [910]. The present model can be considered as an extension of this work. The continuous equations of this model read:

$$\partial_t Y_i + \mathbf{u}^T \nabla Y_i = 0, \quad i \in \Phi_1 \quad (34)$$

$$\partial_t Z_i + \mathbf{u}^T \nabla Z_i = 0, \quad i \in \Phi_2 \quad (35)$$

$$\partial_t \rho + \text{div}(\rho \mathbf{u}) = 0, \quad (36)$$

$$\partial_t(\rho \mathbf{u}) + \text{div}(\rho \mathbf{u} \mathbf{u}^T) + \nabla p = 0, \quad (37)$$

$$\partial_t(\rho E) + \text{div}(\rho E \mathbf{u} + p \mathbf{u}) = Q, \quad (38)$$

with  $Y_i$  the mass fractions of the species in phase 1,  $Z_i$  the mass fractions of the species in phase 2,  $u$  the mixture velocity,  $\rho$  the mixture density,  $p$  the mixture pressure,  $E$  the mixture total energy,  $Q$  the energy source term.

Within each phase, the species obey the Dalton's law, while the total pressure of both phases are equal:

$$\sum_{\Phi_1} p_i = \sum_{\Phi_2} p_i.$$

Let  $\rho_i$  and  $e_i$  be respectively the density and the specific internal energy of one species, then, this species equation of state reads:

$$p_i = \rho_i(\gamma_i - 1)(e_i - q_i) - \gamma_i p_i^\infty.$$

Let  $T$  be the mixture temperature, the equation of state of one species can also read:

$$p_i = \rho_i(\gamma_i - 1)C_{v,i}T - p_i^\infty,$$

where  $C_{v,i}$  is the heat capacity at constant volume.

Let  $\alpha$  be the volume fraction of the phase 1, then one considers the following relations:

$$\begin{aligned}\rho Y_i &= \alpha \rho_i, & i \in \Phi_1, \\ \rho Z_i &= (1 - \alpha) \rho_i, & i \in \Phi_2.\end{aligned}$$

The volume fraction  $\alpha$  can easily be computed from the conservative variables, the equations of state and the pressure equality.

The total energy is defined by the sum of the mixture kinetic energy and its internal energy as follows:

$$E = \frac{1}{2} |u|^2 + \sum_{\Phi_1} Y_i e_i + \sum_{\Phi_2} Z_i e_i.$$

The energy source term  $Q$  reads :

$$Q = \sum_{\Phi_1} \rho Y_i Q_i + \sum_{\Phi_2} \rho Z_i Q_i$$

where the  $Q_i$  can take one of the following forms:

1. the energy source term per species is constant, thus  $Q_i$  get constant values,
2. the energy source term is a function  $F$  of time and is defined for the whole volume of a species:

$$\int_{\mathcal{D}} \alpha_i(t, \mathbf{x}) Q_i(t, \mathbf{x}) = F(t),$$

where  $\mathcal{D}$  is the spatial domain,  $\alpha_i$  is the volume fraction of the species:

- $\alpha_i = \alpha$  if  $i \in \Phi_1$  and its mass fraction within phase 1 is above 0.9,
  - $\alpha_i = 1 - \alpha$  if  $i \in \Phi_2$  and its mass fraction within phase 2 is above 0.9,
3. the energy source term is a function  $G$  of the volume of the species ( $V_i = \int_{\mathcal{D}} \alpha_i$ ):

$$\int_{\mathcal{D}} \alpha_i(t, \mathbf{x}) Q_i(t, \mathbf{x}) = G(V_i(t)).$$

Finally, we assume that the mixture sound velocity  $c$  satisfies a Wood's relation and reads:

$$c = \left( \rho \left( \frac{\alpha}{\rho_{\Phi_1} c_{\Phi_1}^2} + \frac{1 - \alpha}{\rho_{\Phi_2} c_{\Phi_2}^2} \right) \right)^{-1/2},$$

with  $\rho_{\Phi_1} = \sum_{\Phi_1} \rho_i$ ,  $\rho_{\Phi_2} = \sum_{\Phi_2} \rho_i$ ,  $c_{\Phi_1}$  and  $c_{\Phi_2}$  are the phasic sound velocities which can be computed from the equations of states and the temperature equality.

### Outputs:

- ECR(1) : mixture pressure  $p$
- ECR(2) : mixture density  $\rho$
- ECR(3) : mixture sound velocity  $c$
- ECR(4) : velocity  $\mathbf{u}$  along x-axis
- ECR(5) : velocity  $\mathbf{u}$  along y-axis

ECR(6) : velocity  $\mathbf{u}$  along z-axis (if existing)

ECR(4 + ndim) : volume fraction of phase 1  $\alpha$

ECR(5 + ndim) : mixture temperature  $T$

ECR(6 + ndim) : mass fraction of the first species

ECR(7 + ndim + ncp1 + ncp2) : FV cell volume fraction of the first species

### Examples:

A first example can be found in the verification test named `bm_vfcc_hmem_0D.epx`. A unique FV cell is composed of two phases and two species per phase. All species are ideal gases. The FV cell is contracted, implying the isentropic compression of the mixture. Four cases are studied:

1. all species are the same ideal gas ( $q = p^\infty = 0$ ), the cell keeps a constant volume. We thus check that the initial conditions provide the same values as an ideal gas law:

$$p = p_0, \quad \rho = \rho_{1,0}, \quad c = \sqrt{\frac{\gamma_1 p_0}{\rho_0}}, \quad \alpha = \alpha_0, \quad T = T_0,$$

2. all species are the same ideal gas, the cell is contracted from volume  $V_0 = 1$  to volume  $V = 0.81$ . The final values should be the same as an ideal gas isentropic compression:

$$p = \frac{p_0}{V_1^\gamma}, \quad \rho = \frac{\rho_{1,0}}{V}, \quad c = \sqrt{\frac{\gamma_1 p}{\rho}}, \quad \alpha = \alpha_0, \quad T = \frac{T_0}{V^{\gamma-1}},$$

3. the mixture is made of two identical phases, each one composed of two ideal gases with different properties ( $\gamma_1 \neq \gamma_2$ ,  $C_{v,1} \neq C_{v,2}$ ). The cell is contracted from volume  $V_0 = 1$  to volume  $V = 0.81$ . The mixture should behave like an ideal gas with the following parameters:

$$C_v = Y_1 C_{v,1} + Y_2 C_{v,2}, \quad \gamma = \frac{Y_1 \gamma_1 C_{v,1} + Y_2 \gamma_2 C_{v,2}}{C_v},$$

$$p = \frac{p_0}{V^\gamma}, \quad \rho = \frac{\rho_0}{V}, \quad c = \sqrt{\frac{\gamma p}{\rho}}, \quad \alpha = \alpha_0, \quad T = \frac{T_0}{V^{\gamma-1}},$$

4. the mixture is made of two different phases, each one composed of a unique ideal gas ( $\gamma_1 \neq \gamma_3$ ,  $C_{v,1} \neq C_{v,3}$ ). The volume fraction  $\alpha$  is chosen so that the initial density is identical to the previous test. The cell is contracted from volume  $V_0 = 1$  to volume  $V = 0.81$ . The results must be the same as the previous test, except for the sound velocity, which reads:

$$c = \sqrt{\frac{\tilde{\gamma} p}{\rho}}, \quad \tilde{\gamma} = \left( \frac{\alpha}{\gamma_1} + \frac{1-\alpha}{\gamma_3} \right)^{-1}.$$

===== llllllll master

## 7.9 IMPEDANCES

### Object :

This directive enables impedances for elements with CLxx boundary conditions to be input.

There are three forms of the directive, depending upon whether the impedance concerns:

- Finite Elements (IMPE), or
- Elements using the Van Leer formulation (IMPV), or
- Cell-Centred Finite Volumes (VFCC) (CLVF)

The available options for IMPE are the following:

name	law of behaviour
ABSI	absorbing boundary (JRC implementation)
ABSO	absorbing boundary
ABST	total absorbing material
ABSZ	absorbing boundary (Zienkiewicz for geotechnical materials)
AIRB	air blast wave
DCRI	critical mass flow rate
DIAP	diaphragm with imposed pressure
FOND	closed bottom
FPLT	fragile plate
FSUI	following force
GRFS	grid model with fluid/structure coupling
GRIL	grid model
MEMB	safety membrane
NAH2	coupling of the water mass flow rate in sodium-water reaction
PCHA	head loss
PIMP	imposed pressure
POMP	pump model
PPLT	perforated plate (JRC implementation)
RDK2	rupture disk (JRC implementation 2)
RSEA	coupling of the water mass flow rate in sodium-water reaction (new model)
RUDI	rupture disk (JRC implementation)
RDMC	MC rupture disk (JRC implementation)
SFSI	NTNU's simplified Fluid Structure Interaction model
STAC	Stacey's 1st-order absorbing boundary (JRC implementation)
SVAL	safety valve (JRC implementation)
SWVA	swing check valve with fluid-structure coupling(EDF implementation)
VANN	closure of a valve
VISU	record data (e.g. FSI pressure) for visualization

Not all these options are available for all boundary condition elements. See the following

table :

Option	CL1D	CL2D	CL3D	CL3T	CLTU	CL22	CL3Q	CL3I
ABSI						x	x	x
ABSO	x	x	x	x	x			
ABST								
ABSZ								
AIRB		x	x	x		x	x	x
DCRI	x		x		x			
DIAP	x	x			x			
FOND					x			
FPLT						x	x	x
FSUI					x			
GRFS		x	x	x				
GRIL	x	x	x	x	x			
MEMB	x	x			x			
NAH2	x				x			
PCHA	x	x	x	x	x			
PIMP	x	x	x	x	x	x	x	x
POMP	x				x			
PPLT						x	x	x
RDK2				x		x	x	x
RDMC				x			x	
RSEA	x				x			
RUDI						x	x	x
SFSI		x	x	x		x	x	x
STAC						x	x	
SVAL						x	x	x
SWVA	x				x			
VANN	x				x			
VISU		x	x	x		x	x	x

The available options for IMPV are the following:

name	law of behaviour
<b>ABSO</b>	absorbing boundary
<b>INFI</b>	conditions at infinity for a fluid
<b>PIMP</b>	imposed pressure
<b>DEGP</b>	imposed mass flow rate for a perfect gas
<b>MUR</b>	total reflexion (rigid obstacle)

Not all these options are available for all boundary condition elements. See the following table:

Option	CL1D	CL2D	CL3D	CL3T	CLTU	CL22	CL3Q	CL3I
INFI		x						
ABSO	?	?	?	?	?			

PIMP		?		?		?		?		?							
DEGP		?		?		?		?		?							
MUR		?		?		?		?		?							
-----		-----		-----		-----		-----		-----		-----		-----		-----	

The available options for CLVF are the following:

name	law of behaviour
<b>ABSO</b>	absorbing boundary
<b>INFI</b>	Conditions at infinity for a fluid
<b>PIMP</b>	Imposed pressure
<b>DEBI</b>	Imposed mass flow rate
<b>ESUB</b>	Imposed mass flow rate, sub-sonic inflow
<b>LOD1</b>	Lodi, quasi-1D condition
<b>FOUR</b>	Fourier modes in 2D
<b>RIEM</b>	Riemann in 3D
<b>LODG</b>	Lodi 3D with generalized coordinates
<b>CARM</b>	Lodi 3D with multi-dimensional characteristics
<b>ASYM</b>	Asymptotic 3D
<b>LIBR</b>	Free impedance (can be programmed by the user)
<b>FOND</b>	Closed bottom
<b>PFCT</b>	Pressure function of time
<b>DCRI</b>	Critical mass flow rate
<b>DIAP</b>	Diaphragm
<b>SVAL</b>	Safety valve

Not all these options are available for all boundary condition elements. See the following table:

Option		CL1D		CL2D		CL3D		CL3T		CLTU		CL22		CL3Q		CL3I	
-----		-----		-----		-----		-----		-----		-----		-----		-----	
ABSO																	
INFI																	
PIMP																	
DEBI																	
ESUB																	
LOD1																	
FOUR																	
RIEM																	
LODG																	
CARM																	
ASYM																	
LIBR																	
FOND																	
DCRI																	
DIAP																	
SVAL																	
-----		-----		-----		-----		-----		-----		-----		-----		-----	

Syntax :

"IMPE"	\$	"ABSO"	. . .	\$
	\$	"PCHA"	. . .	\$
	\$	"PIMP"	. . .	\$
	\$	"DIAP"	. . .	\$
	\$	"GRIL"	. . .	\$
	\$	"MEMB"	. . .	\$
	\$	"GRFS"	. . .	\$
	\$	"NAH2"	. . .	\$
	\$	"RSEA"	. . .	\$
	\$	"VANN"	. . .	\$
	\$	"SWVA"	. . .	\$
	\$	"DCRI"	. . .	\$
	\$	"FOND"	. . .	\$
	\$	"FSUI"	. . .	\$
	\$	"POMP"	. . .	\$
	\$	"PPLT"	. . .	\$
	\$	"RUDI"	. . .	\$
	\$	"STAC"	. . .	\$
	\$	"RDMC"	. . .	\$
	\$	"SVAL"	. . .	\$
	\$	"RDK2"	. . .	\$
	\$	"ABSI"	. . .	\$
	\$	"AIRB"	. . .	\$
	\$	"SFSI"	. . .	\$
	\$	"FPLT"	. . .	\$
	\$	"VISU"	. . .	\$
"IMPV"	\$	"ABSO"	. . .	\$
	\$	"INFI"	. . .	\$
	\$	"PIMP"	. . .	\$
	\$	"DEGP"	. . .	\$
	\$	"MUR"	. . .	\$
"CLVF"	\$	"ABSO"	. . .	\$
	\$	"INFI"	. . .	\$
	\$	"PIMP"	. . .	\$
	\$	"DEBI"	. . .	\$
	\$	"ESUB"	. . .	\$
	\$	"LOD1"	. . .	\$
	\$	"FOUR"	. . .	\$
	\$	"RIEM"	. . .	\$
	\$	"LODG"	. . .	\$
	\$	"CARM"	. . .	\$
	\$	"ASYM"	. . .	\$
	\$	"LIBR"	. . .	\$
	\$	"FOND"	. . .	\$
	\$	"PFCT"	. . .	\$
	\$	"DCRI"	. . .	\$
	\$	"DIAP"	. . .	\$
	\$	"SVAL"	. . .	\$

**Comments :**

These key-words can be repeated as many times as necessary with different options each time (if need be). The keyword `IMPE`, `IMPV` or `CLVF` should not be separated from the options : `ABSO` , `PCHA` , `PIMP` , etc.

Beware that the `CLVF` models are still experimental and under development. Only the `ABSO`, `LOD1`, `FOUR` and `RIEM` models have been tested somewhat (the `ABSO` being by far the most used one), but are for the moment available only in 3D, for perfect gas material and for first-order in space and in time VFCC formulations. Furthermore, they may function with only some of the flux solvers available. For an overview of the state of the art of these developments see references [\[937\]](#) and [\[938\]](#).



### 7.9.1 ABSORBING MATERIAL

#### Object :

This option enables to specify absorbing or partially absorbing boundary conditions for 1-D, 2-D or 3-D elements.

This model is appropriate for CLxx elements developed at CEA, namely CL1D, CL2D, CL3D or CL3T. There exists a similar, but not identical, absorbing boundary model developed at JRC which is appropriate for their CLxx elements (CL22, CL3I or CL3Q), see page C.880.

Only pressure waves normal to the boundary are absorbed. The model consists simply in applying a fictitious external pressure  $p = -\rho c v_n$ , where  $\rho$  is the density of the material at the boundary,  $c$  its sound speed and  $v_n$  the normal component of the particle velocity at the boundary, in Lagrangian calculations, or of the relative (particle minus mesh) velocity in Eulerian or ALE calculations. The “internal” forces due to the absorbing boundary are finally computed by spatial integration of a modified pressure  $\pi = (p + p_{\text{old}})/2$ , where  $p_{\text{old}}$  is the value of  $\pi$  at the previous time integration step.

#### Syntax:

```
"IMPE"  "ABSO"  <"R0" rho>  <"C" c>    /LECTURE/
```

#### rho

Fixed, user-imposed value of the density. If omitted, the code will try to determine the density automatically.

#### c

Fixed, user-imposed value of the sound speed. If omitted, the code will try to determine the sound speed automatically. Since for a structural material the code is sometimes unable to determine the sound speed automatically, the value becomes useful in this case (and is of course constant). However, since the physical sound speed is fairly constant in such a case, the behaviour of the model should be quite good. A notable exception are materials CAMC and CLAY, for which the sound speed varies considerably: for these materials the code is indeed able to retrieve the current sound speed automatically, so that specifying  $c$  in these cases is unnecessary.

#### LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T)

#### Comments :

If the acoustic waves are to be absorbed, the `rho` and `c` parameters must be the same on both sides of the boundary. The effect will then be that of an infinite medium. In the opposite case, there will be partial reflections.

If the user has omitted `c` and the code is unable to determine it automatically, an error message is issued and the calculation is stopped.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : density times normal velocity in the local reference frame

### 7.9.2 TOTAL ABSORBING MATERIAL

#### Object :

This option enables to specify absorbing boundary conditions for 3D elements. This model is based on a paraxial formulation and is available for CL3D or CL3T elements.

Pressure waves normal to the boundary and shearing waves parallel to the boundary are absorbed. The model consists simply in applying a fictitious external pressure  $p = -\rho(c_p v_n + c_s v_p)$ , where  $\rho$  is the density of the material at the boundary,  $c_p$  its longitudinal sound speed,  $c_s$  its transverse sound speed,  $v_n$  the normal component of the particle velocity at the boundary and  $v_p$  the parallel component, in Lagrangian calculations only. The “internal” forces due to the absorbing boundary are finally computed by spatial integration of a modified pressure  $\pi = (p + p_{\text{old}})/2$ , where  $p_{\text{old}}$  is the value of  $\pi$  at the previous time integration step.

#### Syntax:

```
"IMPE"  "ABST"  <"RO" rho>  <"CP" cp>  <"CS" cs> /LECTURE/
```

**rho**

Fixed, user-imposed value of density. If omitted, the code will try to determine it automatically.

**cp**

Fixed, user-imposed value of the longitudinal sound speed. If omitted, the code determines it using material parameters of the neighboring element through the following formula:

$$c_p = \sqrt{(\lambda + 2G)/\rho}$$

with

$$G = E/(2(1 + \nu)), \lambda = E\nu/((1 + \nu)(1 - 2\nu))$$

**cs**

Fixed, user-imposed value of the transverse sound speed. If omitted, the code determines it using material parameters of the neighboring element through the following formula:

$$c_s = \sqrt{G/\rho}$$

**LECTURE**

Reading procedure of the elements composing the boundary (CL3D or CL3T)

#### Comments:

This model works correctly only if 3D-elements in the neighborhood of the boundary have a linear elastic behavior and are provided with the laws LINE or VMJC.

**Warning:** If the user has omitted  $c_p$  and(or)  $c_s$  and the neighbor is not provided with the "LINE" or "VMJC" laws, the code is unable to determine it automatically. An error message is issued and the calculation is stopped.

**Outputs:**

The different components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : component of shearing in the first direction of boundary plan

ECR(3) : component of shearing in the second direction of boundary plan

ECR(4) : density

ECR(5) : density times normal velocity in the local reference frame

ECR(6) : density times normal velocity in the local reference frame

ECR(7) : density times first component of parallel velocity in the local reference frame

ECR(8) : density times second component of parallel velocity in the local reference frame

### 7.9.3 HEAD LOSS

**Object :**

This instruction enables localized head losses to be input.

**Syntax :**

```
"IMPE"  "PCHA"  "R0" rho  "K"  k  /LECTURE/
```

rho

Density.

k

Coefficient of a localized head loss.

LECTURE

Reading procedure of the element numbers (CL1D, CL2D or CL3D).

**Comments :**

The head loss (DP) is deduced from the density (rho) and the velocity (V) up-stream of the singularity:

$$DP = 0.5 * k * rho * V * V$$

The result is a resisting force which is always opposed to the velocity.

It is evident that this model may not be applied to the extremities of pipelines.

**Outputs:**

The different components of the ECR are as follows:

ECR(1) : pressure

ECR(2) : density

#### 7.9.4 GRID

**Object:**

This instruction enables to model the influences of grids or rigid perforated plates on a fluid.

See also on page C4.650 directive GRFS.

**Syntax:**

```
"IMPE"  "GRIL"  "R0" rho  "C" c  "ALP" alpha  ...  
        ...  "TAU" tau  /LECTURE/
```

rho

Density.

c

Velocity of sound in the fluid.

alpha

Dissipative impedance.

tau

Time constant.

LECTURE

Reading procedure of the element numbers (CL2D or CL3D).

**Comments:**

The model is based on the hypothesis of an acoustic propagation of plane waves.

The meaning of the parameters alpha and tau is as follows:

Let:

L : equivalent length of the grid holes

ST : total cross section

s : cross section open to flow

M : Mach number of the permanent flow up-stream of the singularity

xi : head loss coefficient

Then:

$$\alpha = \xi * M \quad \tau = \frac{ST}{s} * \frac{L}{2*c}$$

### Remarks:

The ratio  $s / ST$  represents the perforation ratio of the plate.

In the steady-state regime the pressure drop across the plate assumes the form:

$$DP = \alpha * \rho * c * V$$

Recall that for an absorbing boundary  $\alpha = i$ , since the pressure and mass flow rate fluctuations are in quadrature.

The equivalent length  $L$  is not equal to the plate thickness. To take into account three-dimensional effects, add a length equivalent to the diameter of the holes to that thickness.

### Warning:

This directive allows to transmit the stresses from the plate to the fluid. It allows to modify the flow by taking into account the pressure drops introduced by a perforated plate. The directives "IMPE" "GRFS" should be used in order to correctly transfer the fluid stresses onto an equivalent shell structure representing the plate.

For an A.L.E. computation with a fluid-structure coupling of the perforated plate, the user has to mention twice the "FS2D" elements, which link together the plate and the fluid : first, in the instruction "GRILLE" "ALE" ... "FS" ..., in order to make the fluid nodes follow the motion of the grid; and then in the instruction "LIAISON" "FS" ..., in order to transmit correctly the forces of the fluid towards the plate. The fluid has to be meshed continuously through the plate. In this case, the elements "FS2D" are located inside the fluid and not in its surrounding area. All the other "FS2D" elements, normally located at the boundaries of the fluid, are mentioned only once in the instruction "GRILLE" "ALE" ... "FS" ..., for in A.L.E. the fluid-structure coupling is done automatically on the boundaries.

In a Lagrangian computation, for the same mesh this problem does not exist, because only the following instruction exists : "LIAS" "FS"...

### Outputs :

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

### 7.9.5 IMPOSED PRESSURE

#### Object:

This instruction enables a pressure at the boundary of different elements to be imposed by the means of a "CLxx" element.

#### Syntax:

```
"IMPE"  "PIMP"  < "RO" rho >  "PRES" pres                ...
          ... < "PREF" pref >    < "ENTH" enth >            ...
          ... < $[ "FONC" nufo ;                               ...
          ... obsolete:  "TABP" npt*( t , p ) ]$ >          ...
          ... /LECTURE/
```

rho

Density.

pres

Constant imposed pressure, or multiplying factor for the ordinates of the following table or function if it exists.

pref

Reference pressure. Note that if a zero reference pressure is desired, it is mandatory to specify 'PREF 0' in the input file. This is because, if no value for PREF is specified, the code assumes  $PREF = PRES(t=0)$  so the initial imposed pressure has no effect, since  $p = PRES - PREF = 0$ !

enth

Input enthalpy.

npt

Number of points defining the pressure curve.

t , p

Coordinates of a point on the curve (time , pressure).

nufo

Number of the function to be used to describe the pressure versus time.

LECTURE

Reading procedure of the number of the "CLxx" element defining the boundary.



**Comments:**

For the meaning of pref, see [GBC\\_0300](#).

If the input enthalpy is zero, the code uses the enthalpy value of the neighbour element. Otherwise, the user's imposed value is taken into account. This possibility works only in Van Leer.

**ATTENTION !** The keyword "TABP", that introduces the time function of the pressure, is obsolete. Use preferably the directive "FONC". Keyword "TABP" is maintained just to ensure compatibility with old input data sets.

The keywords "TABP" and "FONC" are mutually exclusive.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(9) : enthalpy per unit volume (Van Leer only)

### 7.9.6 DIAPHRAGM

**Object:**

This directive introduces, at the end of a pipeline, a diaphragm that causes a localised pressure drop, and an external imposed pressure.

**Syntax:**

```
"IMPE"  "DIAP"  "RO" rho  "PFIN" pfin  <"PREF" pref> ...  
        ...  "PINI" pini  "TAU" tau  "K" k  /LECTURE/
```

rho

Density.

pfin

External imposed pressure in steady-state flow.

pref

Reference pressure.

pini

Initial pressure at the diaphragm level.

tau

Time constant of the exponential function that transforms pinit into pfin.

k

Diaphragm head loss coefficient ( $>1$  or  $= 1$ ).

LECTURE

Reading procedure of the CL1D element forming the boundary.

**Comments:**

The meaning of pref is given on page C.300.

The imposed pressure passes from pini to pfin following an exponential function whose time constant is tau. It is possible to take  $\tau = 0$  to represent an abrupt change (depressurisation). However, by giving  $\tau \neq 0$  it is possible to simulate a finite opening time of the diaphragm.

The K coefficient allows to account for the diaphragm cross section, that may be smaller than the tube diameter.

$$K = \sup( K_o , \text{ksi} * R * R )$$

$\begin{array}{c} \text{-----} \\ | \_ | \\ S \quad \_ \quad s \\ \text{-----} | \_ | \end{array}$

R is the ratio of cross-sections:  $R = S / s$

$$\begin{aligned} \text{ksi} = & 1 + 0.5 * ( 1 - 1/R ) \\ & + \text{eta} * \text{sqr} ( 1 - 1/R ) \\ & + \text{lambda} * \text{long} / D_h \end{aligned}$$

$K_o$  is the coefficient corresponding to  $S = s$  :  $1.06 < K_o < 1.10$

The preceding formula is taken from IDEL'CIK for the openings with a thick diaphragm, and for large Reynolds numbers.

The eta coefficient varies along with the ratio  $\text{long} / D_h$ . It passes from 1.35 to 0 when this ratio goes from 0 to 2.4.

The long parameter is the length of the diaphragm (thickness of the bottom), and  $D_h$  the hydraulic diameter ( $h = 4 * \text{area} / \text{perimeter}$ ).

The lambda parameter allows to define the head loss along the small tube equivalent to the diaphragm.

If the distribution of velocities at the outlet would be uniform, one would have  $K_o = 1$ .

Most often, it may suffice to use the IDEL'CIK formula for a thin diaphragm with sharp border:

$$\text{ksi} = (1 + 0.707 * \text{sqrt}( 1 - 1/R ) ) ** 2$$

By full opening, take  $K = K_o$ , with  $K_o = 1.06$ .

### Outputs:

The various components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : density

ECR(3) : mass velocity ( $\rho * v$ )

### 7.9.7 MEMBRANE

**Object:**

Introduces a safety membrane to the extremity of a pipeline (1D), or on the axis of an axisymmetric reservoir (2D). The membrane rupture occurs: either when the pressure in the neighbouring element exceeds the rupture pressure, or when the time exceeds a prescribed value.

**Syntax:**

```
"IMPE"  "MEMB"  "RO" rho  "PINI" pini  "PFIN" pfin  ...  
... <"PREF" pref>  "TAU" tau  "K" k  $[ "PRUP" prup ;  
                                     "TRUP" trup ]$ /LECTURE/
```

rho

Density.

pini

Initial pressure.

pfin

Imposed external pressure in steady-state flow.

pref

Reference pressure.

tau

Time constant of the exponential function that leads pini to pfin.

k

Head loss coefficient ( $>1$  or  $= 1$ ).

prup

Rupture pressure of the membrane.

trup

Rupture instant.

LECTURE

List of the concerned elements.

**Comments:**

In the "GEOM" directive, the elements with a material "MEMB" must be listed after the adjacent fluid elements.

**Outputs:**

The various components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : density

ECR(3) : mass velocity

ECR(5) : rupture instant of the membrane

ECR(6) : rupture indicator (0=intact, 1=broken)

### 7.9.8 CRITICAL MASS FLOW RATE

**Object:**

Computes the critical mass flow rate for a perfect gas or for a water-steam mixture at the extremity of a pipeline (1D).

**Syntax:**

```
"IMPE"  "DCRI"  "PINI" pini  "PFIN" pfin < "PREF" pref >  ...  
        "RAP" rapport  < "KSI" ksiz >  < "MOD" mode >      ...  
        < "TAU" tau >  < "TRUP" trupt >  < "FONC" nufo >      ...  
        /LECTURE/
```

**pini**

Initial pressure.

**pfin**

Imposed external pressure in steady-state conditions.

**pref**

Reference pressure.

**rapport**

Ratio of the cross-sections :  $S(\text{exit})/S(\text{upstream})$ .

**ksiz**

Head loss coefficient at the outlet for the fluid.

**mode**

Choice of the critical mass flow rate model (for the two-phase water only : see below)

**tau**

Time constant of the exponential function leading from pini to pfin.

**trupt**

Rupture instant of the membrane. By default, trupt = 0.

**nufo**

function number for a progressive opening.

**LECTURE**

List of the concerned elements.

**Comments:**

The user may choose among four modes for the equilibrated two-phase water, which differ in the phase-to-phase sliding:

mode = 1 : homogeneous model  
 mode = 2 : MOODY model  
 mode = 3 : FAUSKE model  
 mode = 4 : DEMENT model (developed by M. Lepareux)

For the three first modes, the results do not depend upon the size of the opening. On the contrary, the DEMENT mode developed by Michel Lepareux [692] accounts for this effect provided the following value is not exceeded (rapport=ratio):

$$\text{rapport} < \text{or} = 0.75$$

For the liquids, the **ksiz** coefficient allows to account for the form of the outlet. By default, the value suggested by IDEL'CIK is retained for **ksiz**, corresponding to a straight pipe outlet and a thin diaphragm with sharp sides, i.e.:

$$ksiz = (1 + 0.707\sqrt{1 - rapport})^2$$

Of course, with the data of the upstream medium, the formula becomes:

$$ksi = \frac{ksiz}{rapport^2}$$

The old data input files included a parameter K, equal to 'ksi' with ksiz = 1. In order to better reproduce the phenomena, it is suggested to no longer use K, and to declare the two parameters 'ksiz' and 'rapport' instead.

## Outputs:

The different components of the ECR table are as follows:

ECR(1) : pressure at the opening  
 ECR(2) : density of the donor element  
 ECR(3) : mass velocity ( $Q = \rho \cdot V_n$ ) of the donor  
 ECR(4) : outlet pressure (Pext or Pcrit)  
 ECR(6) : indicator (0 if liquid; 1 if gas; mass title if two-phase mixture)  
 ECR(7) : Mach number at the outlet (only for perfect gases and two-phase water)  
 ECR(9) : maximum mass velocity (Qmax), if ECR(4) = Pcrit  
 ECR(10) : indicator (0 = virgin membrane, 1 = ruptured membrane)

### 7.9.9 CLOSED BOTTOM

**Object:**

This option allows to impose a closed bottom condition at the extremity of a pipeline, by means of a CLTU element.

**Syntax:**

```
"IMPE"  "FOND"  /LECTURE/
```

**LECTURE**

List of the elements concerned.

**Comments:**

This directive automatically ensures the fluid-structure coupling between the pipe and the internal fluid.

**Warning**

The couplings between degrees of freedom are done directly in the CLTU element, therefore it is necessary that the number of the CLTU element concerned be greater than all other elements that arrive in the same point (TUBE, TUYA, POUT). This in order to have, at the moment of computing the coupling, the resultant of the applied forces. A very simple way of proceeding is to declare the CLTU elements as the last ones in the GEOM directive.



### 7.9.10 PUMP

**Object:**

This option allows to impose on a pipeline, by means of a CL1D or CLTU element, a pump model represented by its characteristic curve.

**Syntax:**

```
"IMPE"  "POMP"  "RO" rho  "COEF" coef  "NUFO" nf  /LECTURE/
```

rho

Density.

coef

Multiplicative coefficient allowing to define the functioning direction of the pump, and to convert the units.

nf

Number of the function to be used to describe the characteristic curve of the pump as a function of the volume flow rate.

LECTURE

List of the concerned elements.

**Comments:**

The pressure difference is of the form:

$$\Delta P = C f(Q_v)$$

where  $C$  is coef,  $Q_v$  is the volume flow rate and  $f$  the function that models the characteristic.

**Warning**

The characteristic curve of the pump must be given by means of the keyword **FUNCTION** in the following way: in abscissa the volume flow rate and in ordinate the height of the pump. Be careful that the chosen units are coherent! For example, pump height in Pascal and flow rate in  $m^3s^{-1}$ . In the case of a normal functioning of the pump, all these values (pressure and flow rate) are positive.

The coefficient **COEF** allows to use different units for the pressures, and its sign allows to orient the pump as a function of the positive orientation associated with the node where the pump is located.

Thus, if the numbers of the elements located at either part of the node where the pump is located are in growing order along the direction of the flow, then the **COEF** coefficient is positive.

If the flow is in the opposite direction than the normal (the fluid flows back through the pump), it is assumed that the head loss is zero. Furthermore, in accident calculations, it is prudent to foresee a characteristic (even zero) for very large flow rates.

### **Outputs:**

The different components of the ECR table are as follows:

ECR(1) : pressure difference

ECR(2) : density

ECR(3) : mass velocity

**7.9.11 FOLLOWING FORCE****Object:**

This directive allows to impose, at the extremity of a pipeline, by means of a CLTU element, a “following force” applied along the direction of the last element of the pipeline.

**Syntax:**

```
"IMPE"  "FSUI"  "RO" rho  "FORC" fs  "NUFO" nf  /LECTURE/
```

rho

Density.

fs

Multiplicative coefficient for the force.

nf

Number of the function used to describe the force as a function of time.

LECTURE

List of the concerned elements.

**Comments:**

A positive force will be directed towards the interior of the pipe.

The function to be used will be defined by means of the "FONC" directive described on page E.10.

**Outputs:**

The various components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : density

ECR(3) : following force

**7.9.12 CRITICAL MASS FLOW RATE COUPLING (NAH2)****Object:**

Computation of the critical mass flow rate of a water-steam mixture at the extremity of a pipeline (1D) accounting for the pressure produced by the hydrogen bubble generated by a sodium-water reaction. This material uses a model with 2 components (sodium and hydrogen) and two phases. The starting instant of the tube opening may be prescribed in advance (case of several pipes whose rupture times are different)

**Syntax:**

```
"IMPE"  "NAH2"  "R0" rho  "PINI" pini  <"PREF" pref>
...  "TAU" tau  "K" k  "TUBE" ntube  "MOD" mode
...    <"TRUP" trup> <"PFIN" pfin> /LECTURE/
```

rho

Density.

pini

Initial pressure.

pref

Reference pressure.

tau

Time constant of the exponential (see DIAP page C.550).

k

Head loss coefficient (single-phase flow).

ntube

Total number of ruptured tubes.

mode

Choice of the critical flow rate model.

trup

Rupture instant (zero by default).

pfin

Imposed final pressure.

**LECTURE**

List of the concerned elements.

**Comments:**

The user may choose among four modes:

mode = 1 : homogeneous model

mode = 2 : MOODY model

mode = 3 : FAUSKE model

mode = 4 : DMT model (developed by M. Lepareux)

This directive is similar to "DCRI" and must be used in conjunction with the "NAH2" material within a "TUBE" or "TUYA" element.

**Outputs:**

The various components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : density

ECR(3) :  $Q = \rho \cdot V_n$  at the rupture (mass velocity).

ECR(4) : pressure at the outlet ( $P_{ext}$  or  $P_{crit}$ )

ECR(6) : indicator (0= single-phase ; 1= two-phase)

ECR(7) : status of the tube (0 = intact, 1 = ruptured)

ECR(9) :  $Q_{max} = (\rho \cdot V_n)$  maximum if one is in the critical regime. This parameter only makes sense for water or for gases.

### 7.9.13 CRITICAL MASS FLOW RATE COUPLING (RSEA)

**Object:**

Computation of the critical mass flow rate of a water-steam mixture at the extremity of a pipeline (1D) accounting for the pressure produced by the hydrogen bubble generated by a sodium-water reaction. This material is similar to the previous one ("NAH2" page C4.710), but is coupled to the "RSEA" material, which uses three components (sodium, hydrogen and argon) and 2 phases.

**Syntax:**

```
"IMPE"  "RSEA"  "RO" rho  "PINI" pini  <"PREF" pref>
...  "TAU" tau  "K" k  "TUBE" ntube  "MOD" mode
...    <"TRUP" trup> <"PFIN" pfin> /LECTURE/
```

rho

Density.

pini

Initial pressure.

pref

Reference pressure.

tau

Time constant of the exponential (see DIAP page C.550).

k

Head loss coefficient (single-phase flow).

ntube

Total number of ruptured tubes.

mode

Choice of the critical flow rate model.

trup

Rupture instant (zero by default).

pfin

Final prescribed pressure.

**LECTURE**

List of the elements concerned.

**Comments:**

The user may choose among four modes:

mode = 1 : homogeneous model

mode = 2 : MOODY model

mode = 3 : FAUSKE model

mode = 4 : DMT model (developed by M. Lepareux)

This directive is similar to the "DCRI" directive and must be used in conjunction with the "RSEA" material within a "TUBE" or "TUYA" element.

**Outputs:**

The various components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : volumetric mass

ECR(3) :  $Q = \rho \cdot V_n$  at the rupture (mass velocity).

ECR(4) : outlet pressure ( $P_{ext}$  or  $P_{crit}$ )

ECR(6) : indicator (0 = single-phase ; 1 = two-phase)

ECR(7) : tube status (0 = intact, 1 = ruptured)

ECR(9) :  $Q_{max} = (\rho \cdot V_n)$  maximum if one is in the critical regime. This parameter makes sense only for water or for a gas.

### 7.9.14 “VANNE” - SAFETY AND REGULATING VALVES

#### Object:

This directive allows the user to model the behaviour of a safety and a regulating valve (“vanne”) placed within a pipeline or at its extremity. It introduces a localised pressure drop with a variable pressure loss coefficient depending on the opening cross section of the valve. This impedance, available for the elements of type CL1D and CLTU, is useful to model water hammer effects (“coup de belier”) in the pipeline systems.

For a safety valve, the beginning and the duration of the closure must be specified. For a regulating valve, the user has to specify a tabulated function allowing to govern the valve opening during the calculation (partial or complete opening or closure).

#### Syntax:

for a safety valve:

```
"IMPE"  "VANN"  "TFER" tferm  "TAU" tau  "SMIN" smin  "PREF" pref
/LECTURE/
```

for a regulating valve:

```
"IMPE"  "VANN"  "SMIN" smini  "PREF" pref  "NFSL" numfo
/LECTURE/
```

**tferm**

Initial time of the valve closure.

**tau**

Duration of the closure.

**smin**

Minimum cross section, below which the valve is considered as closed.

**pref**

Reference pressure.

**numfo**

Number of the function prescribing the time variation of the ratio  $\frac{S_{free}}{S_{upstream}}$ .

**LECTURE**

List of the elements concerned.

#### Comments:

The parameters **SMIN** and **PREF** are compulsory.



**Safety valve:**

Initially the valve is considered to be open. It starts to close at time **tferm**, and the closure has a duration of **tau**. It is assumed that the opening cross-section varies linearly between time **tferm** and **tferm + tau**, at which the valve is completely closed.

During the closure, the pressure drop varies as a function of the opening cross-section:

$$K(t) = \left( \frac{S_{upstream}}{S_{free}} \right)^2 - 1$$

Its value therefore varies from zero (fully open valve) to  $K_{max}$ , which is determined by assuming that the closure is complete when  $S_{free} < \mathbf{SMIN}$ . In most cases the value 0.01 may be assumed for **SMIN**.

Once the valve closed, a blockage condition (CL1D) or closed end condition (CLTU) is applied to the node to which the element is attached. In order to know whether the valve is open or closed, EUROPLEXUS checks the value of ECR(6).

**Regulating valve:**

For the regulating valve a tabulated function given by the user prescribes the time variation of the valve cross section ( $\frac{S_{free}}{S_{upstream}}$  ratio) allowing thus to govern the valve opening during the calculation.

**Outputs:**

The components of the ECR table are as follows:

ECR(1) : pressure drop

ECR(2) : density of the upstream element

ECR(4) : ratio of cross-sections ( $\frac{S_{free}}{S_{upstream}}$ )

ECR(6) : if =0 the valve is open, if =1 the valve is closed

ECR(7) : time

ECR(8) : pressure drop coefficient  $K(t)$

### 7.9.15 SWING CHECK VALVE WITH FLUID-STRUCTURE COUPLING

**Object:**

This directive allows the user to enter along a pipeline an anti-backflow valve which can close rapidly when the flow is inversed. The valve disc dynamics is governed by its angular inertia as well as by different moments due to the disc weight and hydrodynamic forces. The disc motion causes a localised pressure drop taken into account via a variable head loss coefficient depending on the aperture of the valve disc. The model is attached to CL1D elements.

**Syntax:**

```
"IMPE"  "SWVA"  "MASS" mass <"RO"  rho>  "ITOT" itot  <"PREF" pref>
          "STUB" stub "DIST" dist <"AINI" aini> <"AMAX" amax>
          <"POPE" pope> "FNUM" nume /LECTURE/
```

**mass**

Mass of the valve disc.

**rho**

Fluid density.

**itot**

Total moment of inertia accounting for the disc and added fluid inertia.

**pref**

Reference pressure.

**stub**

Flow-tube cross section.

**dist**

Length of the valve disc moment arm.

**aini**

Initial opening angle.

**amax**

Maximum opening angle.

**pope**

Opening pressure.

nume

Number of the singular head loss  $K(\alpha)$  curve.

LECTURE

Reading procedure of the CL1D element forming the boundary.

**Comments:**

The meaning of pref is given on page C.300.

The head loss (DP) is deduced from the density ( $\rho$ ) and the velocity (V) up-stream of the singularity and it is function of variable head loss coefficient depending on the aperture angle of the disc:

$$DP = 0.5 * k(\alpha) * \rho * V * V$$

The result is a resisting force which is always opposed to the velocity.

The integration of the disc motion equation is abandoned after closure of the valve and EUROPLEXUS replaces the boundary condition by a zero velocity condition (case of CL1D). The valve can reopen when DP exceeds the opening pressure specified by the user.

**Outputs:**

The various components of the ECR table are as follows:

ECR(1) : pressure drop

ECR(2) : fluid density

ECR(3) : mass velocity ( $\rho * v$ )

ECR(4) : reference pressure

ECR(5) : current angle of disc (degrees)

ECR(6) : current angular velocity of disc (rad/s)

ECR(7) : current time then  $t+dt$

ECR(8) : current head loss coefficient

**7.9.16 FLUIDE-STRUCTURE GRID****Object:**

This directive allows to model for a fluid the influence of grids or perforated plates and to apply the resulting pressure drop to the structure (grid).

See also on page C.530 the GRILLE directive.

**Syntax:**

```
"IMPE"  "GRFS"  "RO" rho  "C" c  "ALP" alpha  ...  
        ...  "TAU" tau  /LECTURE/
```

rho

Density.

c

Sound speed in the fluid.

alpha

Dissipative impedance.

tau

Time constant.

LECTURE

Lecture procedure of the elements concerned.

**Comments:**

The model assumes that plane acoustic waves are propagated.

The meaning of the alpha and tau parameters is as follows:

Let:

L : equivalent length of the grid holes

ST : total cross-section

s : flow cross-section

M : Mach number of the permanent upstream flow

k : head loss coefficient

Then:

$$\alpha = 0.5 * k * M \qquad \tau = \frac{ST}{s} * \frac{L}{2*c}$$

### Remarks:

The ratio  $s / ST$  represents the perforation ratio of the plate.

The head loss coefficient  $k$  takes the form:

$$DP = 0.5 * k * \rho * V^2$$

Recall that for an absorbing boundary  $\alpha = i$ , since the pressure and flow rate fluctuations are in quadrature.

The equivalent length  $L$  is not equal to the plate thickness. To account for three-dimensional effects, to this thickness a further length must be added, of the order of the orifice diameter.

### Warning:

EUROPLEXUS automatically searches for the structure nodes that ‘touch’ the boundary condition elements. It is then mandatory to mesh the structure in the same way as the boundary condition elements (same mesh density and same topology of the faces in contact).

### Outputs:

The various components of the ECR table are as follows:

ECR(1) : pressure

ECR(2) : density

### 7.9.17 PERFORATED PLATE (JRC)

**Object:**

This instruction enables the modelling of a perforated structure (e.g. a plate) embedded in a fluid.

The pressure drop across the plate is computed according to the expression:

$$\text{Deltap} = \text{zeta} * \text{rho} * \text{v} * \text{v} / 2.0$$

Here zeta is the resistance coefficient, rho the fluid density and v the velocity normal to the plate in the undisturbed upstream region of the fluid.

**Syntax :**

```
"IMPE"  "PPLT"  "ZETA" zeta  
/LECTURE/
```

zeta

Resistance coefficient (assumed as constant in the present model).

/LECT/

Concerned elements.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : current pressure drop across the plate

ECR(2) : density upstream the plate

ECR(3) : resistance coefficient

ECR(4) : structural node 1

ECR(5) : structural node 2

ECR(6) : structural node 3

ECR(7) : structural node 4

ECR(8) : unused

ECR(9) : unused

Note that the positions 4 to 7 contain up to 4 indexes of the structural nodes corresponding to the CLxx element's (fluid) nodes. These quantities are determined only once at the beginning of the calculation and never change. Furthermore, the resistance coefficient is also assumed as constant in the present implementation.

**Comments:**

Normally, the determination of the above mentioned node correspondence is performed automatically. However, in case of problems the user may either change the tolerance for node matching (see OPTI TOLC on page H.40) or force the node correspondence by using the directive COMP CNOD, see page C.92.

### 7.9.18 RUPTURE DISK (JRC)

#### Object:

This instruction enables the modelling of a rupture disk structure (e.g. a plate) embedded in a fluid.

The disk reacts to the pressure drop caused by a fluid on the two sides of the disk, until a certain  $\Delta p$  is reached, which causes the rupture of the disk.

Usually, the disk is modelled by a series of structural finite elements (plate, shell) and the fluid on both sides is discretized by continuum finite elements.

Special boundary condition elements (CLxx) are attached to the fluid nodes in order to connect the disk with the fluid. These elements are assigned the present IMPE RUDI material.

The disk rupture is caused by a fixed (nominal)  $\Delta p$ , which can occur at any point of the disk, in case this is discretized by several finite elements. When one element reaches this condition, failure is initiated also in all other elements at the same time.

A certain rupture time interval can be prescribed from failure initiation to failure completion in order to simulate the fact that in reality the disk is a mechanical system with inertia and can not break instantaneously.

Unlike in the models of safety valves, note that once the disk rupture is initiated at a certain point, it continues until full failure even if the  $\Delta p$  is reduced.

#### Syntax :

```
"IMPE"  "RUDI"  "DPRU" dpru      < "TRUP" trup >  
/LECTURE/
```

#### dpru

Pressure difference ( $\Delta p$ ) between the two sides that causes rupture of the disk. The  $\Delta p$  is measured between the fluid elements directly attached to the disk on each side. The  $\Delta p$  includes only the fluid pressure and not the pseudo-viscous pressure. However, the pseudo-viscous pressure is taken into account in the computation of the resistance forces developed by the disk.

#### trup

Rupture time interval of the disk. By default it is  $\text{trup} = 0$  (instantaneous rupture). This can be specified  $> 0$  in order to simulate the inertia of the disk and the gradual opening of the orifice.

```
/LECT/
```



Concerned elements.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : current pressure drop across the disk (at this element), doesn't include the pseudoviscous pressure

ECR(2) : 0 for virgin disk, 1 for broken disk

ECR(3) : time at which failure is initiated

ECR(4) : structural node 1

ECR(5) : structural node 2

ECR(6) : structural node 3

ECR(7) : structural node 4

ECR(8) : unused

ECR(9) : unused

Note that the positions 4 to 7 contain up to 4 indexes of the structural nodes corresponding to the CLxx element's (fluid) nodes. These quantities are determined only once at the beginning of the calculation and never change.

**Comments:**

Normally, the determination of the above mentioned node correspondence is performed automatically. However, in case of problems the user may either change the tolerance for node matching (see OPTI TOLC on page H.40) or force the node correspondence by using the directive COMP CNOD, see page C.92.

**7.9.19 STACEY'S 1ST ORDER ABSORBING BOUNDARY (JRC)****Object:**

This instruction enables the modelling of an absorbing boundary according to Stacey's 1st-order law, see R. Stacey, "Improved transparent boundary formulations for the elastic-wave equation", Bull. Seis. Soc. Am., Vol. 78, pp. 2089-2097, December 1988.

The model is only available in conjunction with spectral elements and can be applied only to CL22 elements, in 2D, and CL3Q elements, in 3D.

**Syntax :**

```
"IMPE"  "STAC"  /LECTURE/
```

```
/LECT/
```

Concerned elements.

**Outputs:**

The components of the ECR table are as follows. If the material belongs to a CL22 element, then:

```
ECR(1) : unused  
ECR(2) : unused  
ECR(3) : unused  
ECR(4) : unused  
ECR(5) : unused  
ECR(6) : unused  
ECR(7) : unused  
ECR(8) : unused  
ECR(9) : unused
```

If the material belongs to a CL3Q element, then:

```
ECR(1) : x-component of 1st tangent vector  
ECR(2) : y-component of 1st tangent vector  
ECR(3) : z-component of 1st tangent vector  
ECR(4) : x-component of 2nd tangent vector  
ECR(5) : y-component of 2nd tangent vector
```

ECR(6) : z-component of 2nd tangent vector

ECR(7) : x-component of normal vector (outwards the associated MS38)

ECR(8) : y-component of normal vector (outwards the associated MS38)

ECR(9) : z-component of normal vector (outwards the associated MS38)

**Remarks:**

It has been noted that the use of this type of absorbing boundary conditions can in some cases reduce the stability step. If needed, the safety coefficient of the calculation may be reduced (from the default value of 0.5) by using the directive "OPTI CSTA", see Group H (Options).

Usually it has been found that OPTI CSTA 0.25 is sufficient to prevent instabilities (especially in 2D), but in some 3D cases it has been necessary to use OPTI CSTA 0.125.

### 7.9.20 RUPTURE DISK FOR MC FORMULATION (JRC)

**Object:**

This instruction enables the modelling of a rupture disk structure (e.g. a plate) embedded in a fluid modeled with the MC finite volume formulation.

The disk reacts to the pressure drop caused by a fluid on the two sides of the disk, until a certain  $\Delta p$  is reached, which causes the rupture of the disk.

The mechanical behaviour (deformability) of the disk is not taken into account, so there is no need to introduce structural elements.

The fluid mesh is not continuous, i.e. the rupture disk separates two distinct zones of the fluid domain, and the elements facing each other have coincident (same coordinates) fluid nodes on the discontinuity.

Special boundary condition elements (CLxx) are attached to the fluid element at one of the two sides of the aforementioned discontinuity, no matter which one of them. These elements are assigned the present IMPE RDMC material.

The disk rupture is caused by a fixed (nominal)  $\Delta p$ , which can occur at any point of the disk, in case this is discretized by several CLxx elements. When one element reaches this condition, failure is initiated also in all other elements at the same time.

Unlike in the models of safety valves, note that once the disk rupture is initiated at a certain point, it continues until full failure even if the  $\Delta p$  is reduced.

**Syntax :**

```
"IMPE"  "RDMC"  "DPRU" dpru
```

```
/LECTURE/
```

dpru

Pressure difference ( $\Delta p$ ) between the two sides that causes rupture of the disk. The  $\Delta p$  is measured between the fluid elements directly attached to the disk on each side.

```
/LECT/
```

Concerned elements.

**Outputs :**

The different components of the ECR table are as follows :

- ECR(1) : current pressure drop across the disk (at this element)
- ECR(2) : 0 for virgin disk, 1 for broken disk
- ECR(3) : time at which failure is initiated
- ECR(4) : fluid node opposite to CLxx element node 1
- ECR(5) : fluid node opposite to CLxx element node 2
- ECR(6) : fluid node opposite to CLxx element node 3
- ECR(7) : fluid node opposite to CLxx element node 4
- ECR(8) : index of fluid element attached to this CLxx
- ECR(9) : index of fluid element opposed to this CLxx

Note that the positions 4 to 7 contain up to 4 indexes of the fluid nodes opposite to the CLxx element's (fluid) nodes. These quantities are determined only once at the beginning of the calculation and never change.

**Comments:**

Normally, the determination of the above mentioned node correspondence is performed automatically. However, in case of problems the user may either change the tolerance for node matching (see OPTI TOLC on page H.40) or force the node correspondence by using the directive COMP CNOD, see page C.92.

**7.9.21    ABSORBING MATERIAL VAN LEER****Object :**

This directive enables an absorbing boundary conditions for Van Leer elements to be input.

**Syntax:**

```
"IMPV"   "ABSO"   "R0" rho /LECTURE/
```

rho

Density.

LECTURE

Reading procedure of the numbers of the elements concerned.

**7.9.22 CONDITION AT INFINITY VAN LEER****Object:**

This directive allows to impose a rest condition at infinity for a fluid by means of a CL2D element.

**Syntax:**

```
"IMPV" "INFI" "R0" ro "PRES" press "PREF" pref "GAMA" ga /LECTURE/
```

R0

Density at infinity.

PRES

Pressure at infinity.

PREF

Reference pressure.

GAMA

Value of the ratio of specific values for perfect gases.

LECTURE

List of the elements concerned.

**Comments:**

The coupling with the GZPV material is automatically ensured.

### 7.9.23 IMPOSED PRESSURE VAN LEER

**Object:**

This instruction enables a pressure at the boundary of different Van Leer elements to be imposed by the means of a "CLxx" element.

**Syntax:**

```
"IMPV"  "PIMP"  "RO" rho "PRES" pres < "PREF" pref >  
... "GAMA" gamma < "IMPO" impo > /LECTURE/
```

rho

Density.

pres

Constant imposed pressure.

pref

Reference pressure.

gama

Ratio of specific heats.

impo

The value impo=0 corresponds to free pressure, density and velocity (absorbing), impo=1 to constant entropy and mass flow rate, and impo=2 to constant entropy and velocity.

**Comments:**

For the meaning of pref, see page C.300.



### 7.9.24 IMPOSED PERFECT GAS MASS FLOW RATE VAN LEER

**Object:**

This directive allows to impose a mass flow rate of perfect gas in case of Van Leer elements.

**Syntax:**

```
"IMPV" "DEGP" "R0" ro "PRES" press "PREF" pref "GAMA" ga  
"DEBX" dx "DEBY" dy "DEBZ" dz /LECTURE/
```

ro

Initial density.

press

Imposed initial pressure.

pref

Reference pressure.

ga

Value of the ratio of specific values for perfect gases.

dx dy dz

Components of mass flow rate.

LECTURE

List of the elements concerned.

**7.9.25 RIGID OBSTACLE VAN LEER****Object:**

This directive allows to impose a total reflection (rigid obstacle) condition in case of Van Leer elements.

**Syntax:**

"IMPV" "MUR" /LECTURE/

**LECTURE**

List of the elements concerned.

### 7.9.26 SAFETY VALVE (JRC)

#### Object

This instruction enables the modelling of fluid discharge from a pressurized vessel through a generic safety valve or (by a suitable choice of the parameters) through an orifice. The fluid passing through the valve or orifice can be either incompressible (typically a liquid) or compressible (a gas).

Alternatively, the model can also be used in the opposite flow direction in order to fill up (pressurize) a vessel from an external source. Therefore, we distinguish between two modes: **discharge** mode or **loading** mode. The two modes cannot be combined in the same device, since no reverse flow is allowed. However, one could of course attach two **SV** devices, one in discharge mode and the other in loading mode, to the same vessel (at different locations along the wall) if needed.

The input syntax includes also a set of (optional) parameters that allow a sort of “pressure regulation” functionality whereby the opening of the valve is subjected to a series of constraints which may also depend upon the (independent) pressure in a “master” fluid element not belonging to the zone of the fluid domain to which the valve is attached.

#### A - Discharge mode

The valve is characterized by three main parameters:

- $A^{\text{tube}}$ , the area of the tube in which the valve is mounted;
- $\Delta p^{\text{min}}$ , the pressure difference (between internal and external pressures) at which the valve starts to open;
- **mode** the functioning mode of the device: 0 means discharge mode, 1 means loading mode. The default value is 0, so this parameter may be omitted if discharge mode is required.

Other **optional** parameters are:

- $\Delta p^{\text{max}}$ , the pressure difference at which the valve is fully open, by default  $\Delta p^{\text{max}} = \Delta p^{\text{min}}$ ;
- $\Delta t^{\text{ope}}$ , the time interval needed to completely open the valve under a positive step increment of the pressure difference  $\Delta p > \Delta p^{\text{max}}$ , by default  $\Delta t^{\text{ope}} = 0$ ;
- $\Delta t^{\text{clo}}$ , the time interval needed to completely close the valve under a negative step increment of the pressure difference  $\Delta p < -\Delta p^{\text{max}}$ , by default  $\Delta t^{\text{clo}} = 0$ ;
- $q^{\text{max}}$ , the maximum mass flow rate through the valve, by default  $q^{\text{max}} = \infty$ ;
- $p^{\text{ext}}$ , the external pressure (assumed constant), by default  $p^{\text{ext}} = 0$ ;
- $\rho^{\text{ext}}$ , the external density (assumed constant); this quantity must be specified in loading mode, but it must not be specified in discharge mode.
- $i^{\text{ext}}$ , the external specific internal energy (assumed constant); this quantity must be specified in loading mode, but it must not be specified in discharge mode.
- $A^{\text{max}}$ , the maximum opening area of the valve which must be  $A^{\text{max}} \leq A^{\text{tube}}$ , by default  $A^{\text{max}} = A^{\text{tube}}$ ;

- $C_c$ , the contraction coefficient of the fluid flow through the valve opening, i.e. the ratio between the area of the flow in the *vena contracta* and the opening area of the valve (assumed constant, independent of the current opening area), by default  $C_c = 1$ ;
- $\gamma$ , the ratio of specific heats  $\gamma = C_p/C_v$  in case of compressible fluid flow. If omitted, the fluid being discharged is considered as incompressible;
- $t^{\text{act}}$ , the activation time of the valve. The valve is guaranteed to be inactive (closed) for  $t < t^{\text{act}}$ , whatever be the conditions of the fluid. If not specified, the code assumes  $t^{\text{act}} = -\infty$ .
- $t^{\text{dea}}$ , the de-activation time of the valve. The valve is guaranteed to be inactive (closed) for  $t > t^{\text{dea}}$ , whatever be the conditions of the fluid. If not specified, the code assumes  $t^{\text{dea}} = \infty$ .
- **emas**, the *master* fluid element whose pressure  $p^{\text{mas}}$  has to be monitored for pressure regulation purposes. Note that the pressure  $p^{\text{mas}}$  is independent from the opening or not of the current valve, i.e. it is not influenced (at least directly) from the opening or not of the valve. If specified, the master element should therefore belong to a zone of the fluid domain not directly connected with the fluid part to which the current valve is attached (but the code does not check this requirement).
- **esla**, the *slave* fluid element whose pressure  $p^{\text{sla}}$  has to be monitored for pressure regulation purposes. Note that the pressure  $p^{\text{sla}}$  directly depends upon the opening or not of the current valve. If specified, the slave element should therefore belong to the zone of the fluid domain directly connected with the fluid part to which the current valve is attached (but the code does not check this requirement).
- $p^{\text{dis}}$ , the value of pressure in the slave element that, when reached, triggers the discharge through the current valve, i.e. starts the opening of the valve. This parameter must not be specified in loading mode. By default  $p^{\text{dis}} = \infty$ .
- $p^{\text{max}}$ , the value of pressure in the slave element that, when reached, triggers the closing of the current valve. This parameter must not be specified in discharge mode. By default  $p^{\text{max}} = \infty$ .
- $p^{\text{ms1}}$ , the value of pressure difference between the master and the slave elements at which the valve starts to open (in an attempt to keep the pressure difference below the chosen value:  $\Delta p = p^{\text{mas}} - p^{\text{sla}} \leq p^{\text{ms1}}$ .) By default  $p^{\text{ms1}} = \infty$ .
- $p^{\text{ms2}}$ , the value of pressure difference between the master and the slave elements at which the valve becomes fully open. If  $\Delta p = p^{\text{mas}} - p^{\text{sla}} \geq p^{\text{ms2}}$  then the valve is fully open, in an attempt to keep the pressure difference below the chosen value. By default  $p^{\text{ms2}} = p^{\text{ms1}}$ .
- **rval**, a *reference* valve (**SVAL**) element. The reference valve must be in discharge mode and the current valve must be in loading mode. If the opening area of the reference valve is  $a^{\text{ref}} > 0$ , then the (current) valve closes immediately ( $a = 0$ ).

This model is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC.

## References

More information on the formulation of this model may be found in reference [94].

## Syntax

```

"IMPE"  "SVAL"  <"MODE" mode> "ATUB" atub "DPMI" dpmi
        < "DPMA" dpma "TOPE" tope "TCLO" tclo "QMAX" qmax
        "PEXT" pext "ROEX" roex "IEXT" iext
        "AMAX" amax "CC"   cc   "GAMM" gamm "TACT" tact
        "TDEA" tdea
        "EMAS" /LECT_emas/ "ESLA" /LECT_esla/
        "PDIS" pdis "PMAX" pmax "PMS1" pms1 "PMS2" pms2
        "RVAL" /LECT_rval/                                     >
/LECTURE/

```

## mode

Functioning mode of the device: 0 means discharge mode, 1 means loading mode. If omitted, the code assumes value 0 (discharge mode).

## atub

Cross-section of the safety valve flow tube.

## dpmi

Pressure difference at which the valve starts to open.

## dpma

Pressure difference at which the valve is fully open; by default, it is equal to  $\Delta p^{\min}$ .

## tope

Opening time interval; by default, it is 0.

## tclo

Closure time interval; by default, it is 0.

## qmax

Maximum mass flow rate; if omitted, by default it is infinite (no limit).

## pext

External pressure (assumed constant); by default it is 0.

## roex

External density (assumed constant); this quantity must be specified in loading mode, but it must not be specified in discharge mode.

## iext

External specific internal energy (assumed constant); this quantity must be specified in loading mode, but it must not be specified in discharge mode.

## amax

Maximum opening area of the valve. It must be  $A^{\max} \leq A^{\text{tube}}$ . By default, it is  $A^{\max} = A^{\text{tube}}$ .

cc

Flow contraction coefficient ( $0 < C_c \leq 1$ ). By default it is  $C_c = 1$ .

gamm

Ratio of specific heats for the fluid being discharged ( $\gamma = C_p/C_v$ ). If specified, the fluid is considered as compressible. If omitted, the fluid is considered as incompressible ( $\gamma$  is unused in that case).

tact

Activation time of the valve. The valve is guaranteed to be inactive (closed) for  $t < t^{\text{act}}$ , whatever be the conditions of the fluid. If not specified, the code assumes  $t^{\text{act}} = -\infty$ .

tdea

De-activation time of the valve. The valve is guaranteed to be inactive (closed) for  $t > t^{\text{dea}}$ , whatever be the conditions of the fluid. If not specified, the code assumes  $t^{\text{dea}} = \infty$ .

EMAS

Introduces the definition (/LECT\_emas/) of the *master* fluid element whose pressure  $p^{\text{mas}}$  has to be monitored for pressure regulation purposes. Note that the pressure  $p^{\text{mas}}$  is independent from the opening or not of the current valve, i.e. it is not influenced (at least directly) from the opening or not of the valve. If specified, the master element should therefore belong to a zone of the fluid domain not directly connected with the fluid part to which the current valve is attached (but the code does not check this requirement).

ESLA

Introduces the definition (/LECT\_esla/) of the *slave* fluid element whose pressure  $p^{\text{sla}}$  has to be monitored for pressure regulation purposes. Note that the pressure  $p^{\text{sla}}$  directly depends upon the opening or not of the current valve. If specified, the slave element should therefore belong to the zone of the fluid domain directly connected with the fluid part to which the current valve is attached (but the code does not check this requirement).

pdis

The value of pressure  $p^{\text{dis}}$  in the slave element that, when reached, triggers the discharge through the current valve, i.e. starts the opening of the valve. This parameter must not be specified in loading mode. By default,  $p^{\text{dis}} = \infty$ .

pmax

The value of pressure  $p^{\text{max}}$  in the slave element that, when reached, triggers the closing of the current valve. This parameter must not be specified in discharge mode. By default,  $p^{\text{max}} = \infty$ .

pms1

The value of pressure difference  $p^{\text{ms1}}$  between the master and the slave elements at which the valve starts to open (in an attempt to keep the pressure difference below the chosen value:  $\Delta p = p^{\text{mas}} - p^{\text{sla}} \leq p^{\text{ms1}}$ .) By default,  $p^{\text{ms1}} = \infty$ .

pms2

The value of pressure difference  $p^{\text{ms2}}$  between the master and the slave elements at which the valve becomes fully open. If  $\Delta p = p^{\text{mas}} - p^{\text{sla}} \geq p^{\text{ms2}}$  then the valve is fully open, in an attempt to keep the pressure difference below the chosen value. By default,  $p^{\text{ms2}} = p^{\text{ms1}}$ .

RVAL

Introduces the definition (/LECT\_rval/) of a *reference* valve (SVAL) element. The reference valve must be in discharge mode and the current valve must be in loading mode. If the opening area of the reference valve is  $a^{\text{ref}} > 0$ , then the (current) valve closes immediately ( $a = 0$ ).

LECTURE

Reading procedure of the number of the "CLxx" element defining the boundary.

### Determination of the current opening area of the valve

The calculation of the current opening area of the valve consists of three phases. In the *first phase*, we check whether at the current time  $t$  the valve is active or not: if  $t < t^{\text{act}}$  or  $t > t^{\text{dea}}$  then the valve is closed irrespective of fluid conditions ( $a = 0$ ) and we skip the following two phases. Then, if rval has been specified, we also check the opening area of the reference valve: if  $a^{\text{ref}} > 0$  then the current valve is immediately closed ( $a = 0$ ) and we skip the following two phases.

Otherwise ( $t^{\text{act}} \leq t \leq t^{\text{dea}}$ ), the valve is active. The *second phase* consists of checking whether there are any pressure regulation constraints specified for the current valve, since such constraints override the normal opening calculations to be performed in the third phase. If no slave element esla was specified, there are no regulation constraints and so we go directly to the third phase. Otherwise there are some constraints, which depend upon the chosen functioning mode of the device.

- In *discharge mode*, if  $p^{\text{dis}}$  was given and  $p^{\text{sla}} \geq p^{\text{dis}}$ , then we set  $a = A$  (valve fully open) and skip the third phase.
- In *loading mode*, if  $p^{\text{max}}$  was given and  $p^{\text{sla}} \geq p^{\text{max}}$ , then we set  $a = 0$  (valve fully closed) and skip the third phase. Otherwise, we check any constraints on the master-slave pressure difference. If  $p^{\text{ms1}}$  was given, we compute the pressure difference  $\Delta p = p^{\text{mas}} - p^{\text{sla}}$ : if  $\Delta p < p^{\text{ms1}}$  the valve is completely closed ( $a = 0$ ); else if  $\Delta p > p^{\text{ms2}}$  the valve is completely open ( $a = A$ ); else  $p^{\text{ms1}} \leq \Delta p \leq p^{\text{ms2}}$  and the valve is only partially open ( $a = \frac{\Delta p - p^{\text{ms1}}}{p^{\text{ms2}} - p^{\text{ms1}}} A$ ). Having computed  $a$ , we skip the third phase. Otherwise ( $p^{\text{ms1}}$  was not given) we go to the next phase.

The *third and last phase* is as follows. The *nominal* opening area of the valve  $a'$  is assumed to be a linear function of the differential pressure  $\Delta p$  between the internal fluid ( $p$ ) and the external medium ( $p^{\text{ext}}$ ):

$$\Delta p = p - p^{\text{ext}}$$

$$\begin{aligned} &\text{for } \Delta p \leq \Delta p^{\text{min}} & a' &= 0 \\ &\text{for } \Delta p^{\text{min}} < \Delta p < \Delta p^{\text{max}} & a' &= \frac{\Delta p - \Delta p^{\text{min}}}{\Delta p^{\text{max}} - \Delta p^{\text{min}}} A \\ &\text{for } \Delta p \geq \Delta p^{\text{max}} & a' &= A \end{aligned}$$

The *actual* opening area of the valve  $a$  at a given time depends on the opening or closure times, and on the valve opening reached at the previous time,  $a^{\text{old}}$ . Let  $v_a$  and  $v_c$  represent the velocity of aperture and closure of the valve:

$$\begin{aligned} v_a &= A/\Delta t^{\text{ope}} \\ v_c &= -A/\Delta t^{\text{clo}} \end{aligned}$$

Then:

$$\begin{aligned} \text{for } a' > a^{\text{old}} \text{ and } \Delta t^{\text{ope}} > 0 & \quad a = \min[(a^{\text{old}} + v_a \Delta t), a'] \\ \text{for } a' < a^{\text{old}} \text{ and } \Delta t^{\text{clo}} > 0 & \quad a = \max[(a^{\text{old}} + v_c \Delta t), a'] \\ \text{in all other cases} & \quad a = a' \end{aligned}$$

The value of  $a$  is constrained to be between 0 and  $A$ :

$$\begin{aligned} \text{if } a > A & \quad a = A \\ \text{if } a < 0 & \quad a = 0 \end{aligned}$$

In order to compute the mass flow rate through the valve, we distinguish two cases, depending upon the compressibility of the discharged fluid.

### Incompressible or nearly incompressible fluid

The fluid being discharged is incompressible or nearly incompressible (e.g., a liquid). This situation is determined by the fact that the user has *not* specified a value for  $\gamma = C_p/C_v$  in the input data. In this case the following relation is assumed to hold:

$$\Delta p = p - p^{\text{ext}} = \zeta \rho \frac{v_{\text{tube}}^2}{2} \quad (39)$$

where  $\zeta$  is a dimension-less *resistance coefficient* which depends upon the geometry of the valve,  $\rho$  is the density of the fluid upstream the valve and  $v_{\text{tube}}$  is the velocity of the fluid in the tube upstream the valve.

By assuming steady state flow and a jet that expands back to fill the entire cross-section of the tube downstream the valve, the resistance coefficient  $\zeta$  is evaluated by:

$$\zeta = \left( \frac{1}{r_p C_c} - 1 \right)^2$$

where  $r_p$  is the current *perforation ratio*  $r_p$  of the valve, given by:

$$r_p = a/A^{\text{tube}}$$

Since the pressure drop  $\Delta p$  is known (we assume  $\Delta p = p - p^{\text{ext}}$ , without any limitations), from (39) we obtain the (nominal) fluid velocity in the tube upstream the valve:

$$v'_{\text{tube}} = \sqrt{\frac{2}{\zeta} \frac{p - p^{\text{ext}}}{\rho}}$$

and the current (nominal) mass flow rate is given by:

$$q' = A^{\text{tube}} \rho v'_{\text{tube}} = A^{\text{tube}} \sqrt{\frac{2}{\zeta} (p - p^{\text{ext}}) \rho}$$



### Compressible fluid

The fluid being discharged is compressible. This situation is determined by the fact that the user has specified a value for  $\gamma = C_p/C_v$  in the input data. (The given value of  $\gamma$ , assumed constant, should be equal to that of the fluid being discharged, i.e. the fluid upstream the safety valve.)

In this case the code assumes that the expansion of the fluid across the valve can be represented by a simple adiabatic transformation:

$$\frac{p}{\rho^\gamma} = \text{const.} \quad (40)$$

This is reasonable for a perfect gas but not, for example, if the fluid is a superheated vapor that becomes saturated during the expansion.

We compute the current *critical pressure*  $p_c$  corresponding to the current pressure  $p$  upstream the valve and to the chosen value of  $\gamma$ :

$$p_c = p \left( \frac{2}{\gamma + 1} \right)^{\frac{\gamma}{\gamma - 1}}$$

For example, if  $\gamma = 1.4$  like for air,  $p_c/p = 0.528$ . The pressure in the valve orifice (i.e., in the zone where the cross-section area of the duct is minimum) cannot drop below the critical value. We therefore distinguish two cases, depending on the value of the external pressure  $p^{\text{ext}}$  with respect to  $p_c$ :

- if  $p^{\text{ext}} \geq p_c$ , then the fluid pressure in the valve orifice is  $p_2 = p^{\text{ext}}$ , the fluid velocity in the orifice is

$$v_2 = \sqrt{2 \frac{\gamma}{\gamma - 1} \frac{p}{\rho} \left[ 1 - \left( \frac{p_2}{p} \right)^{\frac{\gamma - 1}{\gamma}} \right]}$$

and the (nominal) mass flow rate is given by

$$q' = S_2 v_2 \rho_2 = S_2 v_2 \rho \left( \frac{p_2}{p} \right)^{\frac{1}{\gamma}} = S_2 \sqrt{2 \frac{\gamma}{\gamma - 1} p \rho \left[ \left( \frac{p_2}{p} \right)^{\frac{2}{\gamma}} - \left( \frac{p_2}{p} \right)^{\frac{\gamma + 1}{\gamma}} \right]}$$

where  $p$  and  $\rho$  are the pressure and density of the fluid upstream the orifice, respectively, use has been made of (40) to explicitate the density  $\rho_2$  at the orifice and  $S_2$  is the flow area of the *vena contracta* at the orifice  $S_2 = C_c a$ .

- if  $p^{\text{ext}} < p_c$ , then the fluid pressure in the valve orifice is equal to the critical pressure  $p_2 = p_c$  and the (nominal) mass flow rate is given by

$$q' = \psi S_2 \sqrt{p \rho}$$

where  $S_2$  has the same meaning as above and the coefficient  $\psi$  is given by

$$\psi = \left( \frac{2}{\gamma + 1} \right)^{\frac{1}{\gamma - 1}} \sqrt{2 \frac{\gamma}{\gamma + 1}}$$

### Flow limitation and velocity

Finally, in both cases (incompressible or compressible flow) the current (effective) mass flow rate  $q$  is computed by taking into account a possible user-imposed maximum value  $q^{\max}$  and by inhibiting any back flow through the valve:

$$\begin{array}{ll} \text{for } q^{\max} > 0 \text{ and } q' > q^{\max} & q = q^{\max} \\ \text{for } q' < 0 & q = 0 \text{ (no back flow)} \\ \text{in all other cases} & q = q' \end{array}$$

The current (effective) fluid velocity in the upstream tube is then:

$$v_{\text{tube}} = \frac{q}{A^{\text{tube}} \rho}$$

### Modelling of an orifice

In order to model a simple orifice in the wall of the pressurized tank, set  $A^{\text{tube}}$  to the area of the orifice and  $\Delta p^{\min}$  to a very small value.

As concerns the optional parameters, leave out  $\Delta p^{\max}$  and  $A^{\max}$  so that it will be  $\Delta p^{\max} = \Delta p^{\min}$  and  $A^{\max} = A^{\text{tube}}$ , leave  $\Delta t^{\text{ope}}$  to its default value of 0 and specify a huge value for  $\Delta t^{\text{clo}}$  so that the orifice will never close in practice.

### B - Loading mode

Loading mode is activated by specifying MODE 1 in the input data. In this mode,  $p^{\text{ext}}$  (assumed constant) must be specified since the default value of 0 is not appropriate. In order to completely define the external state (which in this case becomes the donor state) one must also give  $\rho^{\text{ext}}$  and  $i^{\text{ext}}$  (which are also assumed to be constant).

### Outputs:

The different components of the ECR table are as follows :

- ECR(1) : current pressure in the fluid element to which the valve is attached
- ECR(2) : current nominal opening area
- ECR(3) : current actual opening area
- ECR(4) : previous time this element was treated
- ECR(5) : previous actual opening area
- ECR(6) : total ejected mass (in discharge mode) or injected mass (in loading mode)
- ECR(7) : total ejected energy by mass transport (in discharge mode) or injected energy (in loading mode)
- ECR(8) : current flow tube velocity
- ECR(9) : current density in the fluid element to which the valve is attached
- ECR(10) : current specific internal energy in the fluid element to which the valve is attached

### 7.9.27 RUPTURE DISK (JRC NEW)

#### Object

This instruction enables the modelling of a rigid rupture disk structure (e.g. a plate) embedded in a fluid.

This model is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC.

The disk reacts to the pressure drop caused by a fluid on the two sides of the disk, until a certain  $\Delta p$  is reached, which causes the rupture of the disk.

In this particular model the disk structure is NOT represented, while the fluid on both sides is discretized by continuum finite elements.

Special boundary condition elements (CLxx) are attached to the fluid nodes in order to represent the action of the disk on the fluid. These elements are assigned the present IMPE RDK2 material.

The disk rupture is caused by a fixed (nominal)  $\Delta p$ , which can occur at any point of the disk, in case this is discretized by several CLxx elements. When one CLxx element reaches this condition, failure is initiated also in all other elements at the same time.

As long as the disk is not ruptured, it acts as a rigid boundary with infinite friction. In other words, all fluid nodes attached to the disk are completely blocked.

The blocking forces are instantly removed as the disk breaks.

#### References

More information on the formulation of this material model may be found in reference [\[131\]](#).

#### Syntax

```
"IMPE"  "RDK2"  "DPRU" dpru  /LECTURE/
```

#### dpru

Pressure difference ( $\Delta p$ ) between the two sides that causes rupture of the disk. The  $\Delta p$  is measured between the fluid elements directly attached to the disk on each side. The  $\Delta p$  includes only the fluid pressure and not the pseudo-viscous pressure.

/LECT/

Concerned elements.

**Outputs :**

The different components of the ECR table are as follows :

ECR(1) : current pressure drop across the disk (at this element), doesn't include the pseudoviscous pressure

ECR(2) : 0 for virgin disk, 1 for broken disk

ECR(3) : time at which failure occurred

ECR(4) : unused

ECR(5) : unused

ECR(6) : unused

ECR(7) : unused

ECR(8) : unused

ECR(9) : unused

### 7.9.28 ABSORBING MATERIAL (JRC implementation)

#### Object :

This option enables to specify absorbing or partially absorbing boundary conditions for 2-D or 3-D elements developed at JRC (CL22, CL3I or CL3Q).

There exists a similar, but not identical, absorbing boundary model developed at CEA which is appropriate for their CLxx elements (CL1D, CL2D, CL3D or CL3T), see page C.610.

Only pressure waves normal to the boundary are absorbed. The model consists simply in applying a fictitious external pressure  $p = -\rho cv_n$ , where  $\rho$  is the density of the material at the boundary,  $c$  its sound speed and  $v_n$  the normal component of the particle velocity at the boundary, in Lagrangian calculations, or of the relative (particle minus mesh) velocity in Eulerian or ALE calculations. The “internal” forces due to the absorbing boundary are finally computed by spatial integration of the pressure  $p$  (and not, like in the IMPE ABSO material, of a modified pressure  $\pi = (p + p_{old})/2$ , where  $p_{old}$  is the value of  $\pi$  at the previous time integration step).

#### Syntax:

```
"IMPE"  "ABSI"  <"R0" rho>  <"C" c>  /LECTURE/
```

#### rho

Fixed, user-imposed value of the density. If omitted, the code will try to determine the density automatically.

#### c

Fixed, user-imposed value of the sound speed. If omitted, the code will try to determine the sound speed automatically. Since for a structural material the code is sometimes unable to determine the sound speed automatically, the value becomes useful in this case (and is of course constant). However, since the physical sound speed is fairly constant in such a case, the behaviour of the model should be quite good. A notable exception are materials CAMC and CLAY, for which the sound speed varies considerably: for these materials the code is indeed able to retrieve the current sound speed automatically, so that specifying  $c$  in these cases is unnecessary.

#### LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL22, CL3I or CL3Q).

#### Comments :

If the acoustic waves are to be absorbed, the `rho` and `c` parameters must be the same on both sides of the boundary. The effect will then be that of an infinite medium. In the opposite case, there will be partial reflections.

If the user has omitted `c` and the code is unable to determine it automatically, an error message is issued and the calculation is stopped.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity in the local reference frame

**7.9.29 ABSORBING MATERIAL (Zienkiewicz for geotechnical materials)****Object :**

This option enables to specify absorbing or partially absorbing boundary conditions for 2-D or 3-D elements developed at JRC (CL22, CL3I or CL3Q).

The formulation of this model is due to Zienkiewicz (Computational Geomechanics) and is related to geotechnical materials only.

Both the pressure waves normal to the boundary and those tangential to the boundary are absorbed. The model consists simply in applying a fictitious external pressure  $p = -\rho cv$ , where  $\rho$  is the density of the material at the boundary,  $c$  its sound speed and  $v$  the appropriate component (normal or tangential) of the particle velocity at the boundary (in Lagrangian calculations). The “internal” forces due to the absorbing boundary are finally computed by spatial integration of the pressure  $p$  (and not, like in the IMPE ABSO material, of a modified pressure  $\pi = (p + p_{\text{old}})/2$ , where  $p_{\text{old}}$  is the value of  $\pi$  at the previous time integration step).

For the normal component one has:

$$p_n = \rho c_n v_n$$

where  $v_n$  is the particle velocity normal to the boundary and the normal sound speed  $c_n$  is computed by:

$$c_n = \sqrt{k/\rho}$$

where  $k$  is given by:

$$k = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}$$

.

For the tangential component (2 components in 3D) one has:

$$p_t = \rho c_t v_t$$

where  $v_t$  is the particle velocity tangential to the boundary and the tangential sound speed  $c_t$  is computed by:

$$c_t = \sqrt{G/\rho}$$

where  $G$  (the shear modulus) is given by:

$$G = \frac{E}{2(1+\nu)}$$

.

**Syntax:**

```
IMPE ABSZ <R0 rho> <CN cn> <CT ct> /LECTURE/
```

**rho**

Fixed, user-imposed value of the density. If omitted, the code will try to determine the density automatically.

**cn**

Fixed, user-imposed value of the normal sound speed. If omitted, the code will try to determine the sound speed automatically. Since for a structural material the code is sometimes unable to determine the sound speed automatically, the value becomes useful in this case (and is of course constant). However, since the physical sound speed is fairly constant in such a case, the behaviour of the model should be quite good. A notable exception are materials **CAMC** and **CLAY**, for which the sound speed varies considerably: for these materials the code is indeed able to retrieve the current sound speed automatically, so that specifying **cn** in these cases is unnecessary.

**ct**

Fixed, user-imposed value of the tangential sound speed. In 3D, this is the tangential speed in both tangential directions. If omitted, the code will try to determine the sound speed automatically. Since for a structural material the code is sometimes unable to determine the sound speed automatically, the value becomes useful in this case (and is of course constant). However, since the physical sound speed is fairly constant in such a case, the behaviour of the model should be quite good. A notable exception are materials **CAMC** and **CLAY**, for which the sound speed varies considerably: for these materials the code is indeed able to retrieve the current sound speed automatically, so that specifying **cn** in these cases is unnecessary.

#### LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL22, CL3I or CL3Q).

#### Comments :

If the acoustic waves are to be absorbed, the **rho**, **cn** and **ct** parameters must be the same on both sides of the boundary. The effect will then be that of an infinite medium. In the opposite case, there will be partial reflections.

If the user has omitted **cn** or **ct** and the code is unable to determine them automatically, an error message is issued and the calculation is stopped.

#### Outputs:

The different components of the ECR table are as follows :

ECR(1) : normal pressure

ECR(2) : density

ECR(3) : normal sound speed

ECR(4) : normal velocity



ECR(5) : tangential sound speed

ECR(6) : tangential velocity

ECR(7) : tangential pressure

### 7.9.30 AIR BLAST WAVE

#### Object :

This directive simulates an explosion in the air (see References below). It allows to load the structures without having to model the fluid domain. It does not take into account **multiple** wave reflections on structural walls, but optionally allows to take into account in a very simplified way the first wave reflection at a wall (see below).

The position of the charge may be specified either by giving its coordinates ( $x$ ,  $y$  and  $z$ ) or the node at which the charge is placed.

#### Syntax:

```
"IMPE"  "AIRB"  |[ "X" x "Y" y <"Z" z> ; "NODE" /LEC1/ ]|
            "MASS" m $[ "TINT" t ; "TAUT" ]$ <"OPOS">
            <"ANGL">
            <"CUBE">
            <"COEF" cf>
            <"CONF" c>
            <"DECA" d>
            <"PMAx" pmax "TD" td "B" b>
            /LECTURE/
```

**x**

X-coordinate of the explosive source.

**y**

Y-coordinate of the explosive source.

**z**

Z-coordinate of the explosive source. This is 0 by default.

**NODE /LEC1/**

Introduces the node where the explosive charge is located. Typically, a PMAT element may be located at the charge position, so as to be able to visualize it.

**m**

Mass of the explosive in Kilograms.

**t**

Starting time of the explosion. By default it is equal to the initial time of the calculation.

**TAUT**

Indicates that the starting time is calculated automatically by the code, in such a way that the air blast wave reaches the first CLxx element shortly after the starting of the calculation. This is to avoid an “idle” calculation at the beginning of the transient.

**OPOS**

Indicates that only the part with the positive pressure (overpressure) is regarded. After the time of duration of the positive phase the pressure is set to 0.

**ANGL**

Indicates that the angle of incidence between the charge and the structural element is considered.

**CUBE**

Indicates that the cubic approach will be used for the calculation of the negative phase. By default the bilinear approach is used.

**cf**

The user can input a value to calibrate the decay coefficient of the air blast load. The calculated decay coefficient is multiplied by the inserted value in order to produce a load closer to experimental data.

**c**

Choice between different available explosion models, see the References below. By default it is 1 (unconfined, reflected, Kingery). The term “unconfined” below means that the explosion takes place in an unconfined space, as opposed to “half-confined” where the charge is placed close to a rigid ground and so the wave propagation occurs in a half-space (experimentally, the measured pressure is somewhat lower in this case because some of the energy is absorbed by the ground). The term “reflected” hereafter means that the model accounts for the pressure increase due to (first) wave reflection at a rigid wall as it is typically measured in experiments. The pressure value in this case may be between 2 and 8 times the incident pressure in the “non-reflected” case, i.e. without taking into account this first reflection.

1. unconfined (full space), reflected (Kingery)
2. unconfined (full space), not reflected (Kingery)
3. unconfined (full space), not reflected (Kinney)
4. half-confined (half space), reflected (Kingery)
5. half-confined (half space), not reflected (Kingery)
6. Blast parameters will be directly specified next

**CONF 6** indicates that the blast parameters  $p_{\max}$ ,  $t_d$  and  $b$  appearing in the so-called modified Friedlander equation (see below) will be directly specified next and should not be calculated automatically by the code. In this case, no other parameters (except **CONF** of course) are accepted, only the positive pressure (overpressure) is considered and the pressure-time function is identical in each element. The **modified Friedlander equation** reads:

$$p(t) = p_0 + p_{\max} \left(1 - \frac{t}{t_d}\right)^{-\frac{bt}{t_d}}$$

and expresses the pressure  $p$  as a function of time  $t$ , with  $p_0$  the initial (normally the atmospheric) pressure,  $p_{\max}$  the maximum overpressure (peak overpressure),  $t_d$  the duration

of the positive pressure phase and  $b$  the decay parameter, which defines how rapidly the pressure decays.

d

Choice between different available decay coefficient equation models. Each equation is defined according to the explosion model chosen before (incident, reflected - spherical, hemispherical). The equations based on the Kingery-Bulmash data have been calculated by iteratively solving the Friedlander equation with the set of positive blast parameters proposed by Kingery-Bulmash. There are different equations for reflected or not reflected (incident) cases of unconfined (spherical) and half-confined (hemi-spherical) blast waves. An additional equation for the blast coefficient is available which is based on the Kinney and Baker data. The default blast decay equation is based on the Kingery-Bulmash data. The explosion model, that has already been defined by the parameter c, shows which of the blast wave decay equations (incident, reflected, spherical or hemispherical) will be used.

1. Blast wave decay equation based on Kinney data
2. Blast wave decay equation based on Kingery-Bulmash data (default)

pmax

Maximum overpressure  $p_{\max}$  appearing in the modified Friedlander equation. This should only be given when CONF 6 has been specified.

td

Duration of the positive pressure phase  $t_d$  appearing in the modified Friedlander equation. This should only be given when CONF 6 has been specified.

b

Decay parameter  $b$  appearing in the modified Friedlander equation. This should only be given when CONF 6 has been specified.

/LECT/

Elements concerned. These must be of type CLxx.

## Comments :

This model requires that the user adopts the standard Unit system, i.e. metres, Kilograms, seconds.

Care must be taken in the orientation of the CLxx elements, in such a way that the pressure load resulting from the AIRB model acts in the right sense.

The force generated by a positive AIRB overpressure pushes the CLxx element in the sense of the normal to the element itself. This normal is determined by the numbering of the element in EUROPLEXUS.

Therefore, typically the CLxx elements must be oriented in such a way that their normal points **away from** the AIRB charge, or away from the direction from which the AIRB overpressure is expected to arrive.

This convention has been assumed because of the possibility of using AIRB model to load not directly a structure, but a fluid boundary. In this case, the CLxx elements must be attached to the fluid boundary (i.e. to continuum-like fluid elements) and the code always orients them

(irrespective of the orientation chosen by the user) in such a way that the normal to the CLxx elements points **inside** the fluid.

A final *caveat*: since the orientation of the AIRB load is related to the orientation of the CLxx element, a problem may arise in case of extremely large rotations of the structure (and thus of the attached CLxx element), such as for example in the case of a rupturing structure which breaks up into large flying debris. In fact, if the element rotates by more than, say, 90 degrees, then the AIRB load may appear to act in the wrong direction. This is an inherent limitation of the model which may not be avoided, and the user should be aware of it.

The equations of Kingery are only usable up to a scaled distance of  $Z=40$ . Above this distance, diagrams of Baker are used (linearised in the double logarithmic scale).

## Outputs:

The different components of the ECR table are as follows :

ECR(1) : pressure

## References:

For more information on the physical models, consult the following references:

- Kingery, Charles N., Bulmash, Gerald: *Airblast Parameters from TNT Spherical Air Burst and Hemispherical Surface Burst*, Defense Technical Information Center, Ballistic Research Laboratory, Aberdeen Proving Ground, Maryland, 1984.
- Baker, Wilfrid E.: *Explosions in the Air*. University of Texas Pr., Austin, 1973.
- Kinney, G.F., Graham, K.J.: *Explosive Shocks in Air*. Springer, Berlin, 1985.

### 7.9.31 NTNU's SIMPLIFIED FSI MODEL

#### Object :

This directive allows to load a structure by a blast-like pressure without discretizing also the fluid. It represents a very simplified approach to Fluid-Structure Interaction (FSI) modelling.

The model takes in input the time history of the (absolute) pressure (incident plus reflected wave)  $p_{\text{rigid}}(t)$  that would be measured at the impacted wall if this wall would be rigid and blocked, plus some constants of the gas (typically the atmospheric air) which characterize the undisturbed state before arrival of the blast wave. Then, the model estimates the pressure  $p_{\text{FSI}}$  acting on the impacted wall as this wall moves and deforms with a certain velocity  $v_{\text{wall}}$ . Let:

- $p_{\text{rigid}}(t)$  be the time function of the (absolute) incident plus reflected blast pressure for a rigid wall. This may come from an experiment, from an analytical solution or from theoretical considerations.
- $\gamma = C_p/C_v$  be the ratio of heat capacities of the gas, assumed as constant.
- $p_{\text{atm}}$  be the atmospheric pressure, i.e. the pressure of the gas before the arrival of the blast wave (a constant).
- $\rho_{\text{atm}}$  be the atmospheric density, i.e the density of the gas before the arrival of the blast wave (a constant).

We assume that the wall is initially at rest and has not yet been reached by the blast wave. Therefore the code sets  $v_{\text{wall}}(0) = 0$ ,  $p_{\text{rigid}}(0) = p_{\text{atm}}$  and  $\rho_{\text{rigid}}(0) = \rho_{\text{atm}}$ . At the generic step  $t^{n+1}$  the code computes:

- The normal velocity of the wall (approximated by using the wall's half-step speed  $v^{n+1/2}$ ):

$$v_{\text{wall}}^{n+1} = \underline{v}^{n+1/2} \cdot \underline{\hat{n}}^{n+1}$$

where  $\underline{\hat{n}}^{n+1}$  is the unit vector normal to the wall. This vector is assumed positive when entering the wall material.

- The pressure of the gas at the wall if it would be rigid, extracted from the given time function:

$$p_{\text{rigid}}^{n+1} = p_{\text{rigid}}(t^{n+1})$$

- The density of the gas at the wall if it would be rigid, by assuming that the gas is subjected to an isentropic compression from the initial undisturbed state:

$$\rho_{\text{rigid}}^{n+1} = \rho_{\text{atm}} \left( \frac{p_{\text{rigid}}^{n+1}}{p_{\text{atm}}} \right)^{1/\gamma}$$

- The sound speed of the gas at the wall if it would be rigid, from the EOS of an ideal gas:

$$c_{\text{rigid}}^{n+1} = \sqrt{\gamma \frac{p_{\text{rigid}}^{n+1}}{\rho_{\text{rigid}}^{n+1}}}$$

- Finally, the wall pressure accounting for FSI effects is obtained from:

$$p_{\text{FSI}}^{n+1} = p_{\text{rigid}}^{n+1} \left( 1 + \frac{\gamma - 1}{2} \cdot \frac{-v_{\text{wall}}^{n+1}}{c_{\text{rigid}}^{n+1}} \right)^{\frac{2\gamma}{\gamma-1}}$$

- Additionally, the density of the gas at the moving wall is also computed, only for output and comparison purposes:

$$\rho_{\text{FSI}}^{n+1} = \rho_{\text{atm}} \left( \frac{p_{\text{FSI}}^{n+1}}{p_{\text{atm}}} \right)^{1/\gamma}$$

It should be noted that with the sign conventions assumed above, i.e.  $v_{\text{wall}}$  positive when directed away from the incoming blast, a positive wall velocity generates a reduced FSI pressure compared with a rigid wall. Conversely, if the wall moves towards the incoming blast (negative wall velocity), then the FSI pressure is larger than for a rigid wall.

The value obtained of  $p_{\text{FSI}}^{n+1}$  is used as the pressure of the CLxx element, i.e. the pressure (already accounting for FSI effects) acting on the moving wall. This pressure (diminished by the reference pressure) is used by the code to update the wall deformation and the wall velocity, for the next time step.

### Syntax:

```
"IMPE"  "SFSI" "GAMM" gamm "PATM" patm "RATM" ratm
        < "PREF" pref > < "PFAC" pfac > "PFUN" pfun
        /LECTURE/
```

#### GAMM

Ratio of specific heats  $\gamma = C_p/C_v$  of the gas (atmosphere), assumed as constant.

#### PATM

Atmospheric pressure (absolute), expressed in Pa.

#### RATM

Atmospheric density, expressed in kg/m<sup>3</sup>.

#### PREF

Reference pressure expressed in Pa. Note that if a zero reference pressure is desired, it is mandatory to specify **PREF 0** in the input file. This is because, if no value for **PREF** is specified, the code assumes  $P_{\text{ref}} = P(t = 0)$  so the initial imposed pressure has no effect, since  $p = P - P_{\text{ref}} = 0$ !

#### PFAC

Multiplicative factor  $\phi$  of the pressure given in the following time function. By default  $\phi = 1.0$ . It may be useful to uniformly scale the time function.

#### PFUN

Index of the input time function representing the (absolute) incident plus reflected pressure (in Pa) vs. time (in s). See part E of the manual on how to define a function.

/LECT/

Elements concerned. These must be of type CLxx.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : Pressure (absolute) acting on the moving wall, taking into account the FSI effects ( $p_{\text{FSI}}^{n+1}$ ).

ECR(2) : Density of the gas at the moving wall, taking into account the FSI effects ( $\rho_{\text{FSI}}^{n+1}$ ).

ECR(3) : Pressure (absolute) incident plus reflected acting on the wall assumed as rigid ( $p_{\text{rigid}}^{n+1}$ ), as extracted from the given time function. This can be useful to be compared with the pressure taking into account FSI effects ECR(1).

ECR(4) : Density at the moving wall, assumed as rigid ( $\rho_{\text{rigid}}^{n+1}$ ). This can be useful to be compared with the density taking into account FSI effects ECR(2).

**References:**

For more information on the physical models, consult the following references:

- Aune, V., Casadei, F., Valsamos, G.: *A simplified computational approach to account for fluid-structure interaction effects on blast loaded plates*, Technical Report, in preparation. 2020.



### 7.9.32 FRAGILE PLATE

**Object:**

This instruction enables the modelling of a fragile structure (e.g. a plate) embedded in a fluid.

The pressure drop across the plate is computed according to the expression:

$$\text{Deltap} = p_1 - p_2$$

Here  $p_1$  and  $p_2$  are the pressures on the two sides of the plate.

As long as the plate element holds, it prevents any fluid from passing through it. As soon as the structural element fails, the structure is replaced by a cloud of debris particles and the fluid may start to flow through.

**Syntax :**

```
"IMPE"  "FPLT"  
        /LECTURE/
```

```
/LECT/
```

Concerned elements.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : current pressure drop across the plate

ECR(2) : unused

ECR(3) : unused

ECR(4) : structural node 1

ECR(5) : structural node 2

ECR(6) : structural node 3

ECR(7) : structural node 4

ECR(8) : unused

ECR(9) : unused

Note that the positions 4 to 7 contain up to 4 indexes of the structural nodes corresponding to the CLxx element's (fluid) nodes. These quantities are determined only once at the beginning of the calculation and never change.

**Comments:**

Normally, the determination of the above mentioned node correspondence is performed automatically. However, in case of problems the user may either change the tolerance for node matching (see `OPTI TOLC` on page H.40) or force the node correspondence by using the directive `COMP CNOD`, see page C.92.

### 7.9.33 DATA RECORDING FOR VISUALIZATION

**Object:**

This directive enables the recording of data for subsequent visualization in the form of iso-value maps (on a whole region of the mesh) or of time curves (one element at a time).

For example, one may want to record the fluid (over-)pressure acting on a structure embedded (immersed) in a fluid and coupled with the fluid by means of a FSI algorithm such as FLSW or FLSR. In this case, the CLxx element to which the material **IMPE VISU** is affected should be attached to the structure. By visualizing the CLxx elements and their “recorded” data in the form of iso-maps one obtains an exact view of the structural walls, with the data (e.g. FSI overpressure) acting on such walls.

Another example occurs in fluid-only simulations where the user may want to visualize the fluid (over-)pressure only along certain parts of the fluid domain surface (those where a structure, not represented in the model, is attached). In this case the CLxx element is attached directly to the fluid elements.

**Syntax :**

```
"IMPE"  "VISU" <$ "COUP" ; "DECO" $>
        /LECTURE/
```

**VISU**

Introduces the parameters of the current **IMPE VISU** material.

**COUP**

The FSI coupling is realized by means of a “strong” (coupled) approach (e.g. FLSR algorithm) so that the FSI forces are contained in the FLIA table.

**DECO**

The FSI coupling is realized by means of a “weak” (uncoupled) approach (e.g. FLSW algorithm) so that the FSI forces are contained in the FDEC table.

**/LECT/**

Concerned CLxx elements.

**Outputs:**

The different components of the ECR table are as follows:

- ECR(1) : current FSI overpressure
- ECR(2) : maximum FSI overpressure over time
- ECR(3) : minimum FSI overpressure over time
- ECR(4) : current area of the CLxx element (embedded case)
- ECR(5) : x-component of the current unit normal to the CLxx element (embedded case)
- ECR(6) : y-component of the current unit normal to the CLxx element (embedded case)
- ECR(7) : z-component of the current unit normal to the CLxx element (embedded case)
- ECR(8:10) : unused

Note, however, that the first three components are directly defined only if the CLxx element with IMPE VISU material is directly attached to a fluid element. If the CLxx element is attached to a structural element (embedded FSI) then these components are obtained by averaging of the nodal quantity PFSI over each element and this process is likely to introduce some smootghing of the results (especially if the “element” quantity ECR is then drawn in the form of iso-maps of an element field, whereby the data are usually projected back on the nodes).

The last four components are only defined if the CLxx element with IMPE VISU material is attached to a structural element (and the COUP or DECO keyword is specified).

In the case of a CLxx element with IMPE VISU material attached to a structural element the current, maximum and minimum FSI overpressure **directly evaluated at nodes** can be visualized by using the PFSI, PFMI and PFMA keywords, see Page ED.70 (time curves of nodal variables) and Page O.80 (iso-value maps of a nodal field).

### Comments:

The COUP or DECO optional keywords are only used (and **must** be specified) if the CLxx element to which the IMPE VISU material is affected is attached to a structural element. If the CLxx element is directly attached to a fluid element, then these keywords **must not** be specified.

### 7.9.34 CLVF ABSORBING MATERIAL (SUPERSONIC OUTLET)

#### Object :

This option enables to specify absorbing or partially absorbing boundary conditions for 1-D, 2-D or 3-D Cell-Centred Finite Volumes (VFCC). From a mathematical point of view, this boundary condition well represents a supersonic outlet<sup>1</sup>: the state inside the fluid domain is copied outside (into the so called "ghost" cell) to evaluate the numerical flux. From a practical point of view this boundary condition can be applied also to other cases (even for inlets), but only in the case of a supersonic outlet it is guaranteed that all waves are absorbed.

#### Syntax:

```
"CLVF" "ABSO" "R0" rho /LECTURE/
```

rho

Density (**never used but here for historical reasons**).

LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

#### Outputs:

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

---

<sup>1</sup>For a theoretical overview of the CLVF boundary conditions implemented in Europlexus, one can read [808] from page 187. For this particular case, see Section "Frontière absorbante" at page 190.

### 7.9.35 CLVF CONDITIONS AT INFINITY (INFINITELY HUGE RESERVOIR)

#### Object :

This option works for perfect gases only (GAZP material). It enables to specify conditions at infinity for a fluid modelled by Cell-Centred Finite Volumes (VFCC)<sup>2</sup>. According to the values of the speed normal to the interface  $u_{n,int}$  and of the Mach number  $M_{int}$  (evaluated using the normal speed) inside the fluid domain, we distinguish between different cases.

1.  $M_{int} \leq 1$ . We have a subsonic inlet ( $u_{n,int} \leq 0$ ) or a subsonic outlet ( $u_{n,int} \geq 0$ ). We enforce that in the "ghost" cell we have pressure, density and velocity at infinity (the last one is zero).
2.  $M_{int} \geq 1$  and ( $u_{n,int} > 0$ ). We have a supersonic outlet. Then, as in the case ABSO, the state inside the fluid domain is copied outside into the "ghost" cell.
3.  $M_{int} > 1$  and ( $u_{n,int} < 0$ ). We have a supersonic inlet. Then we stop the computation because we are not able to deal with this case.

#### Syntax:

```
"CLVF"  "INFI" "R0" rho "PRES" pres <"PREF" pref> "GAMA" gama
/LECTURE/
```

rho

Density at infinity.

pres

Pressure at infinity.

pref

Reference pressure at infinity.

gama

Gamma (ratio of specific heats) at infinity.

LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

#### Outputs:

<sup>2</sup>For a theoretical overview of the CLVF boundary conditions implemented in Europlexus, one can read [808] from page 187. For this particular case, see Section "Impédance infinie" at page 191.

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

### 7.9.36 CLVF IMPOSED PRESSURE (SUBSONIC OUTLET)

#### Object :

This option works for perfect gases only (GAZP material). It enables to specify an imposed external pressure  $P_{\text{imp}}$  on a fluid modelled by Cell-Centred Finite Volumes (VFCC)<sup>3</sup>. From a mathematical point of view, this boundary condition well represents a subsonic outlet. Indeed the "ghost" state takes into account one external information (the pressure) and the rest of information is taken inside the domain. According to the value of the parameter "IMPO" of the syntax, we distinguish between 6 different cases.

1. At the "ghost" cell we impose that we have the same entropy and the same momentum as in the internal domain. Namely  $P = P_{\text{imp}}$  and

$$\rho = \rho_{\text{int}} \left( \frac{P}{P_{\text{int}}} \right)^{\frac{1}{\gamma}}$$

$$\vec{u} = \frac{\rho_{\text{int}} \vec{u}_{\text{int}}}{\rho}$$

2. At the "ghost" cell we impose that we have the same entropy and the same velocity as in the internal domain. Namely  $P = P_{\text{imp}}$  and

$$\rho = \rho_{\text{int}} \left( \frac{P}{P_{\text{int}}} \right)^{\frac{1}{\gamma}}$$

$$\vec{u} = \vec{u}_{\text{int}}$$

3. We compute the density in the "ghost" cell by imposing that the "ghost" state and the internal state are on the same Hugoniot curve (while in cases 1 and 2 we impose that the "ghost" state and the internal state are on the same isoentropic curve). Moreover, we impose that the "ghost" state presents the same velocity as the internal state.
4. We compute the density in the "ghost" state by imposing that the "ghost" state and the internal state are on the same Hugoniot curve. Moreover, we impose that the "ghost" state presents the same momentum as the internal state.
5. We enforce that the "ghost" state has the same entropy and the same (outlet) Riemann invariant as the internal state.
6. We enforce that the "ghost" state has the same density and velocity as the internal state.

#### Syntax:

```
"CLVF"  "PIMP" "R0" rho "PRES" pres <"PREF" pref> "GAMA" gama
        <"IMPO" impo>
        /LECTURE/
```

<sup>3</sup>For a theoretical overview of the CLVF boundary conditions implemented in Europlexus, one can read [808] from page 187. For this particular case, see Section "Pression imposée" at page 192.



rho

Density (**never used but here for historical reasons**).

pres

Pressure.

pref

Reference pressure.

gama

Gamma (ratio of specific heats).

impo

By default this is 2.

LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

### Outputs:

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

### 7.9.37 CLVF IMPOSED MASS FLOW RATE (SUPERSONIC INLET)

#### Object :

This option works for perfect gases only (GAZP material). It enables to specify an imposed mass flow rate on the boundary of a fluid modelled by Cell-Centred Finite Volumes (VFCC)<sup>4</sup>. From a mathematical point of view, this boundary condition represents a supersonic inlet. We impose in the "ghost" state the specified boundary conditions (density, pressure and momentum).

#### Syntax:

```
"CLVF"  "DEBI" "R0" rho "PRES" pres "PREF" pref "GAMA" gama
        "DEBX" debx "DEBY" deby <"DEBZ" debz>
        /LECTURE/
```

rho

Density.

pres

Pressure.

pref

Reference pressure.

gama

Gamma (ratio of specific heats).

debx

X-component of the mass flow rate.

deby

Y-component of the mass flow rate.

debz

Z-component of the mass flow rate (3D only).

#### LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

---

<sup>4</sup>For a theoretical overview of the CLVF boundary conditions implemented in EUROPLEXUS, one can read [808] from page 187.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

### 7.9.38 CLVF SUBSONIC INLET

#### Object :

This option works for perfect gases only (GAZP material). It enables to specify a subsonic inlet on the boundary of a fluid modelled by Cell-Centred Finite Volumes (VFCC)<sup>5</sup>. Indeed, we have one internal information taken into account and the rest of information is taken outside the domain. In the "ghost" state we take into account of the internal pressure (information taken inside) and we impose the values of the entropy, total enthalpy and flow direction of the state at infinity.

#### Syntax:

```
"CLVF"  "ESUB" "R0" rho "PRES" pres "PREF" pref "GAMA" gama
        "DEBX" debx "DEBY" deby <"DEBZ" debz>
        /LECTURE/
```

rho

Density at the infinity.

pres

Pressure at the infinity.

pref

Reference pressure.

gama

Gamma (ratio of specific heats).

debx

X-component of the mass flow rate at the infinity.

deby

Y-component of the mass flow rate at the infinity.

debz

Z-component of the mass flow rate at the infinity (3D only).

#### LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

---

<sup>5</sup>For a theoretical overview of the CLVF boundary conditions implemented in Europlexus, one can read [808] from page 187. This specific boundary condition has been applied to study the flow in a nozzle in [839], page 22.

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

**7.9.39 CLVF LODI QUASI 1-D CONDITION****Object :**

This option works for perfect gases only (GAZP material). It enables to specify a Lodi quasi 1-D condition on a fluid modelled by Cell-Centred Finite Volumes (VFCC). It is still experimental and under development. See references [937] and [938] for details.

**Syntax:**

```

"CLVF"  "LOD1" "R0"  rho    "PRES" pres  "GAMA" gama
        "VNOR" vnor  <"VTG1" vtg1> <"VTG2" vtg2>
        <"COXA" coxa> <"COYA" coya> <"COZA" coza>
        <"COXB" coxb> <"COYB" coyb> <"COZB" cozb>
        "TYPE" type  "CELP" celp  "VITP" vitp
        "LONP" lonp  <"ROP"  rop>  <"TABT" tabt>
        <"TABO" tabo>
/LECTURE/

```

**rho**

Density.

**pres**

Pressure.

**gama**

Gamma (ratio of specific heats).

**vnor**

Normal velocity (give a negative value for an undefined velocity at an exit).

**vtg1**

Tangential velocity (first component).

**vtg2**

Tangential velocity (second component).

**coxa, coya, coza**

XA, YA, ZA coefficients.

**coxb, coyb, cozb**

XB, YB, ZB coefficients.

**type**

Type of boundary condition.

**celp**

Reference sound speed.

**vitp**

Reference normal speed.

**lonp**

Reference normal length.

**rop**

Reference density.

**tabt**

Unknown.

**tabo**

Unknown.

#### LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

#### Outputs:

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

### 7.9.40 CLVF FOURIER MODES IN 2D

#### Object :

This option works for perfect gases only (GAZP material). It enables to specify a Fourier modes in 2D condition on a fluid modelled by Cell-Centred Finite Volumes (VFCC). It is still experimental and under development. See references [937] and [938] for details.

#### Syntax:

```
"CLVF"  "FOUR" "RO"  rho  "PRES" pres  "GAMA" gama
          "VNOR" vnor  <"VTG1" vtg1> <"VTG2" vtg2>
          <"COXA" coxa> <"COYA" coya> <"COZA" coza>
          <"COXB" coxb> <"COYB" coyb> <"COZB" cozb>
          "TYPE" type  "CELP" celp  "VITP" vitp
          "LONP" lonp  <"ROP"  rop>  <"TABT" tabt>
          <"TABO" tabo>
/LECTURE/
```

rho

Density.

pres

Pressure.

gama

Gamma (ratio of specific heats).

vnor

Normal velocity (give a negative value for an undefined velocity at an exit).

vtg1

Tangential velocity (first component).

vtg2

Tangential velocity (second component).

coxa, coya, coza

XA, YA, ZA coefficients.

coxb, coyb, cozb

XB, YB, ZB coefficients.

type



Type of boundary condition.

**celp**

Reference sound speed.

**vitp**

Reference normal speed.

**lonp**

Reference normal length.

**rop**

Reference density.

**tabt**

Unknown.

**tabo**

Unknown.

#### LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

#### Outputs:

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

**7.9.41 CLVF RIEMANN 3-D CONDITION****Object :**

This option works for perfect gases only (GAZP material). It enables to specify a Riemann 3-D condition on a fluid modelled by Cell-Centred Finite Volumes (VFCC). It is still experimental and under development. See references [937] and [938] for details.

**Syntax:**

```
"CLVF"  "RIEM" "R0"  rho    "PRES" pres  "GAMA" gama
          "VNOR" vnor  <"VTG1" vtg1> <"VTG2" vtg2>
          <"COXA" coxa> <"COYA" coya> <"COZA" coza>
          <"COXB" coxb> <"COYB" coyb> <"COZB" cozb>
          "TYPE" type  <"TABT" tabt>
/LECTURE/
```

rho

Density.

pres

Pressure.

gama

Gamma (ratio of specific heats).

vnor

Normal velocity (give a negative value for an undefined velocity at an exit).

vtg1

Tangential velocity (first component).

vtg2

Tangential velocity (second component).

coxa, coya, coza

XA, YA, ZA coefficients.

coxb, coyb, cozb

XB, YB, ZB coefficients.

type

Type of boundary condition.

tabt

Unknown.

#### LECTURE

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

#### Outputs:

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

### 7.9.42 CLVF TIME-DEPENDENT PRESSURE

**Object :**

This option works for perfect gases only (GAZP material). It enables to specify an imposed time-dependent outside pressure on a fluid modelled by Cell-Centred Finite Volumes (VFCC). It is still experimental and under development. See references [937] and [938] for details.

**Syntax:**

```
"CLVF"  "PFCT" "COEF" coef "FONC" fonc <"PREF" pref>
      /LECTURE/
```

**coef**

Scaling coefficient.

**fonc**

Index of the time function.

**pref**

Reference pressure.

**LECTURE**

Reading procedure of the numbers of the elements composing the boundary (CL1D, CL2D, CL3D or CL3T).

**Outputs:**

The different components of the ECR table are as follows :

ECR(1) : pressure

ECR(2) : density

ECR(3) : sound speed

ECR(4) : normal velocity

ECR(5) : tangential velocity

ECR(6) : tangential velocity (2nd component in 3d)

### 7.9.43 SAFETY VALVE FOR VFCC (JRC)

#### Object

This instruction enables the modelling of fluid discharge from a pressurized vessel through a generic safety valve or (by a suitable choice of the parameters) through an orifice. The fluid passing through the valve or orifice can be either incompressible (typically a liquid) or compressible (a gas).

Alternatively, the model can also be used in the opposite flow direction in order to fill up (pressurize) a vessel from an external source. Therefore, we distinguish between two modes: **discharge** mode or **loading** mode. The two modes cannot be combined in the same device, since no reverse flow is allowed. However, one could of course attach two **SV** devices, one in discharge mode and the other in loading mode, to the same vessel (at different locations along the wall) if needed.

The input syntax includes also a set of (optional) parameters that allow a sort of “pressure regulation” functionality whereby the opening of the valve is subjected to a series of constraints which may also depend upon the (independent) pressure in a “master” fluid element not belonging to the zone of the fluid domain to which the valve is attached.

#### A - Discharge mode

The valve is characterized by three main parameters:

- $A^{\text{tube}}$ , the area of the tube in which the valve is mounted;
- $\Delta p^{\text{min}}$ , the pressure difference (between internal and external pressures) at which the valve starts to open;
- **mode** the functioning mode of the device: 0 means discharge mode, 1 means loading mode. The default value is 0, so this parameter may be omitted if discharge mode is required.

Other **optional** parameters are:

- $\Delta p^{\text{max}}$ , the pressure difference at which the valve is fully open, by default  $\Delta p^{\text{max}} = \Delta p^{\text{min}}$ ;
- $\Delta t^{\text{ope}}$ , the time interval needed to completely open the valve under a positive step increment of the pressure difference  $\Delta p > \Delta p^{\text{max}}$ , by default  $\Delta t^{\text{ope}} = 0$ ;
- $\Delta t^{\text{clo}}$ , the time interval needed to completely close the valve under a negative step increment of the pressure difference  $\Delta p < -\Delta p^{\text{max}}$ , by default  $\Delta t^{\text{clo}} = 0$ ;
- $q^{\text{max}}$ , the maximum mass flow rate through the valve, by default  $q^{\text{max}} = \infty$ ;
- $p^{\text{ext}}$ , the external pressure (assumed constant), by default  $p^{\text{ext}} = 0$ ;
- $\rho^{\text{ext}}$ , the external density (assumed constant); this quantity must be specified in loading mode, but it must not be specified in discharge mode.
- $i^{\text{ext}}$ , the external specific internal energy (assumed constant); this quantity must be specified in loading mode, but it must not be specified in discharge mode.
- $A^{\text{max}}$ , the maximum opening area of the valve which must be  $A^{\text{max}} \leq A^{\text{tube}}$ , by default  $A^{\text{max}} = A^{\text{tube}}$ ;

- $C_c$ , the contraction coefficient of the fluid flow through the valve opening, i.e. the ratio between the area of the flow in the *vena contracta* and the opening area of the valve (assumed constant, independent of the current opening area), by default  $C_c = 1$ ;
- $\gamma$ , the ratio of specific heats  $\gamma = C_p/C_v$  in case of compressible fluid flow. If omitted, the fluid being discharged is considered as incompressible;
- $t^{\text{act}}$ , the activation time of the valve. The valve is guaranteed to be inactive (closed) for  $t < t^{\text{act}}$ , whatever be the conditions of the fluid. If not specified, the code assumes  $t^{\text{act}} = -\infty$ .
- $t^{\text{dea}}$ , the de-activation time of the valve. The valve is guaranteed to be inactive (closed) for  $t > t^{\text{dea}}$ , whatever be the conditions of the fluid. If not specified, the code assumes  $t^{\text{dea}} = \infty$ .
- **qmom**, a flag that controls the flux of momentum: 0 (the default) means that no flux of momentum is computed, 1 means that the flux of momentum is computed. This parameter is used only when the valve is attached to a VFCC fluid volume.
- **emas**, the *master* fluid element whose pressure  $p^{\text{mas}}$  has to be monitored for pressure regulation purposes. Note that the pressure  $p^{\text{mas}}$  is independent from the opening or not of the current valve, i.e. it is not influenced (at least directly) from the opening or not of the valve. If specified, the master element should therefore belong to a zone of the fluid domain not directly connected with the fluid part to which the current valve is attached (but the code does not check this requirement).
- **esla**, the *slave* fluid element whose pressure  $p^{\text{sla}}$  has to be monitored for pressure regulation purposes. Note that the pressure  $p^{\text{sla}}$  directly depends upon the opening or not of the current valve. If specified, the slave element should therefore belong to the zone of the fluid domain directly connected with the fluid part to which the current valve is attached (but the code does not check this requirement).
- $p^{\text{dis}}$ , the value of pressure in the slave element that, when reached, triggers the discharge through the current valve, i.e. starts the opening of the valve. This parameter must not be specified in loading mode. By default  $p^{\text{dis}} = \infty$ .
- $p^{\text{max}}$ , the value of pressure in the slave element that, when reached, triggers the closing of the current valve. This parameter must not be specified in discharge mode. By default  $p^{\text{max}} = \infty$ .
- $p^{\text{ms1}}$ , the value of pressure difference between the master and the slave elements at which the valve starts to open (in an attempt to keep the pressure difference below the chosen value:  $\Delta p = p^{\text{mas}} - p^{\text{sla}} \leq p^{\text{ms1}}$ .) By default  $p^{\text{ms1}} = \infty$ .
- $p^{\text{ms2}}$ , the value of pressure difference between the master and the slave elements at which the valve becomes fully open. If  $\Delta p = p^{\text{mas}} - p^{\text{sla}} \geq p^{\text{ms2}}$  then the valve is fully open, in an attempt to keep the pressure difference below the chosen value. By default  $p^{\text{ms2}} = p^{\text{ms1}}$ .
- **rval**, a *reference* valve (**SVAL**) element. The reference valve must be in discharge mode and the current valve must be in loading mode. If the opening area of the reference valve is  $a^{\text{ref}} > 0$ , then the (current) valve closes immediately ( $a = 0$ ).

This model is similar to the material **IMPE SVAL** described on page C.860 but can be used with VFCCs instead of Finite Elements in the fluid domain.

## References

More information on the formulation of this model may be found in reference [94].

## Syntax

```
"CLVF"  "SVAL"  <"MODE" mode> "ATUB" atub "DPMI" dpmi
      < "DPMA" dpma "TOPE" tope "TCLO" tclo "QMAX" qmax
      "PEXT" pext "ROEX" roex "IEXT" iext
      "AMAX" amax "CC"  cc  "GAMM" gamm "TACT" tact
      "TDEA" tdea "QMOM" qmom
      "EMAS" /LECT_emas/ "ESLA" /LECT_esla/
      "PDIS" pdis "PMAX" pmax "PMS1" pms1 "PMS2" pms2
      "RVAL" /LECT_rval/
      /LECTURE/
```

### mode

Functioning mode of the device: 0 means discharge mode, 1 means loading mode. If omitted, the code assumes value 0 (discharge mode).

### atub

Cross-section of the safety valve flow tube.

### dpmi

Pressure difference at which the valve starts to open.

### dpma

Pressure difference at which the valve is fully open; by default, it is equal to  $\Delta p^{\min}$ .

### tope

Opening time interval; by default, it is 0.

### tclo

Closure time interval; by default, it is 0.

### qmax

Maximum mass flow rate; if omitted, by default it is infinite (no limit).

### pext

External pressure (assumed constant); by default it is 0.

### roex

External density (assumed constant); this quantity must be specified in loading mode, but it must not be specified in discharge mode.

### iext

External specific internal energy (assumed constant); this quantity must be specified in loading mode, but it must not be specified in discharge mode.

**amax**

Maximum opening area of the valve. It must be  $A^{\max} \leq A^{\text{tube}}$ . By default, it is  $A^{\max} = A^{\text{tube}}$ .

**cc**

Flow contraction coefficient ( $0 < C_c \leq 1$ ). By default it is  $C_c = 1$ .

**gamm**

Ratio of specific heats for the fluid being discharged ( $\gamma = C_p/C_v$ ). If specified, the fluid is considered as compressible. If omitted, the fluid is considered as incompressible ( $\gamma$  is unused in that case).

**tact**

Activation time of the valve. The valve is guaranteed to be inactive (closed) for  $t < t^{\text{act}}$ , whatever be the conditions of the fluid. If not specified, the code assumes  $t^{\text{act}} = -\infty$ .

**tdea**

De-activation time of the valve. The valve is guaranteed to be inactive (closed) for  $t > t^{\text{dea}}$ , whatever be the conditions of the fluid. If not specified, the code assumes  $t^{\text{dea}} = \infty$ .

**qmom**

Flag that controls the flux of momentum: 0 (the default) means that no flux of momentum is computed, 1 means that the flux of momentum is computed. This parameter is used only when the valve is attached to a VFCC fluid volume.

**EMAS**

Introduces the definition (/LECT\_emas/) of the *master* fluid element whose pressure  $p^{\text{mas}}$  has to be monitored for pressure regulation purposes. Note that the pressure  $p^{\text{mas}}$  is independent from the opening or not of the current valve, i.e. it is not influenced (at least directly) from the opening or not of the valve. If specified, the master element should therefore belong to a zone of the fluid domain not directly connected with the fluid part to which the current valve is attached (but the code does not check this requirement).

**ESLA**

Introduces the definition (/LECT\_esla/) of the *slave* fluid element whose pressure  $p^{\text{sla}}$  has to be monitored for pressure regulation purposes. Note that the pressure  $p^{\text{sla}}$  directly depends upon the opening or not of the current valve. If specified, the slave element should therefore belong to the zone of the fluid domain directly connected with the fluid part to which the current valve is attached (but the code does not check this requirement).

**pdis**

The value of pressure  $p^{\text{dis}}$  in the slave element that, when reached, triggers the discharge through the current valve, i.e. starts the opening of the valve. This parameter must not be specified in loading mode. By default,  $p^{\text{dis}} = \infty$ .



**pmax**

The value of pressure  $p^{\max}$  in the slave element that, when reached, triggers the closing of the current valve. This parameter must not be specified in discharge mode. By default,  $p^{\max} = \infty$ .

**pms1**

The value of pressure difference  $p^{\text{ms1}}$  between the master and the slave elements at which the valve starts to open (in an attempt to keep the pressure difference below the chosen value:  $\Delta p = p^{\text{mas}} - p^{\text{sla}} \leq p^{\text{ms1}}$ .) By default,  $p^{\text{ms1}} = \infty$ .

**pms2**

The value of pressure difference  $p^{\text{ms2}}$  between the master and the slave elements at which the valve becomes fully open. If  $\Delta p = p^{\text{mas}} - p^{\text{sla}} \geq p^{\text{ms2}}$  then the valve is fully open, in an attempt to keep the pressure difference below the chosen value. By default,  $p^{\text{ms2}} = p^{\text{ms1}}$ .

**RVAL**

Introduces the definition (/LECT\_rval/) of a *reference* valve (SVAL) element. The reference valve must be in discharge mode and the current valve must be in loading mode. If the opening area of the reference valve is  $a^{\text{ref}} > 0$ , then the (current) valve closes immediately ( $a = 0$ ).

**LECTURE**

Reading procedure of the number of the "CLxx" element defining the boundary.

**Determination of the current opening area of the valve**

The calculation of the current opening area of the valve consists of three phases. In the *first phase*, we check whether at the current time  $t$  the valve is active or not: if  $t < t^{\text{act}}$  or  $t > t^{\text{dea}}$  then the valve is closed irrespective of fluid conditions ( $a = 0$ ) and we skip the following two phases. Then, if **rval** has been specified, we also check the opening area of the reference valve: if  $a^{\text{ref}} > 0$  then the current valve is immediately closed ( $a = 0$ ) and we skip the following two phases.

Otherwise ( $t^{\text{act}} \leq t \leq t^{\text{dea}}$ ), the valve is active. The *second phase* consists of checking whether there are any pressure regulation constraints specified for the current valve, since such constraints override the normal opening calculations to be performed in the third phase. If no slave element **esla** was specified, there are no regulation constraints and so we go directly to the third phase. Otherwise there are some constraints, which depend upon the chosen functioning mode of the device.

- In *discharge mode*, if  $p^{\text{dis}}$  was given and  $p^{\text{sla}} \geq p^{\text{dis}}$ , then we set  $a = A$  (valve fully open) and skip the third phase.
- In *loading mode*, if  $p^{\max}$  was given and  $p^{\text{sla}} \geq p^{\max}$ , then we set  $a = 0$  (valve fully closed) and skip the third phase. Otherwise, we check any constraints on the master-slave pressure difference. If  $p^{\text{ms1}}$  was given, we compute the pressure difference  $\Delta p = p^{\text{mas}} - p^{\text{sla}}$ : if  $\Delta p < p^{\text{ms1}}$  the valve is completely closed ( $a = 0$ ); else if  $\Delta p > p^{\text{ms2}}$  the valve is completely open ( $a = A$ ); else  $p^{\text{ms1}} \leq \Delta p \leq p^{\text{ms2}}$  and the valve is only partially open ( $a = \frac{\Delta p - p^{\text{ms1}}}{p^{\text{ms2}} - p^{\text{ms1}}} A$ ). Having computed  $a$ , we skip the third phase. Otherwise ( $p^{\text{ms1}}$  was not given) we go to the next phase.

The *third and last phase* is as follows. The *nominal* opening area of the valve  $a'$  is assumed to be a linear function of the differential pressure  $\Delta p$  between the internal fluid ( $p$ ) and the external medium ( $p^{\text{ext}}$ ):

$$\begin{aligned} \Delta p &= p - p^{\text{ext}} \\ \text{for } \Delta p &\leq \Delta p^{\min} & a' &= 0 \\ \text{for } \Delta p^{\min} < \Delta p < \Delta p^{\max} & a' &= \frac{\Delta p - \Delta p^{\min}}{\Delta p^{\max} - \Delta p^{\min}} A \\ \text{for } \Delta p &\geq \Delta p^{\max} & a' &= A \end{aligned}$$

The *actual* opening area of the valve  $a$  at a given time depends on the opening or closure times, and on the valve opening reached at the previous time,  $a^{\text{old}}$ . Let  $v_a$  and  $v_c$  represent the velocity of aperture and closure of the valve:

$$\begin{aligned} v_a &= A/\Delta t^{\text{ope}} \\ v_c &= -A/\Delta t^{\text{clo}} \end{aligned}$$

Then:

$$\begin{aligned} \text{for } a' > a^{\text{old}} \text{ and } \Delta t^{\text{ope}} > 0 & a = \min[(a^{\text{old}} + v_a \Delta t), a'] \\ \text{for } a' < a^{\text{old}} \text{ and } \Delta t^{\text{clo}} > 0 & a = \max[(a^{\text{old}} + v_c \Delta t), a'] \\ \text{in all other cases} & a = a' \end{aligned}$$

The value of  $a$  is constrained to be between 0 and  $A$ :

$$\begin{aligned} \text{if } a > A & a = A \\ \text{if } a < 0 & a = 0 \end{aligned}$$

In order to compute the mass flow rate through the valve, we distinguish two cases, depending upon the compressibility of the discharged fluid.

### Incompressible or nearly incompressible fluid

The fluid being discharged is incompressible or nearly incompressible (e.g., a liquid). This situation is determined by the fact that the user has *not* specified a value for  $\gamma = C_p/C_v$  in the input data. In this case the following relation is assumed to hold:

$$\Delta p = p - p^{\text{ext}} = \zeta \rho \frac{v_{\text{tube}}^2}{2} \quad (41)$$

where  $\zeta$  is a dimension-less *resistance coefficient* which depends upon the geometry of the valve,  $\rho$  is the density of the fluid upstream the valve and  $v_{\text{tube}}$  is the velocity of the fluid in the tube upstream the valve.

By assuming steady state flow and a jet that expands back to fill the entire cross-section of the tube downstream the valve, the resistance coefficient  $\zeta$  is evaluated by:

$$\zeta = \left( \frac{1}{r_p C_c} - 1 \right)^2$$

where  $r_p$  is the current *perforation ratio*  $r_p$  of the valve, given by:

$$r_p = a/A^{\text{tube}}$$

Since the pressure drop  $\Delta p$  is known (we assume  $\Delta p = p - p^{\text{ext}}$ , without any limitations), from (41) we obtain the (nominal) fluid velocity in the tube upstream the valve:

$$v'_{\text{tube}} = \sqrt{\frac{2}{\zeta} \frac{p - p^{\text{ext}}}{\rho}}$$

and the current (nominal) mass flow rate is given by:

$$q' = A^{\text{tube}} \rho v'_{\text{tube}} = A^{\text{tube}} \sqrt{\frac{2}{\zeta} (p - p^{\text{ext}}) \rho}$$

### Compressible fluid

The fluid being discharged is compressible. This situation is determined by the fact that the user has specified a value for  $\gamma = C_p/C_v$  in the input data. (The given value of  $\gamma$ , assumed constant, should be equal to that of the fluid being discharged, i.e. the fluid upstream the safety valve.)

In this case the code assumes that the expansion of the fluid across the valve can be represented by a simple adiabatic transformation:

$$\frac{p}{\rho^\gamma} = \text{const.} \quad (42)$$

This is reasonable for a perfect gas but not, for example, if the fluid is a superheated vapor that becomes saturated during the expansion.

We compute the current *critical pressure*  $p_c$  corresponding to the current pressure  $p$  upstream the valve and to the chosen value of  $\gamma$ :

$$p_c = p \left( \frac{2}{\gamma + 1} \right)^{\frac{\gamma}{\gamma - 1}}$$

For example, if  $\gamma = 1.4$  like for air,  $p_c/p = 0.528$ . The pressure in the valve orifice (i.e., in the zone where the cross-section area of the duct is minimum) cannot drop below the critical value. We therefore distinguish two cases, depending on the value of the external pressure  $p^{\text{ext}}$  with respect to  $p_c$ :

- if  $p^{\text{ext}} \geq p_c$ , then the fluid pressure in the valve orifice is  $p_2 = p^{\text{ext}}$ , the fluid velocity in the orifice is

$$v_2 = \sqrt{2 \frac{\gamma}{\gamma - 1} \frac{p}{\rho} \left[ 1 - \left( \frac{p_2}{p} \right)^{\frac{\gamma - 1}{\gamma}} \right]}$$

and the (nominal) mass flow rate is given by

$$q' = S_2 v_2 \rho_2 = S_2 v_2 \rho \left( \frac{p_2}{p} \right)^{\frac{1}{\gamma}} = S_2 \sqrt{2 \frac{\gamma}{\gamma - 1} p \rho \left[ \left( \frac{p_2}{p} \right)^{\frac{2}{\gamma}} - \left( \frac{p_2}{p} \right)^{\frac{\gamma + 1}{\gamma}} \right]}$$

where  $p$  and  $\rho$  are the pressure and density of the fluid upstream the orifice, respectively, use has been made of (42) to explicitate the density  $\rho_2$  at the orifice and  $S_2$  is the flow area of the *vena contracta* at the orifice  $S_2 = C_c a$ .

- if  $p^{\text{ext}} < p_c$ , then the fluid pressure in the valve orifice is equal to the critical pressure  $p_2 = p_c$  and the (nominal) mass flow rate is given by

$$q' = \psi S_2 \sqrt{p\rho}$$

where  $S_2$  has the same meaning as above and the coefficient  $\psi$  is given by

$$\psi = \left( \frac{2}{\gamma + 1} \right)^{\frac{1}{\gamma-1}} \sqrt{2 \frac{\gamma}{\gamma + 1}}$$

### Flow limitation and velocity

Finally, in both cases (incompressible or compressible flow) the current (effective) mass flow rate  $q$  is computed by taking into account a possible user-imposed maximum value  $q^{\text{max}}$  and by inhibiting any back flow through the valve:

$$\begin{array}{ll} \text{for } q^{\text{max}} > 0 \text{ and } q' > q^{\text{max}} & q = q^{\text{max}} \\ \text{for } q' < 0 & q = 0 \text{ (no back flow)} \\ \text{in all other cases} & q = q' \end{array}$$

The current (effective) fluid velocity in the upstream tube is then:

$$v_{\text{tube}} = \frac{q}{A^{\text{tube}} \rho}$$

### Modelling of an orifice

In order to model a simple orifice in the wall of the pressurized tank, set  $A^{\text{tube}}$  to the area of the orifice and  $\Delta p^{\text{min}}$  to a very small value.

As concerns the optional parameters, leave out  $\Delta p^{\text{max}}$  and  $A^{\text{max}}$  so that it will be  $\Delta p^{\text{max}} = \Delta p^{\text{min}}$  and  $A^{\text{max}} = A^{\text{tube}}$ , leave  $\Delta t^{\text{ope}}$  to its default value of 0 and specify a huge value for  $\Delta t^{\text{clo}}$  so that the orifice will never close in practice.

### B - Loading mode

Loading mode is activated by specifying MODE 1 in the input data. In this mode,  $p^{\text{ext}}$  (assumed constant) must be specified since the default value of 0 is not appropriate. In order to completely define the external state (which in this case becomes the donor state) one must also give  $\rho^{\text{ext}}$  and  $i^{\text{ext}}$  (which are also assumed to be constant).

### Outputs:

The different components of the ECR table are as follows :

- ECR(1) : current pressure in the fluid element to which the valve is attached
- ECR(2) : current nominal opening area
- ECR(3) : current actual opening area
- ECR(4) : previous time this element was treated

ECR(5) : previous actual opening area

ECR(6) : total ejected mass (in discharge mode) or injected mass (in loading mode)

ECR(7) : total ejected energy by mass transport (in discharge mode) or injected energy (in loading mode)

ECR(8) : current flow tube velocity

ECR(9) : current density in the fluid element to which the valve is attached

ECR(10) : current specific internal energy in the fluid element to which the valve is attached

## 7.10 MECHANISMS

### Object:

This option allows to assign behaviour laws to the elements of a mechanical joint "MECA" or "LIGR" that link together two sub-structures.

For the "MECA" element : There are currently 5 directives, with several sub-directives.

FORC : Imposed force as a function of time over a slider ('glissière').

COUP : Imposed couple as a function of time over a pivot.

FOCO : Force and couple imposed on a sliding pivot.

MOCC : Electric motor with continuous current, on a pivot.

RESS : Linear or non-linear spring.

For the "LIGR" element : There is only 1 directive.

LIGR : Linear or non-linear spring.

Options	Sub-directives		Element
	mandatory	optional	
FORC	FONC	INER	MECA
COUP	FONC	INER	MECA
		ASSE	MECA
FOCO	FONC	INER	MECA
MOCC	FONC	INER	MECA
	ELEC	REDU	MECA
		TACH	MECA
		ASSE	MECA
RESS	/	/	MECA
LIGR	/	/	LIGR

### Syntax:

```
$ "FORC" | <"INER" ...> | /LECTURE/ $
```

	\$		"FONC" ...			\$
	\$					\$
	\$	"COUP"		<"INER" ...>		/LECTURE/
	\$			"FONC" ...		
	\$			<"ASSE" ...>		
	\$					
"MECA"	\$	"FOCO"		<"INER" ...>		/LECTURE/
	\$			"FONC" ...		
	\$					
	\$	"MOCC"		<"INER" ...>		/LECTURE/
	\$			"ELEC" ...		
	\$			"FONC" ...		
	\$			<"REDU" ...>		
	\$			<"TACH" ...>		
	\$			<"ASSE" ...>		
	\$					
	\$	"RESS"			/LECTURE/	
	\$					
	\$	"LIGR"			/LECTURE/	

## LECTURE

Index of the associated mechanism element.

**Comments:**

It is not necessary to systematically give a material law to elements of type "MECA" or "LIGR". If this is missing, one will have a simple kinematic joint.

### 7.10.1 IMPOSED FORCE

**Object:**

This directive allows to introduce a driving force on a junction of type 'glissière' (slider).

**Syntax:**

```
"FORC"      |      "FONC" ...      |  /LECTURE/  
             |      <"INER" ...>    |
```

**Comments:**

The behaviour law is defined by an imposed force  $F(t)$  un the axis of the slider.

The force is computed by the function:  $A : F(t) = \text{Function } A$ .

The B function is redundant.

**Outputs:**

ECR(5) : relative displacement of the slider since the start of the calculation

ECR(6) : relative velocity

ECR(7) : applied force



### 7.10.2 MOTOR COUPLE

#### Object:

This directive allows to define a motor couple on a connection of type 'pivot' (simple pin joint).

#### Syntax:

```
"COUP"      |      "FONC" ...      |  /LECTURE/
              |      <"INNER" ...>  |
              |      <"ASSE" ...>   |
```

#### Comments:

The law of behaviour is define by a motor couple  $C(t)$  on the axis of the pivot.

If the motor is not a servomotor:  $C(t) = \text{Function A}$ . In this case, the B function is redundant.

If the motor is a servomotor:  $C(t)$  is computed by the control function.

#### Outputs:

ECR(1) : angular displacement of the motor from the calculation origin (in radians)

ECR(2) : angular velocity (rad / s)

ECR(3) : couple on the arm

### 7.10.3 IMPOSED FORCE AND COUPLE

**Object:**

This directive is a combination of the 2 preceding ones ("FORC" and "COUP"), and is applied to sliding pivots ("PIGL").

**Syntax:**

```
"FOCO"      |      "FONC" ...      |  /LECTURE/  
             |      <"INER" ...>    |
```

**Comments:**

The behaviour law is defined by an imposed force  $F(t)$  on the slider axis, and by a motor couple  $C(t)$  around this axis.

The force is computed from function A:  $F(t) = \text{Function A}$ .

The couple is computed from function B:  $C(t) = \text{Function B}$ .

**Outputs:**

ECR(1) : angular displacement of the motor from the beginning of the calculation  
(in radians)

ECR(2) : angular velocity (rad / s)

ECR(3) : couple on the arm

ECR(5) : relative displacement of the slider from the beginning of the calculation

ECR(6) : relative velocity

ECR(7) : applied force

### 7.10.4 DIRECT CURRENT MOTOR

#### Object:

This directive allows to specify a motor couple on a junction of type 'pivot' (simple pin joint). The behaviour and the material characteristics are those of a direct current motor.

#### Syntax:

"MOCC"		"INER" ...		/LECTURE/
		"ELEC" ...		
		"FONC" ...		
		<"REDU" ...>		
		<"TACH" ...>		
		<"ASSE" ...>		

#### Comments:

\* U(t) - Tension at the motor poles

\* C(t) - Motor couple at the outlet shaft

\* tetap - Angular velocity of the motor shaft

The other variables are defined in the electric parameters.

The behaviour law is of the following type.

If the motor is not a servo-motor: U(t) = Function A. The B function is then redundant.

If the motor is a servo-motor: U(t) is computed from the control mechanism:

$$C(t) = (K_c * N * U(t) / R) - (N * N * tetap (F_f + K_c * K_c / R)) - (N * F_s)$$

(for the meaning of parameters K<sub>c</sub>, N, R, etc., consult pages C.760 and C.770).

#### Outputs:

ECR(1) : angular displacement of the motor since the beginning of the calculation  
(in radians)

ECR(2) : angular velocity (rad / s)

ECR(3) : couple on the arm

### 7.10.5 SPRING

#### Object:

This directive allows to introduce stiffnesses on the free d.o.f.s of mechanisms in order to model linear or non-linear springs.

#### Syntax:

```
"RESS"      ("K"   " k | "KFON"  kfon)
              ("C"   " c | "CFON"  cfon)
              "STAK"  stak      "STAC"  stac      /LECTURE/
```

**k**

Longitudinal stiffness for the linear spring.

**kfon**

Function number describing the longitudinal non-linear behavior. The function gives the force depending on the displacement.

**c**

Rotational stiffness for the linear spring.

**cfon**

Function number describing the rotational non-linear behavior. The function gives the moment depending on the angular displacement.

**stak**

Estimation of the maximal longitudinal oscillation period.

**stac**

Estimation of the maximal rotational oscillation period.

#### Comments:

This directive may be used in conjunction with a mechanism defined by a 2-nodes "MECA" element and an "ARTI" connection (except a ROTULE) on this very element.

The stiffnesses introduced in the articulated systems are associated with the free d.o.f.s: for example, for a sliding pivot, the stiffness K corresponds to the translational d.o.f. ('glissière'), while the stiffness C corresponds to the free rotational d.o.f. (pivot).

The direction of the displacement for the non-linear spring is defined according to the local axis of the "MECA" element or, in case of merging points, thanks to the axis defined for the connection.

**Outputs:**

ECR(1) : angular displacement since the beginning of the calculation (in radians)

ECR(2) : angular velocity (rad / s)

ECR(3) : applied couple

ECR(5) : relative displacement of the slider since the beginning of the calculation

ECR(6) : relative velocity

ECR(7) : applied force

ECR(8) : initial length of the MECA element

### 7.10.6 LIGR

**Object:**

This directive allows to introduce stiffnesses in rotation around the axis "AXE1" and "AXE2" (local axis of the shell) in order to model linear or non-linear springs. The axis "AXE1" and "AXE2" are defined in the "LINK COUP ARTI TGGR" (See D.270) or "LINK COUP ARTI CRGR" (See D.275) directives.

**Syntax:**

```
"LIGR"          ("KR1   "  kr1 | "KFR1"  kfr1)
                  ("KR2   "  kr2 | "KFR2"  kfr2)
                  <"FROT  "  frot >                /LECTURE/
```

**kr1**

Rotational stiffness for the linear spring around the axis "AXE1".

**kfr1**

Function number describing the rotational non-linear behavior. The function gives the moment depending on the relative angular displacement around the axis "AXE1".

**kr2**

Rotational stiffness for the linear spring around the axis "AXE2".

**kfr2**

Function number describing the rotational non-linear behavior. The function gives the moment depending on the relative angular displacement around the axis "AXE2".

**frot**

Frottement according to the perpendicular axis to the plan defined by "AXE1" and "AXE2".

**Comments:**

This directive may be used in conjunction with a mechanism defined by a multi-nodes "LIGR" element and an "ARTI TGGR" or an "ARTI CRGR" connection on this element.

The frottement according to the perpendicular axis to the plan defined by "AXE1" and "AXE2" make sense only for an "ARTI CRGR" connection.

**Outputs:**

ECR(1) : angular displacement since the beginning of the calculation around the axis "AXE1" (in radians)

ECR(2) : angular velocity (rad / s) around the axis "AXE1"

ECR(3) : applied couple around the axis "AXE1"

ECR(5) : angular displacement since the beginning of the calculation around the axis "AXE2" (in radians)

ECR(6) : angular velocity (rad / s) around the axis "AXE2"

ECR(7) : applied couple around the axis "AXE2"

**7.10.7 FUNCTIONS RELATED TO MECHANISMS****Object:**

This directive allows to specify the functions used for the articulated systems (mechanisms).

**Syntax:**

```
"FONC"          "COEA" coea    "NUFA" nufa    ...  
...             <"COEB" coeb>  <"NUFB" nufb>  ...
```

**coea**

Multiplying coefficient of function A.

**coeb**

Multiplying coefficient of function B.

**nufa**

Index of function A.

**nufb**

Index of function B.

**Comments:**

This directive is mandatory for articulated systems.



### 7.10.8 MECHANISM INERTIA

**Object:**

This directive allows to enter the values of mass and inertia for the articulated systems (mechanisms).

**Syntax:**

"INER"		"MMT1" m1	"IMT1" i1	...
	...	"MMT2" m2	"IMT2" i2	...

m1

Added mass on the first node of the mechanism.

i1

Added rotational inertia on the first node of the mechanism.

m2

Added mass on the second node of the mechanism.

i2

Added rotational inertia on the second node of the mechanism.

**Comments:**

The needed values are the additional inertias which have not been taken into account in the structures connected by the articulated system.

Rotational inertias are added along the axis of the mechanism.

**Warning:**

In the case of the direct current motor "MOCC", it is assumed that the rotor is placed on the first node, and the stator on the second node of the mechanism. Verify that the mesh conforms to this convention.

### 7.10.9 SERVOMECHANISM

**Object:**

This directive allows to specify a servo-mechanism.

The servo-control law is of the type P.I.D. (Proportional - Integrated - Derived):

$$F(t) = K_0 - K_i * (I - I_c) - K_p * (A - A_c) - K_v * (A' - A_c')$$

with:

I = integral of the angular position from 0 to t,

I<sub>c</sub> = integral of the command for the position.

**Syntax:**

"ASSE"    <"KP" kp>    <"KV" kv>    <"KI" ki>    <"K0" k0>

k0

Off-set tension constant.

kp

Parameter for the control in position.

kv

Parameter for the control in velocity.

ki

Parameter for the integral control.

**Comments:**

If this directive is missing, the mechanism is considered non-controlled.

**7.10.10 ELECTRIC PARAMETRES****Object:**

This directive allows to specify the electric parameters of a direct current motor.

**Syntax:**

```
"ELEC"  "R" r  <"KC" kc>  <"TNSN" tnsn> ...  
        ... <"INTS" is> <"FFMT" ffmt>  <"FSMT" fsmt>
```

**r**

Electric resistance of the motor.

**kc**

Torque constant per Ampère.

**tnsn**

Saturation tension in Volt.

**is**

Saturation intensity in Ampère.

**ffmt**

Viscous friction coefficient.

**fsmt**

Torque constant for dry friction.

**Comments:**

The friction values are those of the motor alone.

**7.10.11 REDUCER****Object:**

This directive allows to add the characteristics of a reducer to a mechanism of type motor.

**Syntax:**

```
"REDU"    <"MRD1" m1>    <"MRD2" m2>    <"IRD1" i1> <"IRD2" i2>    ...  
          ...    <"FFRD" ff> <"FSRD" fs> <"N" n>    ...
```

**m1**

Added mass on the first node of the mechanism.

**i1**

Added rotational inertia on the first node of the mechanism.

**m2**

Added mass on the second node of the mechanism.

**i2**

Added rotational inertia on the second node of the mechanism.

**n**

Reduction ratio.

**ff**

Viscous friction coefficient.

**fs**

Torque constant for dry friction.

**Comments:**

The rotational inertias are applied along the axis of the mechanism.

**Warning:**

In the case of the direct current motor "MOCC", it is assumed that the rotor is placed on the first node, and the stator on the second node of the mechanism. Verify that the mesh conforms to this convention.

**7.10.12 TACHYMETRIC GENERATOR****Object:**

This directive allows to add the characteristics of a tachymetric generator to a mechanism of type motor.

**Syntax:**

```
"TACH"    <"MGT1" m1>    <"MGT2" m2>    <"IGT1" i1>    <"IGT2" i2>  
          ... <"JASS" jbras>
```

**m1**

Added mass on the first node of the mechanism.

**i1**

Added rotational inertia on the first node of the mechanism.

**m2**

Added mass on the second node of the mechanism.

**i2**

Added rotational inertia on the second node of the mechanism.

**jbras**

Inertia seen by the mechanism.

**Comments:**

The "JASS" parameter is only used for the stability step calculation. An estimation, even in first approximation, is sufficient.

The rotational inertias are applied along the axis of the mechanism.

**Warning:**

In the case of the direct current motor "MOCC", it is assumed that the rotor is placed on the first node, and the stator on the second node of the mechanism. Verify that the mesh conforms to this convention.

## 7.11 ASSIGNING MATERIALS TO MULTILAYER SHELL ELEMENTS

### Object:

When using sandwiches composed of multiple layers in certain shell elements it is necessary to assign a (possibly different) material to each layer.

Sandwiches and layers are defined in the Geometry Complements section via the **SAND** directive, see page C.45.

In order to assign a given material to certain layers, first declare the material with all its properties as usual and list via the **/LECT/** directive all the elements that have that material, including sandwich elements that possess this material in at least one layer. Then, after the **/LECT/** directive, list all the layers which have the material (each layer is identified by a progressive index, as explained on page C.45).

### Syntax :

```
"MATE" "Material_Definition"  
  
    ( /LECT/    < "LAYE" /LECT_LAY/ > )
```

#### Material\_Definition

Definition of the material and its properties (see preceding sections), except the elements to which it is assigned.

**/LECT/**

Elements possessing this material.

**/LECT\_LAY/**

Layers of the **/LECT/** elements that possess the given material. Each layer is identified by an index, as explained on page C.45.

### Comments:

Note that the **/LECT/** directive (with its optional **LAYE** subdirective) may be repeated more than once for the same material. This allows e.g. to assign the same material to a few unlayered elements, then to a group of layered elements (i.e. a sandwich) in layers 1 and 3, then to another group of layered elements (i.e. another sandwich) in layers 2 and 4, and so on. For example:

```
"MATE" "Material_Definition"  
LECT 1 2 3 TERM  
LECT 4 6 TERM LAYE LECT 1 3 TERM  
LECT 5 8 TERM LAYE LECT 2 4 TERM  
. . .
```

The only element types that accept layers are ED01, COQI, CQD3, CQD4, CQD6 and CQD9. Since version 2005, Q4MC and T3MC are also available.

In order to be accepted in a layer, the material type must be available for the unlayered element type as well.

All layers of multilayer elements (sandwiches) must be explicitly assigned a material.

## 7.12 JOINT PROPERTIES

### 7.12.1 BUSHING ELEMENT

#### Object:

All the characteristics of the bushing element have to be given of the material type called "JOINT PROPERTIES". This sub-directive allow the definition of a stiffness or a damping in providing both the amplitude and the number of the function, which describes the force-displacement (force-velocity) or torque-rotation (torque-angular velocity) law. Nevertheless a simplified syntax is allowed for the case of linear laws, which can be expressed only by a constant stiffness/damping coefficient.

Note that two models have been implemented. The first one is "BSHT" with only translation degrees of freedom and the second one is "BSHR" with rotational degrees of freedom too. In the latter case the stiffness and/or damping can be applied also to the rotational degrees of freedom.

A suitable criteria to properly introduce a model of rupture in the riveted joints can be used. This criteria is based on the rigid behaviour of the joints; the rupture occurs when the following limit curve is reached :

$$(N/N_u)^{**a} + (T/T_u)^{**b} \geq 1$$

Four different laws of behaviour can be used :

- 1) Bushing with elastic behaviour;
- 2) Bushing with plastic behaviour;
- 3) Bushing with elastic behaviour and elliptic rupture criteria;
- 4) Bushing with plastic behaviour and elliptic rupture criteria;

#### Syntax:

##### 1) Generic data

Each behaviour is declared by a different sub-directive.

- 1) Case 1

"JPRP" "BUSH"

- 2) Case 2

"JPRP" "BPLA"

- 3) Case 3

"JPRP" "BELC"

- 4) Case 4

"JPRP" "BPEC"



## - 2) Data concerning the law of behaviour

```

<"KTXC" ktxc <"KTXN" ktxn>> <"KTYC" ktyc <"KTYN" ktyn>> <"KTZC" ktzc <"KTZN" ktzn>>
<"KRXC" krxc <"KRXN" krxn>> <"KRYC" kryc <"KRYN" kryn>> <"KRZC" krzc <"KRZN" krzn>>

<"DTXC" dtxc <"DTXN" dtxn>> <"DTYC" dtyc <"DTYN" dtyn>> <"DTZC" dtzc <"DTZN" dtzn>>
<"DRXC" drxc <"DRXN" drxn>> <"DRYC" dryc <"DRYN" dryn>> <"DRZC" drzc <"DRZN" drzn>>

<"V1X " v1x "V1Y " v1y "V1Z " v1z >
<"V2X " v2x "V2Y " v2y "V2Z " v2z >
<"EX " ex > <"EY " ey > <"EZ " ez >
<"LOCA" loca>
<"DISP" disp>
<"RADB" radb>

```

**ktxr**

Amplitude multiplying the function describing X-translational stiffness or constant X-translational stiffness if the function KTXN is not given (REAL).

**ktxn**

Number of the function describing X-translational stiffness (INTEGER).

**ktyr**

Amplitude multiplying the function describing Y-translational stiffness or constant Y-translational stiffness if the function KTYN is not given (REAL).

**ktyn**

Number of the function describing Y-translational stiffness (INTEGER).

**ktzr**

Amplitude multiplying the function describing Z-translational stiffness or constant Z-translational stiffness if the function KTZN is not given (REAL).

**ktzn**

Number of the function describing Z-translational stiffness (INTEGER).

**krxr**

Amplitude multiplying the function describing X-rotational stiffness or constant X-rotational stiffness if the function KTXN is not given (REAL).

**krxn**

Number of the function describing X-rotational stiffness (INTEGER).

**kryr**

Amplitude multiplying the function describing Y-rotational stiffness or constant Y-rotational stiffness if the function KTYN is not given (REAL).

**kryn**

Number of the function describing Y-rotational stiffness (INTEGER).

krzr

Amplitude multiplying the function describing Z-rotational stiffness or constant Z-rotational stiffness if the function KTZN is not given (REAL).

krzn

Number of the function describing Z-rotational stiffness (INTEGER).

dtxr

Amplitude multiplying the function describing X-translational damping or constant X-translational damping if the function KTXN is not given (REAL).

dtxn

Number of the function describing X-translational damping (INTEGER).

dtyr

Amplitude multiplying the function describing Y-translational damping or constant Y-translational damping if the function KTYN is not given (REAL).

dtyn

Number of the function describing Y-translational damping (INTEGER).

dktzr

Amplitude multiplying the function describing Z-translational damping or constant Z-translational damping if the function KTZN is not given (REAL).

dtzn

Number of the function describing Z-translational damping (INTEGER).

drxr

Amplitude multiplying the function describing X-rotational damping or constant X-rotational damping if the function KTXN is not given (REAL).

drxn

Number of the function describing X-rotational damping (INTEGER).

dryr

Amplitude multiplying the function describing Y-rotational damping or constant Y-rotational damping if the function KTYN is not given (REAL).

dryn

Number of the function describing Y-rotational damping (INTEGER).

drzr

Amplitude multiplying the function describing Z-rotational damping or constant Z-rotational damping if the function KTZN is not given (REAL).

drzn

Number of the function describing Z-rotational damping (INTEGER).

v1x v1y v1z

cartesian components for vector V1 of the user defined frame.

v2x v2y v2z

cartesian components for vector V2 of the user defined frame.

ex ey ez

eccentricity of the user defined frame from node 1

loca

location of the user defined frame origin between node 1 and node 2

disp

flag for taking into account initial position (default 0).

radb

flag for the use of the radial bushing element formulation.

### - 3) Data concerning the elliptic criteria:

```
<"NULT " nult "TULT " tult >
<"COEA " coea "COEB " coeb >
```

NULT

ultimate axial effort

TULT

ultimate radial effort

COEA

coefficient a of the elliptic criteria

COEB

coefficient b of the elliptic criteria

### Comments:

As default the force-position and force-velocity laws are defined in the global reference frame: however a user-defined properties frame (x1, x2, x3) is accepted via the directives V1X, V1Y, V1Z, V2X, V2Y, V2Z. After the two vectors V1 and V2 have been defined, the code applies a normalization to obtain the unit versors v1 and v3, and computes the v3 versor as:

$$\{v3\}=\{v1\}^{\wedge}\{v2\}$$

The laws are assumed to be diagonal in the properties frame ( $x_1, x_2, x_3$ ), that is the force  $F_i$  along  $x_i$  depends only upon the relative position  $u_i$  and relative velocity  $v_i$  along the  $x_i$  local direction.

The user can also define a point P in the properties frame where the force and the torque should be applied: this must be done using the parameter LOCA and, if needed the eccentricity parameters EX, EY, EZ. The effect of the parameter LOCA, which must be a real number in the interval  $[0,1]$ , is to move the application point on the segment AB: it is actually the percentage of the position of P along AB. The default is LOCA 0.0. If offsets  $ex, ey, ez$  are also defined, their effect is to translate along the directions ( $x, y, z$ ) of the global frame, the point P, from its position P' defined by LOCA parameter.

### Outputs:

The components of the ECR table are as follows:

- ECR(1): Translation force  $F_{x1}$  between the two nodes along  $x_1$  local axis
- ECR(2): Translation force  $F_{y1}$  between the two nodes along  $y_1$  local axis
- ECR(3): Translation force  $F_{z1}$  between the two nodes along  $z_1$  local axis
- ECR(4): Torque  $T_{x1}$  between the two nodes around  $x_1$  local axis
- ECR(5): Torque  $T_{y1}$  between the two nodes around  $y_1$  local axis
- ECR(6): Torque  $T_{z1}$  between the two nodes around  $z_1$  local axis
- ECR(7): Relative position between the two nodes along  $x_1$  local axis
- ECR(8): Relative position between the two nodes along  $y_1$  local axis
- ECR(9): Relative position between the two nodes along  $z_1$  local axis
- ECR(10): Relative rotation between the two nodes around  $x_1$  local axis
- ECR(11): Relative rotation between the two nodes around  $y_1$  local axis
- ECR(12): Relative rotation between the two nodes around  $z_1$  local axis
- ECR(13): Relative translation velocity between the two nodes along  $x_1$  local axis
- ECR(14): Relative translation velocity between the two nodes along  $y_1$  local axis
- ECR(15): Relative translation velocity between the two nodes along  $z_1$  local axis
- ECR(16): Relative angular velocity between the two nodes around  $x_1$  local axis
- ECR(17): Relative angular velocity between the two nodes around  $y_1$  local axis
- ECR(18): Relative angular velocity between the two nodes around  $z_1$  local axis
- ECR(25): Value of the rupture elliptic criteria

## 8 GROUP D—LINKS

### Object:

To introduce links between degrees of freedom. Links may be subdivided into three broad categories:

- **Coupled** links, which are treated (implicitly) by a method of Lagrange multipliers. This ensures proper coupling between all the imposed links, provided of course the specified conditions are compatible.
- **Decoupled** links, which are imposed via ad-hoc direct methods. In this case the conditions should be independent from one another, and the user is responsible for ensuring this property.
- **”Liaisons”**, which are treated using the former LIAISON directive for compatibility purposes. These links can be either coupled or uncoupled (see below).

Following the above subdivision, this directive admits three forms, characterized by the respective sub-directives: LINK COUP, LINK DECO or LINK LIAI. The complete syntax is summarized below. For each variation of the main directive (COUP, DECO or LIAI) the available link types are listed in the relevant column, so as to provide a compact overview.

### Syntax:

\$ LINK COUP	\$ LINK DECO	\$ LINK LIAI	\$
\$ <NONP>	\$	\$	\$
\$ <SOLV . . .>	\$	\$ <SOLV . . .>	\$
\$ <RENU ; NORE>	\$	\$ <RENU ; NORE>	\$
\$ <VERI> <ARRA>	\$	\$ <FREQ ifreq>	\$
\$ <SPLT \$ DOF ;	\$	\$ <VERI>	\$
\$       NODE ;	\$	\$	\$
\$       DOMA ;	\$	\$	\$
\$       PART ;	\$	\$	\$
\$       NONE \$>	\$	\$	\$
\$ <SOL2>	\$	\$	\$
\$ <GPCG . . .>	\$	\$	\$
\$ <UPDT /CTIME/>	\$	\$	\$

### 8.1 LIST OF LINKS

	COUP	DECO	LIAI	
8.6	BLOQ	BLOQ	BLOQ	Blockages
8.7	TBLO	TBLO	BLOQ	Time limited blockages
8.8	GUID			
8.9	CONT		CONT	
8.10	RADI		RADI	
8.11	RELA		RELA	Relations
8.12	ARMA	ARMA	ARMA	

8.13	CROS	CROS		
8.14		ACBE		Rebar (FEM)-concrete (DEM) link
8.15	LCAB			Link between prestressing cables and concrete
8.16	ARLQ			Shell-3D mesh coupling (ARLEQUIN framework)
8.17	DEPL	DEPL	DEPL	Imposed motions, displacement
8.17	VITE	VITE	VITE	Imposed motions, velocity
8.17	ACCE	ACCE	ACCE	Imposed motions, acceleration
8.18	COQM		COQM	Connections between shells and solid elements
8.19	INTE			Interfaces
8.20	FLST			Fluid-structure coupling
8.21	FLSR	FLSR		Fluid-structure coupling
8.22	FLSX	FLSX		Fluid-structure coupling
8.23.1	FS		FS	
8.25	IMPA	IMPA	IMPA	Impacts and unilateral conditions
8.26			JEUX	
8.27			BUTE	
8.28	GLIS	GLIS	GLIS	
8.29			MPEF	Method of particles and forces
8.30			SPHY	Smoothed particle hydrodynamics method (SPH)
	TYPL			???????
8.31	EDEF			Connecting finite and discrete element models
8.32	BIFU		BIFU	Bifurcation connection
8.33	ADHE			Adhesion connection
8.34	TUBM		TUBM	Tubm connection (3d-1d junction)
8.35	TUYM		TUYM	Tuym connection (3d-1d junction)
8.36	TUYA		TUYA	Tuya connection (3d-1d junction)
8.37	SOLI		SOLI	Rigid body (solide indeformable)
	COMP		COMP	????
8.38	ARTI		ARTI	Articulation
8.39	ROTA		ROTA	Rotation
8.40	MENS		MENS	Imposed time-dependent rotational motion
8.41	TMEN			Time-limited imposed rotational motion
8.42	DIST		DIST	Constant distance connection
8.43	BARY		BARY	Barycentric junction
8.44	RIGI	RIGI	RIGI	Rigid junction
8.45	GLUE			Gluing together two meshes (glue)
8.46			SPLI	Contacts defined by spline functions
8.47			COLL	Collisions
8.48	FSA		FSA	Fluid-structure sliding of ale type (FSA)
8.49	FSR		FSR	Rigid-boundary/fluid sliding of ale type
8.50	PINB	PINB	PINB	Impact/contact by pinball model (PINB)
8.51	GPIN	GPIN		Contact/impact by generalized pinball model (GPIN)
8.52		FSS		Fluid-structure sliding (FSS)
8.53	SH3D		SH3D	Node to shell connector

---

8.54		FLSW	Weak fluid-structure coupling (FLSW)
8.55	MAP2	MAP2	Node on facet element
8.55	MAP3	MAP3	Node on facet element
8.55	MAP4	MAP4	Node on facet element
8.55	MAP5	MAP5	Node on facet element
8.55	MAP6	MAP6	Node on facet element
8.55	MAP7	MAP7	Node on facet element
8.56	FESE		Finite-element/spectral-element interface
8.57	NAVI		Navier-stokes (incompressibility)
8.58	BREC		Pipeline rupture connection
8.59		PELM	Surface pressure measured in an element (PELM)
8.60		ADAP	Uncoupled hanging links
8.61	ENGR	ENGR	Prescribed damage for gradient damage materials
8.62		DRAG	Drag force on fluid-embedded 3D beam/bar elements

---

**COUP**

Introduces the set of coupled links. All coupled links must be declared within this set (the keyword **COUP** may not be repeated).

**DECO**

Introduces the set of decoupled links. All decoupled links must be declared within this set (the keyword **DECO** may not be repeated).

**LIAI**

Introduces the set of "liaison" links. All "liaison" links must be declared within this set (the keyword **LIAI** may not be repeated).

**Comments:**

The **COUP** and **LIAI** subdirectives are mutually exclusive. However, they may be combined with the **DECO** links within the same calculation, with the following syntax:

```
LINK $[COUP ; LIAI]$
    (declare all coupled/"liaison" links here ...)
LINK DECO
    (declare all uncoupled links here ...)
```

The subdirective **LIAI** is not compatible with the other types of links. Only the coupled links declared with **LINK COUP** may be used also in conjunction with *domain decomposition* (see the **STRU** directive), while this is not the case for the **LINK LIAI** directive.

Beware that, for the moment, only the **BLOQ**, **DEPL**, **VITE** and **ACCE** models are accepted in calculations with sub-domains.

Note that the availability of each link formulation introduced above with one or more of the link types is given in their specific manual page below.

**Warning:**

The `LINK COUP` directive allows to build a coupling matrix between the different degrees of freedom appearing in the connections. This matrix must be invertible, and a problem occurs in this sense if the different connections are not independent from each other.

In principle, EUROPLEXUS is able to eliminate the redundant relations in order to be able to invert the connections matrix. But since this elimination is a trial-and-error process, it is preferable when possible to avoid this situation. If this is not possible, it is recommended to start by giving the most complex connections (solids or articulations), and to finish by giving the simplest ones (relations or blockages).

**Comments:**

Be sure to check the various options available in relation to connections: see Page H.160.



## 8.2 LINK CATEGORY

### Object:

Choose between the available categories for links (see [GBD\\_0010](#)).

### Syntax:

```
$[ COUP ; DECO ; LIAI ]$
```

#### COUP

Introduces the set of coupled links. All coupled links must be declared within this set (the keyword **COUP** may not be repeated).

#### DECO

Introduces the set of decoupled links. All decoupled links must be declared within this set (the keyword **DECO** may not be repeated).

#### LIAI

Introduces the set of "liaison" links. All "liaison" links must be declared within this set (the keyword **LIAI** may not be repeated).

### 8.3 OPTIONS FOR COUPLED LINKS

**Object:**

**Syntax:**

```

LINK COUP
  <NONP>
  <SOLV  |[ CHOL
                                ;
                PARD
                                ;
                SPLI < TYPE  imet > < PCON ipre >
                    < PITE  prec > < IPA1 ip1  >
                    < IPA2  ip2  > < RPAR rp   >
                    < INIS  inig >                                ]| >
  $[ RENU ; NORE ]$ <VERI>
  <ARRA>
  <SPLT |[ DOF   ; NODE ; DOMA ; PART ; NONE ]| >
  <SOL2>
  <GPCG < PREC prec > < APRE apre > < DUMP > >
  <UPDT /CTIME/>

```

**NONP**

Accept the occurrence of non-permanent links (such as Lagrangian contacts for example) during the calculation even though there are no non-permanent links explicitly declared in the input. This optional keyword should be used only in the special case that there are no non-permanent links explicitly declared in the input, but at the same time there is some other model which might introduce non-permanent links during the course of the calculation. For example, the Lagrangian contact between SPH particles and structures treated by the **SPHY COQU** or **SPHY STRU** directive, when the **LAGC** keyword is also specified in the calculation type. In this case the model may add coupled, non-permanent links during the course of the calculation, which is only accepted if either some non-permanent links are explicitly declared, or if the present **NONP** keyword is specified. In the second case, just add **LINK COUP NONP** to your input.

**imet**

iterative solver number i, default 8

**ipre**

preconditioner number k, default 3

**prec**

tolerance on residual, default 1.D-6

**ip1**

first integer parameter, default 20

**ip2**

second integer parameter, default 10

**rp**

real parameter, default 1.D-4

**inig**

initial guess, default 0.0

**GPCG**

MPI Only - Use a Global Preconditioned Conjugate Gradient algorithm for the solver.

**PREC** **prec**

MPI Only - Relative precision for GPCG solver (see comment below)

**APRE** **apre**

MPI Only - Absolute precision for GPCG solver (see comment below)

**DUMP** Additional information is printed on the listing output during the iterations of the GPCG algorithm if the keyword **DUMP** is used.

### Comments:

The **SOLV** directive is fully described in [8.4](#) (page D.20).

Concerning imposed motions (directives **DEPL**, **VITE** and **ACCE**), note that no dimensions are needed relative to the motions themselves. However, dimensioning relative to the time tables describing such motions is still necessary (see keywords **FNOM**, **FTAB**).

The **ARMATURE** directive is described in [8.12](#) (page D.125).

The optional keyword **RENU** makes it possible to renumber the links in order to minimize the size of the matrix.

By default, links are renumbered (option **RENU**).

If the optional keyword **NORE** is specified, the links are taken in the order of their definition, and the matrix can be very large and ill-conditioned. If **RENU** (or nothing) is specified, then the connections are renumbered in an attempt to minimize the size of the matrix.

The optional keyword **VERI** can be used to verify a posteriori that the imposed links are effectively satisfied. This option produces heavy output and CPU overhead and should therefore be used only for debugging purposes.

The optional keyword **ARRA** can be used to choose storage of the links in a dynamic array rather than in a doubly linked list of doubly linked lists. This may increase efficiency (but is still under development).

The optional keyword **SPLT** can be used to choose the desired strategy for splitting the constraints into groups. The following possibilities are currently available:

- **DOF** requests dof-based splitting. This is the default and normally needs not being specified explicitly.
- **NODE** requests node-based splitting.
- **DOMA** requests subdomain-based splitting. Of course, this is only effective in calculations with domain decomposition.
- **PART** requests splitting based upon intrinsic node level factor. Of course, this is only effective in calculations with space partitioning.
- **NONE** requests no splitting. All constraints end up in a single, big group.

The optional keyword **SOL2** can be used to choose closed-form solution for groups of links containing just two links (in addition to groups containing just one link). By default, all groups of links containing more than one links are solved by the general numerical method (Choleski's method). In some cases, closed-form solution may be more efficient.

The optional keyword **GPCG** toggles the use of a specific Preconditioned Conjugate Gradient solver for links coupling several subdomains within parallel MPI framework. The precision of the solution is computed in terms of relative residual with respect to the norm of the right-hand side vector. Default precision is  $10^{-5}$  and it can be changed using the keyword **PREC**. An additional precision acting on the absolute norm of the residual can be entered using the keyword **APRE**. It is useful in the rare cases where the norm of the right-hand side vector is very small. Additional information is printed on the listing output during the iterations of the algorithm if the keyword **DUMP** is used.

The optional keyword **UPDT** can be used to introduce an update frequency for time-varying links, in order to save CPU time. This is especially useful for fluid-structure interaction, when links are classically updated at each time-step, with a frequency obtained from the CFL condition, whereas their update should follow the physical structural velocity, often much smaller than sound speed in the different media.

As far as **GLIS** models are concerned, only 3D models (sliding surface) are currently available. For 2D models (sliding lines), please use decoupled or "liaison" links.

The **EDEF** directive is described in [8.31](#) (page D.189).

## 8.4 OPTIONS FOR "LIAISON" LINKS

### Object:

The option "SOLV" makes it possible to change the resolution method in order to reduce the time spent to solve large matrix systems. For iterative solvers ("SPLI" keyword) the data structure uses the CSR format (Compressed Sparse Rows), well suited for iterative solution and used in the SPLIB library.

The option "RENUM" makes it possible to renumber the connections in order to minimize the size of the matrix.

The optional keyword "FREQ" can be used to avoid the inversion of the connection matrix at each computation step, in the cases where this is possible (see below).

The optional keyword "VERI" can be used to verify a posteriori that the imposed liaisons are effectively satisfied. This option produces heavy output and CPU overhead and should therefore be used only for debugging purposes.

### Syntax:

```
< "SOLV"  |[ "CHOL"                                ;
              "PARD"                                ;
              "SPLI" < "TYPE"  imet > < "PCON" ipre > ...
              ...  < "PITE"  prec > < "IPA1" ip1  > ...
              ...  < "IPA2"  ip2  > < "RPAR" rp   > ...
              ...  < "INIS"  inig >                                ]| >

< $[ "RENU" ; "NORE" ]$ >  < "FREQ"  ifreq > < "VERI" >
```

**imet**

iterative solver number i, default 8

**ipre**

preconditioner number k, default 3

**prec**

tolerance on residual, default 1.D-6

**ip1**

first integer parameter, default 20

**ip2**

second integer parameter, default 10

rp

real parameter, default 1.D-4

inig

initial guess, default 0.0

ifreq

The matrix will be inverted each ifreq computation step; by default, ifreq is 1, i.e. the matrix is inverted at each step. This has only effect in Lagrangian computations. In ALE or Eulerian cases, the matrix is inverted anyway at each time step (like if ifreq=1) because the nodal masses are continuously changing due to transport.

### Comments:

If the option "CHOL" is specified, the standard direct solver is used. This is the default option.

When the keyword "PARD" is specified, the library **PARDISO** is used. The package **PARDISO** is a thread-safe, high-performance, robust, memory efficient and easy to use software for solving large sparse symmetric and unsymmetric linear systems of equations on shared-memory and distributed-memory multiprocessors. This package is include in MKL library provided by the Intel compiler.

<http://www.pardiso-project.org/>

<http://software.intel.com/>

When the keyword "SPLI" is specified, the iterative solver is used. SPLIB is a library of iterative solvers with preconditioners for symmetric and nonsymmetric systems. It has been adapted for large matrix systems in the EUROPLEXUS code and uses the Compressed Sparse Row format (CSR).

<http://www.netlib.org/utk/papers/iterative-survey/node61.html>

The integer parameter `imet` specifies which SPLIB solver to use :

`imet = 1` : Bi-Conjugate Gradients

`imet = 2` : Conjugate Gradients with  $AA'y = b$ ,  $x = A'y$

`imet = 3` : Conjugate Gradients with  $A A' x = A'b$

`imet = 4` : Conjugate Gradients Squared (CGS)

`imet = 5` : Conjugate Gradients Stabilized

`imet = 6` : GMRES(ip2)

`imet = 7` : Transpose free QMR

`imet = 8` : Template version of Conjugate Gradients Stabilized

`imet = 9` : Template version of GMRES(ip2)

The integer parameter `ip2` used in GMRES solvers defines the Krylov subspace size (default value 10). When an `imet` of zero or less is specified, CG-Stabilization is used.

The integer parameter `ipre` specifies which preconditioner to use :

`ipre = 0` : no preconditioner

`ipre = 1` : ILU(`ip1`)

`ipre = 2` : MILU(`ip1`,`rp`)

`ipre = 3` : ILUT(`ip1`,`rp`)

`ipre = 4` : SSOR(`rp`)

`ipre = 5` : TRID(`ip1`)

`ipre = 6` : ILU0

`ipre = 7` : ECIMGS(`rp`)

The integer parameter `ip1` indicates the levels of fill-in to allow for ILU and MILU and the block size to use in the tridiagonal preconditioner TRID. For ILUT, `ip1` is the maximum additional entries allowed per row in the preconditioner compared to the original matrix. The real parameter `rp` is the relaxation parameter, the amount of multiply discarded fill-in entries before adding them to the diagonal. For SSOR it is the relaxation parameter. For ILUT, it is the drop tolerance.

For ECIMGS, `rp` specifies the sparsity pattern of the preconditioner :

- `rp = 0` : use the non zero pattern of the matrix
- $0 < rp < 1$  : use a sparser pattern than that of the matrix
- `rp = 1` : use a diagonal pattern
- $rp > 1$  : use a denser pattern with `int(rp)` levels of fill-in

It is greatly recommended to use default values by entering only the following key words : "LIAIS" "SOLV" "SPLIB".

More informations to use SPLIB options can be found in this paper : "SPLIB : A library of iterative methods for sparse linear systems" by R. Bramley and X Wang, department of computer science - Indiana University, 1995.

For information about methods implemented, see for example the following reference by Y. Saad : "Iterative Methods for Sparse Linear Systems". This book can be found at <http://www-users.cs.umn.edu/~saad>

By default, connections are renumbered (option "RENU").

If the option "NORE" is specified, the connections are taken in the order of their definition, and the matrix can be very large and ill-conditioned. If "RENU" (or nothing) is specified, then the connections are renumbered in an attempt to minimize the size of the matrix.

If the connections are simple fixed displacements, a new numeration is useless because the matrix is diagonal.

The option `FREQ` is not compulsory. If it is not specified, a new computation is done at every time step.

When the coefficients of the relations between the degrees of freedom depend on the updated geometry (see `COQM` and `FS`), it is necessary to perform new computations and to invert the matrix at each time step during a `EUROPLEXUS` run. This operation is very costly if there are many coupled degrees of freedom. The keyword `"FREQ"` requests a new computation and an inversion only every `ifreq` computation steps.

In the case of an incompressible fluid or an `A.L.E` or Eulerian computation it is necessary to invert the matrix at each time step because the nodal masses are continuously changing due to transport. Therefore, the code ignores the user-supplied value for `ifreq` in these cases.

The same holds for an incompressible calculation, or for a calculation involving non-deformable sub-structures (keywords `"NAVIER"` and `"SOLIDE"`).



## 8.5 AUXILIARY FILE

### Object:

This directive allows to read the connections data from an auxiliary file.

### Syntax:

```
< "FICHIER"    'nom.fic'  >
```

In certain cases the data may be bulky. It is then recommended to store them on an auxiliary file to shorten the main input data file. The auxiliary file is activated by means of the keyword "FICHIER" that precedes the file name (complete under Unix). In the main data file then only the keywords "LIAISON" "FICHIER" remain.

The auxiliary file (in free format) contains the whole set of connections data, except the keyword "LIAISON". To return to the main input data, the auxiliary file must be terminated by the keyword "RETOUR".

## 8.6 BLOCKAGES

### Object:

To prescribe a zero displacement to (i.e., to block) a degree of freedom, that is to say to ensure the relation  $U(i) = 0$ .

Compatibility: COUP, DECO, LIAI

### Syntax:

```
"BLOQ" ( /LECDDL/ /LECTURE/ )
```

```
/LECDDL/
```

Reading procedure of the degrees of freedom concerned.

```
/LECTURE/
```

Reading procedure of the numbers of the blocked nodes.

### Comments:

It is possible to repeat the same blockage several times. Indeed, when a boundary is described, it is often simpler to use the implicit definition of the procedure `/LECTURE/`; in this case the points which are located at the ends are written twice. The EUROPLEXUS program eliminates these double definitions before it builds up the matrix.

Note, however, that the program is unable to eliminate the repeated points if e.g. several "BLOQ" keywords are used.

A time-limited version (TBLO) of the BLOQ directive, which acts only until a certain time and then is automatically removed, is also available, see Page D.31.

## 8.7 TIME-LIMITED BLOCKAGES

### Object:

To prescribe a zero displacement to (i.e., to block) a degree of freedom, that is to say to ensure the relation  $U(i) = 0$ , up to a certain time or a certain event. After the chosen time (or event), the blockage is automatically released.

Compatibility: COUP, DECO

### Syntax:

```
"TBLOQ" ( /LECDDL/ $ "UPTO" t ; "TRIG" $ /LECTURE/ )
```

/LECDDL/

Reading procedure of the degrees of freedom concerned.

UPTO t

Time up to which the blockage is imposed. After this time, the blockage is automatically released.

TRIG

The blockage is imposed only until a trigger is activated. The trigger refers to the TRIG keyword which activates mesh refinement in some adaptivity models, see OPTI ADAP TRIG on Page H.180. After this time, the blockage is automatically released.

/LECTURE/

Reading procedure of the numbers of the blocked nodes.

### Comments:

It is possible to repeat the same blockage several times. Indeed, when a boundary is described, it is often simpler to use the implicit definition of the procedure /LECTURE/; in this case the points which are located at the ends are written twice. The EUROPLEXUS program eliminates these double definitions before it builds up the matrix.

Note, however, that the program is unable to eliminate the repeated points if e.g. several "BLOQ" keywords are used.

## 8.8 GUIDE (SLIDE CHANNEL)

### Object:

This directive aims to model a piping guide, i.e. the support that blocks the transverse displacement of the piping at a given node, while keeping free its longitudinal motion.

The sliding direction is prescribed by giving 3 parameters DIRX, DIRY and DIRZ, which correspond to 3 components of the direction vector. Rotational degrees of freedom are left free.

Compatibility: COUP

### Syntax:

```
"GUIDE" "DIRX" rx "DIRY" ry "DIRZ" rz /LECTURE/ )
```

rx ry rz

Components that define the direction of the guide.

/LECTURE/

Reading procedure of the number or of the name of the node concerned.

### Comments:

Definition of the local frame (x,y,z) with respect to the global frame (X,Y,Z):

- Local x-axis is collinear with the sliding direction specified by the user through the triplet DIRX, DIRY, and DIRZ.
- Local z-axis is orthogonal to x-axis and it is situated in the plane described by the axes x and Z. Positive projection on Z is used.
- Local y-axis completes the direct orthogonal axis system.

If the slider direction is vertical, the local x-axis is collinear with the sliding direction, the y-axis is collinear with Y-axis, and z-axis completes the direct orthogonal axis system.

To obtain the reaction forces in the local frame corresponding to the guide, one must define the node as a REGION and give the same components of the direction vector.

## 8.9 GEOMETRIC BILATERAL RESTRAINTS (CONTACTS)

### Object:

The following instructions are used to automatically write relations imposed by boundary conditions of geometrical origin. For instance, the user wants certain nodes of an element to stay on a given structure, or to impose symmetry conditions for one part of the boundary.

Compatibility: COUP, LIAI

### Syntax:

```
"CONT"
| "PLAN"  ...  |
| "SPHE"  ...  |
| "CYLI"  ...  |
| "CONE"  ...  |
| "TORE"  ...  |
| "SPLA"  ...  |
```

### Comments:

Do not forget to dimension (see "RELA" n1 n2, page A80).

Here n1 represents the maximum number of nodes in contact and n2 is equal to 2 for 2-D computations and 3 for 3-D computations.

It is very important to note that the behaviour of these directives (except PLAN and SPLA) is different, according to the fact that the constraints coefficients are considered to be constant, or allowed to vary in time (the desired behaviour may be chosen via the OPTI CONT option, described in Section H). By default the constraint coefficients are determined on the initial configuration and are kept constant in time. This treatment is always adequate for the PLAN and SPLA types of constraint (since the normal to the plane does not vary in time anyway). However, for the other directives it is only adequate if the nodes do not move, i.e. for Eulerian nodes. In this case, the directives represent a handy shortcut for specifying constraints with coefficients different from point to point (but constant in time), without having to write such conditions explicitly in the input file.

But when the nodes move in time, i.e. for Lagrangian or ALE nodes, the use of constant coefficients in time is no longer adequate. The coefficients should be recomputed at each time step, which may be a costly operation. The user may require this updating of the coefficients by specifying the OPTI CONT VARI option, see Section H. Using variable coefficients has as effect that the nodes move by remaining on the imposed surface with first-order accuracy.

The instructions are described in detail on the following pages.

### 8.9.1 PLANE/LINEAR RESTRAINT (CONTACT PLAN)

#### Object:

The specified nodes lay (and remain) on a plane normal to a given vector. In 2D, the plane reduces to a straight line. Only translational degrees of freedom are blocked.

Compatibility: COUP, LIAI

#### Syntax:

```
"PLAN"  | [  "NX" x  "NY" y  < "NZ" z > ;
            "NR" r  "NZ" z          ;
            "POIN" /LECT1/          ;
            "AUTO"                  ] | /LECT2/
```

x y

Components of the normal vector (2-D).

x y z

Components of the normal vector (3-D).

r z

Components of the normal vector (Axisymmetric).

POIN /LECT1/

Must specify a node belonging to the mesh, either via its index or via its CASTEM 2000 name. The coordinates of this node are taken as the components of the normal.

AUTO

The components of the normal are determined automatically from the position of the nodes listed in /LECT2/. Therefore, in this case, the nodes contained in the following /LECT2/ list must lie on the same line (in 2D) or plane (in 3D). In the 3D case, of course, the listed nodes must define a plane (not be along the same line).

/LECT2/

Numbers of the nodes concerned.

#### Comments:

It is not necessary that the normal vector be unitary, since it is automatically normalised by the program. Furthermore, it is not necessary that the nodes initially belong to the same line or plane (except in the AUTO case).

The difference between this directive and the CONT SPLA directive (see below) is that CONT PLAN blocks only translational degrees of freedom, while CONT SPLA blocks both translational and rotational degrees. Therefore, the two directives are identical for nodes of continuum elements, which do not possess rotational degrees of freedom. However, for structural nodes (with rotations), CONT PLAN represents a hinge while CONT SPLA represents a symmetry line or plane (the relevant rotations are automatically blocked in that case).

### 8.9.2 SPHERICAL/CIRCULAR RESTRAINT

#### Object:

The specified nodes lay on (a) sphere(s) of given center. In 2D, the sphere reduces to a circle.

Compatibility: COUP, LIAI

#### Syntax:

```
"SPHE"  | [  "CX" x  "CY" y  < "CZ" z  >  ;
           "CR" r  "CZ" z                      ;
           "CENT" /LECT1/                      ] |    /LECT2/
```

x y

Coordinates of the center of the sphere (2-D).

x y z

Coordinates of the center of the sphere (3-D).

r z

Coordinates of the center of the sphere (Axisymmetric).

"CENT" /LECT1/

Node at the center of the sphere. Points should be sufficiently far from the sphere center so as to define the radial direction with sufficient accuracy.

/LECT2/

Nodes located at the surface of the sphere.

#### Comments:

This constraint only ensures that, at each time step, the displacement increment of the specified nodes be tangent to the (current) sphere. For finite displacement increments, therefore, the nodes will only approximately remain on the initial spherical surface. It is not necessary that the nodes initially belong to the same sphere.

This directive blocks only translational degrees of freedom.

In case variable coefficients are specified (via the OPTI CONT VARI option), remember to dimension adequately by the DIME VCON directive). Each sphere/circle requires 3 coefficients.



### 8.9.3 CYLINDRICAL RESTRAINT

#### Object:

The specified nodes lay on (a) circular cylinder(s) of given axis. At each step the displacement increment along the axial direction is free, while that in the plane orthogonal to the axis is tangent to a circle.

The instruction only applies to a 3-D analysis. In 2D, the SPHE directive described in the previous Section may be used to obtain a circular restraint.

Compatibility: COUP, LIAI

#### Syntax:

```
"CYLI"  |[ "P1X" x1  "P1Y" y1  "P1Z" z1 ; "POI1" /LECT1/ ;
           "P2X" x2  "P2Y" y2  "P2Z" z2 ; "POI2" /LECT2/ ]| /LECT3/
```

x1 y1 z1

Coordinates of a point of the cylinder axis.

"POI1" /LECT1/

Node at the first point of the cylinder axis.

x2 y2 z2

Coordinates of another point of the axis.

"POI2" /LECT2/

Node at the other point of the cylinder axis.

/LECT3/

Nodes concerned. Points should be sufficiently far from the cylinder axis so as to define the radial direction with sufficient accuracy.

#### Comments:

This constraint only ensures that, at each time step, the displacement increment of the specified nodes be tangent to the cylinder (current or initial, depending on OPTI CONT option). For finite displacement increments, therefore, the nodes will only approximately remain on the initial cylindrical surface. It is not necessary that the nodes initially belong to the same cylinder.

This directive blocks only translational degrees of freedom.

In case variable coefficients are specified (via the OPTI CONT VARI option), remember to dimension adequately by the DIME VCON directive). Each cylinder requires 6 coefficients.

### 8.9.4 CONICAL RESTRAINT

#### Object:

The specified nodes lay on (a) cone(s) of given axis.

The instruction only applies to a 3-D analysis.

Compatibility: COUP, LIAI

#### Syntax:

```
"CONE"  $[ "SX" x1  "SY" y1  "SZ" z1 ; "APEX" /LECT1/ ]$
          $[ "PX" x2  "PY" y2  "PZ" z2 ; "POIN" /LECT2/ ]$ /LECT3/
```

x1 y1 z1

Coordinates of the apex of the cone.

"APEX" /LECT1/

Node at the cone apex.

x2 y2 z2

Coordinates of a point on the cone axis different from the apex.

"POIN" /LECT2/

Node along the cone axis different from the apex.

/LECT3/

Nodes concerned. Points should be sufficiently far from the cone axis so as to define the radial direction with sufficient accuracy.

#### Comments:

This constraint only ensures that, at each time step, the displacement increment of the specified nodes be tangent to the cone (current or initial, depending on OPTI CONT option). For finite displacement increments, therefore, the nodes will only approximately remain on the initial conical surface. It is not necessary that the nodes initially belong to the same cone.

This directive blocks only translational degrees of freedom.

In case variable coefficients are specified (via the OPTI CONT VARI option), remember to dimension adequately by the DIME VCON directive). Each cone requires 6 coefficients.

### 8.9.5 TOROIDAL RESTRAINT

**Object:**

The specified nodes lay on (a) torus(es) of given axis and center.

This instruction only applies to a 3-D analysis.

Compatibility: COUP, LIAI

**Syntax:**

```
"TORE"  | [ "P1X" x1  "P1Y" y1  "P1Z" z1 ; "POI1" /LECT1/ ;  
            "P2X" x2  "P2Y" y2  "P2Z" z2 ; "POI2" /LECT2/ ;  
            "P3X" x3  "P3Y" y3  "P3Z" z3 ; "CENT" /LECT3/ ] | /LECT4/
```

x1 y1 z1

Coordinates of a point on the torus (circular) axis.

"POI1" /LECT1/

First node on the torus (circular) axis.

x2 y2 z2

Coordinates of another point on the circular axis.

"POI2" /LECT2/

Second node on the torus (circular) axis.

x3 y3 z3

Coordinates of the center of the torus.

"CENT" /LECT3/

Node at the torus center.

/LECT4/

Nodes concerned.

**Comments:**

This constraint only ensures that, at each time step, the displacement increment of the specified nodes be tangent to the torus (current or initial, depending on OPTI CONT option). For finite displacement increments, therefore, the nodes will only approximately remain on the initial torical surface. It is not necessary that the nodes initially belong to the same torus.

This directive blocks only translational degrees of freedom.

In case variable coefficients are specified (via the OPTI CONT VARI option), remember to dimension adequately by the DIME VCON directive). Each torus requires 9 coefficients.

### 8.9.6 PLANE/LINE OF SYMMETRY RESTRAINT

#### Object:

The specified nodes lay (and remain) on (a) plane(s) of given normal vector, that defines the symmetry. In 2D, the plane reduces to a straight line.

Compatibility: COUP, LIAI

#### Syntax:

```
"SPLA"  | [  "NX" x  "NY" y  < "NZ" z  >  ;
            "NR" r  "NZ" z                      ;
            "POIN" /LECT1/                      ;
            "AUTO"                               ] | /LECT2/
```

x y

Components of the normal vector (2-D).

x y z

Components of the normal vector (3-D).

r z

Components of the normal vector (Axisymmetric).

POIN /LECT1/

Must specify a node belonging to the mesh, either via its index or via its CASTEM 2000 name. The coordinates of this node are taken as the components of the normal.

AUTO

The components of the normal are determined automatically from the position of the nodes listed in /LECT2/. Therefore, in this case, the nodes contained in the following /LECT2/ list must lie on the same line (in 2D) or plane (in 3D). In the 3D case, of course, the listed nodes must define a plane (not be along the same line).

/LECT2/

Numbers of the nodes concerned.

#### Comments:

It is not necessary that the nodes initially belong to the same plane (except in the AUTO case).

The difference between this directive and the CONT PLAN directive (see above) is that CONT PLAN blocks only translational degrees of freedom, while CONT SPLA blocks both translational and rotational degrees. Therefore, the two directives are identical for nodes of continuum elements, which do not possess rotational degrees of freedom. However, for structural nodes (with rotations), CONT PLAN represents a hinge while CONT SPLA represents a symmetry line or plane (the relevant rotations are automatically blocked in that case).

Remember to dimension adequately with 'SYME' (see page A.80).

When AUTO is used, the search for enough non-coincident nodes, among those contained in LECT2, so as to define a line in 2D or a plane in 3D is affected by a tolerance. In case of necessity, this tolerance may be set by OPTI TOLC, see page H.40.

## 8.10 IMPOSED CIRCULAR SHAPE

### Object:

The displacements of the specified nodes are constrained to be in the radial direction with respect to a point (center) and to have the same modulus. If the nodes lie initially on the same circle, they remain on a circle, whose radius may vary with time.

The instruction is available for a 2-D and 3-D analysis.

For an Eulerian computation (no mesh displacements), the fluid velocities are radial and of the same modulus.

Compatibility: COUP, LIAI

### Syntax:

```
"RADI" "SPHE" "CENT" /LECTURE/  
      ...  "CONT" /LECTURE/
```

```
"CENT" /LECTURE/
```

Number of the node at the center of the circle.

```
"CONT" /LECTURE/
```

Numbers of the nodes concerned.

### Comments:

The instruction is used to avoid instabilities e.g. when a gas bubble collapses after an initial expansion.

For  $n$  points ( $n > \text{or} = 2$ ), EUROPLEXUS writes  $(\text{idim} * n - 1)$  relations.

## 8.11 RELATIONS

### Object:

Several displacement (or velocity) components are linked by constant coefficients during the whole computation.

Compatibility: COUP, LIAI

### Syntax:

```
"RELA"  ngroup*(
          ... nrel  nterm*( coef  comp $[ nuneu ipas ;
                                /LECTURE/ <SHIFT s> ]$ )
          ... "EGAL" /LECDDL/ /LECTURE/
        )
```

#### ngroup

Number of relation sets.

#### nrel

Number of relations to be generated in a set.

#### nterm

Number of terms in a relation of the set.

#### coef

Coefficient of a term.

#### comp

Displacement component of the node "nuneu" involved in the relation.

#### nuneu

Number of the node concerned.

#### ipas

Increment on the number of the node "nuneu" in order to get the next relation of the set.

#### LECTURE

List of concerned nodes.

#### SHIFT s

Force circular permutation of the list (see example below). The increment in traversing the list circularly is indicated by the **s** quantity (normally 1).



/LECDDL/

Reading procedure of the numbers of the blocked nodes.

EGAL

Indicates the equality along the component /LECDDL/ of the motion of the nodes defined by the following /LECT/.

### Comments:

Each displacement will be specified by the number of the node (nuneu) and its component (icomp). Formula of the relation:

$$0 = \text{coef}(1)*U(1) + \text{coef}(2)*U(2) + \dots + \text{coef}(k)*U(k)$$

There are two ways to define a set of relations. The first is to give the node number **nuneu** and the step **ipas**. The second is to use the procedure /LECTURE/, which allows to use object names created by GIBI. In this latter case, one passes from one relation to the next one in a set by taking the next node in the procedure /LECTURE/ associated with each term. In this case, there must be exactly **nrel** nodes in each one of the lists specified via the /LECT/ procedures (assuming that the optional **SHIF** keyword has not been specified).

The optional **SHIF** keyword can be used to force a circular permutation of the list. In this case, the number of nodes in the lists need not be the same. For example, assume that one wants to impose the same displacement along  $z$  (i.e. global direction 3) to all nodes of an object named "face1". Then the command would be:

```
RELA 1 0 2
      1. 3 LECT face1 TERM
      -1 3 LECT face1 TERM SHIFT 1
```

Note that in this case the number of relations **nrel** can be set to 0 because the code computes it automatically.

### Example:

```
RELA 5 2 2 1. 1 288 1 -1. 1 6 1
      1 3 3.4 2 287 0 -1. 2 5 0 1. 1 5 0
      3 2 0.5 3 115 7 -1. 3 9 5
      5 2 1. 3 LECT toto TERM
          -1. 3 LECT tata TERM
      EGAL 13 LECT 228 321 842 TERM
```

There are five groups. The first group has 2 relations of 2 terms, the second 1 relation of 3 terms, the third 3 relations of 2 terms, the fourth 5 relations of 2 terms and the last one 2 relations of 2 terms.

In the first group, d.o.f 1 of node 288 has been linked to d.o.f 1 of node 6 (first relation), then d.o.f 1 of node 289 to d.o.f 1 of node 7 (second relation). In fact **ipas**=1 for the two terms.

In the second group, d.o.f. 2 of node 287 has been linked to d.o.f. 2 of node 5 and to d.o.f. 1 of the same node 5. There is just one relation since  $ipas = 0$  for the three terms.

On the contrary, in the third group,  $ipas=7$  for the first term and 5 for the second. Therefore, d.o.f 3 of node 115 has to be linked to d.o.f 3 of node 9 (first relation of the group), then d.o.f 3 of node 122 to d.o.f 3 of node 14 (2nd relation), and finally d.o.f 3 of node 129 to d.o.f 3 to node 19 (3rd relation).

In the fourth group, there are 5 relations between the d.o.f. 3 of the nodes belonging to objects 'toto' and 'tata' taken in the order in which they appear.

In the fifth group, there are 4 equalities between the d.o.f. 1 and 3 of nodes 228, 321 and 842.

$$\begin{array}{ll} U_x(228) = U_x(321) & ; \quad U_z(228) = U_z(321) \\ U_x(321) = U_x(842) & ; \quad U_z(321) = U_z(842) \end{array}$$

## 8.12 ARMATURES

### Object:

In calculations of structures made of reinforced concrete, this directive allows to link the displacements of the nodes belonging to continuum-like elements made of concrete, with those of bar-like elements made of steel.

Decoupled treatment of this link consists in introducing a penalty spring between the reference position of a steel node in the corresponding concrete element and its actual position.

The default spring's stiffness is obtained from concrete element through the formula:

$$k = GL$$

with:

$G$  : bulk modulus of concrete element's material,

$L$  : radius of a sphere whose volume equals concrete element's volume.

Compatibility: COUP, DECO, LIAI

### Syntax:

```
"ARMA" <"FAST" <$HGRI hgri ; NMAX nmax ; DELE dele$> <DGRI> >
      <"TSTA" ista> <"CSTI" cstif>
      "BETO" /LECTURE/
      "FERR" /LECTURE/
      <"EACH" n>
      <"FROT" "TABL" npts*(si,taui)>
```

#### FAST

Perform fast search of the couplings instead of the default brute-force search.

#### HGRI

Specifies the size of the fast search grid cell. Each cell has the same size in all spatial directions and is aligned with the global axes.

#### NMAX

Specifies the maximum number of cells along one of the global axes.

#### DELE

Specifies the size of the fast search grid cell as a multiple of the length of the largest coupled concrete (continuum-like) element (see **BETO** below). Element “diameters” are computed only along each global spatial direction and the maximum is taken. For example, by setting **DELE 2** the size of the cell is two times the length of the largest coupled concrete element. By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DELE** are specified, the code takes **DELE 1.01**.

**DGRI**

Dump out the grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed.

**ista**

**DECO** only: flag for stability control over the penalty spring’s stability (default: 1, see comment below)..

**cstif**

**DECO** only: coefficient multiplying default stiffness of the penalty spring (default: 1.0).

**BETO**

Introduces the list of the concrete (continuum-like) elements.

**EACH**

Optional keyword that allows to link only one every **n** of the reinforcement bar elements to be defined next (**FERR**.) Only the first node of each selected element is linked. By default **n=1**, that is each element (and every bar node) is linked. This keyword can be used to tentatively decrease the number of links without un-refining the reinforcement mesh, if there are too many links. However, since the numbering of the elements is arbitrary, linking (the first node of) every **n**-th bar element might be inappropriate (depending on the mesh) so it is advised to visually check the result of taking **n > 1** (e.g. by activating **OPTI LINK VISU**).

**FERR**

Introduces the list of the steel or reinforcement (bar-like) elements. Each node of these elements will be linked to the matching concrete element (if any).

**FROT**

Optional keyword creating a steel-concrete connection in which the tangential efforts depend on the slip between reinforcement and concrete. It requires the definition of the curve  $\tau(s)$  which will be entered in a classical way: first, the number of **npts** points describing the curve and then the list of **npts** couples  $s - \tau(s)$ . The first point must correspond to beginning of the plasticity, thus defining the rigidity per unit area  $k_b$  and the elastic limit of the interface  $\tau_{elas}$ .

This option is available with coupled and decoupled links.

**Remarks:**

If nodes (belonging to concrete elements or steel elements) involved in an “ARMA” connection are also subject to other types of links, such as blockages for example, it’s necessary that all links are treated in the same way : coupled or decoupled.

**Comments:**

The concrete elements must be of continuum-like type 2D or 3D. The steel or reinforcement elements must be of bar-like type (e.g. BR3D in 3D or BARR in 2D).

If **ista** equals 2, stiffness of the penalty spring is limited so that the associated stability time-step is not smaller than the reference concrete element's one.

Up to two named node groups are automatically created for each occurrence of the **ARMA** directive. The first group has the form **\_ARMAnnnn** where **nnnn** is a 4-digit number (0001, 0002 etc.) indicating the index of the current **ARMA** directive in the input file. This group contains the reinforcement bar nodes which were actually linked to the concrete. The second group has the form **\_NARMnnnn** and contains the reinforcement bar nodes which were *not* linked, despite having been declared in the **FERR /LECTURE/** directive, because a matching concrete element could not be found. Obviously, this second group is created only if there are any non-matching nodes. The groups can be visualized in the OpenGL graphical module for direct visual inspection of which nodes were actually glued (or not glued.)

## 8.13 CROSSING

### Object:

This directive allows the user to automatically interconnect crossing longitudinal and transverse reinforcement bars (rebars) to constitute rebar cages (carcasses) frequently used to reinforce the concrete structures. In the real life, the rebars in the cages are usually connected either by welding, tying steel wire, or with mechanical connections.

Links are created between the elements of longitudinal reinforcement and the nodes of transverse reinforcing steel (stirrups).

Both coupled and decoupled links are implemented: the coupled links are treated using Lagrange multipliers method whereas the decoupled ones are solved by the penalty method.

Only the translation degrees of freedom are concerned by this link.

Nodes of the transverse rebars may eventually coincide with the nodes of the longitudinal rebars but should remain distinct.

Compatibility: COUP, DECO

### Syntax:

```
"CROS" < "TSTA" ista> < "CSTI" cstif >  
      "LONG" /LECTURE/  
      "TRAN" /LECTURE/
```

**ista**

DECO only: flag for stability control over the penalty spring's stability (default: 1, see comment below)..

**cstif**

DECO only: coefficient multiplying default stiffness of the penalty spring (default: 1.0).

**LONG**

Introduces the list of longitudinal rebar elements.

**TRAN**

Introduces the list of transverse rebar elements.

### Comments:

Both the longitudinal and transverse rebars must be modelled as POUT elements.

If *ista* equals 2, stiffness of the penalty spring is limited so that the associated stability time-step is not smaller than the reference longitudinal steel element's one.

## 8.14 ACBE: REBAR(FEM)-CONCRETE(DEM) LINK

### Object:

This directive allows creating nonlinear links between a steel reinforcement bar (rebar) modelled as FEM beam and plain concrete modelled by the discrete element method (DEM). Only one link between a given discrete element and a finite element beam may be created. Each link contains a normal and a tangential component.

A decoupled (DECO) link model is implemented only.

### Syntax:

```
"ACBE" < "TSTA" ista> < "CSTI" cstif >
      "BETO" "COEF" c1 /LECTURE/
      "ARMA" "COEF" c2 /LECTURE/
      "YOUN" youn "TN" tn "CN" cn "ADUN" adun
< "AMOR" amor >
      "FTAN" "NUMF" nf
```

**ista**

Flag for stability control over the penalty link's stability (default: 1, see comment below).

**cstif**

Coefficient multiplying default stiffness of the link (default: 1.).

**BETO**

Introduces the list of discrete elements concerned.

**c1**

Coefficient defining the interaction range for the discrete elements.

**ARMA**

Introduces the list of rebar (POUT type) elements concerned.

**c2**

Coefficient defining the interaction range for the beam elements.

**youn**

Young's modulus used to calculate the normal stiffness of the link ( $kn=youn*S/L$ ).

**tn**

Maximum normal tensile strength (perpendicular to the rebar direction)

**cn**

Maximum normal compression strength (perpendicular to the rebar direction)

**adun**

Softening coefficient (ratio between elastic and softening slopes  $>0$ )

**amor**

Reduced damping coefficient applied on steel-concrete links if needed

**nf**

Number of the function describing the tangential behaviour of the link.

**Comments:**

If **ista** equals 2, stiffness of the penalty spring is limited so that the associated stability time-step is not smaller than the reference concrete element's one.



## 8.15 LCAB: LINK BETWEEN PRESTRESSING CABLES AND CONCRETE

### Object:

This directive allows creating kinematic relations between the nodes of prestressing cables modelled as FE bars (BR3D) and the nodes of plain concrete modelled through a FE thick shell model (T3GS,Q4GS). First, a projection of cable nodes onto concrete mesh is done to determine cable node to concrete element correspondence, and then, node by node relations are written.

Adherent and sliding conditions are implemented. In the adherent case, the cable-concrete links act in 3 space directions. For the purely sliding case, only 2 relations per cable node are written (in the normal directions to the cable), the relation in the tangential direction is not written. Those relations are updated at each time step in order to account for the cable direction change when sliding in a curvilinear case.

It is possible to add friction to the sliding case. To do this, RNFR friction-spring elements must be added to the model and declared in the directive GEOM (just after BR3D cables elements). RNFR elements are of SEG2 type with the nodes that coincide geometrically at the beginning of the calculation: the first RNFR node corresponds to cable' node (except for the cables' extremity nodes) and the second one is automatically connected to a concrete point having the same global coordinates and being related to concrete element nodes by ADHE-type relations. The RNFR elements are detected automatically when using LINK LCAB FROT option, thus there is no need to declare them in the present directive. It should be noted that RNFR elements cannot be used outside the LCAB directive.

For theoretical description see [\[949\]](#).

This link model is implemented in coupled (LINK COUP) version only.

### Syntax:

```
"LCAB" $[ "ADHE" ; "GLIS" ; "FROT" ]$
      "BETC" /LECTURE/
      "CABL" /LECTURE/
```

#### ADHE

Fully adherent option.

#### GLIS

Perfectly sliding option.

#### FROT

Sliding with friction.

#### BETC

Introduces the list of concrete shell (T3GS,Q4GS type) elements concerned.

#### CABL

Introduces the list of cable (BR3D type) elements concerned.

### Comments:

This directive may be repeated as many times as necessary.

## 8.16 ARLQ: SHELL-3D MESH COUPLING WITHIN ARLEQUIN FRAMEWORK

### Object:

This directive allows linking in a continuous way two (or more) sub-domains used to model a slender structure, some sub-domains modeled as thick shells (Reissner-Mindlin kinematics) and others represented through a 3D hexahedra-type mesh. The shell and 3D sub-domains are glued in a weak sense within an overlapping zone using the Arlequin method ([950]).

The following hypotheses must be satisfied:

- only Q4GS quadrangular shell and CUBE,CUB8 hexahedron elements can be used,
- the shell and 3D meshes in the gluing zone must be hierarchic.

This link model is implemented in coupled (LINK COUP) version only.

### Syntax:

```
"ARLQ" < "ROTA" >  
      "COQU" /LECTURE/  
      "VOLU" /LECTURE/
```

#### ROTA

Optional key-word to be used to add the gluing of rotations.

#### COQU

Introduces the list of shell (Q4GS) elements of the gluing zone.

#### VOLU

Introduces the list of 3D (CUBE,CUB8) elements of the gluing zone.

### Comments:

This directive may be repeated as many times as necessary.

## 8.17 IMPOSED MOTIONS

### Object:

These instructions define imposed motions (displacements, velocities or accelerations) depending on time, for different degrees of freedom.

Compatibility: COUP, DECO, LIAI

### Syntax:

```
"DEPL" ( /LECDDL/ coef "FONC" ifonc < "TLIM" tlim > /LECTURE/ )  
"VITE" ( /LECDDL/ coef "FONC" ifonc < "TLIM" tlim > /LECTURE/ )  
"ACCE" ( /LECDDL/ coef "FONC" ifonc < "TLIM" tlim > /LECTURE/ )
```

/LECDDL/

Reading procedure of the different d.o.fs concerned.

coef

Multiplying factor of the values of the function.

ifonc

Number of the function to be used.

tlim

Time after which the imposed motion is deactivated.

/LECTURE/

Reading procedure of the numbers of the nodes concerned.

### Comments:

The function to be used will be defined by means of the principal instruction "FONC" which enables the user to choose a tabulated function (linear interpolation between the points), or a function programmed by the user by means of a subroutine.

At a time  $t$ , the imposed motion is :  $\text{coef} \cdot F(t)$ . In this case, only one function is to be defined, if the motions vary only in amplitude.

If the same d.o.f is submitted to several motions, EUROPLEXUS only takes into account the motion which has been defined first.

Motion can be imposed temporarily using the "TLIM" keyword.

## 8.18 CONNECTIONS BETWEEN SHELLS AND SOLID ELEMENTS

### Object:

The purpose is to link together the degrees of freedom at the boundary of two parts of the structure. One part is meshed with shells, the other with solid elements. This link is available only for two-dimensional computations (plane or axisymmetric).

Compatibility: COUP, LIAI

### Syntax:

```
"COQM" ngroup*( nco nma /LECTURE/ )
```

#### ngroup

Number of groups of shell and solid element connections.

#### nco

Number of the shell node linked to a solid element.

#### nma

Number of the solid node linked to a shell.

#### /LECTURE/

Reading procedure to input the other solid element nodes which are connected (no compulsory order).

### Comments:

```

      M1  x-----
          I
(shell)  I
-----x  x  M0 (nma)      R(i) : distance M0-Mi
          /  I            D(i) : normal displacement
      nco  x  Mi            of Mi
          I  (solid element)
      Mp  x-----

```

A "shell-solid element" relation is represented by the following p+2 equations.

2 equations for the displacements:

$$U(1,nco) = U(1,nma)$$

$$U(2,nco) = U(2,nma)$$

p equations for the rotations of p other solid element nodes:

$$U(3,nco) = R(i) * D(i)$$

## 8.19 INTERFACES

### Object:

This directive allows to define an interface between two lines or two surfaces. Link relations are created so that the velocity field is continuous through the interface. In the case of non-matching meshes on both sides of the interface, continuity is imposed in a weak sense.

This directive is very similar to the INTERFACE directive used in the sub-domain calculation framework.

Compatibility: COUP

### Syntax:

```
"INTERFACE"    |[ "COMP"      ;  
                  "MORTAR"    ;  
                  "OPTIMAL" ]| <"TOLE" tole> ...  
  
                ... |[ "SIDE"      ;  
                      "SCOARSE" ;  
                      "SFINE"   ]| /LECTURE/ ...  
  
                ... |[ "SIDE"      ;  
                      "SCOARSE" ;  
                      "SFINE"   ]| /LECTURE/
```

#### "COMP"

Keyword declaring an interface with matching meshes.

#### "MORTAR"

Keyword declaring an interface with non-matching meshes, treated by the mortar method (see comment below).

#### "OPTIMAL"

Keyword declaring an interface with non-matching meshes, treated by the optimal method.

#### tole

Tolerance given to find matching nodes (default=1.E-3).

#### "SIDE"

Keyword introducing the support of both sides of the interface for COMP and OPTI cases (see note below).

#### "SCOARSE"

Keyword introducing the support of the side of the interface with coarse mesh in the **MORTAR** case (see note below).

"**SFINE**"

Keyword introducing the support of the side of the interface with fine mesh in the **MORTAR** case (see note below).

### Comments:

There **MUST NOT** be any coincident nodes between the two sides of an interface.

When using the mortar method, the side of the interface whose mesh is used to discretize Lagrange multipliers has to be specified. It is the mesh introduced by the **SFINE** keyword, the other mesh being introduced by the **SCOARSE** keyword.

When using interfaces with non-matching meshes, so-called **CLxx** elements (see pages INT.90 and INT.100) have to be affected to meshes of both sides of the interface. These elements **must** be given the “phantom” material (**MATE FANT**) with density equal to zero.

The treatment of non-matching meshes with 3D solid elements is restricted to hierarchical meshes. In this case, the mortar method and the optimal method are identical, and a mortar interface has to be declared.

The **mortar** method may be used with any element types in 2D, but only with shell element types in 3D. When using the **mortar** method with linear interfaces (2-noded element sides), there must be at least one geometrical point that has the same coordinates, within the tolerance **tole** defined above, in the two facing meshes. This is necessary because the interface model uses the point’s coordinates internally in order to define a reference frame on the interface.

## 8.20 FLUID-STRUCTURE COUPLING (FLST)

### Object:

This directive allows to specify the coupling between a fluid and a structure modelled by topologically independent meshes.

Compatibility: COUP

### Syntax:

```
FLST <SLID> STRU /LECTS/ FLUI /LECTF/  
      <DGRI> $[ HGRI hgri ; NMAX nmax ; DELE dele ]$
```

/LECTS/

List of structural **nodes** concerned. They must be declared as Lagrangian.

/LECTF/

List of fluid **elements** concerned.

DGRI

Dump out the grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed.

HGRI

Specifies the size of the grid cell. Each cell has the same size in all spatial directions and is aligned with the global axes.

NMAX

Specifies the maximum number of cells along one of the global axes.

DELE

Specifies the size of the grid cell as a multiple of the diameter of the largest coupled fluid element. Element “diameters” are computed only along each global spatial direction and the maximum is taken. For example, by setting DELE 4 the size of the cell is four times the diameter of the largest coupled fluid element. By default, i.e. if neither HGRI, nor NMAX, nor DELE are specified, the code takes DELE 3.



## 8.21 FLUID-STRUCTURE COUPLING (FLSR)

### Object:

This directive allows to specify a “strong” coupling (LINK COUP) between a fluid and a structure modelled by topologically independent meshes. It is similar to FLSW (see page D.555) but uses a strong approach (constraint on velocity imposed by Lagrange multipliers) rather than a weak approach (direct application of the fluid pressure).

The present FLSR directive is (primarily) intended for use with Finite Elements (FE) modeling of the fluid. It can also be used with a node-centred Finite Volume (NCFV) discretization of the fluid domain, but *not* with cell-centred Finite Volumes (VFCC) in the fluid (use FLSW instead).

The fluid mesh may be either fully general (unstructured) or regular (structured), as specified by the STFL directive described on page C.68. In the latter case, the search operations are faster.

The FSI coupling is realized between structural points (ultimately, structural nodes) on one side, and *fluid entities* on the other side. The fluid entities are fluid nodes in this case, since in the FE method the velocities are discretized at the nodes of the FE.

The FLSR command should be typically utilized as a COUP type of link (strong coupling). Note, however, that a decoupled master/slave approach version of the FLSR algorithm also exists and can be activated by specifying FLSR within the (LINK DECO) directive. However, this part of the implementation is still incomplete and experimental. Note also that LINK COUP FLSR and LINK DECO FLSR are mutually exclusive: only one of such directives (at most) can be used in any given calculation.

As indicated by the brackets in the syntax, the STRU data block can be repeated at will, in order to define one or more FSI interaction (structure) zones, each with its own set of parameters. The STRU /LECTS/ keyword *must* be the first one of each zone being defined.

Compatibility: COUP, DECO.

### Syntax:

```

FLSR  |[ FLUI /LECTF/ ; STFL ]|
      $[ HGRI hgri ; NMAX nmax ; DELE dele ]$
      <DGRI>
      <BFLU bflu $[/LECTURE/]$>
      <DVOF dvof $[/LECTURE/]$>

      ( STRU /LECTS/
        $[ R r ; GAMM gamm ; PHIS phis ; GAMI gami ]$
        <FSCP fscp>
        <ADAP LMAX lmax <SCAL scal> >
      )

```

### Basic fluid-related parameters

## FLUI

The fluid mesh to be coupled with the structure is fully general (unstructured). The concerned elements are specified next.

## /LECTF/

List of fluid **elements** concerned. The fluid mesh is unstructured.

## STFL

The fluid mesh to be coupled with the structure is regular (structured). The concerned **elements** need not be specified. In fact, they are simply the elements generated by the STFL directive described on page C.68, which must in this case have been specified previously in the input file.

## Fast search of coupled fluid entities

The next three keywords (HGRI, NMAX or DELE) are used to set the size of the spatial grid used for the *fast search* of fluid nodes contained within the influence domain of the structure. Fast search speeds up the calculation and is *absolutely essential* in medium and even more in large size simulations. For this reason, fast search is *always active* in the present FSI model. Note that this may be unlike other types of search in EPX. For example, in the pinball contact model (PINB) fast search of pinballs contact is *not* active by default (an option has to be activated).

By default, i.e. if neither HGRI, nor NMAX, nor DELE are specified, the code takes DELE 1.01.

A (regular) spatial grid is built up and used for the fast search. The fluid nodes contained in a cell are tested for inclusion in the structural influence subdomains contained either in the same cell or in a direct neighbor cell (there are up to 8 such cells in 2D, up to 26 cells in 3D). The cell grid can be optionally dumped out on the listing by the DGRI keyword.

For the calculation to be as fast as possible, the fast search grid must have the minimum size ensuring correctness of results, i.e. such that a (barely) sufficient number of interacting entities is detected, and thus no spurious fluid passage occurs across the structure. If  $h_F$  denotes the size of the fluid mesh and  $h_S$  the size of the structure mesh, then the grid size  $h_G$  must be:

$$h_G = \phi \cdot \max(h_F, h_S) \quad (43)$$

where  $\phi > 1$  is a safety factor. A value  $\phi = 1.01$  should be sufficient. Since a single grid is used for the search over the whole computational domain,  $h_F$  and  $h_S$  in the above expression must be the *maximum* sizes of the fluid and structural elements which are susceptible of interacting, i.e. which belong to the /LECTF/ and LECTS/ sets defined above, respectively.

In calculations *without adaptivity* one has normally  $h_F < h_S$  for accuracy reasons (especially if shells are used to discretize the structure), so that the grid size is (normally) dictated by the largest coupled structural element. For the case of adaptive calculations, see the Remarks at the end of this manual page.

## HGRI

Specifies the size of the fast search grid cell. Each cell has the same size in all spatial directions and is aligned with the global axes.

## NMAX

Specifies the maximum number of cells along one of the global axes.

## DELE

Specifies the size of the fast search grid cell as a multiple of the length of the largest coupled **structural** element. Element “diameters” are computed only along each global spatial direction and the maximum is taken. For example, by setting **DELE** 2 the size of the cell is two times the length of the largest coupled structural element. By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DELE** are specified, the code takes **DELE** 1.01.

#### DGRI

Dump out the grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed.

### Flux blocking parameters

Next come some additional parameters for the blocking of fluxes.

#### BFLU

Type of treatment of numerical fluxes (density and energy, but **not** momentum) in fluid models, when used in conjunction with the present **FLSR** directive. This directive applies to fluids modelled by multicomponent node-centered Finite Volumes (NCFV, i.e. MCxx ‘elements’) and to any fluid modelled by CEA Finite Elements.

In both cases, the value 0 (default) indicates that fluxes are freely computed.

For MCxx elements, the value 1 indicates that fluxes are blocked between two fluid nodes (or points) which are both within the influence domain of the structure. For CEA fluid elements, it indicates that the fluxes are blocked through a face for which one node at least is within the influence domain of the structure.

For MCxx elements, the value 2 indicates that fluxes are blocked between two fluid nodes (or points) of which at least one lies within the influence domain of the structure. For CEA fluid elements, it indicates that the fluxes are blocked through a face for which all nodes are within the influence domain of the structure.

With CEA fluid elements, the defined treatment can be restricted to the influence domain of a subset of structure elements using the **LECTURE** procedure.

### Combination of FLSR with the VOFIRE algorithm

The next keyword allows to fine-tune the interaction between **FLSR** FSI algorithm and the **VOFIRE** anti-dissipation algorithm (**VOFI**) in case of need.

#### DVOF

Locally deactivate **VOFIRE** anti-dissipative scheme when used in conjunction with the present **FLSR** directive.

The value 0 (default) indicates that no deactivation occurs.

The value 1 indicates that **VOFIRE** anti-dissipation is deactivated for fluid elements, for which one node at least is within the influence domain of the structure.

The value 2 indicates that **VOFIRE** anti-dissipation is deactivated for fluid elements, for which all nodes are within the influence domain of the structure.

The defined deactivation can be restricted to the influence domain of a subset of structure elements using the **LECTURE** procedure.

## Structural influence domain(s)

The following block of data, introduced by the keyword **STRU**, can be repeated any number of times (but it must be specified at least once) to define one or more FSI zones, each with different interaction parameters. For each such zone:

**STRU**

Introduces the structure mesh to be coupled with the fluid. The concerned elements are specified next.

**/LECTS/**

List of structural **elements** concerned. All their nodes must be declared as Lagrangian.

The next four keywords (**R**, **GAMM**, **PHIS** or **GAMI**) are used to set the **size** (thickness) of the **structural influence domain** surrounding the structure elements defined above by **/LECTS/**. All fluid nodes contained within this influence domain will be coupled to the structure.

Therefore, the correct size of the influence domain is related to the size of the fluid mesh in the vicinity of the embedded structure. On one hand, if the influence domain is too thin, then some interactions between the structure and the fluid entities might be overlooked, thus resulting in spurious passage of fluid across the structure (*leakage*). On the other hand, if the influence domain is too thick, too much fluid will be interacting with the structure (excessive added mass effect). The optimal value is then the minimum value which ensures structure tightness (no leakage).

By default, i.e. if neither **R** nor **GAMM** nor **PHIS** nor **GAMI** are specified, the code performs an automatic determination of influence spheres at each coupled structural node by using the default value of **GAMM** ( $\gamma = 1.01$ ). For the choice of **R**, **GAMM**, **PHIS** or **GAMI** in adaptive calculations see the **ADAP** keyword below and the comments at the end of this page.

**R**

Prescribed (fixed) radius  $R$  of influence spheres at each coupled structural node. In the special, but frequent, case of a uniform structured fluid mesh (uniform square or cube elements) it is suggested to take  $R$  slightly larger than the semi-diagonal of a fluid element. This means that, for a 2D uniform square fluid mesh of side  $L_\Phi$  one should take  $R = 0.71L_\Phi$  while for a 3D uniform cube fluid mesh of side  $L_\Phi$  one should take  $R = 0.87L_\Phi$ .

**GAMM**

Coefficient  $\gamma$  for the automatic determination of influence spheres at each coupled structural node, based on the size of the enclosing fluid element (which must thus be found by the code by means of a fast search algorithm, see the remarks at the end of this manual page). The sphere radius is  $R = \gamma R_F = \gamma \delta L_\Phi$  where  $L_\Phi$  is the local length (size) of the fluid mesh,  $\delta$  is a coefficient related to the space dimension  $d$  of the problem ( $\delta = \frac{\sqrt{d}}{2}$ , i.e. about 0.71 in 2D and about 0.87 in 3D calculations). The quantity indicated as  $R_F$  above is the “natural” size of the sphere radius, i.e. the radius of a sphere (circle in 2D) which exactly encompasses all nodes of a regular element (regular cube in 3D or regular quadrilateral in 2D). By default it is  $\gamma = 1.01$ . This value should ensure “tightness” of the structure, at least for a regular mesh. By increasing the value, tightness is safer but the amount of fluid “attached” to the structure also increases. By decreasing the value, some local spurious passage of fluid across a solid structure might occur.

**PHIS**

Coefficient  $\phi_s$  for the automatic determination of influence spheres at each coupled structural node. The sphere radius is equal to  $\phi_s$  times the minimum structural element length at the concerned node. By default it is  $\phi_s = 0.3$ . This option should be rarely used. It is advisable to use **GAMM** instead.

#### GAMI

Same as **GAMM** but radius is computed only at the initial step, that is, the radius is *not* updated during the calculation. This may be convenient (to save some CPU) in calculations with Eulerian fluid mesh (that never changes) and without adaptivity.

### Additional coupling parameters

Next come some additional parameters for the the type of coupling.

#### FSCP

Type of coupling between fluid nodes and corresponding structural points, when used in conjunction with the present **FLSR** directive. The value 0 (default) indicates that coupling occurs only in the direction normal to the structure. The value 1 indicates that coupling occurs along all spatial directions.

### FSI-driven adaptivity

Finally, there are some optional keywords related to automatic (FSI-driven) adaptivity of the fluid mesh near the structure.

#### ADAP

Activates mesh adaptivity for automatic refinement and un-refinement of the fluid mesh specified by **/LECTF/** in the vicinity of the structure specified by **/LECTS/**. Note that this type of mesh adaptivity is at the moment incompatible with other types of adaptivity such as those activated by the **WAVE** or **INDI** directives.

#### LMAX

Introduces **lmax**, the desired maximum adaptive refinement level  $L^{\max}$  of the fluid mesh in the vicinity of the structure. This value should be greater than 1, since level 1 is attributed to the base mesh (no refinement). Each level corresponds to a halving of the mesh size with respect to the immediately previous level.

#### SCAL

Introduces **scal** ( $s$ ), an optional scaling factor to be used in the determination of fluid elements to be refined. By default **scal** is equal to 1. When scaling the structural influence domain by successive powers of two in order to identify, at each refinement level, the fluid elements to be refined or un-refined, the code finally multiplies the result by this factor. Using a value of  $s$  greater than one, e.g. 1.5 or 2, correspondingly enlarges the zone of fluid mesh around the structure which is refined and this may result in a smoother mesh transition (for example, as an alternative to the option **OPTI ADAP RCON**). Note, however, that  $s$  has no influence on the size of the structural influence domain used for the final search of fluid entities (fluid nodes or fluid cell interfaces) interacting with the structure. This search is always done by the smallest influence domain  $R_{L_{\max}} = R_1/2^{L_{\max}-1}$ , i.e. *without* taking into account the  $s$  factor.

In **FSI adaptive calculations**, the size of the structural influence domain specified in input by **R**, **GAMM** or **PHIS** (**GAMI** is not appropriate with adaptivity) is related to the *base* (i.e. the coarsest) fluid mesh size, not to the refined one (for the user's convenience) and is then scaled automatically by the code whenever necessary, up to the maximum chosen refinement value given by the **ADAP LMAX** keyword. Therefore, in order to try out different adaptive refinement levels in the vicinity of the structure the user needs only to change **LMAX** in the input directive (all other parameters **R** etc. remain the same).

In **FSI adaptive calculations**, that is when the **FLSR ADAP LMAX** optional keyword has been specified, one is **certain** that the fluid mesh in the vicinity of the structure will be constantly refined to the maximum level (minimum size) specified for the fluid (**LMAX**), given by:

$$h_F^{\text{refined}} = h_F^{\text{base}} / 2^{L_{\text{max}}-1} \quad (44)$$

For this reason, in the equation (43) for the determination of the grid size **HGRI** ( $h_G$ ) one can use  $h_F^{\text{refined}}$  instead of the base fluid mesh  $h_F^{\text{base}} = h_F$ , obtaining thus:

$$h_G = \phi \cdot \max(h_F^{\text{refined}}, h_S) \quad (45)$$

One should make sure to use (45) instead of (43) since it is likely to be  $h_F^{\text{refined}} < h_S$ , while it is typically  $h_F > h_S$ , so this may lead to important savings of CPU time.

### Remarks:

In case of automatic determination of influence spheres based on the **GAMM** keyword in conjunction with an **unstructured** fluid grid, a fast search over the coupled fluid elements is needed in addition to the normal fast search over the coupled structural elements. Scope of this second search is to determine, for each structural node, which is the fluid element currently containing the node. For this purpose, the code uses a fast search algorithm by means of the same parameters (**DGRI**, **HGRI**, **NMAX**, **DELE**) specified above for the search over structural elements. Note, however, that as concerns this second search if **DELE** is specified it refers to the size of the fluid element rather than to the size of the structural element. However, if a **structured** fluid grid is specified, then no additional search is needed because the containing fluid element can be detected directly.

Make sure you consult the additional **options** related to the functioning of the **FLSR** model in pages H.155 and H.160.

### References

The **FLSR** model was first described in report [250]. A short description of the model is also given in reference [244]. Finally, reference [294] describes the use of **FLSR** in conjunction with Node-Centered Finite Volumes (NCFV).

## 8.22 FLUID/IMMERSED STRUCTURE INTERACTION (FLSX)

### Object:

This directive allows to specify the coupling between a fluid and a structure modelled by topologically independent meshes. The fluid can either be in finite elements or in finite volumes. This directive is more recent than FLSR and FLSW directive and is expected to provide more accurate solutions. At this time, the fluid mesh must have all of its nodes declared as EULE (i.e. Eulerian, **not** ALE). This directive only works in a 3D space. At this time, only thin structures are supported (the mesh of the structure must only contain shell elements).

Compatibility: COUP (for FE fluid meshes), DECO (for FV fluid meshes).

### Syntax:

```
FLSX      STRU /LECTS/
          FLUI /LECTF/
          <LSPC lspc>
          <LORD lord>
```

#### STRU

Introduces the structure mesh coupled with the fluid. This mesh can be meshed totally independently from the fluid mesh. The concerned elements are specified next.

#### /LECTS/

List of the **structural elements** concerned. All their nodes must be declared Lagrangian in the GRIL directive.

#### FLUI

Introduces the fluid mesh coupled with the structure. The concerned elements are specified next.

#### /LECTF/

List of the **fluid elements** concerned. All their nodes must be declared Eulerian in the GRIL directive (note that nodes not explicitly mentioned in the GRIL directive are Eulerian by default).

The two options LSCP and LORD are effective only if the fluid is treated with finite elements. In this case, the FLSX coupling results from a finite-element discretization of the constraint:

$$\int_{\Gamma} [(v_s(x) - v_f(x)) \cdot n_s(x)] \lambda(x) dx, \forall \lambda \in L^2(\Gamma)$$

where  $\Gamma$  is the mid-surface of the structure,  $v_f$  and  $v_s$  the fluid and structure velocity respectively,  $n_s$  the normal to  $\Gamma$ , and  $\lambda$  a test function. The two options LSCP and LORD specify the finite-element space on which the Lagrange multiplier related to the test function  $\lambda$  is discretized. If

LSCP is set to 0 (default), the Lagrange multiplier is defined on a finite element space generated from the mesh of the structure. If LSCP is set to 1, the Lagrange multiplier is defined on the “restriction” to the structure mesh at each time step of a finite-element space generated from the mesh of the fluid. LORD, which can be 0 or 1 (default), is the polynomial degree of interpolation used to generate the finite-element space for the Lagrange multiplier.

**LSCP****Finite-elements fluid only.**

Specifies which mesh is used to generate a finite-element space for the lagrange multiplier enforcing the coupling. If set to 0 (default), the mesh of the structure is used. If set to 1, the mesh of the fluid is used.

**LORD****Finite-elements fluid only.**

Specifies the polynomial degree of interpolation used to generate the finite-element space for the Lagrange multiplier. If set to 0, constant-value elements are used. If set to 1 (default), linear elements are used.



## 8.23 FLUID-STRUCTURE INTERACTIONS

### Object:

This is aimed at linking together the degrees of freedom at the boundary of 2 parts of the structure:

- one part meshed with shells or solid elements;
- one solid element part meshed with a fluid material.

This possibility exists for two- and three-dimensional computations.

This connection may be expressed in two ways:

- By using specific fluid-structure elements FS2D or FS3D: directive "FS";

Without using fluid-structure elements: directive "FSA".

Compatibility: COUP, LIAI

### Comments:

The first directive is available for a Lagrangian or an ALE calculation. Instead, the second is only valid for ALE problems.

The elements FS2D and FS3D behave like incompressible fluids. In order to avoid spurious effects (related to the flow along the boundary), the thickness of this boundary zone must be as small as possible, and possibly 0.

The "FS" directive is described in the next page, while for the "FSA" directive please consult page D.260.

**8.23.1 FLUID-STRUCTURE CONNECTION (FS)****Object:**

The contact between the fluid and the structure is modelled by elements FS2D and FS3D. This directive is available for Lagrangian and ALE calculations.

Compatibility: COUP, LIAI

**Syntax:**

"FS" /LECTURE/

/LECTURE/

Numbers of the FS2D, FS3D or FS3T elements composing the boundary.

**Comments:**

The FS2D, FS3D and FS3T elements are in fact incompressible fluids. In order to avoid any parasitic effects (due to a potential flow along the boundary), the thickness of that boundary zone has to be as small as possible and even equal to zero.

It is strongly advised to use the new directives "FSA" and "FSR".

## 8.24 UNILATERAL RESTRAINT [OBSOLETE]

### Foreword:

This directive is now obsolete, use the IMPACT directive described below (page D.170) which allows to compute at the same time the shock parameters (impulse, reaction, ...).

## 8.25 IMPACT

### Object:

As for unilateral restraints, certain nodes of the structure must remain in the same half-space. However, the boundary is linked to the position of a material point and can be mobile. Impacts are possible in 2D or 3D.

The method of Lagrange multipliers may be activated by adding the keyword "LAGC" in the problem type (see page A.30). This method allows to couple the calculation of contact forces with the permanent connections (relations, boundary conditions, ...). It also allows to take into account the form of a projectile 'nose' in the case of a non-deformable projectile.

Compatibility: COUP, DECO, LIAI

### Syntax:

```
"IMPACT" "DDL" iddl "COTE" alpha      ...

... < "NEZ" |[ "HEMI" "RAYO" rayon1      ;
              "PLAT" "LARG" larg1 < "LONG" long1 > ;
              "CONE" "LARG" larg2 < "ANGL" beta > ;
              "CYLI" "RAYO" rayon2 ]| >

... <"FROT" "MUST" must "MUDY" muddy "GAMM" gamm >

... "PROJ" /LECTURE/ |[ "CIBL" /LECTURE/ ; "CIBD" /LECTURE/]|
```

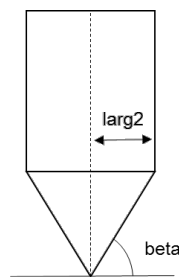


Figure 5: Conical impactor

iddl

Component concerned. Indicates the first direction.

alpha

Enables one to choose between the 2 half-spaces separated by the plane of equation  $x = x_0$ . It must be an integer. Typically, one uses either +1 or -1.

rayon1

Radius of a projectile with hemispherical nose.

larg1

Width of a projectile with rectangular flat nose. It is in the second direction obtained by circular permutation of the Euclidean frame.

long1

Only in 3D, length of the rectangular projectile nose. It is in the third direction (obtained by circular permutation).

larg2

Width of a projectile with a conical nose.

beta

Half-angle of the cone (in degrees).

ray2

Radius of the projectile with cylindrical nose.

FR0T

Introduces the (optional) declaration of friction characteristics.

must

Static friction coefficient  $\mu_s$ : ( $0 < \mu_s < 1$ ).

mudy

High-velocity (dynamic) friction coefficient  $\mu_d$ : ( $0 < \mu_d < \mu_s < 1$ ).

gamm

Coefficient  $\gamma$  of the friction law. This law is similar to the one used for sliding lines and sliding surfaces (see page D.180). The friction coefficient  $\mu$  varies from  $\mu_s$  to  $\mu_d$  as the relative tangential velocity  $V_r$  of the two bodies increases. The passage is governed by the exponential decay law:  $\mu = \mu_d + (\mu_s - \mu_d)e^{(-\gamma V_r)}$ .

PROJ

Introduces the number of the material point located on the boundary (if relevant, it is the tip of the projectile “nose”).

CIBL

Introduces the numbers of the nodes which are submitted to the impact.

CIBD

Used for ELDI elements only to take into account the elements' radius for contact detection. Introduces the numbers of the discrete elements which are submitted to the impact.

**Comments:**

The boundary plane is perpendicular to one of the axes of the general coordinate system. This axis is defined by the component `idd1`, just as for unilateral contacts.

The half-space admissible for a point  $M$  (of the target) of coordinate  $x$  is such that, if  $x_0$  is the abscissa of the material point:

$$\alpha(x - x_0) \geq 0$$

These impacts are available in 2-D or 3-D.

It is suggested to displace the projectile in such a way that the impact occurs after at least one time step.

Do not forget to dimension the keywords "IMPA" and "PSIM" correctly, see (page A.80).

The "NEZ" directive is available **only** with the LAGC option. When it is present, only those "CIBLE" nodes which are in contact with the geometric boundary thus defined, will be considered.

#### **Without the option "LAGC":**

It should be noted that the nodes which undergo shocks may not be connected by other imposed relations (LIAISONS).

The shock between the material point (projectile) and the point(s) of the target is treated elastically. The energy and impulse will therefore be conserved during the impact. This requires a modification of the time step so that the impact instant coincides with the beginning of a time step.

This effect introduces a small error in the work of forces during the impact, of the order:  $dW = F * v * dt$ . It is therefore advisable to shorten the time step in order to obtain better energy conservation.

These recommendation are irrelevant with the option "LAGC".

## 8.26 GAPS ("JEUX")

### Object :

This is an impact between the (uncoupled) nodes along the direction defined by the user. This directive is available in 2D and 3D. In 3D, the gap must be defined also along a direction normal to the first one.

Compatibility: LIAI

### Syntax:

In 2D:

```
"JEUX"      "AXE1" a1x a1y      "JEU1"  jeu1
...         "NOE1" /LECTURE/
...         "NOE2" /LECTURE/
```

In 3D:

```
"JEUX"      "AXE1" a1x a1y a1z  "AXE2" a2x a2y a2z
...         "JEU1"  jeu1      "JEU2" jeu2a jeu2b
...         "NOE1"  /LECTURE/
...         "NOE2"  /LECTURE/
```

a1x,a1y (,a1z)

Components of the first vector of the local reference.

a2x,a2y,a2z

Components of the second vector of the local reference (3D only).

jeu1

Gap along the direction of the first vector.

jeu2a,jeu2b

Gap along the direction of the second vector (3D only).

NOE1

Announces the first group of nodes.

NOE2

Announces the second group of nodes.

**Comments:**

To each node P1 belonging to the first group, is associated one node P2 of the second group, and reciprocally. In 3D, in the local frame of origin P1, defined by vectors AXE1 and AXE2, the impact occurs when: 1) the abscissa of point P2 is less than jeu1; 2) and the ordinate of point P2 lies between jeu2a and jeu2b.

In 2D, in the local reference of origin P1 of which AXE1 is the first axis, the impact occurs when the abscissa of point P2 is less than jeu1.

The direction defined by vector(s) AXE1 (or AXE2) does not change during the calculation.

Do not forget to dimension, by keyword "NBJEUX" (see page A.80).



## 8.27 BUTEE: LIMITED DISPLACEMENT

### Object :

This directive allows limiting the displacement of a node along the direction defined by the user, when a prescribed distance is covered. This directive can be repeated several times. It is operational in MPI.

Compatibility: LINK COUP

### Syntax:

```
"BUTE"    "DIRE" vx vy vz  
          "DMAX" dmax  
          /LECTURE/
```

**vx,vy,vz**

Components of the vector indicating the direction of blocking.

**dmax**

Distance above which the displacement is blocked in the prescribed direction.

**/LECTURE/**

Name of the node for which the displacement is to be limited.

## 8.28 SLIDING LINES AND SLIDING SURFACES

### Object:

This directive defines one or more couple(s) of mutually sliding lines (2D) or sliding surfaces (3D). In 3D the “master” and “slave” objects may be composed of continuum elements or shells. In 2D they are composed by an ordered series of nodes.

In 3D, a self-contact (or auto-contact) model is available. A self-contacting surface is both master and slave at the same time.

When a master surface is composed by shells (CMAI) and the contact should be checked only on one side of the shell, it is necessary to define this side (independently from the inherent orientation of the shell elements) by declaring the “external” or the “internal” half-space of each shell element by using the position of a node or of a point in space (see EXTE or INTE below).

However, this method is not always practical, especially in the presence of shells with large curvature, complex shapes, junctions etc. If contact should be possible from *both* sides of a shell, then *two* sliding surfaces, one using the DIRE (direction of the geometric normal to the shell) and the other using the REVE keywords (reverse direction) can be defined, as an alternative to using the EXTE or INTE method. Note that in this case (i.e., with DIRE or REVE), the keyword SELF is mandatory.

Compatibility: COUP, DECO, LIAI

LIAI only: the method of Lagrange multipliers may be activated by adding the keyword LAGC in the problem type (see page A.30, Section 4.4). This method allows to couple the computation of contact forces with the permanent connections (relations, boundary conditions, ...).

Penalty method is only available in 3D with DECO keyword. If not activated, the same uncoupled algorithm is used as with LIAI and LAGC deactivated (see above and comments below).

### Syntax:

```
"GLIS" nglis * ( < "FROT" "MUST" must "MUDY" muddy "GAMM" gamm >
  < "PENA" > < "PFSI" rfac > < "PGAP" rgap >
  < "SELF" > < "ELIM" < "UPDT" /CTIM/ > >
  < "COHE" >
  |[ |[ "MAIT" <"NODE"> /LECT/ ;
    "CMAI" /LECT/ |[ "EXTE" $ /LECT/ ; "POIN" x y z $ ;
                        "INTE" $ /LECT/ ; "POIN" x y z $ ;
                        "DIRE" ;
                        "REVE"
    ]|
  ]|
  !|[ |[ "ESCL" ; "CESC" ; "PESC" <"PESA"> ]| /LECT/ ]! ;
  "AUTO" "FACE" iface /LECT/ ;
  "DECO" <"SYME"> <"DBLE"> <"ELIM" <"UPDT" /CTIM/>>
  /LECT/
]|
```

```

      < "COPT" 1 >
    )

```

**nglis**

Number of couples of lines (or surfaces) (MAIT, ESCL) or AUTO.

**FROT**

Introduces the (optional) declaration of friction characteristics.

**must**

Static friction coefficient  $\mu_s$ : ( $0 < \mu_s < 1$ ).

**mudy**

High-velocity (dynamic) friction coefficient  $\mu_d$ : ( $0 < \mu_d < \mu_s < 1$ ).

**gamm**

Coefficient  $\gamma$  of the friction law. This law is similar to the one used for the IMPA sirective (see page D.170). The friction coefficient  $\mu$  varies from  $\mu_s$  to  $\mu_d$  as the relative tangential velocity  $V_r$  of the two bodies increases. The passage is governed by the exponential decay law:  $\mu = \mu_d + (\mu_s - \mu_d)e^{(-\gamma V_r)}$ .

**PENA**

DECO only: toggles the use of penalty method to compute contact forces (see comment below).

**rfac**

DECO only: scale factor for the automatically computed contact stiffness (see comment below).

**rgap**

Value of the gap between master surface and slave nodes (3D only). Default value is zero in the case of solid master elements and half of shell's thickness in the case of shell master elements.

**SELF**

Toggles self-contact treatment for shells (3D only, see comment below).

**ELIM**

Toggles the elimination of the not visible faces (for each slave node) of the master elements in the detection of the contact (sometimes necessary option when the 3D master elements contain only a single element in the thickness). This elimination is only once realized during the initialization of the problem. The keyword UPDT can be used to introduce a regular update of this sorting. Indeed, the relative movement of the various elements of structure can require to redefine for each slave node the not visible faces of the master elements.

**COHE**

This keyword enters within the framework of the automatic separation of elements (See page A.63, Section 4.6.1, keyword DECO). Compulsory to update the list of the sliding surfaces and the list of the slave nodes.

MAIT /LECTURE/

Numbers of the master nodes (in 2D) or numbers of the continuum elements (in 3D).

MAIT NODE /LECTURE/

In 3D continuum elements: numbers of the master nodes belonging to the sliding surface.

CMAI /LECTURE/

Numbers of the master elements of the structure meshed by shell elements (“COQUES”) (in 3D).

EXTE /LECT/

A node defining (located in) the half-space external to the shell.

EXTE POIN x y z

Coordinates of a point defining (located in) the half-space external to the shell.

INTE /LECTURE/

A node defining (located in) the half-space internal to the shell.

INTE POIN x y z

Coordinates of a point defining (located in) the half-space internal to the shell.

DIRE /LECTURE/

The half-space external to the shell is in the direction of the geometric normal to the element. The DIRE and REVE keywords are typically used to define *two* sliding surfaces for the *same* set of master shells, one using the geometric normal direction and the other using the opposite direction. The SELF keyword is mandatory in the definition of a sliding surface using either DIRE or REVE.

REVE /LECTURE/

The half-space external to the shell is in the opposite direction of the geometric normal to the element. The DIRE and REVE keywords are typically used to define *two* sliding surfaces for the *same* set of master shells, one using the geometric normal direction and the other using the opposite direction. The SELF keyword is mandatory in the definition of a sliding surface using either DIRE or REVE.

ESCL /LECTURE/

Numbers of the slave nodes (in 2D) or numbers of the slave elements (in 3D).

CESC /LECTURE/

Numbers of the slave elements (in 3D) if the structure is meshed by shell elements (COQUES).

PESC /LECTURE/

3D only. Numbers of the slave nodes.

#### PESA

Optional keyword to choose the strategy used to define the slave nodes updating strategy in adaptivity. It is only related to cases with adaptivity in the contacting parts, where the (base) slave nodes are defined directly via the **PESC** keyword (not via a set of slave elements). A list of base slave elements must be built internally because, unlike for elements, there is no notion of descendent node(s) of a base node in adaptivity. By default a base element is a base slave element if at least one of its nodes is a base slave node in the list **PESC /LECT/** given by the user. When the **PESA** keyword is specified after **PESC**, then a base element is a base slave element if all of its nodes are base slave nodes in the list **PESC PESA /LECT/** given by the user.

#### AUTO FACE iface

Number (local) of the face of the elements submitted to self-contact.

#### DECO /LECTURE/

Numbers of the continuum elements (only CUB8). The keyword "DECO" can appear only once and must be positioned in last position. The keyword "DECO" enters within the framework of the automatic separation of elements (See page A.63, Section 4.6.1, keyword **DECO**). This directive allows to treat self-contact and contacts between various pieces of wood coming from the automatic separation of elements. The creation of a new node causes the activation of a coupled link. In this link the slave node is the new node of the wood and the master faces are the free faces of the wood except the faces containing the new node and the faces already eliminated by the previous links. During the initialization **NPTMAX\_DECO** (See page A.63, Section 4.6.1, keyword **DECO**) links are created but not activated (option by default). For each new node a coupled links is activated. More explanations can be found in [899].

#### SYME

With this keyword the master faces of each new activated coupled link are the free faces of the wood except the faces containing the new node.

#### DBLE

With this keyword (**2\*NPTMAX\_DECO**) links (See page A.63, Section 4.6.1, keyword **DECO**) are created during the initialization but not activated. For each new node two coupled links are activated. In the first link the slave node is the new node. In the second link the slave node is the initial node.

#### ELIM

With this keyword the not visible faces (for the new slave node) of the master elements are eliminated in the detection of the contact. This elimination is only once realized during the initialization of the new activated link (option by default). The keyword **UPDT** can be used to introduce a regular update of this sorting. Indeed, the relative movement of the various elements of structure can require to redefine for the slave node the not visible faces of the master elements.

#### COPT 1

If **COPT 1** is activated, the thickness of the slave surface is taken into account.

**Comments:****Sliding lines (2D):**

The order of the nodes determines the orientation of the contour and defines in that way the inner side of the two domains after a rotation of +90 degrees.

Example. We consider DOM1 and DOM2, in contact on P1P2 and P6P5.

```

P4-----P3
|   DOM1   |
P1-----P2
P5-----P6
|   DOM2   |
P8-----P7

GLISS      1
MAIT       LECT P1P2 TERM
ESCL       LECT P6P5 TERM

```

where P1P2 (respectively P6P5) is an oriented line the first node of which is P1 (respectively P6) and the last one is P2 (respectively P5).

The *slave* nodes must be located just at or above the boundary of the region defined by the line of the *master* nodes.

Without the **LAGC** option, it is preferable that the two lines have similar mesh densities. But, if the *master* domain presents a high convexity, it is better to have *master* segments which are a bit longer than the *slave* segments in front of them. This is aimed at minimizing the interpenetration of the two domains. It is suggested to fix a point of the master line (blocked material point) to avoid the interpenetration of the two domains.

With the **LAGC** option, the recommendations of the preceding paragraph are irrelevant.

When the “erosion” algorithm is activated (See page A.30, Section 4.4, keyword **FAIL**), the sliding surfaces are updated at each time step by eliminating the failed elements.

**Sliding surfaces (3D):**

For the sliding surfaces, the master and slave entities are defined by the elements composing them (possibly these are GIBI objects). If continuum elements are used, then it is not necessary to define the “inner” or “outer” sides of such entities. However, when shell elements are used, it is mandatory to define the outer half-space of the shell structure by choosing a node, whose coordinates are used to identify the half-space. This method is not always practical, e.g. with shells of complicated shape, of large curvature, with junctions, etc. Alternatively, *two* sliding surfaces may be declared, one with the **DIRE** and the other with the **REVE** keyword, using the *same* shell object as master surface, and by specifying the **SELF** keyword for each of these.

The **SELF** keyword is necessary if contact on both sides of a shell should be considered with the same set of slaves (typically, the nodes of the shell itself). It prevents contact from being

detected if a slave node has penetrated the shell by an amount greater than the gap (see **PGAP** keyword). Without this, each slave node would initially be found in contact with either side of the shell.

With the **MAIT NODE** option, the master entity must be defined by the nodes belonging to the sliding surface.

It is not admitted to define master objects (nor slave objects) formed by continuum and shell elements at the same time.

### **Remark (2D and 3D):**

The sliding nodes may not be linked by other imposed relations (**LIAISONS**), except in the case where the treatment of sliding lines (or surfaces) is done by the method of Lagrange multipliers (option **LAGC**).

### **Self-contact:**

This directive indicates that the surface formed by the set of faces defined by the user may be in contact. This surface is both master and slave at the same time.

### **Penalty method:**

When using the penalty method to compute contact forces, contact stiffness is computed automatically from the stiffness of master elements using the following formulae :

$$k = r_{\text{fac}} \frac{GS^2}{V}$$

in the case of solid master elements, with :

$G$  : bulk modulus of master element's material,

$S$  : area of contacting face,

$V$  : volume of master element.

$$k = r_{\text{fac}} \frac{GS}{L}$$

in the case of shell master elements, with :

$G$  : bulk modulus of master element's material,

$S$  : area of master element,

$L$  : maximum length of master element's edges.

## 8.29 METHOD OF PARTICLES AND FORCES

### Object:

This directive activates the (internal) interactions occurring among a set of particles (“billes”) representing a soft body, according to the so-called Method of Particles and Forces (MPEF).

Optionally, the user may require that the particles also interact with some structure, composed either of continuum or of shell elements. By omitting the definition of the structure, interaction occurs only between the particles themselves.

Compatibility: LIAI

### Syntax:

```
"MPEF" nbpef * (      "BILL" /LECTURE/
                      < $[ "STRU" /LECTURE/                ;
                      "COQU" /LECTURE/ "EXTE" /LECTURE/ ]$ > )
```

#### nbpef

Number of pairs (BILL, STRU) or (BILL, COQU) or of single BILL groups (in case no structure is specified).

"BILL" /LECTURE/

Numbers of the nodes belonging to the BILL elements.

"STRU" /LECTURE/

Numbers of the “master” elements of the solid structure, meshed by continuum elements.

"COQU" /LECTURE/

Numbers of the “master” elements of the solid structure, meshed by shell elements.

"EXTE" /LECTURE/

Number of a node defining the “external” half-space to the solid shell structure. Here “external” means simply the side of the shell onto which the particles are going to impact. If the impact is going to occur on both sides of the shell, or if the shape of the shell is such that there exists no point that can be used to uniquely define the “external” space (think e.g. of a spherical structure impacted from outside the sphere), the present algorithm is inappropriate.

### Comments:



If the structure domain presents a large convexity, it is advisable that the faces of the elements of the structure be longer than the diameter of the neighbouring particles. This in order to minimize the interpenetration between the two domains.

The data relative to this method are similar to those of the SPH method, described on page D.187.

### 8.30 SMOOTHED PARTICLE HYDRODYNAMICS METHOD (SPH)

#### Object:

This directive activates the (internal) interactions occurring among a set of particles (“billes”) representing a soft body, according to the so-called Smoothed-Particle Hydrodynamics (SPH) method.

Optionally, the user may require that the SPH particles also interact with some structure, composed either of continuum or of shell elements. By omitting the definition of the structure, interaction occurs only between the particles themselves.

If a structure is specified in the directive described below, the interaction between the particles and the structure is treated by an algorithm of the ‘sliding surfaces’ type. The code offers also other alternative (more general and more robust) methods to describe the interaction between the SPH particles and a structure, see comments below.

Compatibility: LIAI

#### Syntax:

```
"SPHY" nbpef * (      "BILL" /LECTURE/
                    < $[ "STRU" /LECTURE/          ;
                    "COQU" /LECTURE/ "EXTE" /LECTURE/ ]$ > )
```

#### nbpef

Number of pairs (BILL, STRU) or (BILL, COQU) or of single BILL groups (in case no structure is specified).

"BILL" /LECTURE/

Numbers of the nodes belonging to the BILL elements.

"STRU" /LECTURE/

Numbers of the “master” elements of the solid structure, meshed by continuum elements.

"COQU" /LECTURE/

Numbers of the “master” elements of the solid structure, meshed by shell elements.

"EXTE" /LECTURE/

Number of a node defining the “external” half-space to the solid shell structure. Here “external” means simply the side of the shell onto which the particles are going to impact. If the impact is going to occur on both sides of the shell, or if the shape of the shell is such that there exists no point that can be used to uniquely define the “external” space (think e.g. of a spherical structure impacted from outside the sphere), the present algorithm is inappropriate. In such cases the user may utilize one of the alternative contact models mentioned in the comments below.

**Comments:**

The data relative to this method are similar to those of the PEF method, described on page D.185.

If a structure is specified in the directive described above, the interaction between the particles and the structure is treated by an algorithm of the ‘sliding surfaces’ type. Use is made of Lagrange multipliers, but by default the imposed contact constraints are **decoupled** from other constraints imposed e.g. via LIAI or LINK directives.

To force coupling of the SPH contact constraints with other constraints, add the optional **LACG** keyword in the calculation type, see Page A.30.

Sometimes contact detection and enforcement with the above method may be imprecise. In such cases, alternative (more general and robust) contact models can be used.

One possibility is to use the sliding surface algorithm via the LIAI or LINK directives. To this end, specify only the **BILL** keyword in the **SPHY** directive. Then, use the LIAI or LINK directive with the **GLIS** keyword to detect the contact. The LINK form of the directive can use either Lagrange multipliers (strong formulation, either in a coupled or in a decoupled manner, **COUP** or **DECO**), or a penalty method (weak formulation). On the “master” side, the **MAIT** keyword is used to specify a structure made of continuum elements, or the **CMAI** keyword for a shell structure. The SPH particles are then listed after the **PESC** keyword, that treats each particle as a “slave” material point. See page D.180 for further details.

Another possibility is to use the pinball method. To this end, specify only the **BILL** keyword in the **SPHY** directive. Then, embed pinballs both in the SPH particles themselves (with a diameter equal to the diameter of the particles) and in the impacted structure. The LIAI or LINK forms of the pinball contact method can be applied. The latter can use either Lagrange multipliers (strong formulation, either in a coupled or in a decoupled manner, **COUP** or **DECO**), or a penalty method (weak formulation). See page D.480 for further details.

## 8.31 CONNECTING FINITE AND DISCRETE ELEMENT MODELS

### Object:

This directive defines a bridging (recovering) zone allowing to couple a set of discrete elements (ELDI) with a 3D finite element model (meshed with the CUB8 element only) or a shell model (Q4GS elements only).

The coupling equations are solved using Lagrange multipliers. To simplify, a diagonal matrix is used. It's possible to couple discrete elements by using the complete matrix through the LINK procedure.

Compatibility: COUP

### Syntax:

```
"EDEF" nbcoup
      nbcoup*( "NCOU"  ncouches
               "ELDI"  /LECTURE/
               "FRON"  /LECTURE/ )
```

#### nbcoup

Number of combined finite/discrete zones to connect.

#### "NCOU"

Number of finite element range defining the combined finite/discrete element zone.

#### "ELDI" /LECTURE/

List of the discrete elements concerned to research in the combined finite/discrete element zone.

#### "FRON" /LECTURE/

List of nodes forming the border of the finite elements mesh in the bridging finite/discrete element zone.

## 8.32 BIFURCATION CONNECTION

### Object:

Writes the relations that ensure the conservation of mass flow rate for the fluid, and the equality of mechanical d.o.f.s if necessary (case of 1D coupled fluid calculation).

This directive may only be used in 1D.

Compatibility: COUP, LIAI

### Syntax:

```
"BIFU"    < LIBR >    /LECTURE/
```

```
/LECTURE/
```

Numbers of the BIFU elements for which the conservation of flow rate must be imposed.

### Comments:

This directive may only be used in 1D, coupled or not, and for the junctions between the following elements:

		TUBE		TUYA		POUT	
-----							
TUBE		yes		yes		-	
TUYA		yes		yes		no	
POUT		-		no		no	

In the case of a bifurcation linking an element TUBE with an element TUYA, there may be only two nodes connected in the directive /LECTURE/ (no multiple branches).

In the case of bifurcations (even multiple) between TUYA, the 6 mechanical d.o.f.s are connected (continuity of the beam). In order to avoid these connections (for example in the case of a 'soufflet'), add the keyword "LIBR". On the contrary, between a TUBE and a TUYA the 6 mechanical d.o.f.s are left free, and the keyword "LIBR" is irrelevant.

### Outputs:

The various components of the ECR table are as follows:

ECR(1) : density (all materials)

ECR(6) : internal energy (water)

### 8.33 ADHESION CONNECTION

**Object:**

This link can describe adhesion connections between a slave surface and a master surface. The contact can be opened, when a failure criterion is reached. From this point on, the link can not sustain any tension forces. But it can still react to compression forces, when the gap is closed. The model is extended to MPI calculations.

Compatibility: COUP

**Syntax:**

```
"ADHE"  "AUTO"  auto <"CRIT" "TENS" tens >
        "LIST"  "MAST" /LECTURE/
        "SLAV"  /LECTURE/
```

**auto**

Maximum distance for the automatic search.

**tens**

Maximum tensile strength for the definition of the failure.

**/LECTURE/**

Objects which should be taken into account for the automatic search.

**8.34 TUBM CONNECTION (3D-1D JUNCTION)****Object:**

Write the relations ensuring the conservation of mass flow rate for the fluid (Eulerian formulation)

Compatibility: COUP, LIAI

**Syntax:**

"TUBM" /LECTURE/

/LECTURE/

Numbers of the "TUBM" elements (or names of the GIBI objects), which form the junctions.

**8.35 TUYM CONNECTION (3D-1D JUNCTION)****Object:**

Write the relations ensuring the conservation of mass flow rate for the fluid (moving meshes).

Compatibility: COUP, LIAI

**Syntax:**

"TUYM" /LECTURE/

/LECTURE/

Numbers of the "TUYM" elements (or names of the GIBI objects), which form the junctions.



### 8.36 TUYA CONNECTION (3D-1D JUNCTION)

**Object:**

Automatically writes the mechanical relations among d.o.f.s of a pipeline meshed by beams and a pipeline meshed by thin shells.

Compatibility: COUP, LIAI

**Syntax:**

```
"TUYAU"  "CENTRE" /LECTURE/  
          "LISTE"  /LECTURE/
```

"CENT" /LECTURE/

Number of the node (or name of the object) corresponding to the extremity of the pipeline meshed by beams.

"LIST" /LECTURE/

Number(s) of the node(s) (or name of the object) corresponding to the circle, extremity of the pipeline meshed by thin shells.

**Comments:**

This directive automatically writes the relations between the displacements of nodes belonging to the shells and the beam. All rotations are supposed to be equal.

All nodes involved by the link (including the CENT node) must have 6 dofs, since the imposed relations involve also the rotations. Therefore, the CENT node cannot be simply represented by a (stand-alone) PMAT, which has only 3 dofs. In such a case, it is sufficient to attach a (dummy) beam or shell element to the CENT.

### 8.37 RIGID BODY (SOLIDE INDEFORMABLE) CEA Implementation

#### Object:

This directive defines the sub-structures that will be considered as rigid bodies.

It also allows to impose the inertia tensor of the solid, or to leave EUROPLEXUS compute it starting from the mesh, or from a composition of simple homogeneous solids.

The directive may be used in two ways:

- The solid is meshed, i.e. its form is represented by a set of elements
- The solid is not meshed, i.e. one imposes that a small number of points be rigidly connected.

Compatibility: COUP, LIAI

#### Syntax:

```
"SOLI" nsol*( ... )
```

1st case - Solid meshed by elements:

```
"ELEM" /LECTURE/ "PLIE" /LECTURE/ ...
$ < "COMP" ncomp*( "INER" ... ) > $
$ < "INER" ... > $
```

2nd case - Rigidly connected points:

```
"POIN" /LECTURE/
```

**nsol**

Number of non-deformable solids.

```
"ELEM" /LECTURE/
```

Numbers of the elements composing the solid.

```
"PLIE" /LECTURE/
```

Numbers of the points of the solid to be conserved because they take part in a connection (linked points).

**ncomp**

Number of homogeneous simple solids whose combination allows to compute the inertia tensor. In the case that  $ncomp = 1$ , this parameter is optional.

"INER"

This option allows to introduce the parameters of inertia of a solid, that will replace the ones computed starting from the mesh and the initial materials.

"POIN" /LECTURE/

Numbers of the points rigidly linked (case of the non-meshed solid).

### Comments:

A sub-structure described like a non-deformable solid will reduce to a system of four material points. The calculation will be done with these points, and the solid will then be reconstructed to be visualized.

The linked points (participation in a connection) will be conserved in the calculation in order to be able to write down the connection relation.

The other points are not conserved in the calculation. However, they are used for the calculation of the inertia tensor. Care must then be taken that the discretization be sufficient, else the parameters related to the solid will be imprecise, and the computation will be affected by errors.

The "INER" directive is optional. It imposes to the solid inertia values coming from an external calculation. If it is absent, EUROPLEXUS computes inertias from the mesh.

If you impose the inertia tensor via "INER", you may limit the mesh to the minimum indispensable, by directly connecting the linked points (wireframe mesh). In any case, at least ONE free point per solid is necessary, i.e. two linked points will be connected by at least two beam elements.

In the case of complex solids, it is interesting to mesh them finely from the beginning, and to let EUROPLEXUS compute the inertia tensor. The option VERIF is enough for that. For the real dynamic calculation, a coarser mesh (wireframe) will be sufficient, and one will then impose the previously found inertia tensor, by means of the INER directive.

In the case that the solid is not meshed (directive "POIN"), all points of the list will be considered linked. The inertia tensor data is then useless.

Dimension sufficiently by means of directives "SOLI", "PLIE" and "PLIB" (page A.80).

**8.37.1 INERTIA****Object:**

This directive allows to specify inertia parameters for a non-deformable solid. It also allows to compute the inertia tensor starting from simple shapes.

Compatibility: COUP, LIAI

**Syntax:**

```
"COMP"  ncomp*(    "INNER"    "MASS"  m  ...
                  ... <"XG"  xg>    <"YG"  yg>    <"ZG"  zg>    ...
                  ... <"IXX"  ixx>   <"IYY"  iyy>   <"IZZ"  izz>   ...
                  ... <"IXY"  ixy>   <"IXZ"  ixz>   <"IYZ"  iyz> )
```

**"COMP"**

Announces that the inertia tensor is composed by assembly of simple tensors.

**ncomp**

Number of inertia tensors to be read in order to compute the inertia tensor of the composite solid.

**"INNER"**

Announces the beginning of the data relative to an inertia tensor.

**m**

Mass of the isolated solid (without taking into account the added masses).

**xg,yg,zg**

Coordinates of the center of gravity of the solid isolated in the general reference frame (frame of the mesh).

**ixx,iyy,izz**

Diagonal elements of the inertia tensor of the isolated solid, in the general frame translated to the center of gravity of the solid.

**ixy,iyz,ixz**

Off-diagonal elements of the inertia tensor of the isolated solid, in the general frame translated to the center of gravity of the solid.

**Comments:**

If one single inertia tensor is given ( $ncomp = 1$ ), the keywords `< "COMP" ncomp >` are optional. One may start directly by the keyword `"INER"`.

If the `"INER"` directive is absent, the inertia values will be computed from the initial mesh and densities.

The inertia tensor has the following form:

$$I = \begin{pmatrix} i_{xx} & i_{xy} & i_{xz} \\ i_{xy} & i_{yy} & i_{yz} \\ i_{xz} & i_{yz} & i_{zz} \end{pmatrix}$$

If some parameters are not explicitly given, they are supposed to be zero by default.

In case of complex solids, it is interesting to discretise them finely, and let EUROPLEXUS compute the inertia tensor with high precision. When this operation is terminated, one can take a coarser mesh, by imposing the formerly obtained inertia terms. In this way, the output files will be smaller. But the precision of the calculation will be the same.

### 8.38 ARTICULATION

**Object:**

This directive allows to link two sub-structures by means of a kinematic relationship.

Compatibility: COUP, LIAI for VERR, ROTU, PIVO, GLIS, PIGL and DRIT

Compatibility: COUP for TGGR and CRGR

**Syntax:**

```
"ARTI"  
| "VERR"  ...  |  
| "ROTU"  ...  |  
| "PIVO"  ...  |  
| "GLIS"  ...  |  
| "PIGL"  ...  |  
| "DRIT"  ...  |  
| "TGGR"  ...  |  
| "CRGR"  ...  |
```

**Comments:**

Articulations VERR, ROTU, PIVO, GLIS, PIGL and DRIT may only be defined by means of a mechanism element "MECA". It is therefore necessary that such elements be present in the mesh.

Articulations TGGR and CRGR may only be defined by means of a mechanism element "LIGR".

The linked sub-structures may be described as either non-deformable or deformable.

The various types of articulations are described in the following pages.

### 8.38.1 RIGID ARTICULATION (VERR)

**Object:**

This directive allows to join two sub-structures by means of a blocked articulation, i.e. a rigid connection.

Compatibility: COUP, LIAI

**Syntax:**

```
"VERR" /LECTURE/ ...  
  
... ( "NOEU" /LECTURE/ "VOIS" $[ "ABSENT"      ;  
                                "INDEF" isol  ;  
                                /LECTURE/      ]$ )
```

"VERR" /LECTURE/

Number of the "MECA" element to be rigidly connected.

"NOEU" /LECTURE/

Number of the node of the "MECA" element to which the following neighbour will be associated.

"VOIS" "ABSENT"

There is no need to define a neighbour because the nodes of this sub-structure already have 6 d.o.f.s. The node of the mechanism is then sufficient.

"VOIS" "INDEF" isol

The neighbour is part of the non-deformable solid isol. The points resulting from the decomposition will be used. It seems that this directive cannot be used when the solid is defined by "POIN" (not meshed solid).

"VOIS" /LECTURE/

Number of the points forming the neighborhood (the point belonging to the mechanism must be excluded).

**Comments:**

The two sub-structures are rigidly connected. The six degrees of freedom are coupled on both parts of the mechanism.

The couple "NOEU" "VOIS" must be described twice, i.e. for each of the two points of the mechanism.

**8.38.2 PIVOT****Object:**

This option allows to link two sub-structures by a frictionless hinge.

Compatibility: COUP, LIAI

**Syntax:**

```
"PIVOT" /LECTURE/      ...
      ...  "AXE"  "VX" vx  "VY" vy  "VZ" vz  ...

      ... ( "NOEU" /LECTURE/  "VOIS" $[  "ABSENT"      ;
                                           "INDEF"  isol  ;
                                           /LECTURE/      ]$ )
```

"PIVOT" /LECTURE/

Number of the "MECA" element of the hinge.

vx,vy,vz

Components of the initial direction of the hinge axis.

"NOEU" /LECTURE/

Number of the node of the "MECA" element to which the following neighborhood will be associated.

"VOIS" "ABSENT"

There is no need to define a neighborhood because the nodes of this sub-structure already possess 6 d.o.f.s. The first node of the mechanism is then sufficient.

"VOIS" "INDEF" isol

The neighbourhood is part of the non-deformable solid isol. The points resulting from the decomposition will then be used. It seems that this directive cannot be used when the solid is defined by "POIN" (not meshed solid).

"VOIS" /LECTURE/

Numbers of the points forming the neighbourhood (the point belonging to the mechanism must be excluded).

**Comments:**



The pivot axis is modified accounting for the motions of the sub-structures.

The pair "NOEU" "VOIS" must be described twice, once for each of the 2 points of the mechanism.

Special care must be taken for the neighborhood. In fact, these parts will be considered as rigid for the calculations of angular relations.

### 8.38.3 PIN JOINT (ROTU)

#### Object:

This option allows to connect two sub-structures by a friction-less pin joint (“rotule”).

Compatibility: COUP, LIAI

#### Syntax:

```
"ROTU" /LECTURE/ ...

      ... ( "NOEU" /LECTURE/  "VOIS" $[  "ABSENT"      ;
                                           "INDEF"  isol  ;
                                           /LECTURE/    ]$ )
```

"ROTU" /LECTURE/

Number of the "MECA" element of the pin joint.

"NOEU" /LECTURE/

Number of the node of the "MECA" element to which the following neighborhood will be associated.

"VOIS" "ABSENT"

There is no need to define a neighborhood because the nodes of this sub-structure already possess 6 d.o.f.s. The first node of the mechanism is then sufficient.

"VOIS" "INDEF" isol

The neighbourhood is part of the non-deformable solid isol. The points resulting from the decomposition will then be used. It seems that this directive cannot be used when the solid is defined by "POIN" (not meshed solid).

"VOIS" /LECTURE/

Numbers of the points forming the neighbourhood (the point belonging to the mechanism must be excluded).

#### Comments:

The two sub-structures are linked in translation but free in rotation.

The pair "NOEU" "VOIS" must be described twice, once for each of the 2 points of the mechanism.

### 8.38.4 SLIDER (GLISSIERE)

#### Object:

This option allows to connect two sub-structures by a friction-less slider (“glissière”).

Compatibility: COUP, LIAI

#### Syntax:

```
"GLIS" /LECTURE/    ...
      ...  "AXE"  "VX" vx  "VY" vy  "VZ" vz  ...

      ... ( "NOEU" /LECTURE/  "VOIS" $[  "ABSENT"      ;
                                           "INDEF"  isol  ;
                                           /LECTURE/    ]$ )
```

"GLIS" /LECTURE/

Number of the "MECA" element of the pin joint.

vx,vy,vz

Components of the initial direction of the axis.

"NOEU" /LECTURE/

Number of the node of the "MECA" element to which the following neighborhood will be associated.

"VOIS" "ABSENT"

There is no need to define a neighborhood because the nodes of this sub-structure already possess 6 d.o.f.s. The first node of the mechanism is then sufficient.

"VOIS" "INDEF" isol

The neighbourhood is part of the non-deformable solid isol. The points resulting from the decomposition will then be used. It seems that this directive cannot be used when the solid is defined by "POIN" (not meshed solid).

"VOIS" /LECTURE/

Numbers of the points forming the neighbourhood (the point belonging to the mechanism must be excluded).

#### Comments:

The slider axis is modified to account for the motion of the sub-structures.

The pair "NOEU" "VOIS" must be described twice, once for each of the 2 points of the mechanism.

The axis defined by "AXE" is used only in case of a spring ("RESS") on the connection (to compute the forces coming from the spring) or in case of merging points of the MECA element. In general case, the sliding axis is defined by the two points of the MECA element (local axis of the element).

### 8.38.5 SLIDING PIVOT

#### Object:

This option allows to connect two sub-structures by a friction-less sliding pivot.

Compatibility: COUP, LIAI

#### Syntax:

```
"PIGL" /LECTURE/    ...
      ...  "AXE"  "VX" vx  "VY" vy  "VZ" vz  ...

      ... ( "NOEU" /LECTURE/  "VOIS" $[  "ABSENT"      ;
                                           "INDEF"  isol  ;
                                           /LECTURE/    ]$ )
```

"PIGL" /LECTURE/

Number of the "MECA" element of the sliding pivot.

vx,vy,vz

Components of the initial direction of the axis.

"NOEU" /LECTURE/

Number of the node of the "MECA" element to which the following neighborhood will be associated.

"VOIS" "ABSENT"

There is no need to define a neighborhood because the nodes of this sub-structure already possess 6 d.o.f.s. The first node of the mechanism is then sufficient.

"VOIS" "INDEF" isol

The neighbourhood is part of the non-deformable solid isol. The points resulting from the decomposition will then be used. It seems that this directive cannot be used when the solid is defined by "POIN" (not meshed solid).

"VOIS" /LECTURE/

Numbers of the points forming the neighbourhood (the point belonging to the mechanism must be excluded).

#### Comments:

The sliding pivot's axis is modified to account for the motion of the sub-structures.

The pair "NOEU" "VOIS" must be described twice, once for each of the 2 points of the mechanism.

The rotational axis is supposed to be identical with the sliding axis. This single axis is defined with the "AXE" keyword. Nevertheless, for the sliding behavior, the axis defined by "AXE" is used only in case of a spring ("RESS") on the connection (to compute the forces coming from the spring) or in case of merging points of the MECA element. In general case, the sliding axis is defined by the two points of the MECA element (local axis of the element).

### 8.38.6 IMPOSED RELATIVE DISPLACEMENT (DRIT)

#### Object:

This DRIT directive (Déplacement Relatif Imposé en fonction du Temps = Prescribed Time-dependent Relative Displacement) allows to link two sub-structures by an actuator ("vérin") whose length is a prescribed time function.

Compatibility: LIAI

#### Syntax:

```
"DRIT" /LECTURE/      ...
      ...  "AMPLI"  ampli  "FONCTION" ifonc      ...

      ... ( "NOEU" /LECTURE/  "VOIS" $[  "ABSENT"      ;
                                           "INDEF"  isol  ;
                                           /LECTURE/    ]$ )
```

"DRIT" /LECTURE/

Number of the "MECA" element of the "DRIT" mechanism.

ampli

Amplification coefficient.

ifonc

Number of the function defined by the "FONCTION" directive (see page E.10).

"NOEU" /LECTURE/

Number of the node of the "MECA" element to which the following neighborhood will be associated.

"VOIS" "ABSENT"

There is no need to define a neighborhood because the nodes of this sub-structure already possess 6 d.o.f.s. The first node of the mechanism is then sufficient.

"VOIS" "INDEF" isol

The neighbourhood is part of the non-deformable solid isol. The points resulting from the decomposition will then be used.

"VOIS" /LECTURE/

Numbers of the points forming the neighbourhood (the point belonging to the mechanism must be excluded).

**Comments:**

The relative displacement between the two nodes of the element is equal to the product  $\text{ampli} * F(\text{ifonc}, t)$ .

The pair "NOEU" "VOIS" must be described twice, once for each of the 2 points of the mechanism.



**8.38.7 CONNECTION BETWEEN SHELL AND BEAM (TGGR)****Object:**

This option allows to link a node from a shell with a node from a beam. Both nodes are linked in translation. They can be connected in rotation around the axis "AXE1" and "AXE2" (local axis of the shell) by means of two springs ("MECA" "LIGR").

Not available with LIAI.

**Syntax:**

```
"TGGR" /LECTURE/      ...
      ...  "AXE1"  "VX" vx  "VY" vy  "VZ" vz  ...
      ...  "AXE2"  "VX" vx  "VY" vy  "VZ" vz  ...

      ... ( "NOGR" /LECTURE/ )
```

"TGGR" /LECTURE/

Number of the "LIGR" element.

"AXE1" vx,vy,vz

Components of the first vector (local axis of the shell).

"AXE2" vx,vy,vz

Components of the second vector (local axis of the shell).

"NOGR" /LECTURE/

Number of the node of the "LIGR" element which belong to the shell.

**Comments:**

The pivot axis "AXE1" and "AXE2" are modified accounting for the motions of the shell.

**8.38.8 CONNECTION BETWEEN SHELL AND BEAM (CRGR)****Object:**

This option allows to link a node from a shell with the beam's node which is the closer. Both nodes are linked in translation in the plane defined by the vectors "AXE1" et "AXE2" and free in translation in the perpendicular direction of this plane. They can be connected in rotation around the axis "AXE1" and "AXE2" (local axis of the shell) by means of two springs ("MECA" "LIGR") (See C.965).

Not available with LIAI.

**Syntax:**

```
"CRGR" /LECTURE/      ...
...   "AXE1"  "VX" vx  "VY" vy  "VZ" vz  ...
...   "AXE2"  "VX" vx  "VY" vy  "VZ" vz  ...

... ( "NOGR" /LECTURE/ )
... ( "NOCR" /LECTURE/ )

... < "DMAX" dmax  >
```

"CRGR" /LECTURE/

Number of the "LIGR" element.

"AXE1" vx,vy,vz

Components of the first vector (local axis of the shell).

"AXE2" vx,vy,vz

Components of the second vector (local axis of the shell).

"NOGR" /LECTURE/

Number of the node of the "LIGR" element which belong to the shell.

"NOCR" /LECTURE/

List of the nodes of the "LIGR" element which belong to the beam and are candidates for the connection.

"DMAX" dmax

Optional : allows to introduce a test on the distance between both nodes of the connection. If the distance is superior to dmax then the connection is broken.

**Comments:**

The pivot axis "AXE1" and "AXE2" are modified accounting for the motions of the shell.

The relative perpendicular to the plan motion of the beam is free. The beam's node considered in the link is modified accounting for the relative axial motion of the beam.

## 8.39 ROTATION

### Object:

In the case of a rotating structure, this directive allows to define the symmetry condition with respect to a rotating plane, whose axis and rotation velocity are prescribed by the user.

This directive allows, for example, to model just one sector of a rotating disk instead of the whole disk.

Compatibility: COUP, LIAI

### Syntax:

```
"ROTATION"      "ORIG"    x0 y0  < z0 >
                  < "VECT"    vx vy    vz >
                  "FONC"    ifonc
                  ...                               /LECTURE/
```

**x0,y0,z0**

Coordinates of the origin point (z0 is redundant in 2D).

**vx,vy,vz**

Components of the vector defining the rotation axis. These data are not necessary in 2D (see comments below).

**ifonc**

Number of the function defining the rotation velocity (in rad/s) as a function of time.

**LECTURE**

Numbers of the concerned nodes.

### Comments:

This directive may be used at most once in a calculation.

The rotation axis is supposed fixed. The velocity of the rotation varies in time according to the user-specified function.

In 2D plane calculations, the rotation axis is normal to the plane xOy.

## 8.40 IMPOSED TIME-DEPENDENT ROTATIONAL MOTION

### Object:

In the case of a rotating structure, this directive allows to impose a global motion of rotation to a set of nodes. The axis of rotation and the rotation velocity (as a function of time) are prescribed by the user.

Compatibility: COUP, LIAI

### Syntax:

```
"MENS"      "POINT"    x0 y0 < z0 >
              < "VECTEUR"  vx vy    vz >
              "FONCTION"  ifonc
              /LECTURE/
```

**x0,y0,z0**

Coordinates of the origin point (tail of the rotation axis). Note that z0 is redundant in 2D.

**vx,vy,vz**

Components of the vector defining the rotation axis. These data are not necessary in 2D (see comments below).

**ifonc**

Number of the function defining the rotation velocity (in rad/s) as a function of time.

**LECTURE**

Numbers of the concerned nodes.

### Comments:

This directive may be used at most once in a calculation.

The rotation axis is supposed fixed. The velocity of rotation varies in time according to the user-specified function.

In 2D plane calculations, the rotation axis is normal to the plane xOy.

A time-limited version (TMEN) of the MENS directive, which acts only until a certain time and then is automatically removed, is also available, see Page D.321.

## 8.41 TIME-LIMITED IMPOSED ROTATIONAL MOTION

### Object:

In the case of a rotating structure, this directive allows to impose a global motion of rotation to a set of nodes. The axis of rotation and the rotation velocity (as a function of time) are prescribed by the user. The rotation is imposed up to a prescribed time. After that time, the imposed condition is automatically removed.

Compatibility: COUP

### Syntax:

```
"TMEN"      "POINT"   x0 y0 < z0 >
              < "VECTEUR" vx vy   vz >
              "FONCTION" ifonc
              "UPTO"   t
              /LECTURE/
```

$x_0, y_0, z_0$

Coordinates of the origin point (tail of the rotation axis). Note that  $z_0$  is redundant in 2D.

$v_x, v_y, v_z$

Components of the vector defining the rotation axis. These data are not necessary in 2D (see comments below).

*ifonc*

Number of the function defining the rotation velocity (in rad/s) as a function of time.

*t*

Time up to which the rotation is imposed. After this time, the rotation is automatically released.

LECTURE

Numbers of the concerned nodes.

### Comments:

This directive may be used at most once in a calculation.

The rotation axis is supposed fixed. The velocity of rotation varies in time according to the user-specified function.

In 2D plane calculations, the rotation axis is normal to the plane  $xOy$ .

## 8.42 CONSTANT DISTANCE CONNECTION ("DIST")

### Object:

Automatic prescription of the 3D mechanical relations of constant distance between position of a point and that of another point or set of points.

Compatibility: COUP, LIAI

### Syntax:

```
"DISTANCE"  $ /LECTURE/ ; CENT /LECTC/ LIST /LECTL/ $
```

`/LECTURE/`

Numbers of the two nodes (or name of the object, containing only two nodes) whose distance must be kept constant during the calculation.

`CENT /LECTC/`

Master node, serving as a reference for all other node. The `/LECTC/` must contain only one node.

`LIST /LECTL/`

Slave node(s), whose distance with respect to the master node must be kept constant. The `/LECTL/` can contain any number of nodes (1 or greater).

## 8.43 BARYCENTRIC JUNCTION

### Object:

Automatic prescription of mechanical relations (links) such that the displacement of a point equals the **mean value** of the displacements of a set of points, i.e. the displacement of the barycenter of the set of points (considered all with the same weight).

Compatibility: COUP, LIAI

### Syntax:

```
BARY  CENT /LECT/
      LIST /LECT/
      <VECT <VX vx> <VY vy> <VZ vz>>
```

CENT /LECT/

Number of the node (or name of the object) corresponding to the “central” (or reference) node. This node may be located anywhere and does not need to be at (or close to) the true center of the following points set.

LIST /LECT/

Numbers of the nodes (or name of the object) corresponding to the set of nodes, whose *mean* displacement will be identical to that of the reference node.

VECT

Introduces the optional definition of a direction (vector) along which the constraint will act. By default, the constraint acts along all space directions.

VX, VY, VZ

Introduce the optional components of the vector **vx**, **vy**, **vz**. By default, they are zero. At least one non-zero component must be specified. The **vz** component may only be specified in 3D. Note that only the direction, not the norm, of the vector counts. The vector is always normalized to unit length internally.

### Comments:

By default (no **VECT** specified) this directive imposes the following (vectorial) condition on nodal velocities  $\underline{v}$ :

$$\underline{v}_C - (\underline{v}_1 + \underline{v}_2 + \cdots + \underline{v}_N)/N = \underline{0}$$

which corresponds to the following 2 or 3 scalar *independent* links:

$$v_{Cx} - (v_{1x} + v_{2x} + \cdots + v_{Nx})/N = 0$$



$$v_{Cy} - (v_{1y} + v_{2y} + \cdots + v_{Ny})/N = 0$$

$$v_{Cz} - (v_{1z} + v_{2z} + \cdots + v_{Nz})/N = 0 \quad (3D \text{ only})$$

where  $C$  is the “central” node defined by **CENT** and  $1, 2, \dots, N$  are the  $N$  nodes defined by **LIST**.

When a vector  $\underline{V}$  is specified by **VECT**, then the following single condition on nodal velocities is imposed:

$$\underline{v}_C \cdot \underline{V} - \frac{1}{N}(\underline{v}_1 - \underline{v}_2 - \cdots - \underline{v}_N) \cdot \underline{V} = 0$$

which corresponds to the following scalar link (assuming a 2D case):

$$v_{Cx}V_x + v_{Cy}V_y - \frac{1}{N}(v_{1x}V_x + v_{1y}V_y + \cdots + v_{Nx}V_x + v_{Ny}V_y) = 0$$

Note that the above conditions, both without and with the definition of a vector **VECT**, do *not* strictly ensure that the displacements of all nodes in the set will be all equal among them. To obtain this effect, use the **RIGI** link, see page D.326.

## 8.44 RIGID JUNCTION

### Object:

Automatic prescription of mechanical relations (links) such that:

1. the displacement of each point in a certain set of points equals the displacement of a reference point, like if all these points were all rigidly connected among them, or
2. the displacements of each point in a certain set of points are all equal, like if all these points were all rigidly connected among them.

Note that the first definition requires the choice of a reference node while in the second one no reference node is indicated and all nodes play the same role.

The two alternative definitions given above are logically equivalent, but they lead to two different forms of the links matrix. It was found by practical experimentation that in some applications (where the number of points to be rigidly linked is very high) the second form is much more efficient computationally, as far as the solution of the links system is concerned.

If the decoupled form of the link is chosen (DECO), then only the second form of the directive is possible.

Compatibility: COUP, LIAI, DECO

### Syntax:

```
RIGI <CENT /LECT/>
      LIST /LECT/
      <VECT <VX vx> <VY vy> <VZ vz>>
```

CENT /LECT/

(Optional) number of the node (or name of the object) corresponding to the “central” or reference node. This node may be located anywhere and does not need to be at (or close to) the true center of the following points set. Choosing a reference node activates the first form of the rigid junction. This form of the directive is only accepted if coupled links (COUP or LIAI) are chosen.

LIST /LECT/

If CENT has been specified (first form of the directive), these are the numbers of the nodes (or name of the object) corresponding to the set of nodes, whose displacement will be identical to that of the reference node. Otherwise (second form of the directive) these are the numbers of the nodes (or name of the object) corresponding to the set of nodes, whose displacement will be identical among them.

VECT

Introduces the optional definition of a direction (vector) along which the constraint will act. By default, the constraint acts along all space directions.

VX, VY, VZ

Introduce the optional components of the vector  $\mathbf{vx}$ ,  $\mathbf{vy}$ ,  $\mathbf{vz}$ . By default, they are zero. At least one non-zero component must be specified. The  $\mathbf{vz}$  component may only be specified in 3D. Note that only the direction, not the norm, of the vector counts. The vector is always normalized to unit length internally.

### Comments:

Let us define  $N$  as the number of nodes in the LIST sub-directive. Consider the first form of the directive (CENT has been specified). By default (no VECT specified) this directive imposes the following set of  $N$  (vectorial) conditions on nodal velocities  $\underline{v}$ :

$$\underline{v}_C - \underline{v}_1 = \underline{0}$$

$$\underline{v}_C - \underline{v}_2 = \underline{0}$$

...

$$\underline{v}_C - \underline{v}_N = \underline{0}$$

which corresponds to the following  $2N$  or  $3N$  scalar *independent* links:

$$v_{Cx} - v_{1x} = 0$$

$$v_{Cy} - v_{1y} = 0$$

$$v_{Cz} - v_{1z} = 0 \quad (3D \text{ only})$$

...

$$v_{Cx} - v_{Nx} = 0$$

$$v_{Cy} - v_{Ny} = 0$$

$$v_{Cz} - v_{Nz} = 0 \quad (3D \text{ only})$$

where  $C$  is the “central” node defined by CENT and  $1, 2, \dots, N$  are the  $N$  nodes defined by LIST.

When a vector  $\underline{V}$  is specified by VECT, then the following  $N$  conditions on nodal velocities are imposed:

$$\underline{v}_C \cdot \underline{V} - \underline{v}_1 \cdot \underline{V} = 0$$

$$\underline{v}_C \cdot \underline{V} - \underline{v}_2 \cdot \underline{V} = 0$$

...

$$\underline{v}_C \cdot \underline{V} - \underline{v}_N \cdot \underline{V} = 0$$

which correspond to the following  $N$  scalar links (assuming a 2D case):

$$v_{Cx}V_x + v_{Cy}V_y - v_{1x}V_x - v_{1y}V_y = 0$$

$$v_{Cx}V_x + v_{Cy}V_y - v_{2x}V_x - v_{2y}V_y = 0$$

...

$$v_{Cx}V_x + v_{Cy}V_y - v_{Nx}V_x - v_{Ny}V_y = 0$$

In the second form of the directive (**CENT** has *not* been specified), when no **VECT** is specified this directive imposes the following set of  $N$  (vectorial) conditions on nodal velocities  $\underline{v}$ :

$$\begin{aligned}\underline{v}_1 - \underline{v}_2 &= \underline{0} \\ \underline{v}_2 - \underline{v}_3 &= \underline{0} \\ &\dots \\ \underline{v}_{N-1} - \underline{v}_N &= \underline{0} \\ \underline{v}_N - \underline{v}_1 &= \underline{0}\end{aligned}$$

When a vector  $\underline{V}$  is specified by **VECT**, then the following  $N$  conditions on nodal velocities are imposed:

$$\begin{aligned}\underline{v}_1 \cdot \underline{V} - \underline{v}_2 \cdot \underline{V} &= \underline{0} \\ \underline{v}_2 \cdot \underline{V} - \underline{v}_3 \cdot \underline{V} &= \underline{0} \\ &\dots \\ \underline{v}_{N-1} \cdot \underline{V} - \underline{v}_N \cdot \underline{V} &= \underline{0} \\ \underline{v}_N \cdot \underline{V} - \underline{v}_1 \cdot \underline{V} &= \underline{0}\end{aligned}$$

The reason why the solution of the linear system of constraints may become (very) slow for large  $N$  in the first form of the equations is that the same node  $C$  appears in *all* equations. The bandwidth of the constraints matrix might become quite large.

The second (*circular*) form of the equations is computationally more efficient because the nodes involved keep changing from an equation to the other (each node appearing only in two of the vector constraint equations), so that the bandwidth of the constraints matrix can be made small upon proper renumbering of the links and the solution becomes (much) more efficient.

## 8.45 GLUE - Glueing together two meshes

### Object:

Glue together two incompatible (non-conforming) structural meshes. The nodes of the *slave* mesh are linked to the faces of the *master* mesh so that their relative position with respect to the face does not change during motion and deformation. Common rotation is allowed.

Compatibility: COUP, DECO

### Syntax:

```
GLUE  <DMAX d> <LOOS>  <SELF>
      SLAV           /LECT1/
      MAST <ALLF> /LECT2/
```

#### DMAX d

Optional maximum distance  $d$  of a slave node from a master face for the node to be considered on the face. If omitted, only a small tolerance of approximately 1 % of the face size is admitted for the node to be considered on the face. The specification of **DMAX d** may be useful when the two meshes to be glued together are separated by a (small but non-negligible) gap.

#### LOOS

Optional keyword which enforces only a loose (as opposed to a strict) satisfaction of the glueing constraint. If **LOOS** is not specified (as per default) and a slave node has no corresponding master face, then an error message is issued and the code stops. However, in the above case of no matching face, if the **LOOS** keyword has been specified, then no link is imposed to the slave node and the glueing condition is simply ignored for this node.

#### SELF

Optional keyword, which allows the glueing of part of a body upon itself. In this case, a slave node may belong to one or more master elements, which is not the case without this keyword. When the option is active, the search algorithm automatically skips all master elements that contain the slave node under consideration (if any).

#### SLAV /LECT1/

List of the slave *nodes*.

#### MAST /LECT2/

List of the master *elements*. If a master element is of continuum type, then by default only its *external* faces are considered as valid candidates to become master faces, while internal faces are ignored.

#### ALLF

This optional keyword causes *all* faces (both external and internal) of the (continuum) master elements to be considered as valid candidates to become master faces. For shell master elements, no distinction is made between internal and external faces and the ALLF keyword, if present, has no effect.

**Comments:**

Each slave node is checked against all faces of the master elements, until a face is found on which the slave node is initially located (within a small tolerance). Unless LOOS is specified, this face *must* exist and be unique, and is denoted the master face.

The initial position of the slave node with respect to the nodes of the corresponding master face allows to compute the shape functions (coefficients) of the relations (links) that keep together the two entities during the (common) motion and deformation of the model. The relations involve translational degrees of freedom only (two relations in 2D, three in 3D).

Up to two named node groups are automatically created for each occurrence of the GLUE directive. The first group has the form \_GLUEnnnn where nnnn is a 4-digit number (0001, 0002 etc.) indicating the index of the current GLUE directive in the input file. This group contains the nodes which were actually glued. The second group has the form \_NGLUnnnn and contains the nodes which were *not* glued, despite having been declared in the SLAV /LECT1/ directive, because a matching master face could not be found. Obviously, this second group is created only if there are any non-matching nodes and if the user has specified the LOOS keyword. The groups can be visualized in the OpenGL graphical module for direct visual inspection of which nodes were actually glued (or not glued.)

## 8.46 CONTACTS DEFINED BY SPLINE FUNCTIONS

### Object:

In the case of a rotating structure, this directive allows to define the possible contacts between the rotating parts (blades) with the fixed wall (carter). The geometrical forms of these parts are defined by means of spline functions starting from the positions of mesh nodes. This interpolation allows thus to approximate the real geometry of such structures.

Compatibility: LIAI

### Syntax:

```
"SPLINE"
nspline * (      "SURFACE"  /LECTURE/      ...
                ... "COURBE"  ncourbe * ( "LIGNE" /LECTURE/ )      ...
                ... "METC"   metc   "METS"   mets   "NPTT"   nptt   ...
                ... "DEGC"   degc   "DGST"   dgst   "DGSZ"   dgsz   ...
                ... "EPAIS"  epais  "FREQ"   freq                                     )
```

**nspline**

Number of splines.

**SURFACE**

Defines the nodes forming the surface. This surface **MUST** be a cylinder,

**ncourbe**

Number of curves that may get in contact with the surface.

**LIGNE**

Introduces the nodes that compose a curve. **The user must enter these nodes in the order of their position along the curve.**

**metc**

Method for the modelisation of the curve (see comments below).

**mets**

Method for the modelisation of the surface (see comments below).

**nptt**

Number of nodes of the surface lying on the same circumference.

**degc**

Degree for the modelisation of the curve.

**dgst**

Degree for the modelisation of the surface, circumferential direction.

**dgst**

Degree for the modelisation of the surface, axial direction Oz.

**epais**

Thickness of the shell elements composing the surface.

**freq**

Frequency of the updationg of surface nodes.

**Comments:**

The methods for the modelisation (of the curve and of the surface along the circumferential and axial directions) may assume the values: 1 (direct), 2 (interpolation) or 3 (smoothing by least squares).

The surface MUST be a cylinder of axis Oz. Furthermore, the nodes composing it must be regularly spaced.



## 8.47 COLLISIONS

### Object :

This directive allows to simulate the contact and/or shock without friction between the envelopes of 3D rigid bodies.

Compatibility: LIAI

### Syntax :

```
"COLL"      "REST" crest  "SGEO" tolgeo  "SVIT" tolvit
            ( "CHAI"
              ( "SOLI" nusoli
                "SURF" /LECTURE/
                "EPAI" epais
                "ORIE" xp yp zp          )      )
            "CONT"
            ( "CHA1" nucha1  "CHA2" nucha2 )
            "FCON"
```

### COLL

This keyword announces the data relative to collisions.

#### crest

Energy restitution coefficient.

#### tolgeo

Geometric tolerance of the contact.

#### tolvit

Kinematic tolerance of the contact.

### CHAI

This keyword announces the data relative to a chain.

### SOLI

This keyword announces the data relative to one of the solids that define the chain.

#### nusoli

Number of the solid associated to the chain. This number corresponds to the order under which the solid has been listed under the sub-directive "SOLIDE" of the directive "LIAISON".

/LECTURE/

Reading procedure of the triangular elements defining the envelope of the chain, i.e. the contact surfaces.

epais

Thickness of the contact surfaces.

xp, yp, zp

Coordinates of a point interior to the envelope, used to define the orientation of the triangular elements.

CONT

This keyword allows to introduce the list of pairs of chains for which contact may take place.

nucha1

Number of the first chain of the pair.

nucha2

Number of the second chain of the pair.

FCON

This keyword announces the end of the collisions data.

### **Warning :**

It is mandatory:

- to mesh the surfaces by triangular elements;
- to declare these elements as "phantoms" by directive "MATE",
- to define the data block "SOLIDE" before the block "COLLISIONS",
- to specify in "DIME" the dimensioning parameter:

"CSCO" nbpcon

With:

nbpcon

Maximum number of contact points.

### **Comments :**

The coefficient of energy restitution is between 0 and 1. For  $\text{crest} = 0$ , one obtains a perfectly soft shock, while for  $\text{crest} = 1$  one gets a perfectly elastic shock (the energy is conserved).

The thickness of surfaces must be of the order of the size of elements at most. If this value is too small, it is possible that the interpenetration of the two surfaces will not be detected.

The contact geometric tolerance determines the distance starting from which one considers that there is contact.

The kinematic tolerance must be of the order of the time step. The larger this tolerance, the more the discontinuity at the velocity level due to a shock is ignored.

If a contact surface is fixed (instead of being defined via a rigid body), it is sufficient to declare `nusoli = 0`. In this case it is redundant to block the concerned nodes, since it is done automatically by the code.

### References :

For further information, please consult the reference [\[643\]](#).

## 8.48 FLUID-STRUCTURE SLIDING OF ALE TYPE (FSA)

### Object:

To define fluid-structure sliding of the ALE type according to the FSA model developed at JRC Ispra.

The program writes for each node subjected to this type of sliding a 'liaison' that forces the fluid (slave) velocity to be equal to the structure (master) velocity along the normal to the FS interface. In the tangent direction (tangent plane in 3D), the fluid velocity is free.

In the case of a curved interface, the normal direction is determined at each step by taking into account all the element faces that lie along the fluid-structure boundary and include the node under consideration (influence domain) and by imposing that the net flux of mass out of some faces be balanced by the flux entering the other faces.

Since the geometry varies in time, the coefficients of the liaison have to be recalculated and the matrix inverted at each step.

The nodes declared in this directive should be fluid nodes and be declared as Eulerian in the GRIL directive. The program then automatically searches for each slave node a corresponding master node: this is defined as the Lagrangian node having the same coordinates as the slave node (within a small tolerance) and if it exists (nodally **conforming** FS interface), it must be unique. Usually this will be a structural node, but it could be also a fluid (Lagrangian) node, in case the sliding takes place along a fluid-fluid interface.

If no such node exists, then the FS interface is nodally **non-conforming** and the program searches a Lagrangian master **face** on which the slave fluid node lies. The motion of the fluid node is automatically set so as to follow the motion of the master face.

Note that the treatment of non-conforming FS interfaces requires a special optional keyword (NCFS) to be explicitly chosen by the user. If this keyword is not specified and a non-conforming node is found, then an error message is issued and the calculation is stopped. This is to make sure that the user intentionally wanted to specify a non-conforming interface and there was not just an error in mesh specification.

Compatibility: COUP, LIAI

### Syntax:

```
"FSA" <"STRU" /LECT_STRU/> <"NCFS"> /LECTURE/
```

```
STRU /LECT_STRU/
```

Optional sub-directive used to tell the code in which object (/LECT\_STRU/) it should search to determine the "structural" (i.e. the Lagrangian) nodes corresponding to the FSA fluid nodes that will be specified in the final /LECTURE/. By default, the search is extended to the whole mesh.

NCFS

The FS interface may contain non-conforming fluid nodes.

/LECTURE/

List of fluid (slave) nodes subjected to FSA sliding.

### Comments:

The fluid nodes subjected to FSA sliding should preferably be declared Eulerian in the grid movement directive (**GRILL**). The program will automatically consider these nodes as manually rezoned when it encounters the **LIAI FSA** directive. The user might also declare these nodes as automatically rezoned in **GRILL** (e.g., as a consequence of an **AUTO AUTR** directive), with no effect on the results, but in this case the dimensioning for automatically rezoned nodes (**DIME NBLE**) should include these nodes, although this is not necessary for the actual computation.

Beware that the behaviour of the FSA algorithm may be modified by setting appropriate options, see page H.120. In particular, the **FSCR** option activates the correction of normals based on equilibrium considerations (**FSCR** algorithm).

Occasionally, the automatic search for the master node corresponding to a slave node might fail. The code then reports the concerned node number by an appropriate error message. This may happen because either the code finds zero nodes, or it finds more than one Lagrangian nodes matching the slave node.

In the first case, the tolerance for node matching determination might be too small, e.g. due to the fact that mesh coordinates are generated by an external, and not too precise, mesh generator. The user may adjust this tolerance, see **OPTI TOLC** on page H.40.

The second case may occur for example when there are superposed structures (coincident nodes) in the initial mesh. In such cases, there are two possibilities. Either the user specifies the required nodes correspondence by the **COMP CNOD** directive, see page C.92, but this is only practical if there are just a few of these nodes. Or, the user specifies the **STRU /LECT\_STRU/** optional sub-directive, so that the search for matching structural (more precisely, Lagrangian) nodes is confined to the specified object **/LECT\_STRU/** rather than to the whole mesh. This is the method of choice e.g. in case a large shell structure is subjected to FSA on one side, and to Lagrangian sliding (say, by **GLIS**) on the other side, so that the number of “superposed” structural nodes is potentially large.

## 8.49 RIGID-BOUNDARY/FLUID SLIDING OF ALE TYPE

### Object:

To simplify the description of fluid sliding along inviscid, rigid boundaries. The simplification lies in the fact that the program automatically computes the correct sliding conditions, in particular the normal (or possibly the 2 normals, in 3D cases) to the rigid boundary and automatically prescribes the relevant "connections" (liaisons).

For complex geometric shapes this is very convenient with respect to the "manual" prescription of all such connections.

This condition is similar to the "FSA" condition, but with the following differences:

- Since the boundary is rigid, there is no need to represent it by a structure. The sliding condition therefore involves only a fluid node.
- The geometry of the boundary does not vary in time, therefore the coefficients of the liaison are constant and do not need to be recalculated during the transient.
- The program does not search for a Lagrangian node having the same coordinates as the fluid node.

The nodes declared in this directive (/LECT/) should all be fluid nodes and be declared as Eulerian in the GRIL directive.

Compatibility: COUP, LIAI

### Syntax:

```
"FSR"  /LECTURE/
```

```
/LECTURE/
```

List of fluid nodes subjected to FSR sliding.

### Comments:

The fluid nodes subjected to FSR sliding should preferably be declared Eulerian in the grid movement directive (GRILLE). The program will automatically consider these nodes as Eulerian when it encounters the LIAI FSR directive. The user might also declare these nodes as automatically rezoned in GRILLE (e.g., as a consequence of an AUTO AUTR directive), with no effect on the results, but in this case the dimensioning for automatically rezoned nodes (DIME NBLE) should include these nodes, although this is not necessary for the actual computation.

## 8.50 IMPACT/CONTACT BY PINBALL MODEL (PINB)

### Object:

The purpose is to define impact and contact conditions between Lagrangian subdomains (typically two or more solid bodies) by means of the “pinball” model. The model is inspired to a formulation proposed by Belytschko and co-workers in the papers: (i) Ted Belytschko and Mark O. Neal, “Contact-Impact by the Pinball Algorithm with Penalty and Lagrangian Methods”, Int. J. Num. Meths. Eng., Vol. 31, pp. 547-572 (1991), and (ii) T. Belytschko and I.S. Yeh, “The splitting pinball method for contact-impact problems”, CMAME, 105, pp. 375-393, (1993).

The user defines the elements that may enter in contact with one another and a pinball (a sphere or circle) is associated to these elements. Interpenetration is detected by comparing the distance of the centers of two pinballs with the sum of their radii. If this condition is satisfied, equal normal velocity is enforced by the method of Lagrange multipliers and the corresponding contact forces are computed.

Optionally, contact may be verified on a hierarchy of “descendent” pinballs derived from the “parent” pinballs described above by recursively halving the pinball dimensions. This allows finer spatial resolution of the contact conditions.

The uncoupled version of the pinball algorithm (DECO keyword) uses a penalty method instead of (coupled) Lagrange multipliers.

Compatibility: COUP, DECO, LIAI

### Syntax:

```
PINB $[ PENA <SFAC sfac> ]$
  ( ![ $[BODY ; SELF]$
    < "FROT" "MUST" must "MUDY" mudy "GAMM" gamm >
    < $[DMIN dmin ;
      MLEV mlev ;
      DIAM diam < ADAD < UPTO lmax > >
        < ADNP < UPTO lmax > > ]$ >
    < HARD hard > ]!
  /LECT/ )
< EXCL (PAIR n1 n2) >
< ADAP LMAX lmax <SCAL scal> <SCAS scas> <NOUN> >
```

The input consists of several parts. The first part is related to the chosen **solution method**. If LINK COUP or LIAI has been chosen, then this part may be skipped. If LINK DECO has been chosen, this part is mandatory.

### PENA

DECO only: mandatory keyword (ignored with COUP or LIAI), must immediately follow the PINB keyword and indicates that a penalty method is used.

**SFAC sfac**

DECO only: optional scaling coefficient  $\phi$  for the automatic determination of the contact stiffness (see Comments below). By default it is 1.0.

Next, comes the description of the bodies in contact, or more precisely the description of **pinball sets** to be embedded in the contacting bodies. The **BODY** or **SELF** (in order to activate self-contact) sub-directives should be repeated as many times as necessary to define all the contacting pinball sets.

**BODY**

Introduces the declaration of a set of pinballs that form one of the bodies that may come in contact with other bodies. There may not be contact between pinballs belonging to the same body.

**SELF**

Introduces the declaration of a set of pinballs that form one of the bodies that may come in contact with other bodies. In this case, there may be contact between different pinballs belonging to this body (this model is called self-contact or auto-contact).

The next sub-block of data concerns the optional definition of **friction characteristics** of the body (i.e. of the pinballs set).

**FROT**

Introduces the specification of (optional) friction characteristics for the current contacting body. A simple Coulomb dry friction model is assumed.

**MUST must**

Specifies the limiting friction coefficient for the static case  $\mu_S$ . This is the value assumed when no sliding occurs between the contacting surfaces. It must be ( $0 \leq \mu_S \leq 1$ ).

**MUDY muddy**

Specifies the friction coefficient for the dynamic (or kinetic) case  $\mu_K$ . This is the (asymptotic) value assumed at very large (infinite) relative velocity of the contacting surfaces. It must be ( $0 \leq \mu_K \leq \mu_S \leq 1$ ).

**GAMM gamm**

Parameter ( $\gamma$ ) of the law of variation of the friction coefficient ( $\mu$ ) with the relative tangential sliding velocity ( $v_r$ ) of the contacting surfaces. It must be  $\gamma \geq 0$ . The friction coefficient  $\mu$  varies from  $\mu_S$  to  $\mu_K$  as the relative tangential velocity  $v_r$  of the two bodies increases. The transition between the two regimes is governed (smoothly) by the exponential decay law:  $\mu = \mu_K + (\mu_S - \mu_K)e^{-\gamma|v_r|}$ . Note that for  $\gamma = 0$  we have  $\mu = \mu_S$ , *independently* from the relative velocity  $|v_r|$  and from the value chosen for  $\mu = \mu_K$ .

The following sub-block of data basically defines the **size of the pinballs** belonging to the current body (i.e. of the current pinballs set). Three alternatives are possible: choosing the minimum diameter, choosing the maximum refinement level, or choosing a fixed diameter. In the latter case, only one pinball per element is ever generated.

**DMIN dmin**



Minimum diameter of descendent pinballs that will be generated from the set being declared. By default, this value is 0 for continuum elements (the size is then governed by `mlev`, see below), or it is the element thickness for beam or shell elements. For the choice of `DMIN` in adaptive calculations see the `ADAP` keyword below and the comments at the end of this page. Note that a **modification** in the effects of `DMIN` has been introduced recently. Thus, in order to repeat an old calculation which uses `DMIN` made with EPX version 3208 of 20 February 2017 or earlier, one should divide the old input value of `DMIN` by two, in order to obtain “exactly” the same results as previously (in the rare cases where this might have an importance).

#### MLEV `mlev`

Maximum hierarchy level for descendent pinballs that will be generated from the set being declared. The value 0 means that no descendents are generated (contact forces are computed based on interpenetration between parent or 0-level pinballs). The pinball radius is roughly divided by two at each new level produced. If specified, `mlev` must be greater or equal to 0. If not specified, `mlev` has to be computed. If `dmin` ( $d_{\min}$ ) is given, then the maximum level is computed such that the minimum pinball diameter is *of the order* of  $d_{\min}$ . More precisely, the (0-level) pinball diameter is repeatedly halved until the result  $d_{\min}^{\text{eff}}$  is equal to or less than *twice* the chosen value  $d_{\min}$ . This algorithm guarantees that  $2d_{\min} \geq d_{\min}^{\text{eff}} > d_{\min}$ . If `dmin` is not given, for beam/shell elements `mlev` is computed by repeated halvings such that the minimum pinball diameter is of the order of (more precisely: equal to or less than) *twice* the element thickness  $h$ , that is:  $2h \geq d_{\min}^{\text{eff}} > h$ . For continuum elements and for other element types, the default `mlev` value is 0. For the choice of `MLEV` in adaptive calculations see the `ADAP` keyword below and the comments at the end of this page.

#### DIAM `diam`

Fixed pinball diameter, typically to be associated with elements of the material-point type (PMAT). These elements have just one node and thus their pinball radius may not be computed by the code but must be provided by the user. In special cases this keyword can be used to assign a chosen pinball diameter also to elements *not* of the material point type, e.g. continuum elements. By default, the pinball diameter is never updated during the transient calculation even though the associated element undergoes large deformations, unless the `UPDR` option is specified. So make sure *not* to specify the `UPDR` option if you want the imposed pinball diameter to stay constant. When `diam` is specified, `dmin` may not be specified and `mlev` must be 0 (i.e., either unspecified, or specified to be 0). This means that no hierarchic pinballs are generated when `diam` is specified, i.e. only one pinball of the chosen diameter is associated with each element of the body. The pinball is placed at the centroid of the element. For the choice of `DIAM` in adaptive calculations see the `ADAP` keyword below and the comments at the end of this page.

#### ADAD

Adapt the diameter chosen by `DIAM`. Specifying this optional keyword (after choosing a diameter  $D$  by the `DIAM` command) adapts the diameter of pinballs associated with descendents of the elements in the current body, when such elements are refined by adaptivity. That is, first-generation descendents receive one pinball each (no hierarchy is possible with `DIAM`) with a diameter one half of the ancestor’s diameter (i.e.  $D/2$ ), second-generation descendents get a diameter  $D/4$  and so on. The default behaviour when `DIAM` is set but `ADAD` is *not* activated, is that in case of adaptive refinement of the body’s elements, the diameter of the newly generated pinballs (one per element) is constant and equal to  $D$ .

The default rule seems appropriate, for example, if the body is a thin plate discretized by shells and for which a **DIAM** is chosen (as an alternative to other possible pinball strategies, such as hierarchic pinballs for example). In such a case one probably wants the diameter of pinballs to be equal to the thickness of the plate and to remain constant along with mesh adaptive refinement. In other applications, however, one may prefer that the imposed-diameter pinballs be scaled down as the mesh is refined, and this is the purpose of the **ADAD** keyword.

#### UPTO **lmax**

Limit the diameter adaptation mechanism activated by the **ADAD** keyword up to level **lmax** of the hierarchy. By default, adaptation is performed at all levels when **ADAD** is specified. This optional keyword can be used to avoid obtaining too small diameter pinballs in cases with deep mesh refinement (large hierarchy levels).

#### ADNP

Adapt nodal pinballs (more precisely: *propagate* nodal pinballs in adaptivity). This optional keyword has only effect if the pinballs of the current body are so-called *nodal* pinballs, and is ignored otherwise. So-called nodal pinballs are pinballs associated with material point elements (**PMAT**) attached (as typically mass-less geometrical supports) to the nodes of a body which is discretized by continuum or structural elements. They are not *real* nodal pinballs (directly associated with the nodes), because in the current implementation each pinball always requires an associated *element*. When the elements of the body are refined due to adaptivity, new nodes are created. By default, i.e. *without* specifying the **ADNP** optional keyword, no new pinballs would be created, because technically it is the continuum or structural elements which are refined and not the **PMAT** material point elements. By activating **ADNP**, each newly created node will receive a (new) **PMAT** element with an associated pinball. Note that the diameter of the newly created pinballs will be scaled or not, depending on the setting or not of the **ADAD** optional keyword described above for the current body.

#### UPTO **lmax**

Limit the pinball propagation mechanism activated by the **ADNP** keyword up to level **lmax** of the hierarchy. By default, propagation is performed at all levels when **ADNP** is specified. This optional keyword can be used to avoid obtaining too many (descendant) pseudo-nodal pinballs in cases with deep mesh refinement (large hierarchy levels).

Next comes an optional definition of some **additional parameters** (hardness) and the **list of the elements** forming the current body, i.e. the elements into which the pinballs of the current set should be embedded. This completes the definition of the current set of pinballs.

#### HARD **hard**

Optional “hardness” value to be associated with the body. This information is only used in conjunction with options **OPTI PINS MASL** or **OPTI PINS MAS2**, see page H.160, in order to eliminate constraints in multiple flat contact situations. Values of hardness are arbitrary. The only important thing is the relative value of hardness of two bodies that come into flat contact. The body with lower hardness behaves like a “slave”, and the other one as a “master”. It is advised to use simple integer values, e.g. 1, 2, 3 etc.

/LECT/

List of the elements that will be associated with a (parent or 0-level) pinball of the set being described. For continuum-like bodies, these should typically contain only those elements along the body surface which are likely to come in contact with other objects.

Having defined all the pinball sets, next comes an optional definition of **pairs of sets** that should be excluded from contact. By default, the pinballs of each set are checked for contact against all pinballs of any other set (or even with pinballs of the *same* set if the **SELF** keyword has been used to define the current set). Occasionally, the user may want to disable some of these contacts.

#### EXCL

Introduces a list of body pairs to be excluded from contact search.

#### PAIR *n1 n2*

The body pairs of indexes *n1* and *n2* (in the bodies list declared above) are to be excluded from contact search.

The last part of the input is also optional and concerns the activation of **contact-driven mesh adaptivity**.

#### ADAP

Activates *contact-driven mesh adaptivity*, i.e. automatic refinement and un-refinement of the mesh elements containing pinballs, based on contact detection (and on contact anticipation). Note that this type of mesh adaptivity is at the moment incompatible with other types of adaptivity such as those activated by the **WAVE** or **INDI** directives.

#### LMAX *lmax*

Introduces *lmax*, the desired maximum adaptive refinement level  $L^{\max}$  of the structure mesh (elements) in the vicinity of contacting surfaces. This value should be greater than 1, since level 1 is attributed to the base mesh (no refinement). Each level corresponds to a halving of the mesh size with respect to the immediately previous level. The element level should not be confused with the pinball level, see details in the comments below.

#### SCAL *scal*

Introduces **scal**, an optional scaling factor  $\phi_n$  to be used in the determination of elements to be refined belonging to non self-contacting bodies. By default  $\phi_n = 1.0$ . When scaling the structural influence domain by successive powers of two in order to identify, at each refinement level, the structure elements to be refined or un-refined, the code finally multiplies the result by this factor. Using a value of  $\phi_n$  greater than one, e.g. 1.5 or 2, correspondingly enlarges the zone of structure mesh which is refined and this may result in a smoother mesh transition (for example, as an alternative to the option **OPTI ADAP RCON**).

#### SCAS *scas*

Introduces **scas**, an optional scaling factor  $\phi_s$  to be used in the determination of elements to be refined belonging to self-contacting bodies. By default  $\phi_s = 0.55$ . The use of values of  $\phi_s$  lower than 1.0 is necessary in self-contacting bodies in order to avoid a so-called *chain reaction*, i.e. immediate and uniform refinement of all the elements belonging to the self-contacting body up to the maximum chosen level. Theoretical values of  $\phi_s$  can be determined for regular meshes made of continuum elements (see Table in the comments below), but not so easily for other cases. In practice, some experimentation is needed.

## NOUN

When this optional keyword is specified, *no* element *unsplitting* is performed by the contact-driven adaptivity algorithm. That is, the mesh is refined near the contacting surfaces, but never unrefined.

This completes the definition of the input data.

### Choice of the scaling factor for self-contacting bodies

The theoretical maximum scaling factors to be used for self-contacting bodies are shown in the following Table.

Case	Encompassing pinballs	Equivalent pinballs
2D continuum (squares)	$\frac{\sqrt{2}}{2} = 0.707$	$\frac{\sqrt{\pi}}{2} = 0.886$
3D continuum (cubes)	$\frac{1}{\sqrt{3}} = 0.577$	$\sqrt[3]{\frac{\pi}{6}} = 0.806$

Table 14: Maximum scaling factor  $\phi_{\max}$  for self-contact in 2D and 3D regular continuum meshes.

### Comments:

By default, each pinball (belonging to a certain body) is checked for contact with any other pinball belonging to a different body. If the current pinball's body is declared by the **SELF** keyword rather than **BODY**, then the pinball is checked for contact with any other pinball (including those belonging to the same body). A list of non-contacting body pairs can be optionally declared by the **EXCL** keyword.

For example, assume we have the following input:

```
PINB ... BODY ... /LECT1/ ! first body
      SELF ... /LECT2/ ! second body, is self-contacting
      BODY ... /LECT3/ ! third body
      EXCL PAIR 2 3
```

Then, the pinballs in the first body interact with those of the other two bodies, the pinballs of the second body interact with those of the first and second body, while the pinballs of the third body interact with those of the first body.

The exclusion mechanism can be useful, e.g., in the presence of contact on both sides of a (thin) shell, say a thin reservoir filled of liquid, which is impacted externally by a projectile. The user may want to specify that the shell is in contact both with the liquid (internally) and with the projectile (externally), but direct contact between the projectile and the liquid may not occur.

Be sure to consult also the options related to the pinball model in Section H, see Page H.160, and the interactive commands for the visualization of pinballs and of contacts, see Pages A.25 and O.10.

When using penalty method to compute contact forces, contact stiffness is computed automatically from the stiffness of master elements using the following formulae:

$$k = \phi \frac{GS^2}{V}$$

in the case of solid master elements, with :

$\phi$  : optional scaling coefficient `sfac` given in input. By default  $\phi = 1$ .

$G$  : bulk modulus of master element's material,

$S$  : area of contacting face,

$V$  : volume of master element.

$$k = \phi \frac{GS}{L}$$

in the case of shell master elements, with :

$\phi$  : optional scaling coefficient `sfac` given in input. By default  $\phi = 1$ .

$G$  : bulk modulus of master element's material,

$S$  : area of master element,

$L$  : maximum length of master element's edges.

The bulk modulus  $G$  of the material is:

$$G = \frac{E}{3(1 + \nu)}$$

where:

$E$  : Young's modulus of master element's material,

$\nu$  : Poisson's coefficient of master element's material.

### Distinction between element level and pinball level

Note that the above value of  $L_{\max}$  refers to the maximum refinement level *of the elements* (adaptivity)  $L_{\max}^{\text{adap}}$ , and not of the pinballs (contact)  $L_{\max}^{\text{pinb}}$ . This distinction is unfortunate and is only needed due to historical reasons: the pinball models was developed and implemented in EPX long before the adaptivity model. The relation between the levels is as follows: a base (not refined) element in adaptivity has by convention  $L^{\text{adap}} = 1$ , while a base (parent) pinball in the contact model has by convention  $L^{\text{pinb}} = 0$ . Thus, it should be kept in mind that:

$$L^{\text{adap}} = L^{\text{pinb}} + 1$$

It seems preferable and more consistent with other adaptivity directives of EPX to use the LMAX keyword in the PINB ... ADAP directive to define  $L_{\max}^{\text{adap}}$  rather than  $L_{\max}^{\text{pinb}}$ . In any case, it should be rarely necessary to use a hierarchic pinball method *in combination* with contact-driven adaptivity, so the level of the generated pinballs (attached to the smaller and smaller elements) will be zero, and the user can safely ignore this.

### References

Examples of application of the contact model by the pinball method are presented in the following papers: [268].

## 8.51 CONTACT/IMPACT BY GENERALIZED PINBALL MODEL (GPIN)

### Warning:

The present directive is currently still under implementation and validation. It may not be used yet for production runs. It is possible that not all keywords listed below be implemented yet.

### Object:

The purpose is to define contact and impact conditions between Lagrangian subdomains (typically two or more solid bodies) by means of a variant of the “pinball” model, called “generalized pinballs” method. The model is inspired to the original pinball formulation proposed by Belytschko and co-workers in the papers: (i) Ted Belytschko and Mark O. Neal, “Contact-Impact by the Pinball Algorithm with Penalty and Lagrangian Methods”, Int. J. Num. Meths. Eng., Vol. 31, pp. 547-572 (1991), and (ii) T. Belytschko and I.S. Yeh, “The splitting pinball method for contact-impact problems”, CMAME, 105, pp. 375-393, (1993). However, generalized pinballs (GPINs) are not only spherical, but may assume other shapes (rectangles in 2D, cylinders, triangular prisms and hexahedra in 3D).

The user defines the elements that may enter in contact with one another and GPINs of the appropriate shapes are automatically associated with (typically the surface of) these elements. Interpenetration is detected by checking couples of GPINs. If this condition is satisfied, equal normal velocity is enforced by the method of Lagrange multipliers and the corresponding contact forces are computed.

Unlike the standard pinball model (PINB, see page D.480), the generalized pinball model does not admit (and does not need) hierarchical pinballs.

The uncoupled version of the generalized pinball algorithm (DECO keyword) uses a penalty method instead of (coupled) Lagrange multipliers.

Compatibility: COUP, DECO.

### Syntax:

```
"GPIN" $[ "PENA" <"SFAC" sfac> ]$
  ( $[ "BODY" ; "SELF" ]$
    < $[ "NOCG" ; "SHCG" < "ANGL" angl > < "ABS" > ]$ >
    < "FROT" "MUST" must "MUDY" muddy "GAMM" gamm >
    /LECT/ )
  ( "DIAM" diam /LECT/ )
  < "MASL" ("PAIR" m s) >
  < "EXCL" ("PAIR" n1 n2) >
```

PENA

DECO only: mandatory keyword (ignored with COUP), must immediately follow the GPIN keyword and indicates that a penalty method is used.

**sfac**

DECO only: optional coefficient  $\phi$  for the automatic determination of the contact stiffness (see comments below). By default it is 1.0.

**BODY**

Introduces the declaration of a set of generalized pinballs (GPINs) that form one of the bodies that may come in contact with other bodies. There may not be contact between GPINs belonging to the same body. Some restrictions apply to the elements that can be declared, see the comments below.

**SELF**

Introduces the declaration of a set of GPINs that form one of the bodies that may come in contact with other bodies. In this case, there may be contact between different GPINs belonging to this body (this model is called self-contact or auto-contact). Some restrictions apply to the elements that can be declared, see the comments below.

**NOCG**

Do not create corner GPINs (C-GPINs) for this body. This has only effect in 3D. By default, C-GPINs are created in 3D for all corners (both sharp and not sharp) of continuum bodies and for all corners (both sharp and not sharp) and all free edges of plate/shell bodies.

**SHCG**

Create corner GPINs (C-GPINs) only at *sharp* corners and at free edges for this body. For the definition of sharp corners see the description of the ANGL keyword below. This has only effect in 3D.

**ANGL**

Sets the minimum angle  $\alpha_0$  (between two 3D faces with a common side) beyond which the side is considered to be a sharp corner. By default, this angle is 60 degrees. Let  $n_1$  and  $n_2$  be unit normals to the two faces. Then the scalar product  $n_1 \cdot n_2 = \cos \alpha$  is equal to the cosine of  $\alpha$ , the angle between the normals (which is also the angle between the faces). Thus the corner is sharp if  $\cos \alpha < \cos 60^\circ$ , i.e. when  $\alpha < 60^\circ$ .

**ABS**

Consider the absolute value of the above scalar product instead of the signed value. This has the following effect: when two faces have a common side and opposite (or nearly opposite) normals, the side is *not* considered sharp (while by default it would be). This option may be useful in the presence of complex 3D shell structures, because it is not always easy (and sometimes even impossible) to orient them consistently. With this option many “spurious” sharp corners disappear. Thus with this option the rule becomes: the corner is sharp when  $|\alpha| < 60^\circ$ .

**FROT**

Introduces the specification of (optional) friction characteristics for the current contacting body. A simple Coulomb dry friction model is assumed.

**MUST**

Specifies the limiting friction coefficient for the static case  $\mu_S$ . This is the value assumed when no sliding occurs between the contacting surfaces. It must be  $(0 \leq \mu_S \leq 1)$ .

**MUDY**

Specifies the friction coefficient for the dynamic (or kinetic) case  $\mu_K$ . This is the (asymptotic) value assumed at very large (infinite) relative velocity of the contacting surfaces. It must be  $(0 \leq \mu_K \leq \mu_S \leq 1)$ .

**GAMM**

Parameter ( $\gamma$ ) of the law of variation of the friction coefficient ( $\mu$ ) with the relative tangential sliding velocity ( $v_r$ ) of the contacting surfaces. It must be  $\gamma \geq 0$ . The friction coefficient  $\mu$  varies from  $\mu_S$  to  $\mu_K$  as the relative tangential velocity  $v_r$  of the two bodies increases. The transition between the two regimes is governed (smoothly) by the exponential decay law:  $\mu = \mu_K + (\mu_S - \mu_K)e^{-\gamma|v_r|}$ . Note that for  $\gamma = 0$  we have  $\mu = \mu_S$ , *independently* from the relative velocity  $|v_r|$  and from the value chosen for  $\mu = \mu_K$ .

**/LECT/**

List of the **elements** whose nodes (and then faces) will be associated with GPINs of the set being described.

**DIAM**

Introduces the declaration of a “contact diameter” to be associated with the nodes specified next. The nodes specified must be a sub-set of the nodes belonging to the elements listed in the previous **BODY** or **SELF** declarations. Some restrictions apply to the nodes that can be declared, see the comments below.

**diam**

Generalized pinball diameter (contact diameter) to be associated with P-GPINs attached to the nodes specified by the following **/LECT/**.

**/LECT/**

List of the **nodes** concerned.

**MASL**

Introduces a list of body pairs acting as master/slave with respect to each other. By default, body pairs not mentioned in this list act as both master and slave with respect to each other.

**PAIR m s**

The body of indexes **m** acts as a master when contacting body of index **s**, which acts as a slave. It must be  $1 \leq m \leq B$ ,  $1 \leq s \leq B$  and  $m \neq s$ , with **B** the total number of bodies previously declared.

**EXCL**

Introduces a list of body pairs to be excluded from contact search.

**PAIR n1 n2**



The body pairs of indexes  $n1$  and  $n2$  (in the bodies list declared above) are to be excluded from contact search.

### Comments:

A point GPIN (P-GPIN) is associated to each **node** to which a contact diameter has been assigned via the **DIAM** directive. Then, the other GPIN types (L-GPINs in 2D, or L/T/Q-GPINs in 3D) are built for each **element face** whose nodes have *all* received a contact diameter.

The following **restrictions** apply to the elements that are declared in the **BODY** (or **SELF**) directive, and to the nodes that are declared in the **DIAM** directive described above:

- An element cannot belong to more than one body at the same time, therefore each element index can appear in at most one **BODY** or **SELF** declaration.
- A node potentially subjected to contact, and therefore with an assigned **DIAM**, cannot belong to more than one body at the same time.

Since a P-GPIN is attached to each such node, and this P-GPIN (like any other GPIN) must have one and only one associated body index, for obvious reasons, it follows that:

- The nodes which are common to elements belonging to more than one body cannot have an associated **DIAM**, and therefore cannot participate in the contact.
- In other words, (the elements of) two or more bodies can have some nodes in common, but no one of such nodes can have an associated **DIAM**, because it cannot participate in the contact.

By default, each GPIN (belonging to a certain body) is checked for contact with any other GPIN (of suitable type) belonging to a different body. If the current GPIN's body is declared by the **SELF** keyword rather than **BODY**, then the GPIN is checked for contact with any other GPIN of suitable type (including those belonging to the same body). A list of non-contacting body pairs can be optionally declared by the **EXCL** keyword.

For example, assume we have the following input:

```
GPIN ... BODY ... /LECT1/ ! first body
      SELF ... /LECT2/ ! second body, is self-contacting
      BODY ... /LECT3/ ! third body
      DIAM ... /LECT123/ ! same diameter at all nodes
      EXCL PAIR 2 3
```

Then, the GPINs in the first body interact with those of the other two bodies, the GPINs of the second body interact with those of the first and second body, while the GPINs of the third body interact with those of the first body.

The exclusion mechanism can be useful, e.g., in the presence of contact on both sides of a (thin) shell, say a thin reservoir filled of liquid, which is impacted externally by a projectile. The user may want to specify that the shell is in contact both with the liquid (internally) and with the projectile (externally), but direct contact between the projectile and the liquid may not occur.

Be sure to consult also the options related to the generalized pinball model in Section H, see Page H.160, and the interactive commands for the visualization of generalized pinballs and of contacts, see Pages A.25 and O.10.

When using penalty method to compute contact forces, contact stiffness is computed automatically from the stiffness of master elements using the following formulae :

$$k = \phi \frac{GS^2}{V}$$

in the case of solid master elements, with :

$G$  : bulk modulus of master element's material,

$S$  : area of contacting face,

$V$  : volume of master element.

$$k = \phi \frac{GS}{L}$$

in the case of shell master elements, with :

$G$  : bulk modulus of master element's material,

$S$  : area of master element,

$L$  : maximum length of master element's edges.

The bulk modulus  $G$  of the material is:

$$G = \frac{E}{3(1 + \nu)}$$

where:

$E$  : Young's modulus of master element's material,

$\nu$  : Poisson's coefficient of master element's material.

## 8.52 FLUID-STRUCTURE SLIDING BY "FSS"

### Object:

The purpose is to define fluid-structure sliding lines of the ALE, Lagrangian or fixed type according to the models developed at JRC Ispra.

These directives are obsolete and are maintained only for compatibility with old input files. Use the "LINK COUP FSA" or "LINK COUP FSR" directives instead.

Compatibility: DECO

### Syntax:

```
"FSS"  | "ALE"   . . . |  
        | "LAGR"  . . . |  
        | "FIXE"  . . . |
```

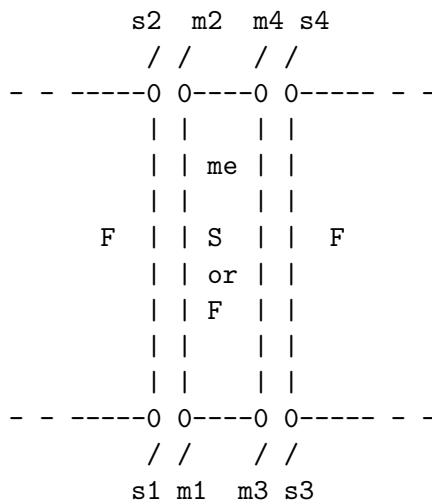
### Comments:

These directives use a rather primitive input syntax that obliges the user to use node indexes and often leads to complex and lengthy input data. A simplification of the input structure to allow the use of GIBI objects is foreseen, but not yet available.

## FLUID-STRUCTURE SLIDING OF THE ALE TYPE

### Object:

Defines fluid-structure sliding lines of the ALE type according to the model developed at JRC Ispra. In this type of sliding, the couples of nodes remain permanently aligned. Thus, there is sliding of the fluid along the structure or with respect to another (master) fluid, but the mesh does not slide. This type of sliding is useful for permanently submerged parts of a structure.



F = fluid element  
S = structural element

### Note:

Nodes (s1, m1) (s2, m2) (s3, m3) (s4, m4) are coincident in the real geometry.

Master (structural or fluid) nodes are Lagrangian, while slave nodes are treated by the ALE formulation and are constrained to follow the corresponding master nodes.

Compatibility: COUP

### Syntax:

```
"ALE"      "NCOT" nasle * ( /LECTURE/ )
           "NPOI" nasln
```

where:

```
/LECTURE/ = LECT me m1 m2 s1 s2 m3 m4 s3 s4 c1 c2 TERM
```

**nasle**

Number of ALE sliding element side couples of the type shown in the above sketch, to be described by the following /LECTURE/.

**me**

Master element index.

**m1, m2**

Nodes defining the first master element side.

**s1, s2**

Nodes defining the first slave element edge.

**m3, m4**

Nodes defining the second master element side (default is 0 0, i.e. sliding occurs along one side only of the master element).

**s3, s4**

Nodes defining the first slave element edge (default is 0 0, i.e. sliding occurs along one side only of the master element).

**c1, c2**

Key to define the type of connection for nodes (s1, m1, m3, s3) and (s2, m2, m4, s4), respectively. Normally these values are both 1, that means ALE sliding. A value of 0 means connection without sliding: this allows to rapidly eliminate a sliding condition, i.e. as if the nodes were rigidly connected, without modifying too much the input. Note that, when a sliding condition is eliminated by posing c1 or c2 equal 0, the corresponding slave node must be declared Lagrangian in the GRILLE directive. Finally a third possibility, indicated by the value -1, is used to model a so-called U-bend ALE sliding. This is useful for situations where a thin structure is permanently submerged in a fluid, in order to model the U-shaped flow around a tip in the structure (represented by the shell element thickness). In this case, the two structural nodes on the tip have different normals (while for 'inner' nodes the normal is unique), so a special treatment is needed.

**nasln**

Total number of nodes defining each of the (slave or master) ale sliding lines.

### Comments:

If a negative value is given for m1, m2, s1, s2, m3, m4 s3 or s4, then the corresponding node is not considered in the ALE sliding process. This feature is useful when modeling e.g. a continuous fluid-structure interface of which one part has a sliding condition of the ALE type, while the rest has a condition of the Lagrangian type. In this case, the element couple at the transition between the two conditions will have one couple of ALE sliding nodes, and the other one Lagrangian. This Lagrangian couple of nodes, say m2 and s2, should have negative indexes.

Finally, note that in this type of sliding the number of nodes in the fluid and in the structure must coincide (the nodes themselves must coincide two by two), so the mesh size is necessarily the same on both sides and it is not possible to use a finer mesh on one of the sides with respect to the other.

## FLUID-STRUCTURE SLIDING OF THE LAGRANGIAN TYPE

### Object:

Defines fluid-structure sliding lines of the Lagrangian type according to the model developed at JRC Ispra. In this type of sliding, the couples of nodes do not remain permanently aligned. Thus, there is sliding of the fluid mesh along the structure. This type of sliding is useful when the interface nodes cannot be kept permanently aligned, e.g. near free surfaces. The first side of the sliding line consists of fluid nodes only; the second side may consist either of structural or of (master) fluid nodes.

```

first side  second side
          f2 s2
          / /
- - ----0 0----0
          | |   |
          fe | | se |
          | |   |
          F  | | S  |
          | | or |
          | | F  |
          | |   |
          | |   |
- - ----0 0----0
          / /
          f1 s1

F = fluid element
S = structural element

```

Compatibility: COUP

### Syntax:

```

"LAGR"  "NCT1" lsle1 * ( /LECTURE1/ )
        "NPOI" lsln1
        "NCT2" lsle2 * ( /LECTURE2/ )
        "NPOI" lsln2

```

where:

```

/LECTURE1/ = LECT fe f1 f2 TERM
/LECTURE2/ = LECT se s1 s2 TERM

```

**lsle1**

Number of element sides on the first side (slave side) of the Lagrangian sliding line.

**fe**

Index of the fluid (slave) element.

**f1, f2**

Indexes of the nodes of the slave edge (first side).

**lsln1**

Total number of nodes defining the slave edges.

**lsle2**

Number of element edges on the second side (master side) of the Lagrangian sliding line.

**se**

Index of the structural (or master fluid) element.

**s1, s2**

Indexes of the nodes of the master edge (second side).

**lsln2**

Total number of nodes defining the master edges.

### Comments:

If a negative value is given for f1, f2, s1 or s2, then the corresponding node is not considered in the Lagrangian sliding process. This feature is useful when modeling e.g. a continuous fluid-structure interface of which one part has a sliding condition of the ALE type, while the rest has a condition of the Lagrangian type. In this case, the element couple at the transition between the two conditions will have one couple of ALE sliding nodes, and the other one Lagrangian. The ALE couple of nodes, say m2 and s2, should have negative indexes.

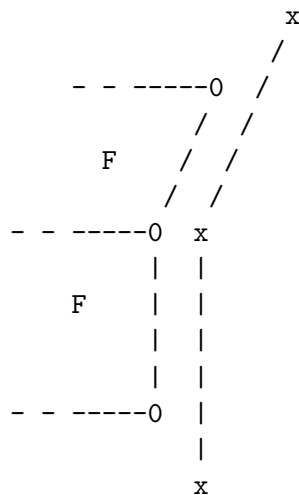
In this type of sliding, the number of nodes on the fluid side may be different from that on the structural side, since the nodes don't have to be aligned in the initial configuration, as it is the case for ALE sliding. It is therefore possible to use meshes of different size for the fluid with respect to the structure.



## FLUID-STRUCTURE SLIDING OF THE FIXED TYPE

### Object:

Defines fluid-structure sliding lines of the fixed type according to the model developed at JRC Ispra. This type of sliding is sometimes useful to model rigid inviscid boundaries. Nodes belonging to a fixed sliding line are treated as Lagrangian. The fixed boundary is defined via a series of points identified by their coordinates.



F = fluid element  
 0 = fluid node  
 x = fixed point defining fixed sliding line

Compatibility: COUP

### Syntax:

```
"FIXE"  "NPOI" n1fsl /LECTURE/
        "NFI"  n2fsl * ( xcoor ycoor )
```

n1fsl

Number of nodes on the fixed sliding line.

n2fsl

Number of fixed points used to define the fixed boundary.

xcoor, ycoor

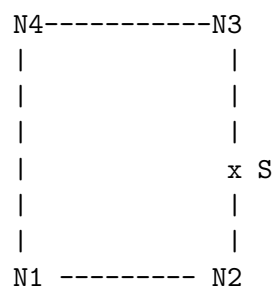
Coordinates of the fixed point.

### 8.53 NODE TO SHELL CONNECTOR

#### Object:

This element is used in order to connect node to a master edge of shell. Note that the "SH3D" directive is a subdirective of the "LIAI" directive but it is needed to define an element which defines the nodes (master and slave) of the liaisons. It is listed in this Section because it consists in the definition of kinematic constraints between the dof of one slave node and 2 master nodes.

Compatibility: COUP, LIAI



N1,N2,N3,N4 = nodes shell  
S = Slave node

#### Syntax:

```
"SH3D OPT 2"  
  ( /LECTURE/)
```

/LECTURE/

Reading procedure of the elements

#### Comments:

Note that in the "GEOM" directive the definition of the element must be in this order : N1 N2 S ie the slave node is defined after the two master nodes.

## 8.54 WEAK FLUID-STRUCTURE COUPLING 2 (FLSW)

### Object:

This directive allows to specify a “weak” coupling (LINK DECO) between a fluid and a structure modelled by topologically independent meshes. It is similar to FLSR (see page D2.143) but uses a weak approach (direct application of the fluid pressure onto the structure) rather than a strong approach (constraint on velocity imposed by Lagrange multipliers).

The present FLSW directive is only intended for use with cell-centered Finite Volumes (CCFV) modeling of the fluid. For a certain period, it was also possible to use FLSW with Finite Element (FE) modeling of the fluid. In this case, the code internally used a decoupled master/slave approach version of the FLSR algorithm. However, this possibility was considered misleading and it was removed as of July 2019 (the code gives an error message and stops if one tries to use FE in the FLSW fluid domain). Note, however, that the decoupled master/slave approach version of the FLSR algorithm can be (directly) accessed via the LINK DECO FLSR directive, see page D.143. Therefore, in order to repeat an old calculation that used the LINK DECO FLSW command with a FE fluid mesh, just replace it by LINK DECO FLSR.

The fluid mesh may be either fully general (unstructured) or regular (structured), as specified by the STFL directive described on page C.68. In the latter case, the search operations are faster. The COMP STFL directive produces by default a Finite Element regular mesh for the fluid domain (which is not suited for use in conjunction with the present FLSW model). To create a regular cell-centred Finite Volume mesh instead, for use with FLSW, add the extra VFCC keyword to the COMP STFL directive (see page C.68).

The FSI coupling is realized between structural points (ultimately, structural nodes) on one side, and *fluid entities* on the other side. The nature of the fluid entities depends upon the chosen options. They are fluid *cell centroids* if the VOLU keyword (or nothing) is specified (this is the default), while they are fluid *cell interfaces* if the FACE keyword is specified (see below for details).

As indicated by the brackets in the syntax, the STRU data block can be repeated at will, in order to define one or more FSI interaction (structure) zones, each with its own set of parameters. The STRU /LECTS/ keyword *must* be the first one of each zone being defined.

Compatibility: DECO

### Syntax:

```
FLSW  |[ FLUI /LECTF/ ; STFL ]|
      $[ HGRI hgri ; NMAX nmax ; DELE dele ]$
      <DGRI>
      <VOLU ; FACE>

      ( STRU /LECTS/
        $[ R r ; GAMM gamm ; PHIS phis ; GAMI gami ]$
        <BFLU bflu> <FSCP fscp>
        <ADAP LMAX lmax <SCAL scal> >
      )
```

## Basic fluid-related parameters

### FLUI

The fluid mesh to be coupled with the structure is fully general (unstructured). The concerned elements are specified next.

/LECTF/

List of fluid **elements** concerned. The fluid mesh is unstructured.

### STFL

The fluid mesh to be coupled with the structure is regular (structured). The concerned **elements** (volumes) need not be specified. In fact, they are simply the elements (volumes) generated by the **COMP STFL** directive described on page C.68, which must in this case have been specified previously in the input file. Since by default the **COMP STFL** directive produces Finite Elements for the fluid domain, make sure to add the **VFCC** optional keyword to that directive (see page C.68), so that cell-centered Finite Volumes are created instead.

## Fast search of coupled fluid entities

The next three keywords (**HGRI**, **NMAX** or **DELE**) are used to determine the size of the spatial grid used for the *fast search* of fluid entities (nodes, or cell interfaces if the **FACE** keyword is specified, see below) contained within the influence domain of the structure. Fast search speeds up the calculation and is *absolutely essential* in medium and even more in large size simulations. For this reason, fast search is *always active* in the present FSI model. Note that this may be unlike other types of search in EPX. For example, in the pinball contact model (**PINB**) fast search of pinballs contact is *not* active by default (an option has to be activated).

By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DELE** are specified, the code takes **DELE 1.01**.

A (regular) spatial grid is built up and used for the fast search. The fluid entities (centroids or interfaces) contained in a cell of the search grid are tested for inclusion in the structural influence subdomains contained either in the same cell or in a direct neighbour cell (there are up to 8 such cells in 2D, up to 26 cells in 3D). The cell grid can be optionally dumped out on the listing by the **DGRI** keyword.

For the calculation to be as fast as possible, the fast search grid must have the minimum size ensuring correctness of results, i.e. such that a (barely) sufficient number of interacting entities is detected, and thus no spurious fluid passage occurs across the structure. If  $h_F$  denotes the size of the fluid mesh and  $h_S$  the size of the structure mesh, then the grid size  $h_G$  must be:

$$h_G = \phi \cdot \max(h_F, h_S) \quad (46)$$

where  $\phi > 1$  is a safety factor. A value  $\phi = 1.01$  should be sufficient. Since a single grid is used for the search over the whole computational domain,  $h_F$  and  $h_S$  in the above expression must be the *maximum* sizes of the fluid and structural elements which are susceptible of interacting, i.e. which belong to the /LECTF/ set defined above and to the /LECTS/ set(s) defined below, respectively.

In calculations *without adaptivity* one has normally  $h_F < h_S$  for accuracy reasons (especially if shells are used to discretize the structure), so that the grid size is (normally) dictated by the largest coupled structural element. For the case of adaptive calculations, see the Remarks at the end of this manual page.

### HGRI

Specifies the size of the fast search grid cell. Each cell has the same size in all spatial directions and is aligned with the global axes.

**NMAX**

Specifies the maximum number of cells along one of the global axes.

**DELE**

Specifies the size of the fast search grid cell as a multiple of the length of the largest coupled **structural** element. Element “diameters” are computed only along each global spatial direction and the maximum is taken. For example, by setting **DELE 2** the size of the cell is two times the length of the largest coupled structural element. By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DELE** are specified, the code takes **DELE 1.01**.

**DGRI**

Dump out the grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed.

**Additional search parameters**

Next come some additional parameters for the geometric search.

**VOLU**

*For use with Cell Centered Finite Volumes only.* The search for fluid entities “contained” in the influence domain of the structure is based upon the element volume, more precisely on the position of the element centroid. This is the default.

**FACE**

*For use with fluid Cell Centered Finite Volumes only.* The search for fluid entities “contained” in the influence domain of the structure is based directly upon the “faces” (interfaces) between neighboring cells. The centroid of the face is considered rather than the centroid of the finite volume. In this case, there is no difference between using **BFLU 1** or **BFLU 2**, see below. However, please note that by omitting **BFLU** or by specifying **BFLU 0** (the default value for **BFLU**) no numerical fluxes are blocked. So, if **FACE** is used and fluxes must be blocked (as is normally the case), one must specify either **BFLU 1** or **BFLU 2** (with no difference in the results).

**Structural influence domain(s)**

The following block of data, introduced by the keyword **STRU**, can be repeated any number of times (but it must be specified at least once) to define one or more FSI zones, each with different interaction parameters. For each such zone:

**STRU**

Introduces the structure mesh to be coupled with the fluid. The concerned elements are specified next.

**/LECTS/**

List of structural **elements** concerned. All their nodes must be declared as Lagrangian.

The next four keywords (**R**, **GAMM**, **PHIS** or **GAMI**) are used to set the **size** (thickness) of the **structural influence domain** surrounding the structure elements defined above by **/LECTS/**. All fluid entities as defined above (cell centroids or cell interfaces) contained within this influence domain will be coupled to the structure.

Therefore, the correct size of the influence domain is related to the size of the fluid mesh in the vicinity of the embedded structure. On one hand, if the influence domain is too thin, then some interactions between the structure and the fluid entities might be overlooked, thus resulting in spurious passage of fluid across the structure (*leakage*). On the other hand, if the influence domain is too thick, too much fluid will be interacting with the structure (excessive added mass effect). The optimal value is then the minimum value which ensures structure tightness (no leakage).

By default, i.e. if neither **R** nor **GAMM** nor **PHIS** nor **GAMI** are specified, the code performs an automatic determination of influence spheres at each coupled structural node by using the default value of **GAMM** ( $\gamma = 1.01$ ). For the choice of **R**, **GAMM**, **PHIS** or **GAMI** in adaptive calculations see the **ADAP** keyword below and the comments at the end of this page.

## R

Prescribed (fixed) radius  $R$  of influence spheres at each coupled structural node. In the special, but frequent, case of a uniform structured fluid mesh (uniform square or cube elements) it is suggested to take  $R$  slightly larger than the semi-diagonal of a fluid element. This means that, for a 2D uniform square fluid mesh of side  $L_\Phi$  one should take  $R = 0.71L_\Phi$  while for a 3D uniform cube fluid mesh of side  $L_\Phi$  one should take  $R = 0.87L_\Phi$ .

## GAMM

Coefficient  $\gamma$  for the automatic determination of influence spheres at each coupled structural node, based on the size of the *enclosing fluid element* (which must thus be found by the code by means of a fast search algorithm, see the remarks at the end of this manual page). The sphere radius is  $R = \gamma R_F = \gamma \delta L_\Phi$  where  $L_\Phi$  is the local length (size) of the fluid mesh,  $\delta$  is a coefficient related to the space dimension  $d$  of the problem ( $\delta = \frac{\sqrt{d}}{2}$ , i.e. about 0.71 in 2D and about 0.87 in 3D calculations). The quantity indicated as  $R_F$  above is the “natural” size of the sphere radius, i.e. the radius of a sphere (circle in 2D) which exactly encompasses all nodes of a regular element (regular cube in 3D or regular quadrilateral in 2D). By default it is  $\gamma = 1.01$ . This value should ensure “tightness” of the structure, at least for a regular mesh. By increasing the value, tightness is safer but the amount of fluid “attached” to the structure also increases. By decreasing the value, some local spurious passage of fluid across a solid structure might occur.

## PHIS

Coefficient  $\phi_s$  for the automatic determination of influence spheres at each coupled structural node. The sphere radius is equal to  $\phi_s$  times the minimum structural element length at the concerned node. By default it is  $\phi_s = 0.3$ . This option should be rarely used. It is advisable to use **GAMM** instead.

## GAMI

Same as **GAMM** but radius is computed only at the initial step, that is, the radius is *not* updated during the calculation. This may be convenient (to save some CPU) in calculations with Eulerian fluid mesh (that never changes) and without adaptivity.

## Additional flux and coupling parameters

Next come some additional parameters for the fluxes and for the type of coupling.

### BFLU

Type of treatment of numerical fluxes (density and energy, but **not** momentum) in fluid models, when used in conjunction with the present **FLSW** directive. The value 0 (default) indicates that fluxes are freely computed. The value 1 indicates that fluxes are blocked between two fluid entities which are both within the influence domain of the structure. The value 2 indicates that fluxes are blocked between two fluid entities of which at least one lies within the influence domain of the structure. If the **FACE** keyword has been specified (see above), there is no difference between using **BFLU** 1 or **BFLU** 2. However, please note that by omitting **BFLU** or by specifying **BFLU** 0 (the default value for **BFLU**) no numerical fluxes are blocked. So, if **FACE** is used and fluxes must be blocked (as is normally the case), one must specify either **BFLU** 1 or **BFLU** 2 (with no difference in the results).

### FSCP

Type of coupling between fluid entities and corresponding structural points, when used in conjunction with the present **FLSW** directive. The value 0 (default) indicates that coupling occurs only in the direction normal to the structure. The value 1 indicates that coupling occurs along all spatial directions.

## FSI-driven adaptivity

Finally, there are some optional keywords related to automatic (FSI-driven) adaptivity of the fluid mesh near the structure.

### ADAP

Activates mesh adaptivity for automatic refinement and un-refinement of the fluid mesh specified by **/LECTF/** in the vicinity of the structure specified by **/LECTS/**. Note that this type of mesh adaptivity is at the moment incompatible with other types of adaptivity such as those activated by the **WAVE** or **INDI** directives.

### LMAX

Introduces **lmax**, the desired maximum adaptive refinement level  $L^{\max}$  of the fluid mesh in the vicinity of the structure. This value should be greater than 1, since level 1 is attributed to the base mesh (no refinement). Each level corresponds to a halving of the mesh size with respect to the immediately previous level.

### SCAL

Introduces **scal** ( $s$ ), an optional scaling factor to be used in the determination of fluid elements to be refined. By default **scal** is equal to 1. When scaling the structural influence domain by successive powers of two in order to identify, at each refinement level, the fluid elements to be refined or un-refined, the code finally multiplies the result by this factor. Using a value of  $s$  greater than one, e.g. 1.5 or 2, correspondingly enlarges the zone of fluid mesh around the structure which is refined and this may result in a smoother mesh transition (for example, as an alternative to the option **OPTI ADAP RCON**). Note, however, that  $s$  has no influence on the size of the structural influence domain used for the final search of fluid entities (fluid nodes or fluid cell interfaces) interacting with the structure. This search is always done by the smallest influence domain  $R_{L_{\max}} = R_1/2^{L_{\max}-1}$ , i.e. *without* taking into account the  $s$  factor.

In **FSI adaptive calculations**, the size of the structural influence domain specified in input by **R**, **GAMM** or **PHIS** (**GAMI** is not appropriate with adaptivity) is related to the *base* (i.e. the coarsest) fluid mesh size, not to the refined one (for the user's convenience) and is then scaled automatically by the code whenever necessary, up to the maximum chosen refinement value given by the **ADAP LMAX** keyword. Therefore, in order to try out different adaptive refinement levels in the vicinity of the structure the user needs only to change **LMAX** in the input directive (all other parameters **R** etc. remain the same).

In **FSI adaptive calculations**, that is when the **FLSW ADAP LMAX** optional keyword has been specified, one is **certain** that the fluid mesh in the vicinity of the structure will be constantly refined to the maximum level (minimum size) specified for the fluid (**LMAX**), given by:

$$h_F^{\text{refined}} = h_F^{\text{base}} / 2^{L_{\text{max}}-1} \quad (47)$$

For this reason, in the equation (46) for the determination of the grid size **HGRI** ( $h_G$ ) one can use  $h_F^{\text{refined}}$  instead of the base fluid mesh  $h_F^{\text{base}} = h_F$ , obtaining thus:

$$h_G = \phi \cdot \max(h_F^{\text{refined}}, h_S) \quad (48)$$

One should make sure to use (48) instead of (46) since it is likely to be  $h_F^{\text{refined}} < h_S$ , while it is typically  $h_F > h_S$ , so this may lead to important savings of CPU time.

### Remarks:

In case of automatic determination of influence spheres based on the **GAMM** keyword in conjunction with an **unstructured** fluid grid, a fast search over the coupled fluid elements is needed in addition to the normal fast search over the coupled structural elements. Scope of this second search is to determine, for each structural node, which is the fluid element currently containing the node. For this purpose, the code uses a fast search algorithm by means of the same parameters (**DGRI**, **HGRI**, **NMAX**, **DELE**) specified above for the search over structural elements. Note, however, that as concerns this second search if **DELE** is specified it refers to the size of the fluid element rather than to the size of the structural element. However, if a **structured** fluid grid is specified, then no additional search is needed because the containing fluid element can be detected directly.

Make sure you consult the additional **options** related to the functioning of the **FLSW** model in pages H.155 and H.160.

### References

The **FLSR** model is similar to **FLSW** in many aspects. It was first described in report [317]. Improvements to the model were proposed in reference [874].



## 8.55 NODE ON FACET ELEMENT

### Object:

Note that the "MAPi" directive ( $i = 2,..7$ ) is a subdirective of the "LIAI" directive but it is needed to define an element which defines the nodes (master and slave) of the liaisons. It is listed in this Section because it consists in the definition of kinematic constraints between the dof of one slave node and master nodes.

The purpose is to glue one slave node to a master face. It can be used in 2-D (the face is a line) or in 3-D.

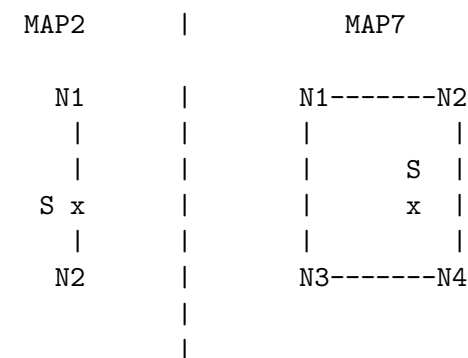
Compatibility: COUP, LIAI

Different cases can be used and are listed below.

Name	Dimension	Npt	Dof	Nb. of liaisons	Remarks
MAP2	2	3	2	2	point on solid line
MAP3	3	4	3	3	point on triangular solid facet
MAP4	3	5	3	3	point on quadrangular solid facet
MAP5	2	3	3	3	point on 2D shell line
MAP6	3	4	6	6	point on triangular shell facet
MAP7	3	4	6	6	point on quadrangular shell facet

### Note:

In 3-D the slave node S should be on the face.



N1,N2,N3,N4 = Master nodes  
S = Slave node

### Syntax:

"MAPi"  
( /LECTURE/ )

/LECTURE/

Reading procedure of the elements.

**Comments:**

Note that in the "GEOM" directive the declaration of the element must be in this order : S N1 N2 (N3 N4), ie the slave node is the first node of the list.

## 8.56 FINITE-ELEMENT/SPECTRAL-ELEMENT INTERFACE

### Object:

This directive allows to specify the interface between a Finite Element domain and a Spectral Element domain in a coupled analysis.

It replaces the former principal directive FESE, which is no longer accepted. The difference is that FE/SE interfacing is now coupled with any other (coupled) links specified in the calculation (LINK COUP), while formerly the FE/SE interface conditions were treated as a separate set of conditions.

Compatibility: COUP

### Syntax:

```
"FESE" "FNOD" /LECT1/  
      "SNOD" /LECT2/
```

/LECT1/

List of Finite Element nodes along the FE/SE interface.

/LECT2/

List of (micro) Spectral Element nodes along the FE/SE interface.

### Remarks:

The model is quite general and accepts the node lists in any order. It is even possible to define interfaces formed by several disjoint lines (or surfaces, in 3D).

The only restriction is that FE and (micro) SE nodes must lie with sufficient precision on the interface, which is defined geometrically by the macro Spectral Element faces.

Furthermore, note that to every macro Spectral Element node on the interface, there must exist one and only one FE node in LECT1 that has the same coordinates. This is necessary in order to ensure that to every FE face on the interface there correspond one and only one opposite macro Spectral Element face (the reverse is not true, in general).

## 8.57 NAVIER-STOKES (INCOMPRESSIBILITY)

### Object:

This directive allows to specify an incompressible or quasi-incompressible behaviour for selected fluid elements. These elements must possess the LIQU material (see page C.390).

It replaces the former NAVI problem type directive (see page A.30) which automatically generated liaison conditions for all elements containing a LIQU material.

Compatibility: COUP

### Syntax:

```
"NAVI" /LECT/
```

```
/LECT/
```

List of Finite Elements concerned. These must possess the LIQU material.

### Remarks:

With this directive, it is not allowed to specify the NAVI keyword in the problem type. Use either the old (NAVIER problem type) directive or the present one, but not together in the same run.

For the moment, only elements of type CAR1, TUBE and TUYA are accepted.

Be aware the verification of a link of type NAVI, as activated by the optional keyword VERI of the LINK directive (see page D2.10), makes sense only when the corresponding LIQU material is perfectly incompressible. In fact, when the material is (even slightly) compressible, as indicated by a finite sound speed C specified in the material parameters, an extra term is added to the diagonal of the assembled links matrix during the solution process. Therefore, it is normal that the original link specification does not hold any more.

## 8.58 PIPELINE RUPTURE CONNECTION

### Object:

This FSI model allows modelling a break of a pipeline discretized with TUYA elements. Prior to the pipeline rupture instant, the conservation of the internal fluid mass flow rate and the continuity of the mechanical degrees of freedom are ensured.

Compatibility: COUP

### Syntax:

```
"BREC"    < "TRUP" trup >    /LECTURE/
```

trup

Rupture instant (no breaking by default).

/LECTURE/

Number or the name of the BREC element.

### Comments:

This directive may only be used to connect two TUYA elements.

### Outputs:

The components of the ECR table are as follows:

ECR(25): pipeline rupture area (water)

ECR(26): mass flow (water)

ECR(27): total ejected mass (water)

## 8.59 SURFACE PRESSURE MEASURED IN AN ELEMENT (PELM)

### Object:

This directive allows to apply the value of ECRO 0 (known as pressure in a fluid) on structural facets (referred to as slave facets), which is measured in a given fluid element of the model (referred to as master element).

It is typically useful when a cavity is modelled by an equivalent pipe network instead of a full 3D mesh, but the pressure on its structural envelop must still be taken into account. Reference element would then be one of the TUBE or TUYA elements used for the cavity and the facets the structural envelop. Compatibility: DECO

### Syntax:

```
PELM  ( MAIT /LECTURE/  
        ESCL /LECTURE/  
        < NOEX /LECTURE/ >  
        < |[ INTE ; EXTE ]| /LECTURE/ >  
        < PREF pref >  
        < VFCC > )
```

#### pref

Reference pressure.

#### VFCC

Option to signal that the fluid is computed with VFCC

### Comments:

Only one master element must be provided for each set of slave facets.

If slave elements are 3D continuum elements, pressure is applied on any of their free facets, along the inward normal direction.

If slave elements are 3D shell elements, keywords INTE or EXTE are used to enter a node defining the internal or external side of the structure respectively and again, pressure is applied along the inward normal direction. Option NOEX allows excluding some slave nodes from applying pressure.

No retroaction occurs from the structure onto the fluid element, which is licit only in the case of a large cavity which imposes its pressure and for limited structural displacements.

The way pressure is applied on the slave nodes in 3D depends on the method used to model the fluid. In FE, the pressure repartition is related to the shape function of the element whereas in FV, the pressure is basically divided by the number of nodes. Then option VFCC is used to tell Europlexus that the fluid is modelled with the Finite Volume Method and then the pressure of the fluid on the structure should be applied accordingly.

## 8.60 UNCOUPLED HANGING LINKS

**Object:**

This directive allows to toggle uncoupled master/slave algorithm to handle hanging links with ADAPTIVITY instead of the fully coupled Lagrange Multipliers approach.[MPI only].

Compatibility: DECO

**Syntax:**

ADAP

**Comments:**

No further subdirective is currently needed.



## 8.61 PRESCRIBED DAMAGE FOR GRADIENT DAMAGE MATERIALS

### Object:

This directive defines imposed damage values for gradient damage materials **ENGR**, see [7.7.21](#). Concretely this routine will simply update the lower and upper bounds of the damage minimization problem.

Compatibility: COUP, DECO

### Syntax:

```
"ENGR" ( alpha0 /LECTURE/ )
```

**alpha0**

Prescribed damage value.

**LECTURE**

List of the nodes concerned.

## 8.62 DRAG FORCES ON 3D BEAMS/BARS EMBEDDED IN A FLUID

### Object:

This directive allows to model the fluid-dynamic drag forces acting on 3D beam/bar elements embedded in a fluid. Such elements are represented by 2-node segments (SEG2 geometrical shapes). The model is somewhat similar to the flying debris model described on Page C.66.

The fluid surrounding the 3D beams/bars may be modeled either as a uniform field with constant properties (velocity, density), or as an evolving fluid field, discretized by Finite Elements or Finite Volumes, or even as a combination of the two (e.g., FE fluid field near the explosive source, uniform field far away).

The drag pressure acting on a 3D beam/bar element (or a part thereof) is computed according to the following expression:

$$p_D = C_D \frac{1}{2} \rho_F (v_R^\perp)^2$$

where  $C_D$  is the drag coefficient (an empirical number given by the user),  $\rho_F$  is the local density of the fluid and  $v_R^\perp$  is the component of the relative (fluid minus structure) velocity in the plane normal to the 3D segment. Then, the drag force  $F_D$  exerted by the fluid on the structure is computed by multiplying the drag pressure by the exposed area of the segment:

$$F_D = p_D A = p_D L D$$

with  $L$  the length of the segment (or part thereof) and  $D$  its diameter.

One may also activate a feed-back mechanism whereby the drag forces generated by the fluid on the structure are applied (with the minus sign) to the fluid itself. This is currently the default when the fluid is discretized by FE (but not by VFCC). Note, however, that in general it is preferable to deactivate the feedback mechanism (see option NOFB below), since it perturbs the fluid flow, while the drag formula (whose coefficient  $C_D$  is determined experimentally) assumes an unperturbed fluid flow. In other words, in order to compute the relative velocity  $v_R$  to be introduced in the formula, the fluid velocity should be taken at a certain distance from the structure, where the effect of the structure is not felt. Furthermore, note that at the moment feedback forces are only applied (by default, i.e. without the NOFB option) when the fluid is discretized by Finite Elements (FE). They cannot be applied (with or without NOFB) to a fluid discretized by Cell-Centred Finite Volumes (VFCC).

In practical applications, the fluid mesh is typically (much) finer than the structural mesh composed of 3D structural members such as beams or bars. Therefore, wherever appropriate, to improve the accuracy of the drag force calculation each structural segment is subdivided into several parts, each one of the length of a typical fluid element.

### Syntax:

```
DRAG <ROF rof> <VFX vfx> <VFY vfy> <VFZ vfz>
      STRU /LECTS/
      <FLUI /LECTF/>
      <(HF hf /LECTS2/)>
      (CD cd /LECTS3/)
```

```
< $ HGRI hgri ; NMAX nmax ; DELE dele $ <DGRI> >  
< $ FBAC ; NOFB $ >
```

**rof**

Density of the (default) uniform fluid field in which the 3D beams/bars are embedded. This value is 0.0 by default, meaning that by default the 3D beams/bars move in vacuum (if they are not coupled with a discretized fluid domain by the **FLUI** keyword). Note that the drag force acting on a 3D beam/bar depends on the density but also on the drag coefficient (**CD**, see below).

**vfx, vfy, vfz**

Components of the velocity of the surrounding uniform fluid field. These values are 0.0 by default.

**STRU**

Introduces the **/LECTS/** of the 3D beam/bar elements. Only elements with a **SEG2** geometrical shape are accepted.

**FLUI**

Introduces the **/LECTF/** of the discretized fluid domain with which the 3D beams/bars motion should be coupled. When a 3D beam/bar traverses this domain, the (local) fluid velocity and density are automatically computed by the code, instead of using the constant user-given values **rof**, **vfx**, **vfy**, **vfz** described above. A fast search algorithm based on a grid of cells (as in bucket sorting) is used to compute the fluid element (if any) encompassing each 3D beam/bar element.

**HF hf**

Size of the fluid elements coupled with each of the 3D beam/bar elements specified in the following **/LECTS2/** directive in order to compute the local drag pressure and then the local drag force. As indicated by the parentheses, the **HF hf** sub-directive may be repeated as many times as necessary to assign a size **hf** to each one of the structure elements previously listed in the **STRU** directive. If omitted, the code computes automatically the local size of the coupled fluid elements to each structural beam/bar element.

**CD cd**

Drag coefficient  $C_d$  assigned to the 3D beam/bar elements specified in the following **/LECTS3/** directive. As indicated by the parentheses, the **CD cd** sub-directive must be repeated as many times as necessary to assign a drag coefficient **cd** to each one of the structure elements previously listed in the **STRU** directive.

**HGRI**

Specifies the size of the grid cell for fast search operations. Each cell has the same size in all spatial directions and is aligned with the global axes. Note that the size of this grid is related to the size of the **structural** elements specified in **STRU**, not of the fluid elements specified in **FLUI**.

**NMAX**

Specifies the maximum number of cells along one of the global axes.

**DELE**

Specifies the size of the grid cell as a multiple of the diameter of the largest coupled **structural** element. Element “diameters” are computed only along each global spatial direction and the maximum is taken. For example, by setting **DELE 2** the size of the cell is twice the diameter of the largest coupled structural element declared in the **FLUI** directive. By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DELE** are specified, the code takes **DELE 1.01**.

**DGRI**

Dump out the initial grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed, that is, at step 0.

**FBAC**

Activate the feed-back mechanism whereby the drag forces generated by the fluid on the structure are also applied (with the minus sign) to the fluid itself. This is the default, although in many cases it is preferable to deactivate the feedback mechanism, since it perturbs the fluid flow while the drag formula assumes an unperturbed fluid flow (the fluid velocity should be taken at a certain distance from the structure, where the effect of the structure is not felt). Furthermore, note that at the moment feedback forces can only be applied when the fluid is discretized by Finite Elements (FE), not by Cell-Centred Finite Volumes (VFCC).

**NOFB**

Deactivate the feed-back mechanism whereby the drag forces generated by the fluid on the structure are also applied (with the minus sign) to the fluid itself. If the fluid is modelled by VFCC, activating or not the feedback mechanism has no influence on the solution because the feedback is not available for this type of fluid discretization.

## 9 GROUP E—FUNCTIONS AND INITIAL CONDITIONS

### Object :

The directives described in this Section allow to introduce functions in a variety of forms, necessary for the description of materials, loads, or initial conditions of a calculation.

We distinguish functions of the form  $y = f(x)$  (directive **FONC**, page E.15), from the functions depending from a parameter  $p$ , which are of the form  $y = f(x, p)$  (directive **ABAQ**, page E.30).

The abscissa  $x$  will most often be the time  $t$  in case of a load function, but it is possible to use arbitrary variables for  $x$ ,  $y$  or  $p$ .

A special energy injection model (developed at JRC) is also described on page E.38.

## 9.1 FUNCTIONS

### Object :

This directive defines functions in the form  $y = f(x)$ . These may be used e.g. for imposed motions or other conditions which depend upon time, and also to define material properties.

### Syntax :

```
"FONC" ( < "NUM" > ifonc |[ "TABL" npts*(xi,yi)          ;
                             "ROUT" <"PARA" n p1 p2 ... pn>  ;
                             < "LSQU" deg > "TABL" npts*(xi,yi) ;
                             "HARM"  nhar*(C c <TYPE type>
                                           $[ OMEG omeg ; FREQ freq ]$
                                           $[ PHIR phir ; PHID phid ]$
                                           <TMIN tmin> <TMAX tmax>)]| )
```

#### "NUM"

Optional keyword introducing the number of the function.

#### ifonc

Number of the function necessary to identify it when it will be used.

#### "TABL"

The function is defined by a table (sequence of pairs). See below page E.20.

#### "ROUT"

The function is computed by means of the subroutine **TABANA** written by the user. See page E.25. Optionally, a list of parameters may be passed to the routine.

#### "LSQU"

The function is defined by a table (sequence of pairs). See below page E.20. But a least square polynomial fitting is performed in order to store the function as a polynomial. The fitting quality can be controled in the listing by the mean of standard deviation. If the least square polynomial fitting fails, the function is stored as a table and the computation goes on.

#### deg

Maximum degree ( $\text{deg} > 0$ ) of the polynomial which fits the table function. Be careful not to use too high degrees: it does not give good results because of too many oscillations.

#### "HARM"

The function is defined as a combination of harmonic functions.

### Comments :

The key-word **FONC** may appear at most once, at the beginning of the sequence relative to the functions.

**Warning :**

If there are imposed displacements, dimension also **FCOE**, see page A.80.

### 9.1.1 TABLE FUNCTION

**Object :**

To define a function  $y = f(x)$  by the means of couples of points.

**Syntax :**

```
"TABL"  npts*( xi , yi )
```

**npts**

Number of couples (xi, yi) defining the table.

**xi , yi**

Abscissa (time for example) and value (ordinate) of the function at point i.

**Comments :**

The value of the function at time  $t$  (or at the abscissa  $x$ ) is determined by a linear interpolation.



### 9.1.2 SUBROUTINE *TABANA*

#### Object :

To define a function by the means of a subroutine written by the user. Optionally, a list of parameters may be passed to the routine.

#### Syntax :

"ROUT" <"PARA" n p1 p2 ... pn>

"PARA"

Optional keyword introducing a list of parameters to be passed to the routine. These parameters will be made available in subroutine *TABANA* by means of the **COMMON** /CTABANA/.

n

Number of parameters to be passed. The maximum is 10.

p1, p2, ... pn

The n parameters to be passed.

#### Comments :

The user has to write a subroutine which computes the function at every time.

```

SUBROUTINE TABANA(IFONC,T,COEF,DERIV)
*
*   cette routine permet d'entrer une table sous forme analytique
*
*   ifonc      : numero de la fonction
*   t          : temps du calcul en sec. ou + generalement abscisse
*   coef       : valeur de la fonction ifonc au temps ou abscisse t
*   deriv      : derivee de la fonction ifonc en t (seulement pour courbe
*               de traction du materiau lem2)
*
* attention ! la fonction 2 est utilisee par le benchmark :
*               bm_rob_smr
*
*   USE M_FUNCTIONS
*
*   IMPLICIT NONE
*
*---  variables globales :
INTEGER, INTENT(IN) :: IFONC
REAL(8), INTENT(IN) :: T
REAL(8), INTENT(OUT) :: COEF, DERIV
*
*---  variables locales :
REAL(8) :: TO,T1,T2,X,XV,ALPHA,ALPHAV,XF,ALPHAF,TF,Y2
REAL(8) :: F_0, T_BAR, TAU, TT, VV, FAC, V_SUR_F
REAL(8), PARAMETER :: PIGR = 3.14159265359D0
*
REAL(8), EXTERNAL :: FOLCO1
*
DERIV=0.D0
*
SELECT CASE (IFONC)
*
CASE(1) ! 0 AVANT TO, RAMPE DE TO A T1, 1 APRES T1
  TO = 0D0
  T1 = 5D-3
  IF(T < TO) THEN
    COEF = 0
  ELSEIF(T >= TO .AND. T <= T1) THEN
    COEF = T/T1
  ELSE
    COEF = 1

```

```

      ENDIF
*
      CASE(2) ! CAS DU BENCH
        T0 = 0D0
        T1 = 0.5D0
        T2 = 1.5D0
        IF(T <= T0) THEN
          COEF = 0D0
        ELSEIF(T > T0 .AND. T <= T1) THEN
          COEF = 673D0*T - 508D0
        ELSEIF(T > T1 .AND. T <= T2) THEN
          COEF = 148D0*EXP(-5.5D0*(T-T1)) + 8D0
        ELSEIF(T > T2) THEN
          COEF = 240D0
        ENDIF
*
      CASE (3) ! FC FOR PARTITIONS PAPER: applied sinusoidal force
*
* p_tabana(1) = f_0      max. value of applied sinusoidal force
* p_tabana(2) = t_bar    period of applied sinusoidal force
*
      IF (N_TABANA < 2) THEN
        CALL ERRMSS ('TABANA', 'TOO FEW PARAMETERS ENTERED')
        STOP 'TABANA : N_TABANA < 2'
      ENDIF
      F_0 = P_TABANA(1)
      T_BAR = P_TABANA(2)
      COEF = FOLCO1 (F_0, PIGR, T, T_BAR)
*
      CASE (4) ! FC FOR PARTITIONS PAPER: velocity for sinusoidal force
*
* p_tabana(1) = f_0      max. value of applied sinusoidal force
* p_tabana(2) = t_bar    period of applied sinusoidal force
* p_tabana(3) = tau      traversal time of bar length
* p_tabana(4) = v_sur_f  ratio between v and F (=c/SE)
*                        (v = velocity, F = applied force,
*                        c = sound speed, S = bar cross-section,
*                        E = Young's modulus)
*
      IF (N_TABANA < 4) THEN
        CALL ERRMSS ('TABANA', 'TOO FEW PARAMETERS ENTERED')
        STOP 'TABANA : N_TABANA < 4'
      ENDIF
      F_0 = P_TABANA(1)
      T_BAR = P_TABANA(2)
      TAU = P_TABANA(3)
      V_SUR_F = P_TABANA(4)
      COEF = FOLCO1 (F_0, PIGR, T, T_BAR)
      FAC = -2.D0
      TT = T - (TAU+TAU)
      DO WHILE (TT >= 0.D0)
        VV = FOLCO1 (F_0, PIGR, TT, T_BAR)
        COEF = COEF + FAC*VV
        FAC = -FAC
        TT = TT - (TAU+TAU)
      END DO
      COEF = V_SUR_F*COEF
*
      CASE (5) ! FC FOR PARTITIONS PAPER: velocity for constant force
*
* p_tabana(1) = f_0      max. value of applied sinusoidal force
* p_tabana(2) = t_bar    period of applied sinus. force (unused here)
* p_tabana(3) = tau      traversal time of bar length
* p_tabana(4) = v_sur_f  ratio between v and F (=c/SE)
*                        (v = velocity, F = applied force,
*                        c = sound speed, S = bar cross-section,
*                        E = Young's modulus)
*
      IF (N_TABANA < 4) THEN
        CALL ERRMSS ('TABANA', 'TOO FEW PARAMETERS ENTERED')
        STOP 'TABANA : N_TABANA < 4'
      ENDIF
      F_0 = P_TABANA(1)
      TAU = P_TABANA(3)
      V_SUR_F = P_TABANA(4)
      COEF = F_0
      FAC = -2.D0
      TT = T - (TAU+TAU)
      DO WHILE (TT >= 0.D0)
        VV = F_0
        COEF = COEF + FAC*VV
        FAC = -FAC
        TT = TT - (TAU+TAU)
      END DO
      COEF = V_SUR_F*COEF
*
      CASE DEFAULT
        CALL ERRMSS('TABANA',
&      'VOUS AVEZ APPELE LE SS-PROGRAMME TABANA SANS LE CREER ')
        STOP 'TABANA NON ECRIT !'
      END SELECT
*
      END SUBROUTINE TABANA
=====
      REAL(8) FUNCTION FOLCO1 (F_0, PIGR, T, T_BAR)
*
      IMPLICIT NONE
*
      REAL(8), INTENT(IN) :: F_0, PIGR, T, T_BAR

```

```
*  
      FOLC01 = 0.5D0*F_0*(1.D0+SIN(PIGR*((2.D0*T/T_BAR)-0.5D0)))  
*  
      END FUNCTION FOLC01
```

The arguments have the following meaning:

IFONC : number of the function (input);

T : computing time (input);

COEF : value of the function at time (abscissa) T (output);

DERIV : value of the function derivative at the abscissa T (output). This is mandatory for some materials.

### **Warning :**

It is strongly advised to foresee adequate error messages, like in the above example on the function number.

### 9.1.3 HARMONIC FUNCTION

#### Object :

To define a harmonic function of the form (sum of  $n$  terms):  $y = C_1 \cos(\omega_1 t + \phi_1) + C_2 \cos(\omega_2 t + \phi_2) + \dots$ , where for the  $i$ -th term  $C_i$  is a coefficient,  $\omega_i$  is the pulsation in rad/s and  $\phi_i$  is the phase in rad. Note, however, that if the user prefers, the frequency  $f$  (in Hz) can be specified, in place of  $\omega$ . Also, the phase can be specified in degrees if so preferred.

#### Syntax :

```
HARM nhar*( VHAR
              C c <TYPE type>
              $OMEG omeg ; FREQ freq$
              <$PHIR phir ; PHID phid$> )
              <TMIN tmin> <TMAX tmax>
```

**nhar**

Number of terms in the sum of harmonic functions.

**VHAR**

Mandatory keyword that introduces the reading of the set of values for the  $i$ -th term.

**c**

Coefficient of the  $i$ -th harmonic term.

**type**

Type of the  $i$ -th harmonic term: 1 means sine, 2 means cosine. By default it is 1 (sine).

**omeg**

Pulsation (angular frequency)  $\omega$  of the  $i$ -th harmonic term in rad/s. Recall that it is  $\omega = 2\pi f$  where  $f$  is the frequency in Hz.

**freq**

Frequency  $f$  of the  $i$ -th harmonic term in Hz. Recall that the pulsation (angular frequency) is then  $\omega = 2\pi f$  in rad/s.

**phir**

Phase  $\phi$  of the  $i$ -th harmonic term *in radians*.

**phid**

Phase  $\phi$  of the  $i$ -th harmonic term *in degrees*.

**tmin**

Time  $t_{\min}$  at which the harmonic function (all terms) starts. The function is 0 for  $t < t_{\min}$ . By default, the function acts over the entire time scale.

tmax

Time  $t_{\max}$  at which the harmonic function (all terms) ends. The function is 0 for  $t > t_{\max}$ .  
By default, the function acts over the entire time scale.

**Comments :**

The value of the function at time  $t$  (or at the abscissa  $x$ ) is determined by computing the above analytical expression.

If both PHIR and PHID are omitted, a phase of 0 is assumed for the concerned harmonic component.

### 9.1.4 SWEEP SINE FUNCTION

**Object :**

To define a sweep sine function of the form:  $y = C \sin \left( \frac{1}{2\pi} \left( f_i + \frac{(f_e - f_i)t}{D} \right) t \right)$ , where  $f_i$  and  $f_e$  are initial and final frequencies,  $D$  is the sweep duration.

The function is extended for time greater than  $D$  with a regular sine with  $f_e$  frequency.

**Syntax :**

```
"SSWP"  "C"      coef
         "FINI"   freq_ini
         "FEND"   freq_end
         "SWPD"   duration
```

**coef**

Multiplicative coefficient.

**FINI**

Initial frequency.

**FEND**

Final frequency.

**SWPD**

Sweep duration.

### 9.1.5 ABAQUE : PARAMETRISED TABLE FUNCTION

**Object :**

To define a set of parametrised functions in the form  $y = f(x, p)$  where  $p$  is a parameter.

**Syntax :**

```
"ABAQ" ( "SET" ifonc "PARA" flot "TABL" npts*( ti , fi ) )
```

"SET"

Mandatory keyword to describe a parametrised function.

ifonc

Number of the function necessary to identify it when it will be used.

"PARA"

Announces the value of the parameter.

flot

Value of the parameter.

npts

Number of couples (ti, fi) defining the table relative to the parameter flot.

ti,fi

Abscissa and ordinate of the function at point i.

**Comments :**

The value of the function for an abscissa  $t$  is obtained by linear interpolation.

The "PARA" sequence must appear at least twice.

**Warning :**

Do not forget to dimension sufficiently (directives "FNOM" and "FTAB", page A.90).

## 9.2 ENERGY INJECTION HISTORY (JRC)

### Object

This instruction defines the energy injected in: i) fluid elements of types FLxx (JRC model) or CUBE, PRIS, TETR (CEA model), or ii) multicomponent fluid finite volumes (MCxx elements). This directive is extended to fluid finite volumes (MCxx elements) using the MCVO keyword.

### References

More information on the formulation of this model may be found in reference [\[131\]](#).

### Syntax

```
"INJE"  "QTAB" ifon  |[ "MASS" ; "VOLU" ;  
                        "MCMA" ; "MCVO" ] |  
                        /LECT/
```

#### ifon

Number of the function (see FONC) used to describe the variation in time of the total injected power.

#### MASS

This keyword applies only in the case of fluid elements, see i) above. The injected power will be distributed among the different elements that form the energy injection zone proportionally to the mass of each element. This is probably the best choice for an energy injection zone bounded by a Lagrangian surface, because in this case the mass of the zone stays constant in time.

#### VOLU

This keyword applies only in the case of fluid elements, see i) above. The injected power will be distributed among the different elements that form the energy injection zone proportionally to the volume of each element. This is probably the best choice for an energy injection zone bounded by an Eulerian surface, because in this case the volume of the zone stays constant in time.

#### MCMA

This keyword applies only in the case of fluid finite volumes, see ii) above. Unlike the MASS model, in this case the energy injection is associated to one (or more) particular component(s) of the gas mixture. The presence of such component(s) identifies, at each instant, the current injection zone. The injected power will be distributed among the different control volumes (nodes) that form the current energy injection zone, proportionally to the mass of the chosen component(s) at each node.



**MCVO**

This keyword applies only in the case of fluid finite volumes, see ii) on page E.38. The injected power will be distributed among the different control volumes that form the energy injection zone, proportionally to the volume of each node. This model is similar to VOLU.

**/LECT/**

In the MASS or VOLU cases, these are the elements defining the injection zone. In the MCMA case, these are the selected gas components. Finally, in the MCVO case, these are the control volumes (nodes) defining the injection zone.

**Comments:**

The INJE directive can be repeated as many times as necessary. An additional injection zone is defined each time. At present, up to 5 different zones can be defined.

One may define zones associated with finite elements, and other zones associated with finite volumes, in the same run.

Note that the FONC of index ifon describes the **total** injected power for the zone, as a function of time. This power is distributed at each instant among the various elements that form the injection zone according to either the mass or the volume of the element in relation to the total mass or volume of the zone.

The following global results can be accessed via TPLLOT, for each injection zone that has been defined:

```
INJVxxxx : Volume of injection zone number xxxx
INJMxxxx : Mass of injection zone number xxxx
INJQxxxx : Power injected in injection zone number xxxx
INJIxxxx : Energy injected in injection zone number xxxx
```

Note that energy injection may also be applied to finite elements (only of FLxx types) having the FLMP multiphase multicomponent material. In that case, the injected energy is distributed among all fluid components currently present in the element proportionally to their respective relative mass fractions.

EUROPLEXUS offers another (completely distinct) mechanism for prescribing energy (or even mass) generation, namely via the GENE and GENM parameters of the user's fluid (FLUT or FLMP) material. The main difference is that in that case, the generation applies to the material rather than to a spatial zone (elements or nodes). Another difference is that in that case the given time function (FONC) represents the specific generated energy per unit time, and not the total energy per unit time.

### 9.3 INITIAL CONDITIONS

**Object:**

The directives in this section enable the input of initial conditions relative to displacements, velocities, stresses, temperatures etc.. The temperature can be entered directly or under the form of injected energy.

**Syntax:**

```
"INIT"  
  < "FICH" 'nom_fich' >  
  < "DEPL" . . . >  
  < "VITE" . . . >  
  < "VRIG" . . . >  
  < "VITC" . . . >  
  < "VFCC" . . . >  
  < "SPHE" . . . >  
  < "CONT" . . . >  
  < "TETA" . . . >  
  < "TNOD" . . . >  
  < "ENER" . . . >  
  < "GRAD" . . . >  
  < "GRAV" . . . >  
  < "DEBI" . . . >  
  < "ROTA" . . . >  
  < "ALIC" . . . >  
  < "MCOM" . . . >  
  < "CQST" . . . >  
  < "CQDF" . . . >  
  < "MEDL" . . . >  
  < "STAT" . . . >  
  < "DMAS" . . . >  
  < "EQUI" . . . >  
  < "CRAK" . . . >  
  < "ADAP" . . . >  
  < "IMAT" . . . >  
  < "SKIP" . . . >  
  < "DELA" . . . >  
  < "MAPB" . . . >  
  < "ENGR" . . . >  
  < "MAPP" . . . >
```

**Comments:**

The key-word "INIT" may appear at most once, at the beginning of the sequence relative to the initial conditions.

### 9.3.1 AUXILIARY FILE

**Object:**

This directive allows to read the initial conditions data from an auxiliary file.

**Syntax:**

```
< "FICHIER"    'nom.fic'  >
```

In certain cases the data may be bulky. It is then recommended to store them on an auxiliary file to shorten the main input data file. The auxiliary file is activated by means of the keyword "FICHIER" that precedes the file name (complete under Unix). In the main data file then only the keywords "INITIAL" "FICHIER" remain.

The auxiliary file (in free format) contains the whole set of initial conditions data, except the keyword "INITIAL". To return to the main input data, the auxiliary file must be terminated by the keyword "RETOUR".

### 9.3.2 INITIAL DISPLACEMENTS

**Object:**

The initial conditions concern the nodes of the mesh.

The coordinates of the original mesh are modified.

**Syntax:**

```
"DEPL" (  icomp  xm  /LECTURE/ )
```

**icomp**

Number of the component to which the initial condition applies.

**xm**

Value of the displacement.

**LECTURE**

List of the numbers of the nodes concerned.

**Comments:**

An initial displacement enables the modification of the geometry for a few points without using the meshing program again.

If several values must be entered, it is not necessary to repeat the number of the component.

Example :

```
"INIT"  "DEPL"  1  3.1  LECTURE  1 2  TERM
          2  1.5  LECTURE  3    TERM
          "VITE" 1  0.2  LECTURE  5 PAS 1 10 TERM
```

### 9.3.3 INITIAL VELOCITIES

#### Object:

These initial conditions are relative to the nodes of the mesh.

- VITE : prescribed initial particle (material) velocity
- VITG : prescribed initial mesh velocity (only available in an ALE calculation)

#### Syntax:

```
|[ "VITE" ; "VITG" ]|
  (  icomp   vi   /LECTURE/                               )
  (  "RADI"  vr   "CENT" /LECTURE/  "SURF" /LECTURE/ )
  $[ "FILE"  ifich ; "LIST" ... ]$
  (  "NOEU"  n   vx  vy  <vz>                               )
```

**icomp**

Number of the component where the initial condition is located.

**vi**

Value of the prescribed velocity according to the component icomp.

**RADI CENT SURF**

Prescribe a radial velocity (see comments below).

**vr**

Modulus of the prescribed radial velocity (see comments below).

**FILE**

Velocities are read from a file, as defined next.

**ifich**

Data-set (logical unit number) or file name (in quotes) of the file from which initial nodal velocities are to be read. The data have to be written on the file according with the following (fixed) format. For 3D calculations there are 4 data per line, i.e. (NODE, VX, VY, VZ), and the format is (I6,3E12.5). For 2D calculations there are 6 data per line, i.e. (NODE, VX, VY, NODE, VX, VY) and the format is 2(I6,2E12.5). Note that **all nodes** of the mesh must be specified in this case, even those having zero initial velocities, because the code reads on from the file until the total number of nodes in the model has been reached.

**LIST**

Velocities are read from a list, which directly follows the **LIST** keyword in the input file (starting on a new line). The format of this list is exactly the same as for the **FILE** option described above. This syntax allows to embed in the input file data originally contained in a separate file, without any format changes, but it should be avoided for newly written inputs.

**n vx vy <vz>**

Velocity components of node **n** are **vx vy** (and **vz** in 3D).

#### LECTURE

List of the numbers of the nodes concerned.

#### Comments:

If several values must be entered, it is possible to repeat the number of the component or the word "RADI".

If the prescribed initial velocity is a radial one the user just has to put the word "RADI" followed by the velocity modulus, and thereafter the word "CENT", to give the number of the central node and to enter the series of nodes having the same radial velocity, after the word "SURF". EUROPLEXUS then automatically computes the initial components of the velocity in the global coordinate system.

Warning: the positive direction goes from the center to the exterior.

Example :

```
"INIT"  "VITE"    1    -0.2  LECTURE  5 PAS 1 10 TERM
              2    -0.3  LECTURE  5 PAS 1 10 TERM
          "RADI"    1.5  "CENT"  LECTURE  pcen  TERM
                      "SURF"  LECTURE  psur  TERM
```

### 9.3.4 INITIAL VELOCITIES FOR RIGID BODIES

**Object:**

To prescribe initial velocities for rigid bodies.

**Syntax:**

```
"VRIG" (  comp vi /LECTURE/ )
```

**comp**

Index of the component of the initial velocity. See comments below for the meaning of each component.

**vi**

Prescribed value of the initial velocity according to the component **comp**. See comments below for details.

**LECTURE**

List of the indexes of the rigid bodies concerned (see directive **COMP RIGI**).

**Comments:**

If several values must be entered, it is possible to repeat the index of the component.

The initial velocities must be expressed in the global reference frame  $X, Y, Z$  (the rotational values are then converted internally to the local reference frame, as appropriate). The meaning of the various components is as follows:

- In 2D calculations, **comp** 1 is  $v_X$ , 2 is  $v_Y$  and 6 is the rotational velocity  $\omega = \omega_Z$ . Components 3 ( $v_Z$ ), 4 ( $\omega_X$ ) and 5 ( $\omega_Y$ ) must be 0 in 2D and may *not* be specified.
- In 3D calculations, **comp** 1 is  $v_X$ , 2 is  $v_Y$ , 3 is  $v_Z$ , 4 is the rotational velocity  $\omega_X$ , 5 is  $\omega_Y$  and 6 is  $\omega_Z$ .

### 9.3.5 INITIAL VELOCITIES FOR CELL-CENTRED FINITE VOLUMES

**Object:**

These initial conditions are relative to the cell centers of a mesh composed of Cell-Centred Finite Volumes.

**Syntax:**

```
("VITC" "VITX" vx "VITY" vy <"VITZ" vz> /LECTURE/)
```

**vx**

Initial velocity along x-axis.

**vy**

Initial velocity along y-axis.

**vz**

Initial velocity along z-axis (3D only).

**LECTURE**

List of the numbers of the elements concerned.



### 9.3.6 INITIAL CONDITIONS FOR CELL-CENTRED FINITE VOLUMES

**Object:**

These initial conditions are relative to the cell centers of a mesh composed of Cell-Centred Finite Volumes. For the moment, we can only impose these initial conditions in GAZP and CDEM materials. These new imposed initial conditions override the ones imposed when applying the material to the concerned elements.

Point-symmetric initial conditions specifying primitive variables can be imposed for perfect gases (GAZP) using an input file.

Point-symmetric initial conditions specifying conservative variables can be imposed for any gas using an input file.

**Syntax** for perfect gas (GAZP) material:

```
("VFCC" "VITX" vx "VITY" vy <"VITZ" vz> "PINI" pini "RHO" rho /LECTURE/)
```

**vx**

Initial velocity along x-axis.

**vy**

Initial velocity along y-axis.

**vz**

Initial velocity along z-axis (3D only).

**pini**

Initial pressure.

**rho**

Initial density.

**LECTURE**

List of the numbers of the elements concerned.

**Syntax** for perfect gas (GAZP) material in the point symmetric case.

```
("SPHE" "VFCC" "GAZP" 'nom.fic')
```

In 'nom.fic' one should specify: the number of centers of symmetry, the number of data for each region, the center of each region and the data: radius and density, radial velocity, pressure (as function of the radius). In the example below we have 2 centers of symmetry, 19 data for the first center ((0. 0.) with  $0 < r < 0.1$ ) and 21 data for the second one ((1. 0.), with  $0 < r < 0.2$ ). EOFI is the last line of 'nom.fic'.

```

2
19
21
1  0.000000E+00  0.000000E+00
  0.000000E+00  0.110000E+01 -0.700000E+03  0.110000E+06
  0.555556E-02  0.110374E+01 -0.687716E+03  0.110374E+06
  0.111111E-01  0.111495E+01 -0.675432E+03  0.111495E+06
  0.166667E-01  0.113364E+01 -0.663148E+03  0.113364E+06
  0.222222E-01  0.115980E+01 -0.650864E+03  0.115980E+06
  0.277778E-01  0.119344E+01 -0.638581E+03  0.119344E+06
  0.333333E-01  0.123456E+01 -0.626297E+03  0.123456E+06
  0.388889E-01  0.128315E+01 -0.614013E+03  0.128315E+06
  0.444444E-01  0.133921E+01 -0.601729E+03  0.133921E+06
  0.500000E-01  0.140275E+01 -0.589445E+03  0.140275E+06
  0.555556E-01  0.147377E+01 -0.577161E+03  0.147377E+06
  0.611111E-01  0.155226E+01 -0.564877E+03  0.155226E+06
  0.666667E-01  0.163822E+01 -0.552593E+03  0.163822E+06
  0.722222E-01  0.173166E+01 -0.540309E+03  0.173166E+06
  0.777778E-01  0.183258E+01 -0.528026E+03  0.183258E+06
  0.833333E-01  0.194097E+01 -0.515742E+03  0.194097E+06
  0.888889E-01  0.205684E+01 -0.503458E+03  0.205684E+06
  0.944444E-01  0.218018E+01 -0.491174E+03  0.218018E+06
  0.100000E+00  0.231100E+01 -0.478890E+03  0.231100E+06
2  0.100000E+01  0.000000E+00
  0.000000E+00  0.170000E+01 -0.300000E+03  0.170000E+06
  0.100000E-01  0.170517E+01 -0.278883E+03  0.170517E+06
  0.200000E-01  0.172068E+01 -0.257766E+03  0.172068E+06
  0.300000E-01  0.174653E+01 -0.236649E+03  0.174653E+06
  0.400000E-01  0.178272E+01 -0.215532E+03  0.178272E+06
  0.500000E-01  0.182925E+01 -0.194415E+03  0.182925E+06
  0.600000E-01  0.188612E+01 -0.173298E+03  0.188612E+06
  0.700000E-01  0.195333E+01 -0.152181E+03  0.195333E+06
  0.800000E-01  0.203088E+01 -0.131064E+03  0.203088E+06
  0.900000E-01  0.211877E+01 -0.109947E+03  0.211877E+06
  0.100000E+00  0.221700E+01 -0.888300E+02  0.221700E+06
  0.110000E+00  0.232557E+01 -0.677130E+02  0.232557E+06
  0.120000E+00  0.244448E+01 -0.465960E+02  0.244448E+06
  0.130000E+00  0.257373E+01 -0.254790E+02  0.257373E+06
  0.140000E+00  0.271332E+01 -0.436200E+01  0.271332E+06
  0.150000E+00  0.286325E+01  0.167550E+02  0.286325E+06
  0.160000E+00  0.302352E+01  0.378720E+02  0.302352E+06
  0.170000E+00  0.319413E+01  0.589890E+02  0.319413E+06
  0.180000E+00  0.337508E+01  0.801060E+02  0.337508E+06
  0.190000E+00  0.356637E+01  0.101223E+03  0.356637E+06
  0.200000E+00  0.376800E+01  0.122340E+03  0.376800E+06
EOF

```

**Syntax** for any gas in the point symmetric case.

```
("SPHE" "VFCC" "CONS" 'nom.fic')
```

In 'nom.fic' one should specify: the number of centers of symmetry, the number of data for each region and the number of conservative variables, the center of each region and the data: radius and conservative variables (as function of the radius). In the example below we have 2 centers of symmetry, 19 data and 3 conservative variables for the first center ((0. 0.) with  $0 < r < 0.1$ ) and 21 data and 4 conservative variables for the second one ((1. 0.), with  $0 < r < 0.2$ ). EOFI is the last line of 'nom.fic'. Concerning the data, in the first column there is the radius, in the second one the density, then the momentum and the total energy follow. Other eventual variables depend on the type of material.

```
2
19 3
21 4
1 0.000000E+00 0.000000E+00
0.000000E+00 0.110000E+01 -0.700000E+01 0.110000E+06
0.555556E-02 0.110374E+01 -0.687716E+01 0.110374E+06
0.111111E-01 0.111495E+01 -0.675432E+01 0.111495E+06
0.166667E-01 0.113364E+01 -0.663148E+01 0.113364E+06
0.222222E-01 0.115980E+01 -0.650864E+01 0.115980E+06
0.277778E-01 0.119344E+01 -0.638581E+01 0.119344E+06
0.333333E-01 0.123456E+01 -0.626297E+01 0.123456E+06
0.388889E-01 0.128315E+01 -0.614013E+01 0.128315E+06
0.444444E-01 0.133921E+01 -0.601729E+01 0.133921E+06
0.500000E-01 0.140275E+01 -0.589445E+01 0.140275E+06
0.555556E-01 0.147377E+01 -0.577161E+01 0.147377E+06
0.611111E-01 0.155226E+01 -0.564877E+01 0.155226E+06
0.666667E-01 0.163822E+01 -0.552593E+01 0.163822E+06
0.722222E-01 0.173166E+01 -0.540309E+01 0.173166E+06
0.777778E-01 0.183258E+01 -0.528026E+01 0.183258E+06
0.833333E-01 0.194097E+01 -0.515742E+01 0.194097E+06
0.888889E-01 0.205684E+01 -0.503458E+01 0.205684E+06
0.944444E-01 0.218018E+01 -0.491174E+01 0.218018E+06
0.100000E+00 0.231100E+01 -0.478890E+01 0.231100E+06
2 0.100000E+01 0.000000E+00
0.000000E+00 0.170000E+01 -0.300000E+01 0.170000E+06 0.170000E+01
0.100000E-01 0.170517E+01 -0.278883E+01 0.170517E+06 0.170517E+01
0.200000E-01 0.172068E+01 -0.257766E+01 0.172068E+06 0.172068E+01
0.300000E-01 0.174653E+01 -0.236649E+01 0.174653E+06 0.174653E+01
0.400000E-01 0.178272E+01 -0.215532E+01 0.178272E+06 0.178272E+01
0.500000E-01 0.182925E+01 -0.194415E+01 0.182925E+06 0.182925E+01
0.600000E-01 0.188612E+01 -0.173298E+01 0.188612E+06 0.188612E+01
0.700000E-01 0.195333E+01 -0.152181E+01 0.195333E+06 0.195333E+01
0.800000E-01 0.203088E+01 -0.131064E+01 0.203088E+06 0.203088E+01
0.900000E-01 0.211877E+01 -0.109947E+01 0.211877E+06 0.211877E+01
0.100000E+00 0.221700E+01 -0.888300E+00 0.221700E+06 0.221700E+01
0.110000E+00 0.232557E+01 -0.677130E+00 0.232557E+06 0.232557E+01
```

---

```

0.120000E+00  0.244448E+01 -0.465960E+00  0.244448E+06  0.244448E+01
0.130000E+00  0.257373E+01 -0.254790E+00  0.257373E+06  0.257373E+01
0.140000E+00  0.271332E+01 -0.436200E-01  0.271332E+06  0.271332E+01
0.150000E+00  0.286325E+01  0.167550E+00  0.286325E+06  0.286325E+01
0.160000E+00  0.302352E+01  0.378720E+00  0.302352E+06  0.302352E+01
0.170000E+00  0.319413E+01  0.589890E+00  0.319413E+06  0.319413E+01
0.180000E+00  0.337508E+01  0.801060E+00  0.337508E+06  0.337508E+01
0.190000E+00  0.356637E+01  0.101223E+01  0.356637E+06  0.356637E+01
0.200000E+00  0.376800E+01  0.122340E+01  0.376800E+06  0.376800E+01
EOF

```

**Syntax** for the CDEM material:

```

("VFCC" "VITX" vx "VITY" vy <"VITZ" vz> "PINI" pini "TINI" tini
    "KSI0" ksi0 "K0" k0 "Y1" y1 "Y2" y2 "Y3" y3 ... /LECTURE/)

```

See the CDEM material for the meaning of each term.

### 9.3.7 INITIAL STRESSES

**Object:**

To prescribe initial stresses to different elements.

**Syntax:**

```
"CONT" (  icomp  sig  /LECTURE/  )
```

**icomp**

Number of the component to which the initial condition applies.

**sig**

Value of the initial stress.

**LECTURE**

List of the numbers of the elements concerned.

**Comments:**

The components of the stress tensor are stored in a one-dimensional array. The number of the components depends on the element. See the description of the elements available page INT.80 .

The initial stresses are provided at the integration points of the element.

If there are several integration points in one element, an initial value is given to the component **icomp** of all the points concerned.

If the initial stresses are the result of a static load, it may be more adequate to use EUROPLEXUS for the necessary computations, by using a quasi static damping (see the instruction "OPTION"). A restart after that first computation, by changing the load, will provide the desired result.

Be careful and respect the writing conventions for the stress tensor of each element (see [GBG\\_0020](#)).

### 9.3.8 INITIAL TEMPERATURES

**Object:**

To prescribe different temperatures on certain elements. The values are either directly read from the input file, or as the results of a DELFINE file.

**Syntax:**

Direct reading:

```
"TETA"    (  ti  /LECTURE/    )
```

Reading from a DELFINE file:

```
"TETA"    "DELFINE" ndelfine    "TDELFINE"  tdelfine
```

**ti**

Initial temperature.

**LECTURE**

List of the numbers of the elements concerned.

**ndelfine**

Logical unit number of the DELFINE file.

**tdelfine**

Time written on file DELFINE. The field of temperatures at that instant is considered as the initial temperature field for the EUROPLEXUS computation.

**Comments:**

The definition of a temperature is compulsory for a isotropic Von Mises material dependant on temperature (VMIS TETA) as well as for the material PUFF. By default it is supposed to be equal to zero.

If the user wants to impose to the material VMIS TETA a field of initial stresses due to heating, he has to use the instruction "ENERGIE" (see page E.90).

### 9.3.9 NODAL TEMPERATURES FOR ADVECTION-DIFFUSION (JRC)

**Object:**

To prescribe initial nodal temperatures for advection-diffusion problems. These can optionally be read from a separate data-set.

**Syntax:**

```
"TNOD"  |[ ( tnod  /LECTURE/) ;  
          "FILE"  ifich      ;  
          $ "LIST"  ...        $
```

**tnod**

Initial temperature.

**/LECTURE/**

List of concerned nodes.

**FILE**

Temperatures are read from a file, as defined next.

**ifich**

Unit number of data-set containing nodal temperatures in the form (node index, T). Values are read using format (4(i6,e12.5)).

**LIST**

Temperatures are read from a list, which directly follows the **LIST** keyword in the input file (starting on a new line). The format of this list is exactly the same as for the **FILE** option described above. This syntax allows to embed in the input file data originally contained in a separate file, without any format changes, but it should be avoided for newly written inputs.

### 9.3.10 INITIAL ENERGY SUPPLY

**Object:**

The directive is used to input the temperature field and the field of the corresponding initial stresses generated by a supply of external energy.

The instruction can be used only in connection with the material VMIS TETA (Von Mises isotrope dependant on temperature).

**Syntax:**

```
"ENERGIE" ( "CV" cv "ALPHA" alpha "DEPOSE" wd /LECTURE/ )
```

cv

Specific heat.

alpha

Coefficient of linear expansion.

wd

Supplied energy.

LECTURE

List of the numbers of the elements concerned.

**Comments:**

The definition of a temperature is compulsory for an isotropic Von Mises material depending on temperature. By default it is supposed to be equal to zero.

For this material, the phenomenon is considered isothermal and the temperature remains constant during the computation.

A supplied energy result in an initial stress sig0:

$$\text{sig0} = E * \alpha * ( \text{wd} / \text{cv} )$$

Here E represents Young's modulus. Obviously, the temperature increment relative to the supplied energy is taken into account:

$$\text{DT} = \text{wd} / \text{cv}.$$



### 9.3.11 INITIAL BENDING STRESSES DUE TO TEMPERATURE GRADIENT

#### Object:

The instruction inputs a field of initial bending stresses, generated by a temperature gradient through the thickness of shell elements.

#### Syntax:

```
"GRADIENT"  gradt          ...
... (  $[  "ALPH"  alpha          ;
        "ALP1"  alp1      "ALP2"  alp2      ]$ /LECTURE/  )
```

**gradt**

Temperature gradient (along the normal to the element).

**alpha**

Coefficient of linear expansion (isotropic materials).

**alp1,alp2**

Coefficients of linear expansion (orthotropic material) in the orthotropy coordinate system relative to the element CMC3.

**LECTURE**

List of the numbers of the elements concerned.

#### Comments:

The orientation the normal depends on the numbering of the nodes of the element (see Maxwell's cork-screw rule).

This option concerns only the following shell elements: COQUE, COQ3, COQ4, CMC3, and the following materials: LINEAIRE, VMIS ISOTROPE, VMIS TETA, ORTHOTROPE.

For isotropic materials, the initial stress sigf0 is :

$$\text{sigf0} = - \text{alpha} * \text{gradt} * \text{epaisseur} * \text{young} / (1-\text{nu})$$

### 9.3.12 GRAVITY LOADING

#### Object:

This directive allows to introduce an initial stress field due to gravity (or to hydrostatic pressure, for the fluids) that equilibrates the body weight at the initial time. In the case of pipelines (elements "TUBE" or "TUYA"), it is preferable to use directive "INIT" "DEBI" (page E.120).

#### Syntax:

```
"GRAVITE"      "PTSL" xcoor ycoor < zcoor >      "PSL" psl
                "G"      gx      gy      <      gz      >      < "STRUC" >
                "COUCH"  ncouch*( "RO" rho      "H" h )
                ...      /LECTURE/
```

xcoor, ycoor, zcoor

Coordinates of the point defining the free surface supposed normal to the vector defining the body weight.

psl

Initial pressure at the free surface.

gx gy gz

Components of the weight.

STRUC

Indicates that the concerned elements are associated to a structural material (the stress field is computed in the global reference frame).

ncouch

Number of layers composed by different materials.

rho

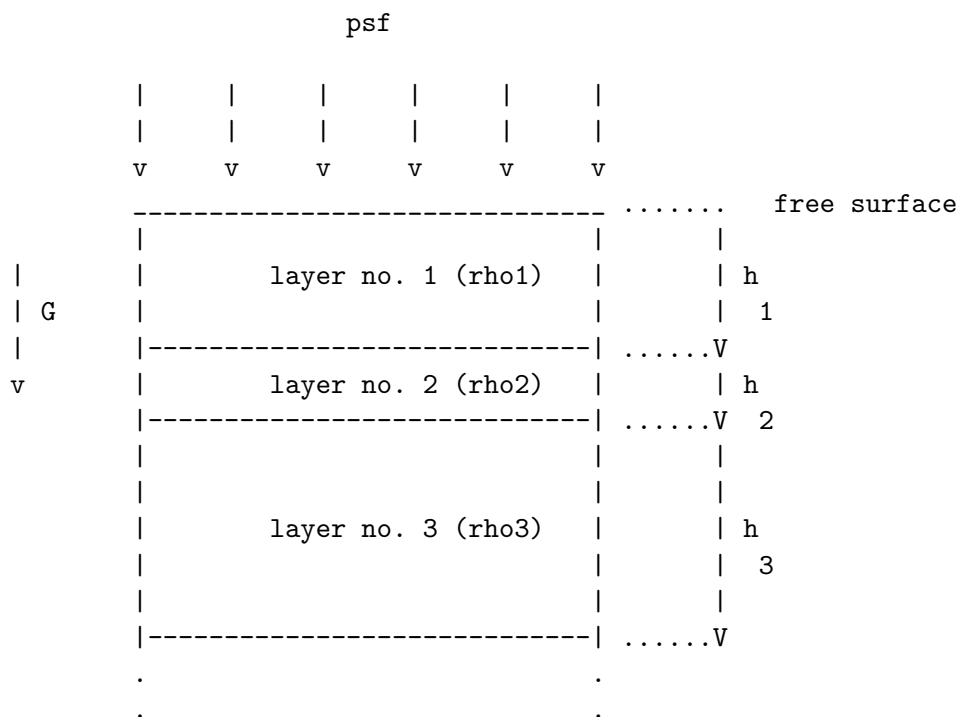
Density of the layer.

h

Height of the layer.

LECTURE

Numbers of the elements subjected to gravity.

**Comments:**

The component  $\langle gz \rangle$  of the acceleration is only used in a 3D calculation.

The pressure is associated to the elements' Gauss points.

The gravity may only be used with continuum elements; the elements of type shell, beam, bar are excluded.

In the case of fluid materials gravity may only be taken into account with the following material laws:

number	name	fluid material law
7	FLUI	isothermal fluid ( c = cte )
8	CAVI	liquid with cavitation
21	ECOU	isothermal fluid flow
10	NAH2	sodium-water reaction
34	ADCR	confinement accident (fast neutrons)
53	ADCJ	confinement accident with JWL gas
22	EAU	two-phase water (liquid + vapour)

**Warning:**

In order to obtain a state of equilibrium at  $t = 0$ , it is necessary to use the directive CHARGE-MENT CONSTANT GRAVITE (see page F.30). The stresses computed will then be equilibrated

by the forces applied to the nodes of the structure defined by the directive "CHARG CONST GRAV".

For the fluid materials for which the density is re-computed, it is necessary to do a first calculation of just one step (for example) so as to obtain the correct values of the density RHO.

### 9.3.13 PRESCRIBED CONSTANT FLOW RATE

#### Object:

This directive allows to specify a field of initial pressures in the branch of a pipeline, which equilibrates at the initial time instant the forces due to head losses, so as to obtain a constant mass flow rate. The hydrostatic pressure is also accounted for, in the case that a gravity load is also prescribed.

This directive may be used only in 1D: elements "TUBE" or "TUYA".

#### Syntax:

```
"DEBIT"
      "LIGN" "PENT" pent    "DEBI" dmas    < "GRAV"  gx gy gz >
      "ORIG"           /LECTURE/
      "LIST"           /LECTURE/
```

#### LIGN

Keyword introducing the reading of the data relative to a pipeline branch.

#### pent

Inlet pressure of the pipeline branch.

#### dmas

Imposed mass flow rate in the pipeline branch.

#### gx,gy,gz

Gravity components.

#### ORIG

Keyword announcing the reading of the node number at the inlet of the pipeline branch.

#### LIST

Keyword announcing the reading of the element numbers of the pipeline branch.

#### Comments:

In the case of a complex pipeline, it is mandatory to decompose it in several elementary branches, for each of which the directive "LIGN" "PENT" ... will be repeated.

The pressure is applied at the element centers.

The velocities in each node of the branch will be computed by EUROPLEXUS, as a function of the mass flow rate (dmas) and of the local diameters.

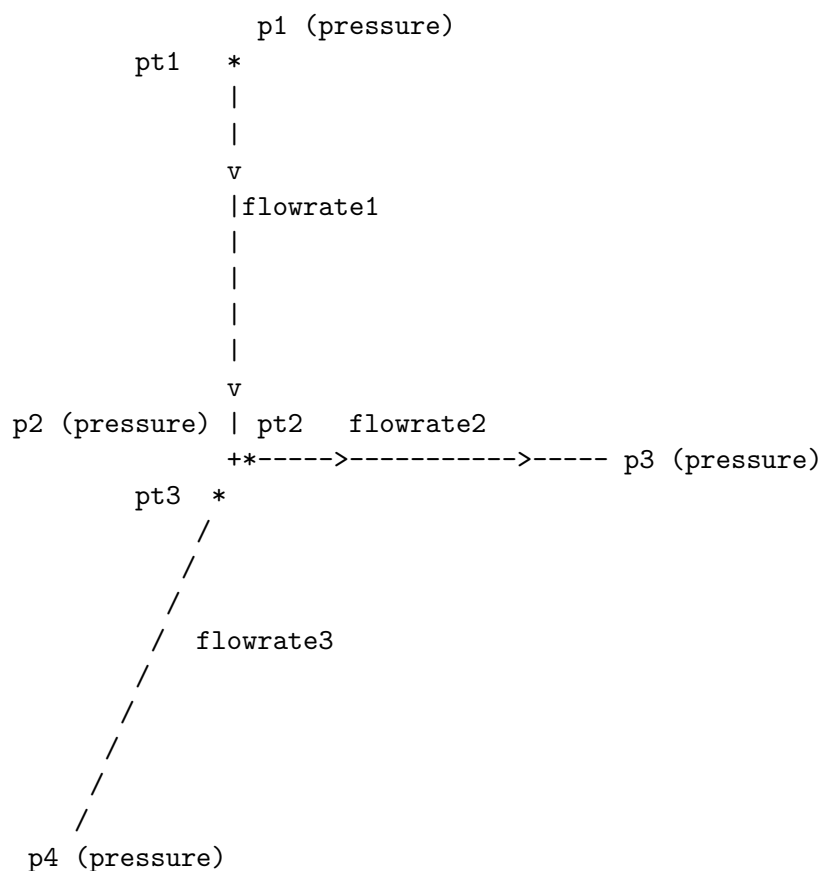
If there is a gravity loading, do not forget to prescribe also "CHARGE" "CONST" "GRAVITE" (page F.30) for the rest of the calculation. In fact, the directive "INIT" "DEBI" computes only the initial status.

There is just one d.o.f. for the elements of type "TUBE", therefore only the weight component along the axis is taken into account for the hydrostatic pressure.

This directive may be used only for the pipelines, i.e. with elements "TUBE", "TUYA", "CAVI", "BIFU", "CL1D" and "CLTU".

It is currently possible to account for the initial mass flow rate only for the following materials:

number	name	fluid material law
7	FLUI	isothermal fluid ( c = cte )
8	CAVI	liquid with cavitation
21	ECOU	isothermal fluid flow
10	NAH2	sodium-water reaction
22	EAU	two-phase water (liquid + vapour)

**Example:**

In this example the directive will become:

"INITIAL"

"DEBIT"

"LIGN"	"PENT"	p1	"DEBI"	debit1	
	"ORIG"		"LECT"	pt1	"TERM"
	"LIST"		"LECT"	ligne1	"TERM"
"LIGN"	"PENT"	p2	"DEBI"	debit2	
	"ORIG"		"LECT"	pt2	"TERM"
	"LIST"		"LECT"	ligne2	"TERM"
"LIGN"	"PENT"	p3	"DEBI"	debit3	
	"ORIG"		"LECT"	pt3	"TERM"
	"LIST"		"LECT"	ligne3	"TERM"

**Remarks:**

In the preceding example, if one just knows the inlet pressure and the mass flow rate in the branch 'ligne1', the following procedure may be followed to obtain the equilibrium status:

- A first EUROPLEXUS calculation is performed for just 1 time step, by taking as initial pressure the inlet pressure  $p1$  for all the pipeline branches with the desired mass flow rates (flowrate1, flowrate2 and flowrate3).

- On the EUROPLEXUS listing, the head losses are printed for each branch of the pipeline. By knowing the head losses in the branch 'ligne1', the outlet pressure for this branch may be deduced:  $p = p1 - pcha1$  which corresponds to the inlet pressures for the branches 'ligne2' and 'ligne3' of our example.

- For the real calculation, the initial conditions to be assumed are then an initial pressure  $p1$  for the branch 'ligne1' and an initial pressure  $p2 = p1 - pcha1$  for the branches 'ligne2' and 'ligne3'. The output pressures of branches 'ligne2' and 'ligne3' are respectively equal to  $p3 = p2 - pcha2 = p1 - pcha1 - pcha2$  and  $p4 = p2 - pcha3 = p1 - pcha1 - pcha3$ ; where  $pcha2$  and  $pcha3$  indicate the head losses in the branches 'ligne2' and 'ligne3'.

- In order to obtain an even better precision, one must take care that the inlet pressure of the branch and the initial pressure of the fluid elements in the branch be coherent. This implies using a "MATERIAU" directive per branch, with the same elements as for the initial mass flow rate.



### 9.3.14 INITIAL ROTO-TRANSLATIONAL VELOCITY

#### Object:

This directive allows to initialise the velocity field for a structure undergoing an initial rotational and an (optional) superposed translational motion. At each node of the structure, the values of initial translational velocity components (resulting from the roto-translational global motion of the body) are evaluated. If a node also possesses rotational degrees of freedom (e.g. because it belongs to a shell or a beam) then also the rotational components of velocity are initialized.

Note that the present directive does *not* initialize the stress field generated by centrifugal forces in the structure. This initialization must be done by a suitable separate input directive, if the structure must be initially in equilibrium.

#### Syntax:

```
"ROTA"      "ORIG" ox oy <oz>
             "AXIS" ax ay az
             "OMEG" omeg
             <$ "VAXE" vaxe ; "VTRA" vx vy <vz> $>
             /LECT/
```

The data consist of two parts. The first part is mandatory and defines the rotational components of the velocity.

ORIG ox oy <oz>

Coordinates  $O_x$ ,  $O_y$ ,  $O_z$  of the “origin” point  $O$  for the rotation ( $O_z$  must not be specified in 2D).

AXIS ax ay az

Components  $a_x$ ,  $a_y$ ,  $a_z$  of the vector defining the rotation axis  $\vec{a}$ . These data *must not* be specified in 2D because in this case the rotation axis is supposed to be normal to the plane and directed “upwards”, i.e. a positive initial rotational velocity produces a counter-clockwise rotation. The defined vector  $\vec{a}$  need not be unitary in length, since it is normalized to unit length internally immediately after reading:  $\hat{\vec{a}} = \vec{a} / \|\vec{a}\|$ . Rotation is then assumed to occur around an axis parallel to  $\hat{\vec{a}}$  and passing through the “origin”  $O$  that has been defined above. The former keyword for this parameter (**VECT** instead of **AXIS**) is obsolescent and deprecated, but it is still accepted for backward compatibility.

OMEG omeg

Rotational velocity  $\omega$  in rad/s around an axis parallel to  $\hat{\vec{a}}$  through  $O$ , that is  $\vec{\omega} = \omega \hat{\vec{a}}$ .

The second part of the directive is optional and defines an initial translational velocity  $\vec{v}_0$  to be combined with the initial rotational velocity  $\vec{\omega}$ . The **VAXE** form is a special syntax that can be used when  $\vec{v}_0$  is directed along the axis of rotation  $\hat{\vec{a}}$ , that is if  $\vec{v}_0 = v_a \hat{\vec{a}}$ , and is maintained for

backward compatibility of old input files. The **VTRA** form is more general (although in principle redundant, as explained in the comments below), since it allows to define a generic translational velocity  $\vec{v}_0$  not necessarily directed along the axis of rotation  $\hat{a}$ . If both keywords are omitted, the initial translational velocity will be zero.

#### VAXE vaxe

Initial *axial* velocity  $v_a$ , i.e. initial translational velocity directed along the axis of rotation  $\hat{a}$  defined above. This keyword is only available in 3D, since in 2D no velocity along the axis of rotation (which is normal to the plane) may occur.

#### VTRA vx vy <vz>

Initial translational velocity  $\vec{v}_0$ , of components  $v_x, v_y, v_z$ . The  $z$  component must be specified in 3D and must *not* be specified (it is zero) in 2D.

The directive is concluded by specifying the nodes of the structure submitted to the initial roto-translational velocity.

#### /LECT/

Numbers of the nodes belonging to the initially roto-translating structure.

#### Comments:

This directive initializes, for each node  $I$  defined by the preceding **/LECT/**, a translational velocity equal to:

$$\vec{v}_I = \vec{\omega} \times \vec{OI} + \vec{v}_0 = \omega \hat{a} \times \vec{OI} + \vec{v}_0 \quad (49)$$

If the **VAXE** form is used for the translational velocity, then this reduces to:

$$\vec{v}_I = \omega \hat{a} \times \vec{OI} + v_a \hat{a} \quad (50)$$

When the concerned node  $I$  has also rotational degrees of freedom, its initial rotational velocity components are initialised as well:

$$\vec{\omega}_I = \vec{\omega} = \omega \hat{a} \quad (51)$$

In principle, *any* rigid body motion can be described by the superposition of the rotation around a given axis (called the instantaneous axis of rotation) plus a translation *along this same axis*. Therefore, the **VAXE** form of this input directive is sufficient to define any roto-translational state of initial velocity, provided the instantaneous axis of rotation is known (i.e., not only as a direction, but also through its position in space, as given by one of its points).

However, in practice a user may prefer to think of the motion as the superposition of a rotation around a particular axis (typically, but not necessarily, passing through the centre of gravity  $G$  of the body) plus a translation in a direction *not* parallel to this axis. Since the analytical determination of the “true” rotational axis (instantaneous axis of rotation as defined above) is not immediate in such a case, the **VTRA** form of the command can come handy, since it can be used directly.

In 2D, it is supposed that the rotation axis is normal to the plane.

It is also necessary to define the initial field of the stresses generated by the centrifuge forces, else the structure will not satisfy the equations of motion at the initial time.

### 9.3.15 INITIALISATIONS FROM A PREVIOUS ALICE FILE

#### Object:

This directive allows to initialise the geometry, the stress field and the field of hardening parameters starting from the ALICE file of a previous EUROPLEXUS calculation.

#### Syntax:

```
"ALICE"  ndfic
         < "MFRO"      omega   >
         < "ECRO"      >
         < "TEMPS"     temps   >
         < "NPAS"      npas    >
         < "POINT"     noe1 noe2 >
         < "ELEMENT"   iel1 iel2 >
```

#### ndfic

Logical unit number of the preliminary ALICE file.

#### omega

Activate “mise a froid” for the structure according to the **omega** parameter.

#### ECRO

Initialize the **ECR** array. By default, **ECR** is not initialized: one makes a preliminary calculation with a very large elastic limit, which yields an elastic solution. In the following calculation, the true material curve is used (one starts with zero plastic deformation).

#### temps, npas

Time or step number of the ALICE file starting from which one reads the fields of displacement, stress and hardening parameters necessary for the initialisation of the present calculation.

#### noe1, noe2

The initialisation will only occur for the nodes between **noe1** and **noe2**.

#### iel1, iel2

The initialisation will only occur for the elements between **iel1** and **iel2**.

#### Comments:

The meshes of the current calculation and of the preliminary one may be different. But it is **MANDATORY** that all nodes and elements of the preliminary calculation **MAINTAIN THE SAME NUMBERS** in the present one. The present mesh must therefore be a superset of the previous one.

When the directives "TEMPS" or "NPAS" are not given, EUROPLEXUS initialises starting from the last step of the preliminary calculation.

When the directive "POINT" (respectively "ELEM") is not present, EUROPLEXUS initialises for all points (respectively all elements).

### 9.3.16 INITIAL STATUS OF MULTICOMPONENT FLOW (JRC)

#### Object:

The instruction defines initial conditions for multicomponent fluid flows.

#### Syntax:

```

"MCOM"    "TEMP"                $ ( val /LECT/ )          $
                                     $ "CHAM" /LCHP/ /LECT/    $

        "PRES"                $ ( val /LECT/ )          $
                                     $ "CHAM" /LCHP/ /LECT/    $

        "VEL1"                $ ( val /LECT/ )          $
                                     $ "CHAM" /LCHP/ /LECT/    $

        "VEL2"                $ ( val /LECT/ )          $
                                     $ "CHAM" /LCHP/ /LECT/    $

        "VEL3"                $ ( val /LECT/ )          $
                                     $ "CHAM" /LCHP/ /LECT/    $

        (      "COMP" 'namecomp'
              "MFRA"  $ ( val /LECT/ )          $
                      $ "CHAM" /LCHP/ /LECT/    $ )

```

#### TEMP

Introduces the value(s) of temperature.

#### PRES

Introduces the value(s) of pressure.

#### VEL1

Introduces the value(s) of  $x$ -velocity.

#### VEL2

Introduces the value(s) of  $y$ -velocity.

#### VEL3

Introduces the value(s) of  $z$ -velocity.

#### COMP

Introduces the component (identified by its name **namecomp**). The name must be spelled exactly as in the declaration of the MCGP material, up to 8 characters.

**MFRA**

Introduces the value(s) of mass fraction.

**val**

Value.

**LCHP**

Values in the form of a CASTEM2000 ‘champoint’.

**LECTURE**

List of the numbers of the nodes concerned.

**Remarks:**

Normally, the `COMP . . . MFRA` directive should be repeated at least `ncom` times, where `ncom` is the total number of components of `THE` multicomponent material of type `MCGP` declared in the `MATE` directive. Note that only one material of type multicomponent (`MCGP`) is allowed within a model.

If values are not specified for a certain zone, they will be set to 0.

### 9.3.17 INITIAL TENSOR OF STRESS

#### Object:

To prescribe initial tensor of stress to different elements.

#### Syntax:

```
"CQST" ( igauss sig /LECTURE/ )
```

**igauss**

Number of the gauss point.

**sig**

Values of the initial components of stress tensor.

**LECTURE**

List of the numbers of the elements concerned.

#### Comments:

The components of the stress tensor are stored in a one-dimensional array. The number of the components depends on the element. See the description of the elements available page INT.80 .

The initial stresses are provided at the integration points of the element. Bending stresses can then be taken into account.

Be careful and respect the writing conventions for the stress tensor of each element (see page G.20).

The table gives the position of the integration points for the Q4GS shell element.

-----					
N. Point Gauss	XSI		ETA		ZETA
-----					
1	-1		-1		-0.9..
2	1		-1		-0.9..
3	1		1		-0.9..
4	-1		1		-0.9..
5	-1		-1		-0.5..
6	1		-1		-0.5..
7	1		1		-0.5..



	8		-1		1		-0.5..	
	9		-1		-1		0	
	10		1		-1		0	
	11		1		1		0	
	12		-1		1		0	
	13		-1		-1		0.5..	
	14		1		-1		0.5..	
	15		1		1		0.5..	
	16		-1		1		0.5..	
	17		-1		-1		0.9..	
	18		1		-1		0.9..	
	19		1		1		0.9..	
	20		-1		1		0.9..	
-----								

For a shell element and a non linear material (such as von Mises material or Hyperelastic material), the stress tensor has 8 components : 3 components of membrane-bending (xx, yy, xy), 3 components that are zero and 2 components of shear (yz, xz).

### 9.3.18 INITIAL TENSOR OF STRAIN

#### Object:

To prescribe initial tensor of strain to different elements.

#### Syntax:

```
"CQDF" ( igauss strain /LECTURE/ )
```

#### igauss

Number of the gauss point.

#### strain

Values of the initial components of strain tensor.

#### LECTURE

List of the numbers of the elements concerned.

#### Comments:

The components of the strain tensor are stored in a one-dimensional array. The number of the components depends on the element. See the description of the elements available page INT.80 .

The initial strains are provided at the integration points of the element. Flexural strains can then be taken into account.

Be careful and respect the writing conventions for the strain tensor of each element (see page G.20).

The table gives the position of the integration points for the Q4GS shell element.

-----					
N. Point Gauss	XSI		ETA		ZETA
-----					
1	-1		-1		-0.9..
2	1		-1		-0.9..
3	1		1		-0.9..
4	-1		1		-0.9..
5	-1		-1		-0.5..
6	1		-1		-0.5..
7	1		1		-0.5..

---

	8		-1		1		-0.5..	
	9		-1		-1		0	
	10		1		-1		0	
	11		1		1		0	
	12		-1		1		0	
	13		-1		-1		0.5..	
	14		1		-1		0.5..	
	15		1		1		0.5..	
	16		-1		1		0.5..	
	17		-1		-1		0.9..	
	18		1		-1		0.9..	
	19		1		1		0.9..	
	20		-1		1		0.9..	

---

For a shell element and a non linear material (such as von Mises material or Hyperelastic material), the strain tensor has 8 components : 3 membrane components (xx, yy, xy), 3 bending components (xx, yy, xy) and 2 components of shear (yz, xz).

### 9.3.19 INITIALISATIONS FROM A MED FILE

**Object:**

This directive allows to read displacements, velocities, stresses, internal variables, strains, fluid velocities and grid velocities from a MED file created by EPX or Code\_Aster only. The coordinates of the original mesh are modified. If reading of stresses is not specified, EUROPLEXUS calculates the initial stress field and the initial internal forces from the displacements (available in sequential version only).

**Syntax:**

```
"MEDL"
      < "CONT" >
      < "ECRO" >
      < "VCVI" >
      < "WGRI" >
      < "NITE"      niter    >
```

niter

Number of iterations to calculate the initial conditions. Default value : NITE = 1

**Comments:**

This directive detects if the file comes from EPX or Code\_Aster with field's names.

The directive "CONT" allows reading both displacements and corresponding stresses stored in the MED file (they must be present). Velocities and strains are also read if present in the file.

If the file comes from Code\_Aster, only BR3D, T3GS, Q4GS, TETR, CUB8, CAR1 and CAR4 elements' values can be initialised. Others values are ignored. When using "CONT" option, no iterations are needed because stresses are not calculated from displacements.

When "ECRO" is present, internal variables are read from the MED file.

"VCVI" is used to take in account initial fluid velocities for VFCC elements. This option is only available with ADCR and EAU materials and must be associated with option "ECRO".

"WGRI" activates the taking into account of the grid velocities field as initial state. It's available in ALE only.

The directive "NITE" allows to initialize a non-linear initial status. At least one material has a non-linear behaviour or the initial displacements, rotations are large.

### 9.3.20 INITIALISATIONS FROM A STATIC ANALYSIS

**Object:**

This directive allows to read results from a former static analysis performed to compute the initial state of the current model. Results are entered by means of displacement fields. Several fields can be given if the static analysis is non-linear, so that the non-linear loading path can be correctly followed. For each field, the corresponding displacement can be applied within a certain number of iterations.

The coordinates of the original mesh are modified.

From displacements EUROPLEXUS calculates the initial stress field and the initial internal forces.

**Syntax:**

```
"STAT"  ifich
```

`ifich`

Number of the logical unit of the formatted file or file name in quotes.

**Comments:**

In order for the initial state to be in equilibrium, keyword (INIT) `EQUI` must also be used (e.g. `INIT STAT ifich EQUI`).

The formatted file is structured as follows:

On the first line, with the format ('NSTP ',I10,' NBNO ',I10), are given the number of displacement fields *nstep* and the number of nodes *nbnodes* for all fields.

The file then contains *nstep* blocks, each block corresponding to one displacement field.

On the first line of each block, with the format ('NCYC ',I10) is given the number of iterations used to apply the corresponding displacement field.

Then, *nbnodes* lines follow, each containing field values for one node of the mesh.

Node line structure is the following:

*Format:* (I10,1X,I2,*ncomp*(1X,A2,1X,E12.5))

*1st integer:* number of the node in the EUROPLEXUS model

*2nd integer:* *ncomp*=number of dofs to be read

For each dof:

*Character:* Name of the dof:

'UX','UY','RZ' for 2D problems

'UR','UZ','RT' for axisymmetrical problems

'UX','UY','UZ','RX','RY','RZ' for 3D problems

*Real:* Value of the displacement field for this dof

**9.3.21 PRESCRIBED MASS FLOW RATE****Object:**

This directive allows to specify a prescribed mass flow rate in a pipeline.

This directive may be used only in 1D.

**Syntax:**

```
"DMAS" dmas /LECTURE/
```

**dmas**

Value of the imposed mass flow rate.

**/LECT/**

List of node numbers.

### 9.3.22 PRESCRIBED INITIAL EQUILIBRIUM

#### Object:

This directive allows to impose initial equilibrium conditions over the whole mesh or in a region specified by the user.

The code computes external applied forces which exactly equilibrate the **internal** forces (e.g. resulting from an initial stress state) in the initial configuration and adds these (constant) force values to the other prescribed external forces at every time step during the transient computation.

In other words, at step 0 the code computes the **equilibrating** forces:

$$F_{\text{eq}} = F_{\text{int}}$$

Then, at every time step (including step 0) these equilibrating forces are added to the other imposed **external** forces:

$$F_{\text{ext}} = F_{\text{ext}} + F_{\text{eq}}$$

Optionally, by specifying the keyword **FTOT**, the above equilibration procedure is applied to the **total** forces (i.e., to the difference between internal and external forces) rather than to the internal forces alone. In this case, at step 0:

$$F_{\text{eq}} = F_{\text{int}} - F_{\text{ext}}$$

and then:

$$F_{\text{ext}} = F_{\text{ext}} + F_{\text{eq}}$$

#### Syntax:

```
"EQUI" $[ "FINT" ; "FTOT" ]$ </LECTURE/>
```

```
/FINT/
```

Equilibrium takes into account only initial internal forces: this is the default behaviour.

```
/FTOT/
```

Equilibrium takes into account both initial internal forces and initial external forces.

```
/LECT/
```

List of node numbers for which initial equilibrium is prescribed. By default (no `/LECT/` specified) all nodes in the mesh are taken.

#### Comments:



The use of /LECT/ to specify the zone subjected to initial equilibrium may be useful in some special cases. For example, assume a stratified (layered) soil subjected to an internal explosion (gas bubble). The soil has initial (hydrostatic-like) stresses due to soil weight, and must be initially equilibrated. However, the fluid (explosive bubble) should not be equilibrated. In this case the /LECT/ would specify just the soil sub-domain (or just its envelope, in case e.g. gravity load is applied to the whole problem).

The use of the FTOT optional keyword may be useful in case e.g. there are some “external” forces that should be equilibrated as well in the initial configuration. For example, fluid-structure interaction forces.

As indicated by the syntax, several zones of imposed equilibrium can be declared, by choosing in each zone the appropriate type of equilibrium (internal or total). For example:

```
EQUI FTOT LECT zone1 TERM
      FINT LECT zone2 TERM
```

Note that the initialization of the equilibrating forces is done at the initial step of a calculation containing the present EQUI directive. In practice, in most cases, this means at step 0 of the calculation.

However, this feature may be exploited, in conjunction with the use of restart, in order to start the equilibration of forces at a time different from the initial time.

For example: perform a first run with saving for restart and without the EQUI directive. In this way, no equilibration takes place at the initial time. Then, restart the calculation by specifying the EQUI directive: the equilibrating forces are computed at the restart time and are applied thereafter (even in case of successive restarts).

1) first run (without EQUI but with QUAS STAT and some blockages):

```
. . .
LINK COUP BLOQ 12 LECT absr TERM
ECRI ... FICH SAUV LAST
OPTI QUAS STAT 1. 105. UPTO 0.E0
CALC TINI -50.E-3 TEND 0.D0
. . .
```

2) restart run (without QUAS STAT but with EQUI, blockages are removed):

```
. . .
REPR 'xxxx.sau' POSI 1
LINK COUP ! empty directive to remove the previous links
EQUI FTOT LECT zone1 TERM
      FINT LECT zone2 TERM
. . .
```

### 9.3.23 PRESCRIBED INITIAL CRACK IN SPHC MODEL

**Object:**

This directive allows to define one or several initial crack(s) in a SPHC model (SPH approach for shells - see Page [GBC\\_0093](#)).

Each crack is defined by a set of segments between given stress-points. Every stress-point located on the path of a crack is supposed to have failed at initial time.

**Syntax:**

```
"CRAK" ncrk * ( /LECTURE/ )
```

**ncrk**

Number of initial cracks.

**/LECTURE/**

Ordered list of stress-points defining the segments of a crack.

### 9.3.24 INITIAL CONDITIONS FOR ADAPTIVITY

#### Object:

This directive allows to impose initial conditions for adaptive calculations, namely mesh refinement operations to be performed in the initialization phase of a calculation (see sub-directives `SPLI` and `ROUT`). These operations are performed starting from the *base* mesh, i.e. the mesh provided in input by means of the `GEOM` directive.

Furthermore, by means of the sub-directive `IMAT` it is possible to fine-tune the initial state (i.e. at step 0) or to re-initialize (at a given time or step subsequent to the beginning of the simulation) the physical state of the material in an adapted mesh by means of the sub-directive `IMAT`.

#### Syntax:

```
ADAP < ( SPLI LEVE leve /LECTURE/ ) >
      <  ROUT CASE ncas >
      <  IMAT <TIME t> <NPAS n>
          nimat*(MATE mat OBJE /LECT/ ( $ INSI ; OUTS $
                                         $ SURF ; VOLU $ /LECT/) ) >
```

#### SPLI

Split the specified object to the specified level of refinement. Recall that the base mesh has level 1. The first splitting (level 2) halves the elements size, the second splitting (level 3) further halves the mesh size (1/4) and so on. Note that splitting can be applied both to base elements and to descendent elements. The splitting operations can be defined in any order, but they are applied in the natural order, i.e. by growing element index.

#### LEVE

Specifies the level *leve* of the split operation.

#### /LECT/

List of element to be split to the specified level.

#### ROUT

Initializations are performed by a user-written routine `INIT_ADAP.ROUT` *after* any split and unsplit operations specified by this same directive.

#### CASE

Introduces the case number *ncas*. Different cases correspond to different initializations which are programmed in the `INIT_ADAP.ROUT` user subroutine.

#### IMAT

Initialize (or re-initialize) the material properties (i.e. the physical state) of the material in an adapted mesh. By default, i.e. if neither **TIME** nor **NPAS** are specified, initializations are performed immediately *after* the split operations done, at step 0, either by the **WAVE** directive or by some FSI-driven adaptivity (**FLSR** or **FLSW**).

**TIME**

Time at which the (re-)initialization of the material state must be performed. If neither **TIME** nor **STEP** are specified, the (re-)initialization is performed at step 0, i.e. at the initial time of the simulation. If both **TIME** and **STEP** are specified, the (re-)initialization is performed at whatever of the two comes first.

**NPAS**

Time step at which the (re-)initialization of the material state must be performed. If neither **TIME** nor **STEP** are specified, the (re-)initialization is performed at step 0, i.e. at the initial time of the simulation. If both **TIME** and **STEP** are specified, the (re-)initialization is performed at whatever of the two comes first.

**nimat**

Number of different **IMAT** zones, to be defined next.

**MATE mat**

Index of the material (as declared in the **MATE** directive) to be assigned to the elements defined in the following.

**OBJE**

The elements in the following **/LECT/** are the candidates for the assignement of the initial material properties (subjected to the following conditions).

**INSI**

The material **mat** has to be assigned to the elements inside the following defined surface or volume. Since the geometrical operations necessary to determine whether a point is or not inside a volume (or surface) are complicated and subjected to errors related to tolerances in the tests, it is always advisable to use *simplex* elements (**TRI3** shape in 2D and **TET4** shape in 3D) to define the volume, instead of using more complex element shapes such as **QUA4**, **CUB8** etc. In fact, for **TRI3** and **QUA4** the inverse mapping is analytical while this may not be the case of the other geometric shapes.

**OUTS**

The material **mat** has to be assigned to the elements outside the following defined surface or volume. Since the geometrical operations necessary to determine whether a point is or not inside a volume (or surface) are complicated and subjected to errors related to tolerances in the tests, it is always advisable to use *simplex* elements (**TRI3** shape in 2D and **TET4** shape in 3D) to define the volume, instead of using more complex element shapes such as **QUA4**, **CUB8** etc. In fact, for **TRI3** and **QUA4** the inverse mapping is analytical while this may not be the case of the other geometric shapes.

**SURF**

The (surfacic) elements in the following `/LECT/` define an oriented surface (not necessarily closed, nor simply connected). All elements belonging to the surface must be consistently oriented. See the comments below for the precise algorithm used to decide whether a given point is “inside” or “outside” the surface.

#### VOLU

The (volumetric) elements in the following `/LECT/` define a volume.

#### Comments:

As indicated by the brackets, the `SPLI` sub-directive can be repeated as many times as necessary to specify all the parts of the mesh to be initially split.

Also, as indicated by the (inner) brackets, the `INSI`, `OUTS`, `SURF`, `VOLU`, `/LECT/` block can be repeated as many times as needed (including zero times) to define the portion of the object `OBJE /LECT/` to which the chosen material properties must be applied. The elements retained are those belonging to the `OBJE` defined and satisfying all the specified conditions (if any). Example 1:

```
... IMAT 1 MATE 3 OBJE LECT toto TERM
```

would assign the properties of material 3 to the entire object `toto` (without any conditions). This is the same as normal material assignment and therefore it is not necessary to use `IMAT` for this.

Example 2:

```
... IMAT 1 MATE 3 OBJE LECT toto TERM INSI VOLU LECT v1 TERM
```

would assign the properties of material 3 to the elements (more precisely, to the active descendent elements) of object `toto` whose centroid lies within the volume `v1` (at the initial time).

Example 3:

```
... IMAT 1 MATE 3 OBJE LECT toto TERM INSI VOLU LECT v1 TERM
                                OUTS SURF LECT s1 TERM
```

would assign the properties of material 3 to the elements (more precisely, to the active descendent elements) of object `toto` whose centroid lies within the volume `v1` *and* outside surface `s1` (at the initial time).

If smooth refinement of the mesh is desired, in addition to the splitting specified, use can be made of the `OPTI ADAP RCON` optional directive.

Note the difference between prescribing, say, initial velocities by means of the `INIT VITE` directive or by means of the `INIT ADAP ROUT` directive. The `INIT VITE` directive can only prescribe velocity values for the *base* mesh nodes. If the mesh is then split by means of the `INIT ADAP SPLI` directive, the velocity values in the descendent nodes are *linearly* interpolated starting from those at the base nodes. On the contrary, the `INIT ADAP ROUT` directive applies the desired initialization to the adapted mesh, i.e. to both the base and the descendent nodes in the same way. No interpolation is done in this case, and therefore the prescription of, say, initial velocities is more precise than in the other case if the initial velocity field is non-linear.

An example of use of the `IMAT` sub-directive is as follows. Assume one wants to simulate the explosion of a solid charge. The bomb domain needs to be discretized very finely in the first steps

of the calculation, until the solid charge has detonated and the pressure waves start to expand. Thereafter, the mesh should be un-refined to recover CPU efficiency (large time increment). The initial mesh refinement of the bomb region can be specified by the **WAVE** directive, until a certain time  $T1$  (see page B.200). The region is discretized coarsely (base mesh) and the refinement is done at step 0 by the **WAVE** directive. In the **MATE** directive, air material (via e.g. **JWLS**) is assigned to the whole fluid coarse mesh, including the bomb. Then when **WAVE** refines the mesh, the air material is automatically propagated to all descendent elements. By using the **IMAT** sub-directive it is then possible to (re-)assign an initial solid explosive material to the effective (fine-meshed) bomb region, while (fine) elements outside the bomb remain with the air material inherited from the base mesh. It is not possible to do this fine-tuning by the **MATE** directive because, when **MATE** is read, mesh refinement has *not* taken place yet.

Another example of use of the **IMAT** sub-directive is the case of a tank with a curved geometry filled with air at higher pressure than the atmosphere. We want to use a regular fluid mesh with the structure immersed in it and to adaptively refine the fluid mesh near the tank walls by means of the **FLSR ADAP** or **FLSW ADAP** directives. The **IMAT** technique may be used to assign high pressure only to the active fluid elements (parents and descendents) which are strictly contained within the curved tank.

Finally, as a practical example of using the **TIME** (or **NPAS**) parameter, consider the case that we want to set the conditions of a gas contained in a tank, but only for the gas elements that are located between two moving and deforming membranes. The gas is modelled in Eulerian so the gas elements that satisfy the above condition are not constant in time, but depend upon the motion of the (Lagrangian) membranes. Assume that the membrane (surfacic) elements are consistently oriented, and that the second membrane is located in the positive (or “external”) half-space of the first membrane while the first membrane is located in the negative (or “internal”) half space of the second membrane.

The positive half-space with respect to a surface is the one to which the (oriented) normal to the surface is pointing to, as indicated by the vectors **n1** and **n2** in the sketch below:



### 9.3.25 MATERIAL RE-INITIALIZATION WITHOUT ADAPTIVITY

**Object:**

This directive allows to re-initialize material properties within a chosen volume or surface. Its syntax is identical to the `INIT ADAP IMAT` directive described on page E.230, but it applies to non-adaptive calculations (although the usefulness of such a directive in non-adaptive cases is perhaps marginal).

For example, it can be used instead of normal material assignment by the `MATE` directive when the mesh zone to which the material belongs has a weird shape that has not been identified as a named geometrical object in the mesh generator, nor can it be simply defined in EPX itself by means of the `COMP GROU` directive. In such cases, an auxiliary (fake) mesh `OBJE` (see below) may be defined in the mesh generator and used to define the domain in which material initialization should occur.

**Syntax:**

```
IMAT <TIME t> <NPAS n>  
      nimat*(MATE mat OBJE /LECT/ ( $ INSI ; OUTS $  
                                     $ SURF ; VOLU $ /LECT/) )
```

For the description of the various parameters, see the `INIT ADAP IMAT` directive described on Page E.230.

**Comments:**

An example of use of the `IMAT` sub-directive is the case of a tank with a curved geometry (a surface) filled with air at higher pressure than the atmosphere. We want to use a regular fluid mesh with the structure (meshed by shells) immersed in it. The `IMAT` technique may be used to assign high pressure only to the fluid elements which are strictly contained within the curved tank surface.



### 9.3.26 SKIPPING ELEMENTS

**Object:**

This directive allows to completely skip from the calculation the elements defined by the command. The skipping can occur from a certain time and/or up to a certain time.

A special form of the command (**VFCC**) is available if one wants to skip all VFCC “elements” (Cell-Centred Finite Volumes). In fact, choosing just part (or all) of the VFCCs present in a calculation with the normal command (**/LECT/**) would not work since most VFCC calculations involve *interfaces* between elements (volumes) and not the elements (volumes) themselves.

**Syntax:**

```
SKIP ( <UPTO upto> <FROM from> $ /LECT/ ; VFCC $ )
```

**UPTO upto**

Time at which skipping of the elements defined next has to cease. By default, skipping never ceases (**upto**= $\infty$ ).

**FROM from**

Time at which skipping of the elements defined next has to begin. By default, skipping begins from the initial time (**from**= $-\infty$ ).

**/LECT/**

List of concerned elements.

**VFCC**

Select all VFCCs present in the calculation.

**Comments:**

Skipping an element has the same effect as if the element would (temporarily) have the phantom (**FANT**) material. The element is not computed, and this allows to save CPU time.

This is different from the effect of the **NOCR** option, whereby the element is computed, but it does not contribute to the definition of the critical time step.

### 9.3.27 INITIAL CONDITIONS BY READING MAP FILE (BLAST LOADING)

**Object:**

This directive allows to read the results of a previous blast simulation (1D, 2D or 3D) and map it to the recent mesh. The centre of the mapping can be defined (POS). Until now only mapping from 1D to 3D is possible.

The file can be written by the ECRI MAPB command (see [11.7](#)).

**Syntax:**

```
MAPB 'file.map'  
  | SPHE POS x y z <TARR>;  
  <ADAP nada > /LECT/ |
```

**file.map**

Name of the file, where the previous blast simulation is stored.

**x, y, z**

Position (x, y, z) where the blast map should be mapped.

**TARR**

The time that is used for the calculation until the mapping file is written is used as initial time. This value will be overwritten by TINI in CALC.

**nada**

Introduces the adaptivity for mapping. In that case, the mapping is performed, and immediately after, the mesh is adapted. In order to get better mapping results, the mapping is performed again. The variable nada gives the number of iterations that should be taken in order to reach the best results. For the adaptivity, the correct dimensioning must be defined ([GBA\\_0062](#)) and an adaptivity procedure must be defined ([GBB\\_0210](#)).

**/LECT/**

List of elements where the pressure is applied.

**Comments:**

The results of a 1D blast simulation can be written by using the mapb operator as part of ECRI (see [GBG\\_0070](#)).

At this time only mapping from 1D (TUBE or TUVF) into the material GAZP are possible. The use of volume centred finite volumes is recommended since the results are more accurate.

### 9.3.28 INITIAL DAMAGE FOR GRADIENT DAMAGE MATERIALS

**Object:**

This directive defines initial damage values for gradient damage materials **ENGR**, see [7.7.21](#). This initial condition introduces an *a priori* state of the damage field. Because of the irreversibility condition, setting the value of 1 to a selection of nodes amounts to model an initial crack in the structure.

**Syntax:**

```
"ENGR" ( alpha0 /LECTURE/ )
```

**alpha0**

Prescribed initial damage value  $0 \leq \alpha_0 \leq 1$ , knowing that 0 corresponds to an undamaged state and 1 to the crack.

**LECTURE**

List of the nodes concerned.

### 9.3.29 INITIALIZATION FROM A MAP FILE

**Object:**

This directive allows to read the results of a previous simulation (stored in a so-called map file) and map it to the current mesh.

The map file must have been previously generated by the **ECRI MAPP** command (see [Section 11.7](#)).

**Syntax:**

```
MAPP <FORM> <nmapp> <MATC>  
    <$HGRI hgri ; NMAX nmax ; DELE dele$> <DGRI>  
    OBJE /LECT/
```

**FORM**

The map file is formatted (ASCII). By default, an unformatted (binary) map file is assumed.

**nmapp**

Name of the map file, included in quotes.

**MATC**

The optional **MATC** keyword declares (under the User's responsibility) that the target object perfectly matches the source object, i.e. the two objects are composed by the same elements (although perhaps with different element and node indexes). This greatly facilitates and speeds up the solution mapping (since no interpolation, and only a relatively simple search is needed) and should be used whenever possible.

**HGRI**

Specifies the size of the fast search grid cell. Each cell has the same size in all spatial directions and is aligned with the global axes.

**NMAX**

Specifies the maximum number of cells along one of the global axes.

**DELE**

Specifies the size of the fast search grid cell as a multiple of the length of the largest element from which the mapping is performed ("from" elements). Element "diameters" are computed only along each global spatial direction and the maximum is taken. For example, by setting **DELE 2** the size of the cell is two times the length of the largest "from" element. By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DELE** are specified, the code takes **DELE 1.01**.

**DGRI**

Dump out the grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed.

**OBJE**

Introduces the object (in the current mesh) to be mapped to the physical conditions present in the map file.

**/LECT/**

List of elements forming the object to be mapped (in the current mesh).

**Comments:**

See directive **ECRI MAPP** on page **GBG\_0070** for a description of how to create the map file.

## 10 GROUP F—LOADS

### Object:

These instructions determine the loads. The directive "LOAD" is an alias of the "CHAR" directive.

### Syntax:

```
$ "CHARGE" ; "LOAD" $
  < "CONSTANTE" . . . >
  < ndcha $[ "FACTORISEE" . . . ;
            "PROGRAMMEE" . . . ]$ >
  < "ADDF" . . . >
  < "SPEC" . . . >
  < "FCTE" . . . >
  < "FIMP" . . . >
  < "FDYN" . . . >
  < "AIRB" . . . >
```

### ndcha

Number of the temporary disc (logical unit number) where the loads are stored (disk of time-dependent loads). **This data is optional:** by default ndcha=1.

The various sub-directives, detailed in the following pages, are summarized hereafter:

## 1/ Constant loads :

```

"CONSTANTE"  $ "GRAVITE"      gx  gy  < gz >      $
              $ "ROTATION"    omega                /LECTURE/ $

```

## 2/ Factorized loads :

```

"FACTORISEE" <ndfact> ( | ( "DEPLA" . . . ) |
                       | ( "FORCE"  . . . ) |
                       | ( "PRESS"  . . . ) |
                       "TABLE" . . . )

```

## 3/ Programmed loads :

```

"PROGRAMMEE" ndprog $ "FORCE" . . . $
                  $ "PRESSION" . . . $
                  "ROUTINE" . . .
                  < "ROUTINE" . . . >

```

## 4/ Generalized loads for advection-diffusion calculations (JRC)

```

"ADDF"  $ "TIMP" . . . $
        $ "FLUX" . . . $
        $ "QGEN" . . . $
        $ "CONV" . . . $
        $ "RADI" . . . $
        $ "PRES" . . . $
        $ "VELO" . . . $
        $ "BLOQ" . . . $
        $ "VPLA" . . . $
        $ "VLIN" . . . $

```

## 5/ Seismic-like loads for use with spectral elements (JRC)

```

"SPEC" | "POIN" . . . |
       | "PLAN" . . . |
       | "SISM" . . . |

```

## 6/ New Constant loads:

```

FCTE  NODE /LECT/ $ FORC f ; MOME m $ VECT x y z

```

## 7/ Imposed time-dependent loads:

```

FIMP  NODE /LECT/ $ FORC f ; MOME m $ VECT x y z NUFO nf

```

## 8/ Dynalpy loads:

```

FDYN  NODE /LECT/ PZER p0 COEF c VECT x y z ELEM e

```

## 9/ Air Blast (AIRB) loading:

```

AIRB  |[ "X" x "Y" y <"Z" z> ; "NODE" /LEC1/ ]|
      "MASS" m $[ "TINT" t ; "TAUT" ]$ <"OPOS">
      <"ANGL">
      <"CUBE">
      <"COEF" cf>
      <"CONF" c>
      <"DECA" d>
      <"PMAX" pmax "TD" td "B" b>

```

```
<"SHAD" /LECS/>
/LECT/
```

### Comments:

#### 1/- CONSTANTES

The constant loads act all along the calculation. It is the case of body weight or of a fixed acceleration.

#### 2/- FACTORISEES

The defined loads are multiplied by a coefficient which varies in time and is interpolated from a table:

$$Q(t) = A * C(t)$$

#### 3/- PROGRAMMEES

The loads will be read and computed for some elementary times which are fixed a priori by the user in a subroutine given by him (FPROG or PPROG). EUROPLEXUS will then perform a linear interpolation to determine the loads at the precise instants of the calculation.

##### 1/ Constant loads:

```
FCTE NODE /LECT/ $ FORC f ; MOME m $ VECT x y z
```

##### 2/ Imposed time-dependent loads:

```
FIMP NODE /LECT/ $ FORC f ; MOME m $ VECT x y z NUFO nf
```

##### 3/ Dynalpy loads:

```
FDYN NODE /LECT/ PZER p0 COEF c VECT x y z ELEM e
```

#### 4/- FCTE

The constant loads act all along the calculation. It is the case of body weight or of a fixed load.

#### 5/- FIMP

The defined loads are multiplied by a coefficient which varies in time and is interpolated from a table:

$$Q(t) = A * C(t)$$

#### 6/- FDYN

These loads are only related to 1-D elements of type TUBE or TUYA.

#### 7/- AIRB

These loads result from an air blast wave. The parameters are similar to those available in the IMPE AIRB directive, see Page C.882, but here the load is applied directly to a region of structural elements rather than by using special CLxx elements. This facilitates the treatment of element erosion and of mesh adaptivity.

These instructions are described in detail on the following pages.



## 10.1 AUXILIARY FILE

### Object:

This directive allows to read the initial conditions data from an auxiliary file.

### Syntax:

```
"CHARGE"    ndcha  < "FICHIER"    'nom.fic'  >
```

In certain cases the data may be bulky. It is then recommended to store them on an auxiliary file to shorten the main input data file. The auxiliary file is activated by means of the keyword "FICHIER" that precedes the file name (complete under Unix). In the main data file then only the keywords "CHARGE" "FICHIER" remain.

The auxiliary file (in free format) contains the whole set of load data, except the keyword "CHARGE". To return to the main input data, the auxiliary file must be terminated by the keyword "RETOUR".

## 10.2 CONSTANT LOADS

### Object:

This directive allows to introduce constant accelerations (most often gravity) during the whole computation. It also gives the possibility to compute a structure in a rotating frame (with a constant rotation speed). Definition of sinusoidal acceleration is available (PERIODE and PHASE). It is also possible that the acceleration is linear and then keeps a constant value (RAMPE).

### Syntax:

```
"CONSTANTE"  | [ "GRAVITE"      gx  gy  < gz >  /LECTURE/  ;
                  "ROTATION"    omega          /LECTURE/  ] |
                  <"PERIODE"> Tx   Ty   Tz
                  <"PHASE">  Phix Phiy Phiz
                  <"RAMPE">   tx   ty   tz
```

$g_x \ g_y \ g_z$

Components of the acceleration or of the gravity.

$\omega$

Rotation speed (rad/s).

$T_x \ T_y \ T_z$

Period of the sinusoidal acceleration

$\text{Phix} \ \text{Phiy} \ \text{Phiz}$

Phase of the sinusoidal acceleration

$t_x \ t_y \ t_z$

Time after which acceleration is constant

LECTURE

Numbers of the concerned nodes.

### Comments:

The component  $\langle g_z \rangle$  of the acceleration only makes sense for a three-dimensional computation.

Forces due to gravity or the acceleration of a moving frame are applied to the nodes of the structure defined in the directive `/LECTURE/`.

In the case of a calculation in a rotating frame, it is assumed that the rotation axis is Oz. The forces applied to the nodes are:

$$F_x = M\omega^2 x$$

$$F_y = M\omega^2 y$$

$$F_z = 0$$

This force applies to the nodes specified by the following `/LECTURE/` directive.

If all nodes are concerned, it is sufficient to put the word `TOUS` in place of the directive `/LECTURE/`.

In the case of pipelines, it is important to couple this directive with `"INIT"` `"DEBIT"` (page E.120) so as to avoid the transient due to sudden application of gravity.

For the tubes and the pipelines, the rotation constant charge does not make sense. Note that, although there is just one d.o.f. for the elements of type `TUBE`, the gravity vector may have an arbitrary orientation.

For a sinusoidal acceleration, the amplitude is defined by the component of the acceleration and is multiply by a sinus function. The user has to define the period of the function and the phase. If a component is not exited, the period and the phase for this component have to be zero.

The ramp acceleration is defined by a linear part that starts from zero at the initial time and then grow linearly to pass by the point `tx=gx` (for example). Then after `tx`, the acceleration is kept constant at a `gx` level. If a component is not exited, the time for this component has to be zero.

### 10.3 FACTORIZED LOADS

**Object:**

This directive allows to input loads varying in time, of the following type:  $Q(t) = A * C(t)$ , with:

- A as a base value (displacement, force or pressure);
- C(t) as a coefficient whose values, depending on time, are supplied by a table.

**Syntax:**

```
"FACT"  ndfact
      (    ( "DEPLA"  . . . )
          ( "FORCE"   . . . )
          ( "PRESS"   . . . )
          ( "ACCE"    . . . )

          "TABLE" . . .
      )
```

**ndfact**

Number of the disc (logical unit number) on which the data of the factorized loads are stored. **This data is optional:** by default ndfact=2.

**Comments:**

**The instruction "FACT" cannot be used more than once.**

On the contrary, the sequence terminating with the "TABLE" directive may be repeated as many times as necessary.

**Warning:**

The instruction "TABL" is mandatory. For the case of charges constant in time, just give a two-entry table with the same value in both entries, i.e. of the form "TABL 2 t1 v t2 v", where v is the (constant) value, t1 is less or equal to the initial time of the computation and t2 is greater or equal to the final time of the computation.

### 10.3.1 DISPLACEMENT

**Object:**

This option enables displacements depending on time to be prescribed.

**Syntax:**

```
"DEPLA"  /LECDDL/  d0  /LECTURE/
```

**LECDDL**

Reading procedure of the numbers of the degrees of freedom concerned.

**d0**

Base value of the imposed displacement. The instruction "TABL" must follow to determine the variation in time, **even for constant displacements** (see note below).

**LECTURE**

Numbers of the nodes concerned.

**Comments:**

The displacements of the nodes defined by the procedure LECTURE and along the directions determined by LECDDL, are of the following type:

$$D(t) = D0 * C(t)$$

C(t) is provided by the first array met after that option (see "TABL").

That option can be used as many times as necessary .

If the imposed displacement is a blockage ( $d0 = 0$ ), it is better to use the option "BLOQUE" of the instruction "LIAISON" (page D.30).

**Warning:**

The instruction "TABL" is mandatory. For the case of charges constant in time, just give a two-entry table with the same value in both entries, i.e. of the form "TABL 2 t1 v t2 v", where v is the (constant) value, t1 is less or equal to the initial time of the computation and t2 is greater or equal to the final time of the computation.

### 10.3.2 FORCE

#### Object:

This option enables nodal forces, varying in time, to be imposed.

#### Syntax:

```
"FORCE" /LECDDL/ f0 /LECTURE/
```

#### LECDDL

Reading procedure of the numbers of the degrees of freedom concerned.

#### f0

Base value of the imposed force. The instruction "TABL" must follow to determine the variation in time, **even for constant forces** (see note below)

#### LECTURE

Numbers of the nodes concerned.

#### Comments:

The forces applied to the nodes defined by the procedure LECTURE and according to the directions determined by LECDDL, have an intensity of:

$$F(t) = F0 * C(t)$$

C(t) is provided by the first array (TABLE) met after that option (see "TABL", page F.150).

That option may be used as often as necessary.

For an textbfaxisymmetric computation, the force must be divided by  $2\pi$ :

$$F0 = \frac{\text{real force}}{2\pi}$$

#### Warning:

The instruction "TABL" is mandatory. For the case of charges constant in time, just give a two-entry table with the same value in both entries, i.e. of the form "TABL 2 t1 v t2 v", where v is the (constant) value, t1 is less or equal to the initial time of the computation and t2 is greater or equal to the final time of the computation.

### 10.3.3 PRESSURE

**Object:**

This option introduces a pressure (expressed in real number form) which is exerted on a segment set (2-dimensional case) or on a surface composed of shell elements (2-dimensional or 3-dimensional case), or on the faces of solid elements (3-dimensional case).

**Syntax:**

```
"PRESSION"  
  | ( "SEGMENT"  . . . ) |  
  | ( "COQUE"    . . . ) |  
  | ( "FACE"     . . . ) |  
  | ( "NODE"     . . . ) |  
  | ( "ELDI"     . . . ) |
```

**Comments:**

The word "PRES" is the first keyword of the option.

There is no need to repeat "PRES" to re-define a new line or surface corresponding to the same table (the one which immediately follows the word "PRES").

On the contrary, it is compulsory to define again the word "PRES" if it is necessary to create lines or surfaces relative to another table.

## SEGMENT PRESSURE

### Object:

This directive is mainly used to apply a pressure to 2D continuum elements (in 3D cases, use **PRES FACE**). It allows to enter a pressure which is exerted on a certain number of adjacent segments or lines, in the case of 2-dimensional computation. The pressure may vary in time (factorized loads) or be hydrostatic.

### Syntax:

```
"SEGMENT"  | [  p0          ;
               "HYDRO" rho  g  z0 ] | /LECTURE/
```

#### $p_0$

Base value  $p_0$  of the pressure. The instruction **TABL** must follow to determine the variation in time, even for constant pressures (see note below).

#### HYDRO

Keyword that announces a hydrostatic pressure. The pressure field is generated by the presence of a “vertical” acceleration of gravity, acting along the  $y$ -axis in 2D or the  $z$ -axis in 3D.

#### rho

Density  $\rho$  of the liquid which generates the pressure.

#### g

Acceleration of gravity  $g$  applied to the fluid in the “vertical” direction, i.e. along the  $y$ -axis in 2D or the  $z$ -axis in 3D. The value should normally be negative (see also comments below).

#### $z_0$

Vertical level  $z_0$  of the free surface of the fluid, supposed horizontal.

#### LECTURE

Numbers of the nodes composing the pressure line. Each couple of subsequent nodes forms a segment. For example, if the list contains the three indexes 25, 4, 39, then two segments are considered: (25, 4) and (4, 39). Special care must be taken if one uses the name of GIBI object(s) to define the line(s) subjected to pressure, see comments below.

### Comments:



The user has the choice between a pressure  $p_0$  which varies in time and a hydrostatic pressure. It is not allowed to define both at the same time without re-using the word **SEGM**.

For a defined basic value  $p_0$ , the pressure intensity is:

$$p(t) = p_0 C(t)$$

$C(t)$  is provided by the first array (**TABL**) met after the option **PRES** (see also **TABL**).

If a hydrostatic pressure is defined (keyword **HYDR**), the pressure is only applied to the segments of the line at levels  $z$ , such that:

$$g(z - z_0) \geq 0$$

From the previous expression it appears that, since the hydrostatic pressure should be exerted on segments at  $z < z_0$ , then the value of  $g$  specified should normally be negative.

Moreover, the intensity of the pressure is:

$$p = \rho g(z - z_0)$$

The values  $\rho$ ,  $g$ ,  $z_0$  may be negative.

Each new option **SEGM** defines a different line.

If the line of segments is defined by giving the name of a **GIBI** object, make sure that the line is oriented, i.e. that the nodes are listed from one extremity to the other in the correct order (not randomly, as it may sometimes occur in some mesh generators).

If there is more than one line, and the lines are disjoint from one another (i.e. the final node of one line is not the initial node of the next one), then the **PRES** directive must be repeated (one directive for each line). Otherwise, a “spurious” segment joining the final node of a line to the initial node of the next one in the list would also be considered as subjected to pressure. This is particularly dangerous if more than one **GIBI** object is listed in the same **LECT**. In case of doubt, it is always safer to use a separate **PRES** directive for each line.

### Sign of the pressure:

The order in which the points of the line are read by means of the procedure **LECT**, defines the orientation of the contour of that line. The normal vector ( $\vec{n}$ ) results from a rotation of  $\pi/2$  of the contour itself.

Positive values will create forces in the orientation of the normal ( $\vec{n}$ ) thus obtained.

### Warning:

The keyword **TABL** is mandatory. For the case of charges constant in time, just give a two-entry table with the same value in both entries, i.e. of the form **TABL 2 t1 v t2 v**, where **v** is the (constant) value, **t1** is less or equal to the initial time of the computation and **t2** is greater or equal to the final time of the computation.

**SHELL PRESSURE****Object:**

This directive is mainly used to apply a pressure to 2D or 3D shell elements. For two- or three-dimensional computations, this option enables a pressure to be entered, which is exerted on a surface composed of shell elements. The pressure may vary in time (factorized loads) or be hydrostatic.

**Syntax:**

```
"COQUE"  |[      p0      ;
           "HYDRO" rho  g  z0      ;
           "HYDRO" rho  gx gy gz  x0 y0 z0 ]|  /LECTURE/
```

**p0**

Base value of the pressure (non hydrostatic case). The instruction "TABL" must follow to determine the variation in time, even for constant pressures (see note below).

**HYDRO**

Hydrostatic pressure.

**rho**

Density of the fluid which generates the pressure.

**For two-dimensional computations:****g**

Acceleration applied to the fluid.

**z0**

Level of the free surface, which is supposed horizontal.

**LECTURE**

Numbers of the shell elements submitted to the pressure.

**For three-dimensional computations:****gx, gy, gz**

Components of the acceleration vector (G) which is applied to the fluid.

**x0, y0, z0**

Coordinates of a point located on the surface supposed horizontal.

#### LECTURE

Numbers of the shell elements submitted to the pressure.

#### Comments:

The user has the choice between a pressure  $p_0$  which varies in time and a hydrostatic pressure. It is impossible to define both at the same time, without re-using the word "COQUE".

If a basic value  $p_0$  has been defined, the pressure intensity is:

$$P(t) = P_0 * C(t)$$

$C(t)$  is provided by the first array met after the option "PRESS" (see TABLE).

If a hydrostatic pressure is defined (keyword HYDR), the pressure is applied to the points on the surface of coordinates X,Y,Z such that :

$$g_x * (x-x_0) + g_y * (y-y_0) + g_z * (z-z_0) > 0 \text{ ( or } = 0 \text{ )}$$

The pressure intensity will be for these points:

$$P = \rho * ( g_x * (x-x_0) + g_y * (y-y_0) + g_z * (z-z_0) )$$

In a two-dimensional case, the definition of the hydrostatic pressure is the same as for the sub-directive "SEGMENT".

For a three-dimensional computation, see the definition of hydrostatic pressure with the sub-directive "FACE".

Each new option "COQUE" defines a new pressure surface.

#### Sign of the pressure:

The normal vector on the surface is oriented according to the numeration of the shell nodes. Positive pressures will create forces in the orientation of that normal.

The orientation of the normal is given by the following rule (Maxwell's cork-screw rule). An observer placed at the centre of the shell element which is crossed by the normal from the bottom to the top, must be able to notice that the shell element is numbered in increasing order, by rotating in a trigonometric sense (anticlockwise).

Warning:

If the mesh developed by means of "COCO" is used, the elements are not necessarily oriented in the same way (this has to be explicitly requested).

If the mesh is entered by the user, all elements have to be numerated so that their orientation is coherent. A shell surface composed of elements which are oriented in a different way, may produce errors and confusion concerning the direction of the pressures from the data point of view as well as from the results.

The instruction "TABL" is mandatory. For the case of charges constant in time, just give a two-entry table with the same value in both entries, i.e. of the form "TABL 2 t1 v t2 v", where v is the (constant) value, t1 is less or equal to the initial time of the computation and t2 is greater or equal to the final time of the computation.

**FACE PRESSURE****Object:**

This directive is mainly used to apply a pressure to 3D continuum elements. For a three-dimensional computation, this option enables a pressure to be entered, which is exerted on a surface composed of the sides of solid elements. The pressure may vary in time (factorized loads) or be hydrostatic.

**Syntax:**

```
"FACE" iface | [ p0
                  "HYDRO" rho gx gy gz x0 y0 z0 ] | /LECTURE/
```

**iface**

Number of the side (face) of the elements read by the procedure. LECTURE

**p0**

Base value of the pressure (non hydrostatic). The instruction "TABL" must follow to determine the variation in time, even for constant pressures. (see note below).

**HYDRO**

Hydrostatic pressure.

**rho**

Density of the fluid which generates the pressure.

**gx, gy, gz**

Components of the acceleration vector (G) applied to the fluid.

**x0, y0, z0**

Coordinates of a point of the free surface which is supposed horizontal.

**LECTURE**

Numbers of the elements submitted to the pressure.

**Comments:**

The user has the choice between a pressure P0 which varies in time and a hydrostatic pressure. It is impossible to define both at the same time without re-using the word "FACE".

For a defined basic value P0, the intensity of the pressure is:

$$P(t) = P_0 * C(t)$$

$C(t)$  is provided by the first array met after the option "PRESS" (see TABLE).

### Hydrostatic pressure:

If a hydrostatic pressure is defined (keyword HYDR), it is applied to the points of the pressure surface of coordinates X,Y,Z such that:

$$g_x * (x-x_0) + g_y * (y-y_0) + g_z * (z-z_0) > 0 \text{ ( or } = 0 \text{ )}$$

For these points, the intensity is:

$$P = \rho * ( g_x * (x-x_0) + g_y * (y-y_0) + g_z * (z-z_0) )$$

The free surface of the liquid is composed of the plane which passes through the point M0 of coordinates (x0, y0, z0) and perpendicular to the vector (G) whose components are : (gx,gy,gz).

If the vector (G) is drawn with the point M0 as origin, the pressure will be applied to the points M of the pressure surface which are located in the half-space containing (G).

The pressure intensity at point M is:

$$P = \rho * g * h$$

Here h represents the distance between M and the free surface; g represents the gravity.

The vector (G) with its 3 components enables the surface of a liquid (horizontal) to be entered, when the vertical axis of the mesh is distinct from the physical vertical line. In this case, the surface is an inclined plane in the coordinate system of the mesh.

Each new definition of the option "PRES" "SEGMENT" generates a new pressure surface.

### Sign of the pressure:

The normal to the face of an element is the outward normal of that element. A positive pressure creates a force in the orientation of that normal.

### Warning:

The instruction "TABL" is mandatory. For the case of charges constant in time, just give a two-entry table with the same value in both entries, i.e. of the form "TABL 2 t1 v t2 v", where v is the (constant) value, t1 is less or equal to the initial time of the computation and t2 is greater or equal to the final time of the computation.

## NODE PRESSURE

### Object:

This directive is mainly used to apply a pressure to continuum elements (2D or 3D). This option enables a pressure to be entered, which is exerted on a surface composed of the nodes belonging to the edge of structure. The pressure may vary in time (factorized loads) or be hydrostatic.

### Syntax:

```
"NODE"      | [      p0      ;
               "HYDRO"  rho  gx gy gz  x0 y0 z0 ] | /LECTURE/
```

#### p0

Base value of the pressure (non hydrostatic). The instruction "TABL" must follow to determine the variation in time, even for constant pressures. (see note below).

#### HYDRO

Hydrostatic pressure.

#### rho

Density of the fluid which generates the pressure.

#### gx, gy, gz

Components of the acceleration vector (G) applied to the fluid.

#### x0, y0, z0

Coordinates of a point of the free surface which is supposed horizontal.

#### LECTURE

Numbers of the nodes belonging to the edge of structure, submitted to the pressure.

### Comments:

The user has the choice between a pressure P0 which varies in time and a hydrostatic pressure. It is impossible to define both at the same time without re-using the word "NODE".

For a defined basic value P0, the intensity of the pressure is:

$$P(t) = P0 * C(t)$$



$C(t)$  is provided by the first array met after the option "PRESS" (see TABLE).

**Hydrostatic pressure:**

This directive is described in [10.3.3](#) (page F.130).

**Sign of the pressure:**

The normal to the face of an element is the outward normal of that element. A positive pressure creates a force in the orientation of that normal.

**Warning:**

The instruction "TABL" is mandatory. For the case of charges constant in time, just give a two-entry table with the same value in both entries, i.e. of the form "TABL 2 t1 v t2 v", where v is the (constant) value, t1 is less or equal to the initial time of the computation and t2 is greater or equal to the final time of the computation.

## PRESSURE ON DISCRETE ELEMENTS

### Object:

This directive is used to apply pressure on the outer surface of a cylinder made of discrete elements (ELDI). The pressure may vary in time (factorized loads).

### Syntax:

```
"ELDI"      | [  p0 ;  
               "PAX1"  x1 y1 z1  
               "PAX2"  x2 y2 z2  
               "RAD "   r  
               <"TOLE"  tol> ] | /LECTURE/
```

p0

Base value of the pressure. The instruction "TABL" must follow to determine the variation in time, even for constant pressures (see note below).

PAX1, x1, y1, z1

Coordinates of the first point defining the axis of the cylindre.

PAX2, x2, y2, z2

Coordinates of the second point defining the axis of the cylindre. WARNING: PAX1 and PAX2 define the direction of the cylinder's axis but are also used as the height to calculate the area of its surface. So, the distance between the two points must be equal to the cylinder's height.

RAD rad

Cylinder's radius.

TOLE tol

Pressure is only applied on the elements that are tangent to the outer surface of the cylinder. Tol value defines the relative precision considered to determine whether the element is tangent or not. Default value: 1E-2.

LECTURE

Group of discrete elements where the search of tangent elements is done.

### Comments:

A procedure selects the elements that are tangent to the outer surface of the cylinder. For each selected element, the pressure application area is calculated with the element's radius  $r$ . In fact, we consider this area is the area of the circle of radius  $r$  ( $A = \pi * r^2$ ).

For a defined peak value  $P_0$ , the intensity of the pressure is:

$$P(t) = P_0 * C(t)$$

So, for each selected element, the applied force modulus is:

$$F(t) = P(t) * A$$

This force is applied in the direction of the normal vector  $V$  to the cylinder at the considered element.  $C(t)$  is provided by the first array met after the option "PRESS" (see TABLE).

### Forces adjustment:

In practice, the sum of selected elements' areas is always significantly lower than cylinder's surface area. An adjustment is done to correct this point by multiplying the calculated force by the ratio of the cylinder's outer surface area and the sum of the selected elements cross section areas.

### Sign of the pressure:

The normal to the cylinder outer surface at the considered element is the outward normal. A positive pressure creates a force in the direction of that normal.

### Warning:

The instruction "TABL" is mandatory. For the case of charges that are constant in time, just give a two-entry table with the same value in both entries, i.e. of the form "TABL 2 t1 v t2 v", where  $v$  is the (constant) value,  $t_1$  is less or equal to the initial time of the computation and  $t_2$  is greater or equal to the final time of the computation.

### 10.3.4 ADDITIONAL ACCELERATION

**Object:**

This option enables additional acceleration depending on time to be prescribed.

**Syntax:**

```
"ACCE"  /LECDDL/  g0  /LECTURE/
```

**LECDDL**

Reading procedure of the numbers of the degrees of freedom concerned.

**g0**

Base value of the acceleration to be added. The instruction "TABL" must follow to determine the variation in time, **even for constant displacements** (see note below).

**LECTURE**

Numbers of the nodes concerned.

**Comments:**

The external forces applied on the nodes defined by the procedure LECTURE and along the directions determined by LECDDL, are of the following type:

$$F(t) = g0 * M * C(t)$$

M is the nodal mass. C(t) is provided by the first array met after that option (see "TABL").

That option can be used to enter gravity like forces depending on time (for example, deceleration forces inside a tank during a crash).

**Warning:**

The instruction "TABL" is mandatory. For the case of charges constant in time, just give a two-entry table with the same value in both entries, i.e. of the form "TABL 2 t1 v t2 v", where v is the (constant) value, t1 is less or equal to the initial time of the computation and t2 is greater or equal to the final time of the computation.

### 10.3.5 TABLE

**Object:**

The tables provide the different values of the coefficients  $C(t)$  which appears in the options of the instruction "CHARGE FACTORISEE" (factorised load).

The  $C(t)$  functions are continuous, linear by parts and therefore defined by points. They can linearly interpolate more complex functions.

**Syntax:**

```
"TABLE"  npts*( tk  ck )
```

**npts**

Number of couples (tk, ck) defining the table.

**tk**

Elementary time.

**ck**

Multiplying factor at the time tk:  $ck=C(tk)$ .

**Comments:**

Each array refers to the options "DEPL", "FORC", "PRES" which are defined in the data set before the table and follow the preceeding table, if it exists. The first table refers to all options defined after the key-word "FACT". It is not allowed to have two consecutive tables in the data set.

If the time  $t_1 = 0$  is not specified in the table, the point of origin (0, 0) is assumed to belong to the curve.

The last time used in the array must be greater than the final time of the computation.

## 10.4 PROGRAMMED LOADS

### Object:

This option enables the user to enter forces or pressures applied to certain nodes or certain elements of the structure, for time instants defined by the user himself.

In this case, the values are defined at each time by linear interpolation.

### Syntax:

```
"PROG"  ndprog    $  "FORCE"    . . .    $
                  $  "PRESSION" . . .    $
                  "ROUTINE"   . . .
                  < "ROUTINE" . . .    >
```

### ndprog

Number of the disc (logical unit number) where the programmed loads are stored. This disc is temporary (the data is destroyed after the creation of a disc of loads depending on time ndcha).

### Comments:

The word "PROG" is the first key-word of the option. It may be used only once in the EUROPLEXUS data set.

There are 2 subdirectives "PRES" and "FORC" which respectively enable pressures or forces to be input.

For each of them, the user has to initially provide a list, which contains according to the circumstances:

- The numbers of the nodes defining the pressure line or the numbers of the elements to which pressures are applied.

- The numbers of the degrees of freedom and the numbers of the nodes to which forces are applied.

The list is entered by the means of the procedures LECTURE and LECDDL.

There are three ways to enter the values of the forces and pressures at the different time increments.

1/- The user directly inserts, into the data set, the cards which each correspond to one time increment. Each card successively contains :

- The value of the time increment concerned.
- The values of the pressures and forces determined according to the preceeding list.

2/- The user has at his disposal or creates a data file with an imposed standard writing format. The file must successively store several sets of values, each corresponding to one time increment. Each set sequentially defines :

- The value of the time increment concerned
- The values of the pressures and forces determined in the same way as the cards.

3/- Regardless of the program, the user provides a subroutine which computes the values of the forces or pressures for each time increment before the program runs. These values are stored in array F or P according to the list.

The data can be read on a file the format of which is not standard, or the values can be directly entered by the means of an analytic formula chosen by the user.

The keywords "ROUTINE" introduce respectively the data associated with each of the subroutines FPROG and PPROG, written by the user.

ATTENTION: the data relative to FPROG are read first.

If there are only forces (or only pressures), a single keyword "ROUTINE" is sufficient to introduce the corresponding data.

**10.4.1 FORCE****Object:**

For each time increment, this option enables nodal forces to be applied to certain nodes of the structure and according to certain directions (degrees of freedom).

**Syntax:**

```

$ "FORCE" "DDL" /LECDDL/ /LECTURE/ ... $
$ "MXTF" ntmax | "CART" ( "INST" ti f1 ... fn ) | $
$ | "BAND" nb | $
$ | "ROUTINE" | $

```

**LECDDL**

Reading procedure of the degrees of freedom to which the forces are applied.

**LECTURE**

Reading procedure of the numbers of the nodes to which the forces are applied.

**ntmax**

Maximum number of time instants for which the loads are defined.

**CART**

Keyword which enables the input of cards.

**BAND**

Keyword which enables the data on a file to be read.

**nband**

File number.

**ROUT**

Keyword which enables the user to provide a computation subroutine. After the word ROUT, and on a new card, the user can, if he wants, provide the data which is read by the subroutine in the order of the writing.

**Comments:**



The directive "FORCE" may be used at most once.

For an axisymmetric calculation, the forces must be divided by  $2\pi$ , since the calculation refers to ONE radians.

The elements of the list defined by LECDDL and LECTURE are stored according to their nodes and each node according to its degree of freedom, in the order of their definitions in the procedures LECTURE and LECDDL.

For example :

"FORC" "DDL" 123 "LECT" 7 8 10 "TERM"

will define the list  $n(1,7)$   $n(2,7)$   $n(3,7)$   $n(1,8)$   $n(2,8)$   $n(3,8)$   $n(1,10)$   $n(2,10)$   $n(3,10)$  where  $n(i,j)$  represents the  $i$  th degree of freedom of the node  $j$  . The 7 th element of this list is the first degree of freedom of the 10 th node.

The parameter  $ntmax$  represents the maximum number of time increments for which the loads are defined. At the most  $ntmax$  cards or  $ntmax$  data sets can be read on the file. If a subroutine is entered, it is used  $ntmax$  times.

The user must choose between 3 input modes for the data :

CARD ( "CARD" )

FILE ( "BAND" )

SUBROUTINE ( "ROUTINE" )

Only one can be used.

### **File:**

The parameter  $nband$  represents the number of the file from which the data is read. The file has been first defined at the level of the control cards.

The different sets of values are written on that file. Each set contains:

- the value of the time increment to which the data of the set is associated;
- the values of the forces  $F(j,t)$ , as for the cards.

The number of the values  $F(j,t)$  must corresponds to the number of the elements defined in the LECDDL-LECTURE list, that is to say  $s$  values (  $s$  : number of nodes ;  $x$  : number of degrees of freedom). The  $j$  th value  $F(j,t)$  represents the value of the force applied to the node at time  $t$ , according to the direction defined by the  $j$  th element of the list.

The values are written on the file unformatted.

If the user reaches the end of the file before he has read all  $nt$  value sets, EUROPLEXUS considers that there are no more values to be read. In this case, the loading is finished and the program goes to the next option or instruction.

If there are more than  $n_t$  data sets, the loading is considered as finished after the reading of the  $n_t$  th set.

In both cases, the last time increment must be greater than the final time of computation defined in the instruction "CALCUL".

The reading of the nband number leads EUROPLEXUS to read the file automatically.

**Subroutine:**

The key-word "ROUT" must be the last data of the card. It automatically calls the user's subroutine. At the most, the latter will be used  $n_t$  times (but if he wants, the user can stop the subroutine before the  $n_t$  th . After each call, the subroutine provides the EUROPLEXUS program with an array (F or P) which contains as many values as elements defined in the LECDDL-LECTURE list. The  $j$  th value of the table is the value of the node to which a force is applied according to the direction determined by the  $j$  th element of the list. If he wants, the user can provide, after the word "ROUTINE", the data which are written on new cards. This data is read sequentially and in the order required by the subroutine. For more explanations, see the chapter "PROGRAMMED LOADS-SUBROUTINE".

### 10.4.2 CARDS

**Object :**

This suboption enables loads (forces or pressures) to be read on cards.

**Syntax :**

```
"CARTES" ( "INSTANT" ti f1 ... fs ) "TERMINE"
```

**INSTANT**

First keyword of each card.

**ti**

Time instant to which the data of the card is associated.

**f1 ... fs**

Value of the force applied to the node and degree of freedom defined at the  $j$  th place of the list entered by LECDDL-LECTURE, at time  $t_i$ . There are  $s$   $f_j$  values per card ( $s$  representing the product of the number of degrees of freedom and the number of nodes).

**TERMINE**

This key-word denotes that there is no more data to be read. The loading is finished and the program takes the next option.

**Comments:**

At the maximum, the word "INSTANT" is written  $nt$  times.

Each card must include, after the value of the time increment  $t_i$ , as many  $f_j$  values as elements defined in the LECDDL-LECTURE list, that is to say  $s$  values. The  $j$  th  $f_j$  value is the value of the force which is applied, at time  $t_i$ , to the node and according to the direction defined by the  $j$  th element of the list. All the numerical values are read in free format.

The word "TERM" indicates that there is no more data to be read. The program considers the loading as finished and takes the next option or instruction.

The last time increment to be read must be greater than to the final time of computation defined in the instruction "CALCUL".

The word "CARTE" leads EUROPLEXUS to read the cards automatically.

### 10.4.3 PRESSURE

#### Object:

This option enables the following pressures (expressed in real number form) to be entered for each time instant:

- a pressure which is exerted on a set of adjacent segments or "pressure line", for two-dimensional computations.
- a pressure which is exerted on shell elements defining a surface, for two- and three-dimensional computations.
- a pressure which is exerted on the faces of solid elements defining a "pressure surface".

#### Syntax:

```

$ "PRESS"          | "SEGM" /LECTURE/          | ...    $
$                  | "COQU" /LECTURE/          |        $
$                  | "FACE" iface /LECTURE/ |        $
$ "MXTP" ntmax     | "CART" ( "INST" ti f1 ... fn ) |        $
$                  | "BAND"  nb                |        $
$                  | "ROUTINE"                  |        $

```

#### SEGM

This concerns two-dimensional computations; the user enters a pressure line. Then, the procedure LECTURE is used for the input of the numbers of the nodes composing the line. On the whole, there are  $s + 1$  nodes defining  $s$  segments.

#### COQU

This concerns two- or three-dimensional computations; the user enters a surface composed of shell elements. Then, the numbers of the shell elements are entered by the procedure LECTURE.

#### FACE

This concerns three-dimensional computations; the user enters a surface composed of the faces of solid elements.

#### iface

Number of the solid element face belonging to the surface. In this case, the procedure LECTURE is used for the input of the solid elements concerned.

#### ntmax

Maximum number of time instants where the pressures are defined.

**Comments:**

The input syntax of the data is the same as for the "PROGRAMMED FORCES".

The words "CART" "BAND" "ROUTINE" and the parameters ntmax and nb have the same signification as for the "PROGRAMMED FORCES".

As for the latter, the j-th value defined after the time increment, on the cards or in the data sets written on the file, gives the value of the pressure which is exerted, at that time increment, on the j-th element of the list provided by the procedure LECTURE.

The j th element of the array provided by the subroutine has the same meaning (see the chapter "PROGRAMMED FORCES").

For a two-dimensional computation, the options "SEGM" and "COQU" can be used one after the other.

For example :

```
"PRES"  "SEGM"  "LECT" 1 2 3 4 "TERM"  
        "COQU"  "LECT" 9 13 18 "TERM"
```

The list given by the two procedures LECTURE is respectively composed by the 3 segments 1-2 2-3 3-4, and by the 3 shells 9 13 18. The 6-th element of the list represents the shell number 18.

For three-dimensional computation, the options "COQU" and "FACE" can be used one after the other.

For example :

```
"PRES"  "COQU"  "LECT" 9 10 25 "TERM"  
        "FACE" 3 "LECT" 16 2 12 "TERM"
```

The list given by the two procedures LECTURE is respectively composed of the shell elements 9 10 25, and the solid elements 16 2 12. The 4-th element of the list represents the solid element number 16.

If there are two procedures LECTURE, the list which must take into account the values of the cards, data sets or tables may be obtained by putting the 2 lists defined by the procedures end to end, in the order of their definitions ( the second list follows the first).

If he wants, the user can define several times the words "SEGM" "COQU" or "COQU" "FACE" with the corresponding lists, and in any order. The principle used to obtain the final list on which the loads are defined is the same one used if there are only two LECTURE procedures.

Examples:

```
1/ 2-D    "PRES"  "SEGM"  /LECTURE/  
          "COQU"  /LECTURE/  
          "SEGM"  /LECTURE/  
  
2/ 3-D    "PRES"  "FACE" 1 /LECTURE/  
          "COQU"  /LECTURE/  
          "FACE" 4  /LECTURE/
```

For the conventions relative to the pressure signs, see the chapter on FACTORIZED LOADS and the corresponding options.

#### 10.4.4 ROUTINE

**Object:**

This is a FORTRAN subroutine provided by the user and compiled independently from the program.

There is one subroutine for the forces (FPROG) and one for the pressures (PPROG). For each time increment, these subroutines create an array (F for forces and P for pressures), containing the values of the forces and the pressures according to the list supplied by the LECDDL - LECTURE procedures of the options "PROG" "FORC" or "PROG" "PRES".

**Syntax:**

1. Programmed forces :

```
SUBROUTINE FPROG(KAR,IMP,IPT,T,N,NUF,FEP,TAB)
IMPLICIT DOUBLE PRECISION(A-H,O-Z)
DIMENSION FEP(*),TAB(*),NUF(2,*)
. . .
  FORTRAN data set to compute the forces
. . .
RETURN
END
```

2. Programmed pressures :

```
SUBROUTINE PPROG(KAR,IMP,IPT,T,N,IPOS,NUP,PRP,PRF,TAB)
IMPLICIT DOUBLE PRECISION(A-H,O-Z)
DIMENSION IPOS(N),NUP(*),PRP(*),PRF(N),TAB(*)
. . .
  FORTRAN data set to compute the pressures
. . .
RETURN
END
```

**KAR**

Input variable. At the origin, it is a number of the file the value of which is 0 by default (card reader). It can be read or defined again inside the subroutine. If KAR is negative at the exit of the subroutine EUROPLEXUS considers that the loading is finished and stops the operation.

**IMP**

Input variable, number of the output (listing) file. It may be used to write messages to the listing. The value of IMP cannot be defined again inside the subroutine.

**IPT**

Input variable. It is a pointer which is increased by 1 by PLEXIS-3C after each use of the subroutine. IPT is equal to 1 at the first call. The value of IPT cannot be defined again inside the subroutine.

**T**

Output variable. At the exit of the subroutine T must be equal to the value of the elementary time linked to the values of the forces and the pressures (array FEP or PRP).

**FEP or PRP**

Output arrays. At the exit of the subroutine, they must respectively mention the values of the forces and the pressures, according to the list provided by the procedures LECDDL and LECTURE of the options "PROG" "FORC" and "PROG" "PRES".

**N**

Input variable equal to the number of the elements on the list. The arrays FEP and PRP are automatically dimensioned at N. The value of N must not be defined again inside the subroutine.

**NUF**

Indexes of nodes and ddls where a force is applied.  $NUF(1,K)$  = number of k-th node in the list;  $NUF(2,K)$  = associated ddl.

**IPOS**

Pointer on NUP and PRP:  $KDEB = IPOS(I)$  : address of beginning of face I;  $KFIN = IPOS(I+1) - 1$  : address of the end of face I;  $NBNF = IPOS(I+1) - IPOS(I)$  : number of nodes of face I.

**NUP(KDEB:KFIN)**

Indexes of nodes of face I.

**PRP(KDEB:KFIN)**

Pressures on nodes of face I.

**PRF**

Work array (dimension N). For example, pressures applied on the N faces at time T.

**TAB**

Work array. It enables (if necessary) to store values during the run of the subroutine (see example).

**Comments:**

The user's subroutine is called automatically nt times (nt is the maximum number of elementary times). It has been defined in the options "PROG" "FORC" and "PROG" "PRES".



However the user can stop the calling sequences before having achieved  $nt$  calls, by giving a negative value to  $NB$  at the exit of the subroutine (the call relative to the negative definition of  $NB$  is not taken into account).

If the user wants to define loads for  $n$  elementary times, with  $n$  inferior to  $nt$ , the value of  $NB$  must be negative at the  $(n+1)$ -th call. In any case, the last elementary time which has been computed must be superior to the final time of computation defined in the instruction "CALCUL".

## SAMPLE ROUTINE FPROG

Here is the sample routine FPROG contained in the program:

```
C FPROG      SOUPLEX   LPRE      91/06/07    19:19:31
      SUBROUTINE FPROG(KAR,IMP,IPT,T,N,NUF,FEP,TAB)
C
C  OBJET : LIRE LES FORCES AUX NOEUDS POUR UN INSTANT CONSIDERE
C  -----
C
C ==> EN RETOUR : CHARGER T ET REMPLIR FEP CORRECTEMENT
C
C   KAR : NUMERO DU FICHIER DE LECTURE ET INDICATEUR D'ARRET
C   IMP : NUMERO DU FICHIER DE SORTIE (IMPRIMANTE)
C   IPT : NUMERO DE L'INSTANT CONSIDERE
C   T   : VALEUR DU TEMPS CONSIDERE
C   N   : NOMBRE DE NOEUDS OU UNE FORCE EST APPLIQUEE
C   NUF(1,K) : NUMERO DU KIEME NOEUD DE LA LISTE
C   NUF(2,K) : DDL CONCERNE
C   FEP(K)  : FORCE CORRESPONDANTE
C   TAB : TABLEAU DE TRAVAIL (DIM AJUSTABLE AVEC "TRAV" )
C
C ==> IMPORTANT : NE PAS MODIFIER IPT, N, NUF.
C
C   SI KAR EST MIS < 0 , ON ARRETE LA LECTURE AVANT D'AVOIR EPUISE
C   TOUS LES INSTANTS PREVUS
C
C
C   IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C   CHARACTER*(10) MOT
C
C   PARAMETER (NFORC=20)
C
C   DIMENSION NUF(2,*),FEP(*),TAB(NFORC,2)
C   DIMENSION X(3)
C
C
C   IF(KAR.LE.0) RETURN
C
C   READ (KAR,*) MOT
C   IF(MOT.EQ.'TERMINE') GOTO 91
C   IF(MOT.NE.'INSTANT') GOTO 90
C
C---  LECTURE D'UN BLOC (A T DONNE : NP COUPLES R,F )
      READ (KAR,*) T,NP
      IF(NP.GT.NFORC) GOTO 88
      READ (KAR,*) (TAB(K,1),TAB(K,2),K=1,NP)
C
      WRITE (IMP,*) ' ==> INSTANT : ',T
      CALL DPRINT(IMP,NP,TAB(1,1),'RAYON')
```

```
      CALL DPRINT(IMP,NP,TAB(1,2),'VALEUR')
C
C---  COORDONNEES DU CENTRE :
      XC=0
      YC=0
C
C---  CALCUL DES FORCES PAR INTERPOLATION LINEAIRE
      DO 50 K=1,N
      NUPO=NUF(1,K)
      CALL QUIDNE(0,NUPO,LON,X)
      IF(LON.NE.3) GOTO 89
      DX=X(1)-XC
      DY=X(2)-YC
      R=SQRT(DX*DX+DY*DY)
      CALL DITPL1(NP,TAB(1,1),TAB(1,2),R,F,DFSDR,NX,IER)
      IF(IER.NE.0) GOTO 87
      FEP(K)=F
50    CONTINUE
      RETURN
C
C---  ERREURS ET SORTIE
87    CALL ERRMSS('FPROG','INTERPOLATION INCORRECTE')
      STOP
88    CALL ERRMSS('FPROG','IL Y A TROP DE VALEURS')
      STOP
89    CALL ERRMSS('FPROG','IL N Y A PAS 3 COORDONNEES')
      STOP
90    CALL ERRMSS('FPROG','SYNTAXE INCORRECTE')
      STOP
91    KAR=-KAR
      RETURN
C
      END
```

## SAMPLE ROUTINE PPROG

Here is the sample routine PPROG contained in the program:

```
C PPROG      SOUPLEX   LPRE      91/06/07      19:19:12
      SUBROUTINE PPROG(KAR,IMP,IPT,T,N,IPOS,NUP,PRP,PRF,TAB)
C
C  OBJET : LIRE LES PRESSIONS AUX NOEUDS POUR UN INSTANT CONSIDERE
C  -----
C
C ==> EN RETOUR : CHARGER T ET REMPLIR PRP CORRECTEMENT
C
C      KAR : NUMERO DU FICHIER DE LECTURE ET INDICATEUR D'ARRET
C      IMP : NUMERO DU FICHIER DE SORTIE (IMPRIMANTE)
C      IPT : NUMERO DE L'INSTANT CONSIDERE
C      T   : VALEUR DU TEMPS CONSIDERE
C      N   : NOMBRE DE FACES SOUS PRESSION
C      IPOS : POINTEUR SUR NUP ET PRP :
C              KDEB = IPOS(I) : ADRESSE DU DEBUT DE LA FACE I
C              KFIN = IPOS(I+1)-1 : ADRESSE DE LA FIN DE LA FACE I
C              NBNF = IPOS(I+1)-IPOS(I) : NBR DE NOEUDS DE LA FACE I
C      NUP(KDEB:KFIN) : NUMEROS DES NOEUDS DE LA FACE I
C      PRP(KDEB:KFIN) : PRESSION AUX NOEUDS DE LA FACE I
C      PRF : TABLEAU DE TRAVAIL (DIM N) : PAR EXEMPLE :
C              PRESSIONS APPLIQUEES SUR LES N FACES AU TEMPS T
C      TAB : TABLEAU DE TRAVAIL (DIM AJUSTABLE AVEC "TRAV" )
C
C ==> IMPORTANT : NE PAS MODIFIER IPT, N, IPOS, NUP.
C
C      SI KAR EST MIS < 0 , ON ARRETE LA LECTURE AVANT D'AVOIR EPUISE
C      TOUS LES INSTANTS PREVUS
C
C      SI NUP(K) EST > 10000 : IL S'AGIT D'UN NOEUD APPARTENANT A UN
C      ELEMENT DE COQUE (SINON EL. MASSIF), LE NUMERO EST ALORS
C      NUCO=MOD(NUP(K),10000)
C
C
C      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C      CHARACTER*(10) MOT
C
C      PARAMETER (NPRES=20)
C
C      DIMENSION IPOS(N),NUP(*),PRP(*),PRF(N),TAB(NPRES,2)
C      DIMENSION X(3)
C
C
C      IF(KAR.LE.0) RETURN
C
C      READ (KAR,*) MOT
C      IF(MOT.EQ.'TERMINE') GOTO 91
```

```
      IF(MOT.NE.'INSTANT') GOTO 90
C
C---  LECTURE D'UN BLOC  (A T DONNE : NP COUPLES R,P )
      READ (KAR,*) T,NP
      IF(NP.GT.NPRES) GOTO 88
      READ (KAR,*) (TAB(K,1),TAB(K,2),K=1,NP)
C
      WRITE (IMP,*) ' ==> INSTANT : ',T
      CALL DPRINT(IMP,NP,TAB(1,1),'RAYON')
      CALL DPRINT(IMP,NP,TAB(1,2),'VALEUR')
C
C---  COORDONNEES DU CENTRE :
      XC=0
      YC=0
C
C---  CALCUL DES PRESSIONS PAR INTERPOLATION LINEAIRE
      DO 50 IFA=1,N
      KDEB=IPOS(IFA)
      KFIN=IPOS(IFA+1)-1
      NBNF=IPOS(IFA+1)-KDEB
      PRF(IFA)=0
      DO 40 K=KDEB,KFIN
      NUCO=MOD(NUP(K),10000)
      CALL QUIDNE(0,NUCO,LON,X)
      IF(LON.NE.3) GOTO 89
      DX=X(1)-XC
      DY=X(2)-YC
      R=SQRT(DX*DX+DY*DY)
      CALL DITPL1(NP,TAB(1,1),TAB(1,2),R,P,DPSDR,NX,IER)
      IF(IER.NE.0) GOTO 87
      PRP(K)=P
      PRF(IFA)=PRF(IFA)+P
40    CONTINUE
      PRF(IFA)=PRF(IFA)/NBNF
50    CONTINUE
      RETURN
C
C---  ERREURS ET SORTIE
87    CALL ERRMSS('PPROG','INTERPOLATION INCORRECTE')
      STOP
88    CALL ERRMSS('PPROG','IL Y A TROP DE VALEURS')
      STOP
89    CALL ERRMSS('PPROG','IL N Y A PAS 3 COORDONNEES')
      STOP
90    CALL ERRMSS('PPROG','SYNTAXE INCORRECTE')
      STOP
91    KAR=-KAR
      RETURN
C
      END
```

**EXAMPLE 1****Problem:**

The user wants to enter FORCES into the directions x and y (degrees of freedom 1 and 2). These forces are applied to the nodes 1, 3, 5, 7, 9, 11.

Their values are the same for all the nodes, for both directions; these values are defined by analytical formulas:

$$F = 2.5 * \sin(\pi * t) \text{ For } t \leq 20 \text{ milliseconds}$$

$$F = 2.9 * \sin(2 * \pi * t) \text{ For } t > 20 \text{ milliseconds}$$

The programming concerns 51 elementary times, a value is calculated for each millisecond from 0 to 50.

At each elementary time, the user must provide  $2*6=12$  values stored in array F (2 degrees of freedom and 6 nodes).

**The user's subroutine:**

```

      SUBROUTINE FPROG(KAR,IMP,IPT,T,N,NUF,F,TAB)
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
C
      DIMENSION F(*),TAB(*)
      DATA PI/3.1416/
C
      T=1E-3*(IPT-1)
      IF(IPT.GT.21) GOTO 20
      DO 10 I=1,N
10    F(I) = 2.5 * DSIN( PI * T )
      RETURN
20    IF(IPT.GT.51) GOTO 40
      DO 30 I=1,N
30    F(I) = 2.9 * DSIN( 2*PI * T )
      RETURN
40    KAR=-KAR
      RETURN
      END

```

**Data of the instruction "CHARGE":**

```

"CHARGE" 1 "PROG" 2 "FORCE" "DDL" 12 "LECT" 1 PAS 2 11 "TERM"
          "MXTF" 60 "ROUTINE"

```

**Comments:**

The user can directly define these coefficients inside the subroutine. There are no data cards of the user behind the word "ROUTINE".

The value of ntmax must be superior or equal to 51, since the subroutine has performed a test in order to stop the loading at this value.

**EXAMPLE 2****Problem:**

The user has at his disposal a file containing data sets (elementary times and loads). It is supposed that there are a hundred values of pressure per elementary time and each data set is written under the form : (F10.5,100F15.7).

The user wants to enter only one elementary time out of ten (numbers 10, 20, 30, ...) from the initial file. For each elementary time, the user selects the pressures occupying the ranks 1, 3, 7, 9, 11, 12 and 15 among the initial data sets. Therefore, 7 values must be taken into account. The file has to be read completely and contains no more than 100 value sets. The file has been defined by number 9 at the level of the control cards.

**The user's subroutine:**

```

      SUBROUTINE PPROG(KAR,IMP,IPT,T,N,IPOS,NUP,P,PRF,TAB)
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
C
      DIMENSION P(*),TAB(*),IPOS(*),NUP(*),PRF(*)
      DIMENSION NV(7)
      DATA NV/1,3,7,9,11,12,15/
C
      IF(IPT.GT.1) GOTO 10
C----- THE PRESSURES ARE EQUAL TO ZERO AT T=0
      T=0
      DO 5 I=1,N
        5 P(I)=0
C----- READING OF THE NUMBER OF THE FILE AND THE LENGTH OF THE SETS
      READ(KAR,*) NB,NMAX
      RETURN
      10 DO 20 I=1,10
        READ(KAR,1000,END=50) T,(TAB(I),I=1,NMAX)
      1000 FORMAT(F10.5,100F15.7)
      20 CONTINUE
C----- AT THE END OF THE LOOP ON RECORDS HAVE BEEN SKIPPED
      DO 30 I=1,7
      30 P(I)=TAB(NV(I))
      RETURN
      50 KAR=-KAR
      RETURN
      END

```

**Data of the instruction "CHARGE":**



```
"CHARGE" 1 "PROG" 2 "PRESS" "COQUE" "LECT" 1 2 3 5 "TERM"  
          "FACE" 3 "LECT" 7 9 11 "TERM"  
          "MXTP" 101 "ROUTINE"  
          9 100
```

**Comments:**

The subroutine gives an example of use of the work array TAB.

The data are read at the first call and the value of the pressures at  $t = 0$  is fixed on zero. The user selects 7 values ( $N = 7$ ).

At the following calls, 9 records on the file are skipped in order to get positionned on the sets 10, 20, 30, ... 100, at the end of the reading loop. Then, the right values supplied by the work array TAB must be stored in another array (P).

At the end of the file, the value of NB is -9 to inform the EUROPLEXUS program that the loading is finished.

## 10.5 ADVECTION-DIFFUSION "LOADS" (JRC)

### Object:

This option enables the user to enter generalized loads for advection-diffusion calculations (see also keyword "ADDF" in Section A). These 'loads' include:

- imposed time-dependent temperatures
- imposed time-dependent heat fluxes
- imposed time-dependent heat generation
- imposed time-dependent heat convection
- imposed time-dependent heat radiation
- imposed time-dependent external pressure
- imposed time-dependent velocity
- imposed zero velocity
- imposed velocity parallel to a plane
- imposed velocity parallel to a line

### Syntax:

```
"ADDF"  
$ "TIMP" . . . $  
$ "FLUX" . . . $  
$ "QGEN" . . . $  
$ "CONV" . . . $  
$ "RADI" . . . $  
$ "PRES" . . . $  
$ "VELO" . . . $  
$ "BLOQ" . . . $  
$ "VPLA" . . . $  
$ "VLIN" . . . $
```

The single sub-options are described below.

### Comments:

The word "ADDF" is the first key-word of the option . It should be used only once in the EUROPLEXUS data set.

### 10.5.1 PRESCRIBED TEMPERATURE

**Object:**

To prescribe time-dependent nodal temperatures in an advection-diffusion calculation.

**Syntax:**

```
"TIMP"  nti*( "NOEU" /LECTURE/  
              "TPOI" ntp*(T,t))
```

**nti**

Number of groups of nodes using the same time function to describe a prescribed temperature.

**/LECTURE/**

List of nodal point indexes.

**ntp**

Number of (Temperature, time) couples used for the time evolution.

**T**

Temperature.

**t**

Associated time.

**Comments:**

Remember to dimension sufficiently for the number of groups (nti) and of time points (ntp), see Section A.

### 10.5.2 PRESCRIBED HEAT FLUX

**Object:**

To prescribe a time-dependent normal heat flux on element faces in an advection-diffusion calculation.

**Syntax:**

```
"FLUX"  ntf*( "NELE" nel /LECTURE/  
              "TPOI" ntp*(flux,t))
```

**ntf**

Number of groups of nodes using the same time function to describe a prescribed edge heat flux.

**nel**

Number of elements in the group.

**/LECTURE/**

The 4\*nel nodes defining element faces, in the form nel \* (n1, n2, n3, n4).

**ntp**

Number of (flux, time) couples used for the time evolution.

**flux**

Heat flux.

**t**

Associated time.

**Comments:**

Remember to dimension sufficiently for the number of groups (ntf) and of time points (ntp), see Section A.

### 10.5.3 PRESCRIBED HEAT GENERATION

**Object:**

To prescribe a time-dependent volumetric heat generation in an advection-diffusion calculation.

**Syntax:**

```
"QGEN"  ntq*( "FIRS" n1 "LAST" n2  
              "TPOI" ntp*(Q,t))
```

**ntq**

Number of groups of elements using the same time function to describe a prescribed volumetric heat generation. Each group must contain elements with consecutive indexes.

**n1**

Index number of the first element in the group.

**n2**

Index number of the last element in the group.

**ntp**

Number of (heat gen., time) couples used for the time evolution.

**Q**

Volumetric heat generation.

**t**

Associated time.

**Comments:**

Remember to dimension sufficiently for the number of groups (ntq) and of time points (ntp), see Section A.

#### 10.5.4 PRESCRIBED CONVECTIVE HEAT TRANSFER

**Object:**

To prescribe a convective heat transfer condition in an advection-diffusion calculation.

**Syntax:**

```
"CONV"  ntb*( "NELE" nel /LECTURE/  
              "TPOI" ntp*(H,T,t))
```

**ntb**

Number of groups of elements using the same time function to describe a prescribed time-dependent convective heat transfer on one edge.

**nel**

Number of elements in the group.

**/LECTURE/**

The 4\*nel nodes defining element faces, in the form nel \* (n1, n2, n3, n4).

**ntp**

Number of (heat transf. coef., flux, time) triples used for the time evolution.

**H**

Heat transfer coefficient.

**T**

Temperature.

**t**

Associated time.

**Comments:**

Remember to dimension sufficiently for the number of groups (ntb) and of time points (ntp), see Section A.

### 10.5.5 PRESCRIBED RADIATION HEAT TRANSFER

**Object:**

To prescribe a radiation heat transfer condition in an advection-diffusion calculation.

**Syntax:**

```
"RADI"  nrad*( "NELE" nel /LECTURE/  
              "TPOI" ntp*(Hr,T,t))
```

**nrad**

Number of groups of elements using the same time function to describe a prescribed time-dependent radiation heat transfer on one face.

**nel**

Number of elements in the group.

**/LECTURE/**

The 4 \* nel nodes defining element faces, in the form nel \* (n1, n2, n3, n4).

**ntp**

Number of (radiation heat transf. coef., flux, time) triples used for the time evolution.

**Hr**

Radiation heat transfer coefficient.

**T**

Temperature.

**t**

Associated time.

**Comments:**

Remember to dimension sufficiently for the number of groups (nrad) and of time points (ntp), see Section A.

### 10.5.6 PRESCRIBED TIME-DEPENDENT PRESSURE

**Object:**

To prescribe a time-dependent external pressure in an advection-diffusion calculation.

**Syntax:**

```
"PRES"  npres*( "NELE" nel /LECTURE/  
                "TPOI" ntp*(p,t))
```

**npres**

Number of groups of elements using the same time function to describe a prescribed time-dependent external pressure on one face.

**nel**

Number of elements in the group.

**/LECTURE/**

The 4\*nel nodes defining element faces, in the form nel \* (n1, n2, n3, n4).

**ntp**

Number of (pressure, time) couples used for the time evolution.

**t**

Associated time.

**Comments:**

Remember to dimension sufficiently for the number of groups (npres) and of time points (ntp), see Section A.



### 10.5.7 PRESCRIBED VELOCITIES

#### Object:

To prescribe time-dependent nodal fluid velocities in an advection-diffusion calculation.

#### Syntax:

```
"VTIM"  nvel*( "NOEU" /LECTURE/
               "TPOI" ntp*(v,t)
               $[ "PERP"                      ;
                 "PARA" "ANG1" a1 "ANG2" a2 ]$ )
```

#### nvel

Number of groups of nodes using the same time function to describe a prescribed time-dependent velocity.

#### /LECTURE/

Indexes of the nodes in the group.

#### ntp

Number of (velocity, time) couples used for the time evolution.

#### v

Velocity.

#### t

Associated time.

#### "PERP"

The nodal velocity should be perpendicular to the boundary.

#### "PARA"

The velocity should be parallel to a line defined by the following angles.

#### a1,a2

Angles of the velocity direction in spherical coordinates: a1 is the angle between the global x axis and the projection of the velocity vector on the xy plane, a2 is the angle between the z axis and the velocity vector.

#### Comments:

Remember to dimension sufficiently for the number of groups (nvel) and of time points (ntp), see Section A.

### 10.5.8 PRESCRIBED PARALLEL VELOCITIES

**Object:**

To prescribe null nodal fluid velocities or velocities parallel to a plane or to a line in an advection-diffusion calculation.

**Syntax:**

```
"VELO" $[ "BLOQ" /LECTURE/                ;  
          "VPLA" /LECTURE/                  ;  
          "VLIN" "NOEU" /LECTURE/           ;  
          $[ "PERP"                          ;  
            "PARA" "ANG1" a1 "ANG2" a2 ]$ ]$
```

**"BLOQ"**

The following nodes will have null velocity at all times.

**"VPLA"**

The following nodes will have a velocity parallel to a plane tangent to the local boundary.

**"VLIN" "NOEU"**

The following nodes will have a velocity parallel to a line defined as follows.

**"PERP"**

The nodal velocity should be perpendicular to the boundary.

**"PARA"**

The velocity should be parallel to a line defined by the following angles.

**a1,a2**

Angles of the velocity direction in spherical coordinates: a1 is the angle between the global x axis and the projection.

**Comments:**

Sub-directives "BLOQ", "VPLA" and "VLIN", if present, should appear in this order.

## 10.6 SEISMIC-LIKE LOADS FOR USE WITH SPECTRAL ELEMENTS

### Object:

To impose seismic-like loads in a domain discretized by spectral elements, for the simulation of earthquakes and wave-propagation problems.

There are two main classes of loadings:

- punctual sources, introduced by the keyword "POIN";
- plane wave sources, introduced by the keyword "PLAN";
- seismic moment sources, introduced by the keyword "SISM".

### 10.6.1 PUNCTUAL SEISMIC LOAD SOURCES

#### Object:

This directive specifies punctual sources of loadings for spectral elements. The load is computed as the product of a time function and a function of space:

$$f(x,y,z,t) = h(t) * g(x,y,z)$$

#### Syntax:

```
"POIN" <"NODP" /LECT/> |[ "BET" ; "COS" ]|
                        |[ "DELT" ; "SPRE" ; "PRES" ; "SHEA" ]|
                        "SOUR" amp "T0" t0
                        "BETA" beta "ALFA" alfa
                        <"X" x0 "Y" y0 <"Z" z0>>
                        <"NX" nx "NY" ny <"NZ" nz> >
```

#### "NODP"

The load will be applied to the node specified in the following /LECT/. Only one nodal point must be specified. This directive may be used in alternative to the load point coordinates X, Y, Z.

#### "BET"

The time function  $h(t)$  has the form:

$$h(t) = \text{amp} * (1 - 2*\text{beta}*(t - t_0)^2) * \exp(-\text{beta} * ((t - t_0)^2))$$

#### "COS"

The time function  $h(t)$  has the form:

$$h(t) = \text{amp} * \cos(\text{Pi}*\text{beta}*(t - t_0)) * \exp(-0.5*\text{beta}^2 * ((t - t_0)^2))$$

#### "DELT"

The spatial distribution is punctual (delta function), i.e. the force only acts in the point (node) specified below by (x0, y0, z0). Note that this force is an absolute one, i.e. it does not depend on the nodal volume.

#### "SPRE"

The spatial distribution is a radial function:

$$g(x,y,z) = \exp(- \text{alfa}^2 * R^2)$$

.

where  $R^2$  is the distance from point  $(x_0, y_0, z_0)$ . Note that this force is a volume force, i.e. it is scaled for each node by the nodal volume. All points at the same distance  $R$  from  $(x_0, y_0, z_0)$  have the same value of  $g(x,y,z)$ , but note that the direction of the force is NOT radial, but is defined by  $(n_x, n_y, n_z)$ .

#### "PRES"

Similar to SPREAD but the direction is radial, thus there is no need to define  $(n_x, n_y, n_z)$ . Like SPRE, this force is volumetric, i.e. scaled by the nodal volume:

$$f_x = h(t) * 2 * \alpha^2 * (x - x_0) * \exp(-\alpha^2 * R^2)$$

$$f_y = h(t) * 2 * \alpha^2 * (y - y_0) * \exp(-\alpha^2 * R^2)$$

$$f_z = h(t) * 2 * \alpha^2 * (z - z_0) * \exp(-\alpha^2 * R^2)$$

#### "SHEA"

Similar to PRES but the force is directed along the tangential direction (and not along the radial one). Also here  $(n_x, n_y, n_z)$  are redundant. Furthermore, this kind of loading is only available in 2D. Like SPRE, this force is volumetric, i.e. scaled by the nodal volume. The components are:

$$f_x = -h(t) * 2 * \alpha^2 * (y - y_0) * \exp(-\alpha^2 * R^2)$$

$$f_y = h(t) * 2 * \alpha^2 * (x - x_0) * \exp(-\alpha^2 * R^2)$$

#### "SOUR"

Introduces a source, that will be characterized by its parameters AMP, T0 etc.

**amp**

Amplitude of the time function.

**t0**

The value appearing in the expression of  $h(t)$ . This time corresponds to the maximum of the time function:  $h(t_0) = \text{amp}$ .

**beta**

The beta value appearing in the expression  $h(t)$ . It governs the exponential decay of the time function. Note that it is possible to avoid the exponential decay in time of the load by specifying a negative value for beta. In this case, the exponential function is eliminated from the expression of  $h(t)$ , and the absolute value of beta is used. In the BET case:

$$h(t) = \text{amp} * (1 - 2 * \text{abs}(\text{beta}) * (t - t_0)^2)$$

.

In the COS case:

$$h(t) = \text{amp} * \cos(\text{Pi} * \text{abs}(\text{beta}) * (t - t_0))$$

**alfa**

The alfa value appearing in the expression  $g(x,y,z)$ . It governs the exponential decay of the space function.

**x, y, z**

Coordinates of the source. The load will be applied to the node closest to point x,y,z. If these are given, use of NODP is not permitted.

**nx, ny, nz**

Define the normal direction for DELTA or SPREAD.

**Comments:**

These loads are all volumetric (specific) forces, except in the case of DELT. This is in order to avoid mesh dependency of the total energy introduced in the system.

Therefore, amp represents the total force (per unit volume) acting on all affected nodes at time  $t_0$ , and this quantity varies in time according to  $h(t)$ .

In the special case of DELT, the user is responsible of calibrating the (non-volumetric) force with respect to the nodal mass of the application point.

## 10.6.2 PLANE WAVE SEISMIC LOAD SOURCES

### Object:

This directive specifies loading sources in the form of distributed loadings for spectral elements. The load is applied to all nodal points (typically) along a plane (or a line in 2D) and varies in time according to the same time function in all loaded points (except in the case "DLEN", see below).

### Syntax:

```
"PLAN" |[ "BET" ; "COS" ]|
        "SOUR" "AMP" amp "T0" t0
                "BETA" beta <"DLEN" dlen>
                "NX" nx "NY" ny <"NZ" nz>
                "NOEU" /LECTURE/
```

#### "BET"

The time function  $h(t)$  has the form:

$$h(t) = \text{amp} * (1 - 2*\text{beta}*(t - t_0)^2) * \exp(-\text{beta} * ((t - t_0)^2))$$

#### "COS"

The time function  $h(t)$  has the form:

$$h(t) = \text{amp} * \cos(\text{Pi}*\text{beta}*(t - t_0)) * \exp(-0.5*\text{beta}^2 * ((t - t_0)^2))$$

#### "SOUR"

Introduces a source, that will be characterized by its parameters AMP, T0 etc.

#### amp

Amplitude of the time function at  $t = t_0$ .

#### t0

The value appearing in the expression of  $h(t)$ . This time corresponds to the maximum of the time function:  $h(t_0) = \text{amp}$ .

#### beta

The beta value appearing in the expression  $h(t)$ . It governs the exponential decay of the time function.

#### dlen

If this value is given, then the loading model of ELSE is strictly used. In this model the load applied to nodal points is proportional to the time derivative of the function  $h(t)$ , in order to obtain  $h(t)$  as an imposed displacement. The nodal force is then:

$$F = 2 * \rho * c * (dh/dt) * vol * fac$$

.

where  $\rho$  is the density,  $c$  is the wave propagation speed,  $vol$  is the nodal volume and  $fac$  is a scaling factor, defined as:

$$fac = 2 / (dlen * W)$$

.

where  $W$  is the Gauss-Lobatto weight. The physical meaning of  $dlen$  is the maximum height of the spectral elements strip in which the load is prescribed. Note, however, that in this case the following restriction applies (coming from the ELSE implementation): the line of nodal points affected by the load must coincide with the one given in ELSE (typically, an horizontal row of nodes, in a region where the mesh must be structured). Therefore, this option makes only sense for strict comparisons between ELSE and Ahnse.

**nx, ny, nz**

Define the direction for the loading source.

**/LECT/**

Nodal points in which the loading acts. The code does NOT require that the given nodes actually lie on a line or a plane (3D). The user may therefore specify a non-linear or non-planar source if so desired.

### Comments:

These loads are volumetric (specific) forces. This is in order to avoid mesh dependency of the total energy introduced in the system.

Therefore,  $amp$  represents the total force (per unit volume) acting on all affected nodes at time  $t_0$ , and this quantity varies in time according to  $h(t)$ .



### 10.6.3 SEISMIC MOMENT LOADS

#### Object:

This directive specifies loading sources in the form of seismic moment loadings for spectral elements. The load is applied to a 'fault' defined by a series of nodes, introduced by the keyword NOEU. The program verifies the alignment of the given nodes along a line in 2D and on a plane in 3D. As a special case, a single fault node can be specified (pointwise loading).

A slipping vector must be defined by keywords SX, SY, SZ. The user can also specify the normal to the fault by the keywords NX, NY, NZ. If the normal is missing, the program assumes it coincident with the slipping vector.

The load applied to each nodal point varies in time according to the distance from the hypocenter, defined by its coordinates X, Y, Z. The time delay in each point depends upon the above mentioned distance and on the speed of transversal wave propagation. This speed is automatically evaluated by the code, or can alternatively be imposed by the keyword VRUP.

#### Syntax:

```
"SISM" |[ "BET" ; "COS" ] |
        "SOUR" "AMP" amp "T0" t0
          "BETA" beta
          "SX" sx "SY" sy <"SZ" sz>
          <"NX" nx "NY" ny <"NZ" nz>>
          "X" x "Y" y <"Z" z>
          <"VRUP" vrup>
          "NOEU" /LECTURE/
```

#### "BET"

The time function  $h(t)$  has the form:

$$h(t) = \text{amp} * (1 - 2*\text{beta}*(t - t_0)^2) * \exp(-\text{beta} * ((t - t_0)^2))$$

#### "COS"

The time function  $h(t)$  has the form:

$$h(t) = \text{amp} * \cos(\text{Pi}*\text{beta}*(t - t_0)) * \exp(-0.5*\text{beta}^2 * ((t - t_0)^2))$$

#### "SOUR"

Introduces a source, that will be characterized by its parameters AMP, T0 etc.

#### amp

Amplitude of the time function at  $t = t_0$ .

**t0**

The value appearing in the expression of  $h(t)$ . This time corresponds to the maximum of the time function:  $h(t_0) = \text{amp}$ .

**beta**

The beta value appearing in the expression  $h(t)$ . It governs the exponential decay of the time function.

**sx, sy, sz**

Components of the slipping vector.

**nx, ny, nz**

Components of the normal vector. If omitted, the normal direction is assumed coincident with the slipping vector.

**x, y, z**

Coordinates of the ipocenter.

**vrup**

Velocity of propagation of loading (by default, this is the speed of transverse wave propagation).

**NOEU /LECT/**

Nodal points in which the loading acts. Unlike in the **PLAN** case, the code requires that the given nodes actually lie on a line or a plane (3D).

### **Comments:**

The present model has been borrowed from the **ELSE** code of **CRS4** and is mesh-dependent in that the applied loadings depend on mesh size. A slightly modified, mesh-independedent implementation is also available in **EUROPLEXUS**, and is activated by choosing a negative value for **VRUP**.

## 10.7 NEW CONSTANT LOADS

### Object:

This directive allows to introduce constant forces or moments during the whole computation. The loads may be applied either to the nodes of a deformable body, or to the centroid of a rigid body.

### Syntax:

```
FCTE  $[ NODE ; RIGI ]$ /LECT/  
      $[ FORC f ; MOMO m ]$ VECT x y z
```

#### NODE

The loads will be applied at the following specified nodes.

#### RIGI

The loads will be applied at the following specified rigid bodies.

#### /LECT/

Concerned nodes or concerned rigid bodies (see directive **COMP RIGI**).

#### f

Force modulus.

#### m

Moment modulus.

#### x y z

Components of the vector or moment-vector. See comments below for the meaning of each component.

### Comments:

Three vector components must be specified, even in 2-D calculations. The vector is used to determine the direction of the applied force or moment. It does not need to be unitary, since it is automatically normalized immediately after reading.

The applied forces or moments must be expressed in the global reference frame  $X, Y, Z$  (for rigid bodies, the rotational values are then converted internally to the local reference frame, as appropriate). The meaning of the vector components (**VECT x y z**) is as follows:

1. For an applied force (**FORC**):

- In 2D calculations, **x** is component  $X$ , **y** is component  $Y$  and **z** must be 0 (component  $Z$ ).
- In 3D calculations, **x** is component  $X$ , **y** is component  $Y$  and **z** is component  $Z$ .

2. For an applied moment (**MOME**):

- In 2D calculations, **x** is component  $X$  and must be 0, **y** is component  $Y$  and must be 0, and **z** is component  $Z$ .
- In 3D calculations, **x** is component  $X$ , **y** is component  $Y$  and **z** is component  $Z$ .

## 10.8 IMPOSED TIME-DEPENDENT LOADS

### Object:

This directive allows to introduce time-dependent forces or moments during the transient computation. The loads may be applied either to the nodes of a deformable body, or to the centroid of a rigid body. The time-dependency of the load is specified by a function (see directive FONC on page E.15 and following ones).

### Syntax:

```
FIMP  $[ NODE ; RIGI ]$ /LECT/
      $[ FORC f ; MOME m ]$ VECT x y z NUFO nf
```

#### NODE

The loads will be applied at the following specified nodes.

#### RIGI

The loads will be applied at the following specified rigid bodies.

#### /LECT/

Concerned nodes or concerned rigid bodies (see directive COMP RIGI).

#### f

Force modulus (base value, to be multiplied by the time function).

#### m

Moment modulus (base value, to be multiplied by the time function).

#### x y z

Components of the vector or moment-vector. See comments below for the meaning of each component.

#### nf

Number of the function defining the time-dependency of the imposed load.

### Comments:

Three vector components must be specified, even in 2-D calculations. The vector is used to determine the direction of the applied force or moment. It does not need to be unitary, since it is automatically normalized immediately after reading.

The applied forces or moments must be expressed in the global reference frame  $X, Y, Z$  (for rigid bodies, the rotational values are then converted internally to the local reference frame, as appropriate). The meaning of the vector components (VECT x y z) is as follows:

1. For an applied force (**FORC**):
  - In 2D calculations, **x** is component *X*, **y** is component *Y* and **z** must be 0 (component *Z*).
  - In 3D calculations, **x** is component *X*, **y** is component *Y* and **z** is component *Z*.
2. For an applied moment (**MOME**):
  - In 2D calculations, **x** is component *X* and must be 0, **y** is component *Y* and must be 0, and **z** is component *Z*.
  - In 3D calculations, **x** is component *X*, **y** is component *Y* and **z** is component *Z*.

The defined load *A* is multiplied by a coefficient *C(t)* which varies in time and is interpolated from the specified table:

$$Q(t) = A * C(t)$$

## 10.9 DYNALPY LOADS

### Object:

This directive allows to introduce time-dependent dynalpy loads in 1-D elements of type TUBE or TUYA.

### Syntax:

```
FDYN  NODE /LECT/ PZER p0 COEF c VECT x y z ELEM e
```

/LECT/

Concerned node. Only one single node is possible to define.

p0

Reference pressure.

c

Coefficient.

x y z

Components of the vector or moment-vector.

e

Number of the concerned element.

### Comments:

Three vector components must be specified, even in 2-D calculations. The vector is used to determine the direction of the applied load. It does not need to be unitary, since it is automatically normalized immediately after reading.

These loads are only related to 1-D elements of type TUBE or TUYA and are defined as:

$$\underline{F} = c(p - p_0)\underline{v}$$

where  $p$  is the current pressure in the specified element.

## 10.10 AIR BLAST (AIRB) LOADING

### Object:

This directive allows to apply air blast loading directly to a set of structural elements (continuum, shells) without using specialized CLxx elements with an associated IMPE AIRB material. This simplifies the treatment of element erosion and mesh adaptivity.

### Syntax:

```
AIRB  |[ "X" x "Y" y <"Z" z> ; "NODE" /LEC1/ ] |
      "MASS" m $[ "TINT" t ; "TAUT" ]$ <"OPOS">
      <"ANGL">
      <"CUBE">
      <"COEF" cf>
      <"CONF" c>
      <"DECA" d>
      <"PMAX" pmax "TD" td "B" b>
      <"SHAD" /LECS/>
      /LECT/
```

**x**

X-coordinate of the explosive source.

**y**

Y-coordinate of the explosive source.

**z**

Z-coordinate of the explosive source. This is 0 by default.

**NODE /LEC1/**

Introduces the node where the explosive charge is located. Typically, a PMAT element may be located at the charge position, so as to be able to visualize it.

**m**

Mass of the explosive in Kilograms.

**t**

Starting time of the explosion. By default it is equal to the initial time of the calculation.

**TAUT**

Indicates that the starting time is calculated automatically by the code, in such a way that the air blast wave reaches the first loaded element shortly after the starting of the calculation. This is to avoid an “idle” calculation at the beginning of the transient.



## OPOS

Indicates that only the part with the positive pressure (overpressure) is regarded. After the time of duration of the positive phase the pressure is set to 0.

## ANGL

Indicates that the angle of incidence between the charge and the structural element is considered.

## CUBE

Indicates that the cubic approach will be used for the calculation of the negative phase. By default the bilinear approach is used.

## cf

The user can input a value to calibrate the decay coefficient of the air blast load. The calculated decay coefficient is multiplied by the inserted value in order to produce a load closer to experimental data.

## c

Choice between different available explosion models, see the References below. By default it is 1 (unconfined, reflected, Kingery). The term “unconfined” below means that the explosion takes place in an unconfined space, as opposed to “half-confined” where the charge is placed close to a rigid ground and so the wave propagation occurs in a half-space (experimentally, the measured pressure is somewhat lower in this case because some of the energy is absorbed by the ground). The term “reflected” hereafter means that the model accounts for the pressure increase due to (first) wave reflection at a rigid wall as it is typically measured in experiments. The pressure value in this case may be between 2 and 8 times the incident pressure in the “non-reflected” case, i.e. without taking into account this first reflection.

1. unconfined (full space), reflected (Kingery)
2. unconfined (full space), not reflected (Kingery)
3. unconfined (full space), not reflected (Kinney)
4. half-confined (half space), reflected (Kingery)
5. half-confined (half space), not reflected (Kingery)
6. Blast parameters will be directly specified next

CONF 6 indicates that the blast parameters  $p_{\max}$ ,  $t_d$  and  $b$  appearing in the so-called modified Friedlander equation (see below) will be directly specified next and should not be calculated automatically by the code. In this case, no other parameters (except CONF of course) are accepted, only the positive pressure (overpressure) is considered and the pressure-time function is identical in each element. The **modified Friedlander equation** reads:

$$p(t) = p_0 + p_{\max} \left(1 - \frac{t}{t_d}\right)^{-\frac{bt}{t_d}}$$

and expresses the pressure  $p$  as a function of time  $t$ , with  $p_0$  the initial (normally the atmospheric) pressure,  $p_{\max}$  the maximum overpressure (peak overpressure),  $t_d$  the duration of the positive pressure phase and  $b$  the decay parameter, which defines how rapidly the pressure decays.

d

Choice between different available decay coefficient equation models. Each equation is defined according to the explosion model chosen before (incident, reflected - spherical, hemispherical). The equations based on the Kingery-Bulmash data have been calculated by iteratively solving the Friedlander equation with the set of positive blast parameters proposed by Kingery-Bulmash. There are different equations for reflected or not reflected (incident) cases of unconfined (spherical) and half-confined (hemi-spherical) blast waves. An additional equation for the blast coefficient is available which is based on the Kinney and Baker data. The default blast decay equation is based on the Kingery-Bulmash data. The explosion model, that has already been defined by the parameter *c*, shows which of the blast wave decay equations (incident, reflected, spherical or hemispherical) will be used.

1. Blast wave decay equation based on Kinney data
2. Blast wave decay equation based on Kingery-Bulmash data (default)

*p*<sub>max</sub>

Maximum overpressure *p*<sub>max</sub> appearing in the modified Friedlander equation. This should only be given when CONF 6 has been specified.

*t*<sub>d</sub>

Duration of the positive pressure phase *t*<sub>d</sub> appearing in the modified Friedlander equation. This should only be given when CONF 6 has been specified.

*b*

Decay parameter *b* appearing in the modified Friedlander equation. This should only be given when CONF 6 has been specified.

SHAD /LECS/

Check for shadowing: an element is not loaded by AIRB pressure if it is “shadowed” by another element belonging to the set specified by the /LECS/ list (which may or may not coincide with the /LECT/ set of elements subjected to the AIRB loading).

/LECT/

Elements concerned. These must be of type continuum or shell.

**Comments:**

This model requires that the user adopts the standard Unit system, i.e. metres, Kilograms, seconds.

The equations of Kingery are only usable up to a scaled distance of Z=40. Above this distance, diagrams of Baker are used (linearised in the double logarithmic scale).

**References:**

For more information on the physical models, consult the following references:

- Kingery, Charles N., Bulmash, Gerald: *Airblast Parameters from TNT Spherical Air Burst and Hemispherical Surface Burst*, Defense Technical Information Center, Ballistic Research Laboratory, Aberdeen Proving Ground, Maryland, 1984.

- Baker, Wilfrid E.: *Explosions in the Air*. University of Texas Pr., Austin, 1973.
- Kinney, G.F., Graham, K.J.: *Explosive Shocks in Air*. Springer, Berlin, 1985.

## 11 GROUP G—PRINTOUT AND STORAGE OF RESULTS

### Object:

The **ECRITURE** directive enables the user to specify the requested printouts and data storages during a computation (including data saving for successive restart).

The **REGION** directive enables to define certain “regions” of the mesh, on which the printout of results will then be performed.

The **MEASURE** directive enables to request the printout of various “measurements” on the current geometrical mesh.

### Syntax:

```
< ECRITURE . . . /CTIM/
    < NOPO ; POIN /LECTURE/ >
    < NOEL ; ELEM /LECTURE/ >
    < FICH . . . > >

< REGION ( 'nom region' . . . ) >

< MEASURE ... >
```

### Comments:

The keyword **ECRI** must only appear once in an input sequence. It allows to specify the values to be printed on the listing file, and the nodes and elements for which these values must be printed.

Furthermore, the directive allows to define which results files should be produced, in view of a subsequent post-processing.

If the keyword **ECRI** is absent, a printout is performed for all the time steps, all the nodes and all the elements of the mesh.

In the following subsections, first a general description of the **ECRI** directive is given. Then, all the optional keywords (**NOPO** ...) and optional sub-directives (**FICH** ...) are described below. Output regions created by the directive **REGI** are described on page G.100. Finally, the **MEAS** directive is described on page G.105.

## 11.1 SELECTIVE PRINTOUTS ("ECRITURE")

### Object:

The "ECRITURE" directive can be used to select specific quantities to be printed out in the output listing at user-chosen times. It allows also to choose the nodes and elements for which the quantities will be printed.

The various quantities are associated to nodes or elements as described on page G.30 and following ones.

### Syntax:

```
"ECRITURE"  
    < "COOR" > < "DEPL" > < "VITE" >  
    < "ACCE" > < "FINT" > < "FEXT" > < "FLIA" >  
    < "FDEC" > < "CONT" > < "EPST" > < "ECRO" >  
    < "ENER" > < "MCVA" > < "MCVC" > < "MCVS" >  
    < "FAIL" > < "VFCC" >
```

#### COOR

Printout of nodal coordinates.

#### DEPL

Printout of nodal displacements.

#### VITE

Printout of nodal velocities. In ALE cases, both fluid and grid velocities are printed out.

#### ACCE

Printout of nodal accelerations.

#### FINT

Printout of nodal internal forces.

#### FEXT

Printout of (total) nodal external forces.

#### FLIA

Printout of nodal forces due to liaisons (coupled links: LINK COUP).

#### FDEC

Printout of nodal forces due to decoupled links: LINK DECO.

**CONT**

Printout of element stresses.

**EPST**

Printout of element total strains.

**ECRO**

Printout of element material parameters (hardening for solids, pressure and density for fluids, etc.).

**ENER**

Printout of energies.

**MCVA**

Printout of nodal quantities related to multicomponent fluids: pressure, density, temperature, sound speed and mass fractions. Note that this type of printout is incompatible with **MCVC** and **MCVS**.

**MCVC**

Printout of conserved variables (nodal quantities) related to multicomponent fluids: partial densities ( $\rho_i$ ) of the various components  $i$ , momentum ( $\rho \underline{u}$ ) (each spatial component separately), energy ( $\rho E$ ). Note that this type of printout is incompatible with **MCVA**.

**MCVS**

Printout of secondary variables (nodal quantities) related to multicomponent fluids: total density ( $\rho$ ), total pressure  $p$ , sound speed  $c$ , pressure derivative ( $\frac{\partial p}{\partial(\rho e)}$ ), absolute temperature ( $T$ ), pressure derivative ( $\frac{\partial p}{\partial(\rho_i)}$ ) for each component, mass fraction ( $\mu_i$ ) for each component. Note that this type of printout is incompatible with **MCVA**.

**MCFL**

Printout of MC numerical fluxes: partial densities (first two components), momentum (each spatial component separately) and total energy. In case of FLSR fluid-structure interaction, print out also the list of blocked MC fluxes (in the form of node couples).

**MCEF**

Printout of MC numerical "external" fluxes: partial densities (first two components), momentum (each spatial component separately) and total energy.

**MCMU**

Printout of MC "MUSCL" conserved variables (2nd order in space and time): partial densities (first two components), momentum (each spatial component separately) and total energy.

**MCVM**

Printout of MC volumes (at  $n$ ,  $n+1/2$  and  $n+1$ ) and masses (at  $n+1/2$ ,  $n+1$  and  $n+3/2$ ).

**FAIL**

Printout of reached failure level of the elements. This is only available if the element erosion model has been activated by means of the **EROS** keyword in the problem type declaration, see page A.30. A value of 0 indicates a virgin element, a value of 100 indicates a completely failed element, an intermediate value indicates the ratio (in per cent) between the number of failed Gauss points and the total number of Gauss points in the element.

**VFCC**

Printout at each selected output time of “element” quantities related to cell-centred Finite Volumes: various volume-related quantities and conserved variables.

**Comments**

The keyword “**ECRITURE**” should only appear once in an input data sequence. Keywords “**COOR**”, “**DEPL**”, etc. should immediately follow the “**ECRI**” keyword.

**Warning:**

If none of the preceding keywords is specified, nothing will be printed.

In a standard calculation (not a restart), EUROPLEXUS always prints the last computed time step.

Take care when choosing output frequencies, because the size of listing files may grow very fast.

Note that the results file of type “**ALICE**” allows to re-construct a listing. You may therefore choose to print out the bare minimum. Later on, if additional results need to be printed, you may do so by re-reading the “**ALICE**” file (which must have been specified, of course).

## 11.2 PRINTABLE QUANTITIES

### 11.2.1 NODE-RELATED QUANTITIES

For each chosen node, one may ask to print:

- current coordinates;
- displacements;
- velocities;
- accelerations;
- internal forces;
- total external forces;
- external reaction (coupled link) forces for the nodes subjected to “coupled” conditions (see LINK COUP);
- external reaction (decoupled link) forces for the nodes subjected to “decoupled” conditions (see LINK DECO);
- multi-component flow-related data (finite volumes).

**Coordinates** For each chosen node, the code prints X,Y or R,Z o X,Y,Z.

**Displacements, velocities, accelerations and forces** The chosen nodes are grouped by increasing number of degrees of freedom (d.o.f.). First all the nodes with 1 d.o.f. are printed, then those with 2 d.o.f.s, etc.

### 11.2.2 ELEMENT-RELATED QUANTITIES

For each chosen element and each integration point one may ask to print:

- the stress components (SIG);
- the deformation (EPST);
- the hardening parameters (ECR).

**Stresses and material parameters** The stress tensor and the total deformation tensor are related to the element, and independent from the material.

The material parameters, contained in the ECR table, sre independent of the element, and are only function of the material.

The choices done in EUROPLEXUS for these two quantities are detailed in the following pages.



### 11.3 STRESSES AND DEFORMATIONS

For a given element type, the stress components have always the same meaning, for whatever material is assigned to the element. On the contrary, the hardening values (ECR) are strictly related to the chosen material, and do not depend upon the element type.

The stress tensor is stored and printed in vector form, and is printed for each integration point of the element.

#### Remark 1:

In 2D, it will be necessary to distinguish the axisymmetric case (AXIS) from the plane strain (DPLA) and plane stress (CPLA) cases.

#### Remark 2:

For continuum-like elements, the stresses are written in the global reference frame, while for the other types of elements (shells, beams, bars) they are expressed in a local frame attached to the element.

#### Remark 3:

Instead of computing a bending moment, one computes a “bending stress” (sigf), that may be directly compared with the membrane stresses. This bending stress is related to the bending moment as follows:

$$\text{Moment} = E * I * K_{hi} \quad K_{hi} : \text{curvature}$$

$$\text{sigf} = E * (h/2) * K_{hi} \quad h : \text{thickness}$$

$$\text{Hence:} \quad \text{Moment} = 2 * (I / h) * \text{sigf}$$

$$\text{For a shell:} \quad \text{Moment} = (h * h / 6) * \text{sigf}$$

### 2D ELEMENTS

#### BARR and PONC

These elements may only work in traction and compression. There is just one stress component (scalar).

#### COQU

This element works in membrane and bending. There are 4 stress components per element, expressed in a local reference frame.

The first direction of the local frame is along the element from node 1 to node 2. The second, located in the mesh plane, is normal to the first. The third direction is such that the reference (u, v, w) be right-handed.

sig(1) : membrane (u)	sig(3) : bending (v)
sig(2) : membrane (w)	sig(4) : bending (w)

## COQC

Also this element works in membrane and bending. But besides the 4 preceding stress components, there is a fifth one for the shear, which is treated elastically.

sig(1) : membrane (u)	sig(3) : bending (v)
sig(2) : membrane (w)	sig(4) : bending (w)
sig(5) : shear	

## CONTINUUM ELEMENTS

The stress components are expressed in the global frame. For a calculation in plane stress, there are three stresses expressed in the (x, y) frame. For an axisymmetric calculation or a plane strain calculation there are 4 stress components, expressed in the frame (x, y, z). The z direction is the normal to the mesh plane, and such that (x, y, z) be right-handed.

1) CPLA:

$$(\text{sig}) = \begin{pmatrix} \text{SIG}(1) & \text{SIG}(3) \\ & \\ \text{SIG}(3) & \text{SIG}(2) \end{pmatrix}$$

2) AXIS or DPLA:

$$(\text{sig}) = \begin{pmatrix} \text{SIG}(1) & \text{SIG}(3) & 0 \\ & & \\ \text{SIG}(3) & \text{SIG}(2) & 0 \\ & & \\ 0 & 0 & \text{SIG}(4) \end{pmatrix}$$

## 3D ELEMENTS

### BR3D

Like for the BARR and PONC elements in 2D, these elements may only work in traction and compression. The stress tensor has just one component.

**POUT**

This element works in traction, torsion and bending. There are always 4 stresses per Gauss point, expressed in the local frame.

The first direction of the local frame (u) is along the element, from node 1 to node 2. The second (v) is in the plane defined by u and the local vector V, on the same side as V. The third one (w) is deduced from the others.

Here, due to beam assumptions, the bending stresses are expressed in the frames (u, v) and (u, w):

sig(1) : traction (u)	sig(3) : bending (u,v)
sig(2) : torsion (u)	sig(4) : bending (u,w)

**Important:**

In order to determine the local state of the beam, only the moments and the deformation have a sense! It is therefore mandatory to estimate the moments starting from the stresses, by the following relation:

$$M = \sigma \frac{I}{h}$$

The value of  $\sigma$  is read on the listing,  $I$  and  $h$  are specified in the input data set (see Chapter C1). The deformations are read directly from the listing.

For an elastic calculation ONLY, it is then possible to compute the stresses in any point of the cross-section.

**COQ3 and COQ4**

These elements work in membrane and bending. There are always 6 stress components per Gauss point, expressed in a local frame.

For the triangular elements COQ3, the first direction of the local frame (u) is along the first side of the element, from node 1 to node 2. The second (v) lies on the element plane, such that node 3 is on the positive side.

Because of shell hypotheses, the stresses are expressed in the (u, v) frame.

The quadrangular elements COQ4 are composed by 4 triangles:

1-2-3 3-4-1 1-2-4 3-4-2

Each of these triangles has a local reference frame as defined above. If the quadrangle is a parallelogram, the 4 local frames are identical.

The 4 Gauss points of element COQ4 are at the centers of the triangles mentioned above. If the element has an irregular shape, the stresses at the various Gauss points will not be directly comparable.

sig(1) : membrane (u)	sig(4) : bending (u)
sig(2) : membrane (v)	sig(5) : bending (v)
sig(3) : membrane (uv)	sig(6) : bending (uv)

### Continuum elements

The stresses are expressed in the global frame (x, y, z).

$$(\text{sig}) = \begin{pmatrix} \text{SIG}(1) & \text{SIG}(4) & \text{SIG}(6) \\ \text{SIG}(4) & \text{SIG}(2) & \text{SIG}(5) \\ \text{SIG}(6) & \text{SIG}(5) & \text{SIG}(3) \end{pmatrix}$$

#### 11.3.1 TOTAL DEFORMATIONS

The tensor of total deformations is the dual of the stress tensor. Its structure is therefore the same as that of the stresses (see the previous Section).

## 11.4 MATERIAL PARAMETERS ("ECROU")

All internal variables pertaining to the different materials are stored in the ECR table. Initially reserved only for the hardening parameters, this table has been considerably enlarged, as more complex materials have been implemented in EUROPLEXUS.

Only the simplest materials use just the first hardening quantities. For the others, the meaning of the ECR components are described within each material law description (see page C.100 and following ones).

ECR	shells	continua (solids)	continua (fluids)
ECR(1)	V.M. membrane	Pressure	Pressure
ECR(2)	V.M. memb. + bend.	Von Mises	Density
ECR(3)	plast. deform.	plast. deform.	-

### Remarks:

The equivalent plastic deformation (ECR(3)) is only printed for elasto-plastic calculations.

The Von Mises criterion for the shells is expressed as:

$$\text{sig}(\ast) = \text{SQRT}(\text{sig}(\text{m}) \ast \text{sig}(\text{m}) + (\text{alpha} \ast \ast 2) \ast \text{sig}(\text{f}) \ast \text{sig}(\text{f}))$$

with **sig(m)** and **sig(f)** Von Mises stresses in membrane and bending and **alpha** = 2/3 by default.

The Von Mises criterion for the beams is expressed as:

$$\text{sig}(\ast) = \text{SQRT}(\text{ap} \ast \text{press} \ast \text{press} + \text{am} \ast \text{sig}(1) \ast \text{sig}(1) + \text{at} \ast \text{sig}(2) \ast \text{sig}(2) + \text{af} \ast (\text{sig}(3) \ast \text{sig}(3) + \text{sig}(4) \ast \text{sig}(4)))$$

The **sig(i)** are defined above, and **press** is the internal pressure, if the beam is a pipe. The coefficients **ap** (pressure), **am** (membrane), **at** (torsion) and **af** (bending) are computed by EUROPLEXUS according to the type of beam, of the existence or not of a curvature, etc.

## 11.5 TIME CHOICE (PROCEDURE /CTIME/) FOR THE PRINTOUTS

### Object:

The /CTIME/ procedure, described in the introduction (see page INT.57) is used to specify when printouts should take place during a computation.

### Comments:

If nothing is specified, the printouts are performed for all time steps.

If the keyword "NUPA" is used, do not forget to dimension adequately by means of the word "MNTI" as described on page A.100.

If the keyword "TIME" is used, do not forget to dimension adequately by means of the word "MTTI" as described on page A.100.

Be aware that printout times specified via "TFRE" or "TIME" are rounded to the closest time unit, that can be chosen via the "OPTI TION" directive.

### Warning:

Be careful in the choice of your printouts if you do not want to produce unnecessarily large listings. In general, it is advisable to use graphics post-processing in order to analyse the results, instead of reading values on a listing.

It is useful to know that the output file of the results ( "FICH" "ALICE" ), enables to print selected results on a listing after completion of a calculation. Therefore it is advisable to print only the bare minimum, although it will be necessary to read the results file again, in order to output interesting things after a calculation has been completed.

## 11.6 NODES OR ELEMENTS TO BE PRINTED

### Object:

The user can choose the nodes and/or elements where he wants to print the results.

### Syntax:

```
$["NOPOINT" ; "POINT" /LECTURE/ ]$  
$["NOELEM" ; "ELEM" /LECTURE/ ]$
```

"NOPOINT"

No point is printed.

"POINT"

Selection of the points which have to be printed.

"NOELEM"

No element is printed.

"ELEMENT"

Selection of the elements which have to be printed.

/LECTURE/

List of the points or elements for which the results are printed.

### Comments:

The two options "POINT" and "NOPOINT" are mutually exclusive, and the same is true for the two options "ELEMENT" and "NOELEM".

If none of the two options is specified, the results are printed for all nodes and for all elements.

## 11.7 RESULT FILES

### Object:

This directive is aimed at creating files for the post-processing of the computation results or a saving file to restart the calculation. The following file types are available:

- SAUVER file (saving file for successive restart)
- ALICE file (postprocessor: EUROPLEXUS)
- ALIT (ALICE TEMP) file (postprocessor: EUROPLEXUS)
- PVTk file (postprocessor: PARAVIEW VTK format)
- TABL file (a simple formatted table)
- POCH file (for Pochhammer-Chree post-treatment by EPX)
- MAPB file (to store a blast wave for later mapping)
- MED file (compatible with many softwares)
- MAPP file (to store a solution for later mapping onto a different model)
- SNAP file (to store a snapshot of of some fields at given times)

The following file types are also available but are not being further developed:

- SPTAB file (postpr.: SUPERTAB (old I-DEAS interface))
- TPLot file (postprocessor: TPLot)
- XPLOT file (postprocessor: TPLot)
- K2000 file (postprocessor: CAST3M)
- AVS file (postprocessor: AVS and old ParaView)
- PLOT-MTV file (postprocessor: PLOT-MTV)
- UNIV file (postprocessor: I-DEAS)

The SAUVER file is used to restart the calculation. It is described in detail on [GBG\\_0110](#).

CAST3M is the product of CEA (see <http://www-cast3m.cea.fr>), SUPERTAB is a commercial software by SDRC, TPLot is a software by JRC, AVS is a commercial product by Advanced Visual Systems, PLOT-MTV is a public utility (2D plotting only), I-DEAS is a commercial software by SDRC. The MED format is a format co-developed by the CEA and EDF which is compatible with many softwares. PARAVIEW is an open-source multi-platform application designed to visualize data sets of size varying from small to very large (see <http://www.paraview.org>).

Note that, besides the I-DEAS interface (UNIV keyword) described hereafter, another version is available, that had been independently developed by the CESI (formerly ENEL) group, and which is described on [GBG\\_0072](#).

### Syntax:

```
< FICHER      | [ "SAUV" <ndsauv> <"PROT" 'maclef'> <"LAST"> </CTIME/> ;
               "ALICE" <"FORMAT"> <"SPLIT"> <ndgrap>    /CTIME/ ;
               "ALIT"   <"FORMAT">          <ndalic>      /CTIME/  ...
```



```

... < "POINT" /LECTURE/ > ...
... < "ELEM" /LECTURE/ > ;

"PVTK" < $["FORM" ; "FOLD"]$ > <ndpara> /CTIME/
<"PINB"> <"MPI"> <"FLSW">
<"GROU" |[ "AUTO" ; nobj*("OBJE" <'groupname'>
    $[ "GAUS" ngaus ; "GAUZ" ngauz ]$ /LECT/) ]| >
<"VARI" <"DEPL"> <"VITE"> <"ACCE"> <"FEXT">
    <"FINT"> <"FLIA"> <"MCXX"> <"SIGN">
    <"ECRN"> <"RISK"> <"FAIL"> <"VCVI">
    <"CONT"> <"EPST"> <"ECRO"> <"XLVL">
    <"DTST"> <"EPAI"> <"PCLD"> <"PL2T">
    <"VFLU"> <"EFSI" efsi_params> >
<"ECRC" /LECT/>
<"SHEL" $[vx vy vz ; "ORTS"]$ /LECTURE/ > ;

"TABL" <ndtabl> /CTIME/
"VARI" <"NUGR"> nv * (
|[ "COOR" "COMP" ic "NOEU" /LECT/ ;
  "DEPL" "COMP" ic "NOEU" /LECT/ ;
  "VITE" "COMP" ic "NOEU" /LECT/ ;
  "ACCE" "COMP" ic "NOEU" /LECT/ ;
  "FINT" "COMP" ic "NOEU" /LECT/ ;
  "FEXT" "COMP" ic "NOEU" /LECT/ ;
  "CONT" "COMP" ic "GAUS" gp "ELEM" /LECT/ ;
  "ECRO" "COMP" ic "GAUS" gp "ELEM" /LECT/ ;
  "EPST" "COMP" ic "GAUS" gp "ELEM" /LECT/ ;
  "FONC" ifon ]| ) ;

"POCH" <"FORMAT"> <"SPLIT"> <ndpoch> /CTIME/ ...
... "NLIN" nl * ( /LECT/ ) ...
... "VARI" $[ "DEPL" ; "VITE" ; "ACCE" ]$ ;

"MAPB" | "MSPA" ; "MTIM" | "DIPR" dipr "PCHE" pche /CTIME/

"MAPP" <"FORM"> <"SPLI"> <nmapp> "OBJE" /LECT/ /CTIME/

"SNAP" /CTIME/ <"ELEM" /LECT/> <"POIN" /LECT/> ...
<"VARI" <"ECRO"> <"DEPL"> <"VITE"> ...
    <"ACCE"> <"FEXT"> <"FINT"> > ...
<"ECRC" /LECT/>
"SPTAB" <ndspta> /CTIME/ ;

"TPLOT" <"FORMAT"> <ndtplot> /CTIME/ ...
... "DESC" 'dddddd' ...
... < "POINT" /LECTURE/ > ...
... < "ELEM" /LECTURE/ > ;

"XPLOT" <"FORMAT"> <ndxplo> /CTIME/ ...
... "DESC" 'dddddd' ...

```

```

... < "POINT" /LECTURE/ >      ;

"K2000" < $[ "FORM" ; "XDR" ; "BINA" ]$ > <"SPLIT"> ...
... <ndcast> /CTIME/ ...
... < "POINT" /LECTURE/ > ...
... < "ELEM" /LECTURE/ > ...
... <"SHEL" $[vx vy vz ; "ORTS"]$ /LECTURE/ >
<"CHAMELEM">
<"VARI" < "DEPL"> <"VITE"> <"FEXT"> <"ACCE"> <"MCXX">
<"SIGN"> <"ECRN">
<"CONT"> <"EPST"> <"ECRO">
<"ECRC" /LECT/> >      ;
"AVS" "FORMAT" <"PRVW"> <ndavs> /CTIME/
<"VARI" <"DEPL"> <"VITE"> <"FEXT"> <"ACCE"> <"MCXX">
<"CONT"> <"EPST"> <"ECRO"> <"XLVL">
<"ECRC" /LECT/> >      ;

"PMTV" "FORMAT" <npmtv> /CTIME/
<"VARI" <"DEPL"> <"VITE"> <"SIGN"> <"ECRN"> >      ;

"UNIV" <FORMAT> $["CURR" ; "OBSO"]$ <nuniv>
/CTIME/ ;

"MED" /CTIME/
< "POINT" /LECTURE/ >
< "ELEM" /LECTURE/ >
<"SHEL" $[vx vy vz ; "ORTS"]$ /LECTURE/ > ;
]| >

```

**FORMAT**

If this keyword is present, the file will be formatted, otherwise it will be unformatted ("BINA") (but only where both possibilities exist).

**XDR**

Only for K2000 file if this keyword is present, the K2000 file will be independent of hardware and operating systems. This is the default option for K2000 file.

**ALICE**

A file of results is written in the standard ALICE format. This file can be read again by the EUROPLEXUS or ALICE programs.

**SPLI**

Split the ALICE or K2000 (formatted file only) or Pochhammer-Chree results into several files, one for each time instant, instead of producing just a single, big file. Useful e.g. to produce animations of results, which require typically many tens or even a few hundred time instants and to overcome the file size limitations that hold under some operating systems (e.g., under 32 bit MS-Windows maximum file size is 2 GB).

**ALIT**

A file of results is written in the ALICE format as a function of time. This file will only contain the results at the nodal points and elements defined with the keywords POIN and ELEM. This file can be read again by the EUROPLEXUS or ALICE programs.

#### SPTAB

An output results file in the SUPERTAB universal file format is produced (old I-DEAS version).

#### TPLOT

A file of results is written in the standard TPLOT format. This file can be further processed by the TPLOT program.

#### XPLOT

A file of results is written in the XPLOT format. This file can be further processed by the TPLOT program.

#### K2000

A file of results is written in the CAST3M format. The default mode of K2000 output is **XDR**. It can be read by CAST3M by using the keyword **RESTITUER**. The file format is the standard format of a CAST3M file obtained with the **SAUVER** directive. It is mandatory in this case to indicate the list of the nodes for which the results have to be stored (possibly **TOUS**). The directive **CHAMELEM** is optional. The values defined on the elements (stresses, hardening quantities, strains) are only stored if this keyword is specified. Note that these element values are averaged on the element GPs (or on the GPs of a specific fibre of the element: see **K2FB** option) and are affected either to the nodes or to the barycentre of the element (see **K2CH** option).

#### CHAM

This keyword introduces the definition of “chamelems” in the K2000 results file. If it is omitted, the latter will only contain the selected “champoints”, defined on the nodes chosen by directive POIN above.

#### VARI

Alternatively to CHAM, one can use the directive VARI which allows finer-grain control over the quantities effectively stored. Each nodal quantity (DEPL, VITE, FEXT, ACCE, MCXX in case of multicomponent gases, SIGN + ECRN in case of spectral elements, VCVI in case of finite volumes) and each element quantity (CONT, ECRO, EPST) may be specified separately. Furthermore, one may specify exactly which components of the ECR vector are to be stored, via the ECRC /LECT/ directive. Up to a maximum of 40 different components can be selected via the /LECT/ directive. This mechanism allows to greatly reduce the size of output files in large complex 3D calculations.

#### DEPL

Nodal displacements.

#### VITE

Nodal velocities.

#### FEXT

Nodal external forces (including reactions).

FINT

Nodal internal forces.

ACCE

Nodal accelerations.

MCXX

Nodal variables for multicomponent fluids (pressure, density, temperature, sound speed and component mass fractions).

SIGN

Nodal stress components (only for spectral elements).

ECRN

Nodal hardening variables (only for spectral elements).

VCVI

Velocity in the centre of finite volumes (CEA formulation, only available for PVTk).

VFLU

Fluid velocity (in 3D) at the nodes of 1D FE pipe elements (TUBE and TUYA) (only available for PVTk).

EFSI

EFSI (Extract embedded FSI fields) field extracted from the fluid. The EFSI keyword must be followed by the EFSI parameters, which are described in the EFSI “interactive” command on Page O.10. This keyword is only available for PVTk output.

FLSW

Face or volume blocked by FLSW (only available for PVTk).

RISK

Risk limit for eardrum failure and death. In addition, the maximum pressure and the total impulse are written to the result file. The risk types can also be split by using the SPLI command when the risk is defined (see A.30). (only available for PVTk).

FAIL

Failure level. Not active debris get a failure level of -1.0. (only available for PVTk).

XLVL

Level set of XFEM (only available for PVTk).

FLIA

Nodal forces due to liaisons (links) (only available for PVTk).

DTST

Stable time step (only available for PVTk).

**EPAI**

Thickness (and other **COMP** parameters) (only available for PVTk).

**PCLD**

Outputs related to point cloud adaptivity: point cloud index, distance to point cloud, adaptivity level (only available for PVTk).

**PL2T**

Treat mesh adaptivity in the visualization by converting elements with hanging nodes to triangles or tetrahedra to achieve correct visualization of sharp corners and other artefacts (only available for PVTk).

**SHEL**

Option which enables to print in the K2000 output file, the PVTk output file or the MED output file the stress (**CONT**) and strain (**EPST**) of shells according to specific axes rather than local axes (default). *vx vy vz* are the coordinates of the vector which is projected onto the shells read in **/LECTURE/** in order to define the first direction of the posttreatment axes. There is also the keyword **ORTS** that can be given in place of a vector, see details below. The third posttreatment direction is identical with the third one in local axes. This definition of the posttreatment axes is made in the initial configuration and is not updated ; that means this option is only relevant for small strains. This command may be repeated as many times as needed. This option is available only for **DST3**, **DKT3**, **T3GS** and **Q4GS** shells.

**ORTS**

For shells with orthotropic materials, the orthotropic axes are given using the directive **COMP ORTS**. In this case (only), the keyword **ORTS** allows to use these orthotropic directions to define the posttreatment axes.

**AVS**

A set of result files (see note below) are written in the AVS format. For the moment, only a formatted output is available for AVS, so the **FORM** option is actually mandatory in this case.

**PRVW**

The AVS files are modified to be imported into PARAVIEW software (see note below). This is only compatible with older versions of ParaView (less than 2.9 or so). For newer versions, use the PVTk type of output, which produces files in the VTK format.

**UNIV**

A results file in “universal” (I-DEAS) format is to be written. This file may be of two types: “current” or “obsolete”. By default it is of “current” type.

**CURR**

The universal results file will be of type “current”. This is the default.

**OBSO**

The universal results file will be of type “obsolete”.

#### PVTK

A set of result files (see note below) are written in the PARAVIEW format (`.pvd` and `.vtu` files). This is the VTK format, compatible with the newer versions of ParaView. By default, use is made of the library `LIB_VTK_IO`, written by Stefano Zaghi (see <http://stefano.zaghi.googlepages.com/lib-vtk-io>), which allows to produce either ASCII or binary output formats. However, if the (obsolete) keyword `FOLD` is specified in place of `FORM`, then formatted output is produced without making use of the `LIB_VTK_IO` library (note that in this case only ASCII format, not binary format, is possible).

#### PINB

The geometry of the pinballs are written in a separate `vtu`-file.

#### MPI

MPI calculations only. One set of `vtu`-files is written for each subdomain, to avoid centralizing data from all subdomains on one thread before writing. Groups must be defined at least via `GROU AUTO`

#### GROU

Groups can be defined automatically (using keyword `AUTO`) or manually by entering `nobj` objects defined with the `OBJE` keyword (which must be repeated exactly `nobj` times). A name can be optionally given to each manually defined group. Automatic groups were generated for each material definition and for each element class (not element type). Each separate group is written as output files (see comments below for syntax). For manually defined groups, the number of the Gauss point `ngaus` or the number of the layer `ngauz`, which is used for output, can be defined for each group with `GAUS` or `GAUZ` respectively. The logic of the VTK format is not the same as the one of CAST3M and EUROPLEXUS. If an element is contained in several groups, the element will be written to the output files several times. This element is also repeated in the output of ParaView. If present, only the elements (and nodes) defined by the `GROU` keyword are written to the output.

#### MSPA

Introduces the writing of a blast field into a map file for one specific time step over a certain geometry.

#### MTIM

Introduces the writing of a blast field into a map file for one specific distance over the given time.

#### DIPR

For `MSPA`: Distance in which the pressure is checked and the output is written when the pressure reaches a certain value.

For `MTIM`: distance where the pressure history is saved.

#### PCHE

Pressure value which is used for the check to write the map file. For the `MTIM` option this is the start point of the output. The `CTIME` parameter should be set small in order to check the limits often.

**MAPP**

Activate creation of a so-called map file, where the solution of the current simulation (limitedly to a specified object part of the mesh) is stored, for successive mapping as initial conditions on a (possibly) different mesh. See directive **INIT MAPP** on page [GBE.0260](#) for a description of how to read back the map file and use it to initialize the solution in a subsequent calculation.

**SNAP**

Activate creation of a so-called snapshot file (with extension “.snp”), where snapshots of some given fields (limitedly to a specified object part of the mesh) are stored at given times of the calculation. For nodal and elementary fields, the user has to specify which nodes or elements to consider with the keywords **POIN** and **ELEM** respectively. At this time, the only elementary field allowed for this snapshot output is **ECRO**. If the field **ECRO** is queried, the keyword **CERC** is mandatory to specify which **ECRO** components to output.

**FORM**

Create a formatted (ASCII) map file, instead of the default unformatted (binary) map file.

**SPLI**

Split the map file into several files in case the map is written at more than one time station. Such files are automatically named `<basename>_0001.map`, `<basename>_0002.map` etc., where `<basename>` is the base name of the input file.

**nmapp**

Number of the logical unit of the map file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.MAP`.

**OBJE**

Introduces the list of elements (specified by the following `/LECT/`) whose solution must be stored on the map file for subsequent mapping on a different (or identical) mesh.

**/CTIM/**

Choice of the time station(s) at which the mapping file should be produced (see Page INT.57).

**ndgrap**

Number of the logical unit of the ALICE file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.ALI`. For the special case of split ALICE files, see comments below.

**ndalic**

Number of the logical unit of the ALICE TEMPS file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.ALT`. If the **SPLI** keyword is specified, potentially many ALIC results files are produced. To place all these files in a results sub-directory (in order to unclutter the current directory) one can use the technique explained on Page A.28.

**ndspta**

Number of the logical unit of the SPTAB file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.SPT`.

**ndtplo**

Number of the logical unit of the TPLOT file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.TPL`.

**ndxplo**

Number of the logical unit of the XPLOT file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.XPL`.

**ndcast**

Number of the logical unit of the CAST3M file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.K20`. For the special case of split CAST3M files, see comments below.

**ndavs**

Number of the logical unit of the AVS file(s) or file name in quotes. If omitted, the program chooses a (base) file name by default (see page A.27). The default extension is `.AVS`. Split files are generated for this type of output. See comments below.

**ndpara**

Number of the logical unit of the PARAVIEW file(s) or file name in quotes. If omitted, the program chooses a (base) file name by default (see page A.27). The default extension is `.pvd`. This file contains links to files with the data (vtu-format, extension `.vtu`). See comments below. To place all these files in a results sub-directory (in order to unclutter the current directory) one can use the technique explained on Page A.28.

**npmtv**

Number of the logical unit of the PLOT-MTV file or file name in quotes. If omitted, the program chooses a (base) file name by default (see page A.27). Split files are generated for this type of output. The default extension is `.MTV`. See comments below.

**nuniv**

Number of the logical unit of the universal file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.UNV`.

**ndtabl**

Number of the logical unit of the table file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is `.TAB`.

**/CTIM/**

This procedure is described in the introduction (page INT.57). If the keywords `NUPA` or `TIME` are used, do not forget to dimension adequately by means of keywords `MNTI`, `MTTI` (see also page A.100).

DESC 'dddddd'



Six-character descriptor to identify the run for the TPLOT (or XPLOT) data-base. Note that this item is in text format, therefore it must be enclosed in quotes. When loading data on the TPLOT database, a prefix PL is automatically placed in front of this descriptor. Thus, the full descriptor on the database will be PLdddddd. For XPLOT, the prefix is XL, so the full descriptor will be XLdddddd.

**POIN /LECTURE/**

List of nodes for which the results have to be stored for successive treatment by TPLOT or XPLOT.

**ELEM /LECTURE/**

List of elements for which the results have to be stored for successive treatment by TPLOT.

**MED**

A file of results is written in the MED format. This file will only contain the results based on the nodes and the elements selected with the keywords POIN and ELEM. For the selected nodes, nodal displacements, nodal velocities, grid velocities (in ALE only), nodal accelerations, nodal external forces (including reactions), nodal internal forces are stored. For selected elements, stresses, strains, internal variables and VCVI field (on VFCC element) are stored. Do not forget to create the MED file before with the keyword MEDE (see page A.30).

**TABL**

A file of results is written in a simple text file in tabular form. This type of output is meant to monitor just a few variables, say a nodal displacement and an element stress component, but with great precision. Results are always written formatted, and with full double precision (16 significant digits). The table contains one line per storage station (typically, each time step). The first two columns of the table always contain the current time step (**npas**) and the current time (**t**). The following columns contain the chosen variables. Obviously, no more than just a few variables can be specified, in practice, else the length of the table line would become excessive. The keyword **VARI** introduces the chosen variables. Their number is given by **nv** and thereafter exactly **nv** variable specifications must be given, chosen among the following possibilities: **COOR** for nodal coordinates, **DEPL** for nodal displacements, **VITE** for nodal velocities, **ACCE** for nodal accelerations, **FINT** for nodal internal forces, **FEXT** for nodal external forces, **CONT** for elemental stresses, **ECRO** for elemental hardening quantities, **EPST** for elemental total strains, **FONC** for the value of a specified function. The keyword **COMP** introduces the component, **GAUS** the Gauss point, **NOEU** the node and **ELEM** the element. Note that the **/LECT/** directives must specify one single node or element. The value **ifon** after **FONC** specifies the function identifier (see the **FUNCTION** directive).

**NUGR**

Optional variable allowing, if present, the output (in the listing file) of the correspondence between the nodes or elements indicated in the header of the ".tab" file with the name (node, element, group, ..) entered by the user in the Europlexus file.

**ndpoch**

Number of the logical unit of the Pochhammer-Chree file(s) or file name in quotes. If omitted, the program chooses a (base) file name by default (see page A.27). The default extension is `.poc`.

#### NLIN

Introduces the number of “lines” along which the results will be sampled for the successive Pochhammer-Chree post-treatment. Each line is formed by an ordered sequence of nodes and is defined by the following `/LECT/`.

#### Comments:

The keyword `FICHIER` is not compulsory. If it is used, the last step is systematically saved.

Do not forget to define the logical unit(s) of the file(s), on the control cards. As an alternative, EUROPLEXUS accepts the name of the file enclosed in quotes.

If one does not pay attention, result files may become very bulky, because the total number of computed time steps is often very large. It is then advisable to estimate the total number of steps from the stability step computed by the program, and then choose a reasonable number of storages on the ALICE file. It is also possible to obtain a smaller results file by using the `ALICE TEMPS (ALIT)` directive: in this case only the variables relative to the nodes and elements given in directives `POIN` and `ELEM` will be stored.

It is rare that one needs more than a dozen of time stations to plot the deformed shapes of the mesh — in this case the `ALICE` directive will be used. It is also infrequent that one needs more than a few hundred points to plot curves as a function of time — it is suggested to use the directive `ALIT` which allows to obtain a file reduced to just the selected points. Therefore it is possible to specify a much larger number of saving stations on an `ALIT` file than on an `ALIC` one.

For ALICE, the `SPLI` option allows to split the results into many small files, one for each stored time instant, rather than producing just one big file. This option is useful for very large computations and/or for producing animations, which typically require many saved instants. In this case, `ndgrap` is just the base name of the output files. The single file names are automatically given progressive numbers (`_0001`, `_0002`, etc.) appended to the name, which identify the storage index. A file with suffix `_0000` is also produced, which contains the initial mesh topology.

In order to post-treat these split results with EUROPLEXUS, proceed exactly as if the results would be in a single file, but remember to specify the `SPLIT` keyword in the `RESU ALIC` directive, see page ED.20.

`XPLOT` storage is intended to perform pseudo-1D visualization of data in a 2D or 3D run. A curvilinear abscissa is built up passing through the nodes defined in `POIN /LECT/`. Then, nodal and element quantities are stored as a function of this abscissa. By using the `TPLOT` program, the relevant quantities can then be plotted along the curvilinear abscissa (either initial or current). Note that nodal quantities (displacements, velocities, etc.) are stored without modification at the specified nodes. Element quantities, however, (such as stresses and hardening parameters), that are usually defined only at points internal to the elements, are first extrapolated to the nodes, then stored. Currently, the extrapolation consists of simply: 1/

averaging each quantity over each element (by using the values at the different Gauss points),  
2/ averaging all neighbour element contributions to obtain nodal values. Neighbour elements to a node are those elements that contain that node.

Note also that, in the extrapolation process, only certain types of elements are considered. For example, shell or beam elements are rejected, because the mean value of the stress components on all the Gauss points is likely to be meaningless for such elements. Only the following element types are considered: TRIA( 2), CAR1( 8), CAR4( 9), CUBE(11), CUB6(13), PR6(20), TETR(21), PRIS(27), FLU1(52), FLU3(53), FL23(64), FL24(65), FL34(66), FL35(67), FL36(68), FL38(69), Q41 (71), Q42 (72), Q41N(73), Q42N(74).

In order to read with CAST3M a file written by EUROPLEXUS, use the following CAST3M commands:

1/ Formatted file:

```
OPTI REST FORM 'file';  
REST FORM;  
. . . (post-treatment commands)
```

2/ Unformatted file:

```
OPTI REST BINA 'file';  
REST BINA;  
. . . (post-treatment commands)
```

3/ XDR file:

```
OPTI REST 'file';  
REST ;  
. . . (post-treatment commands)
```

For K2000, note that two syntaxes are possible. With the 'old' syntax (the keyword **VARI** does not appear), all nodal quantities are always stored. Furthermore, all element quantities are stored if **CHAM** appears. The components of the ECR table which are stored depends in this case from the material: they are the same components of ECR that are printed on the listing.

With the new syntax (the keyword **VARI** appears) only the specified nodal quantities, element quantities and ECR table components are actually stored.

Note that, strictly speaking, it is only possible to produce an output file for K2000 when the input mesh has also been produced (and read into EUROPLEXUS) in this format. However, if this is not the case but you still desire to postprocess your EUROPLEXUS calculation with K2000, consider transforming your mesh in K2000 by the option **K2MS** (see Section H, output

options). Beware, however, that this may require some manual intervention and in any case the obtained mesh will be less flexible to use than a “real” K2000 mesh.

In the case of standard AVS storage, a set of files is written, one for each stored variable. The files basename is given by `ndavs`. If `ndavs` as given by the user contains the extension `.avs` or `.AVS`, this extension is removed by the program. An extension of the form `.VARI.N.inp` is then automatically provided by EUROPLEXUS. Here `VARI` is the variable type (`DEPL`, `VITE`, ...) and `N` an integer counter which is automatically incremented by 1 at each successive storage in time.

Recall that AVS storage can also be requested interactively, i.e. during an interactive execution of EUROPLEXUS (See Group A, Interactive (Foreground) Execution).

In the case of AVS storage modified for usage with PARAVIEW, elements are split into groups with same element topology and same material law. One file is stored for each group of elements at each successive storage in time. This file contains geometry and both nodal fields and elemental fields required by the user. It is named from the name given by `ndavs`, with its extension removed. An extension of the form `_N1_N2.inp` is then added to the name. `N1` is the number of the element group and `N2` is an integer counter as for standard AVS files. If `ndavs` is not defined, the base of the EUROPLEXUS input file name is used to build AVS-PARAVIEW file names.

If a directory name is provided for AVS-PARAVIEW files with the `OPNF` directive, files are written in this directory, excepted if `ndavs` represents a name with full path. In this latter case, the above given directory is ignored.

The AVS-PARAVIEW format is only readable with older versions of PARAVIEW (less than 2.9). For newer versions of PARAVIEW, the PARAVIEW output (see `PVTK`) with a `.pvd` and `.vtu` files is recommended.

Automatic group definition for PARAVIEW output (with keyword `GROU AUTO`) consists in the same splitting of elements as described above for the AVS-PARAVIEW output.

The PLOT-MTV output is only available in 2D and for spectral elements. the program will only include spectral elements and nodes in these files. A separate file with the extension `.mtv` is produced for each nodal quantity and at each selected storage time.

The `SIGN` keyword produces storage of 4 nodal stress (6 in 3D) components (`SGXX`, `SGYY`, `SGZZ`, `SGXY`, `SGYZ`, `SGZX`). The `ECRN` keyword produces storage of 2 nodal hardening quantities: the hydrostatic stress (`HYDR`) and the Von Mises stress (`VMIS`).

When using SPTAB output format, a file named `sptab.param` must exist in the current directory, containing the following key-words useful to declare the variables to print out: `DISP`, `VELO`, `ACCE`, `INTF`, `EXTF`, `MCVAR`, `MCVEL`, `MCMFR`, `FLVAR`, `SCUB8`, `ECOQI`.

For CAST3M, the `SPLI` option allows to split the results into many small files, one for each stored time instant, rather than producing just one big file. In this case, the chosen output type MUST be formatted (`FORM`). This option is useful for very large computations and/or for producing animations, which typically require many saved instants. In this case, `ndcast` is just the base name of the output files. The single file names are automatically given progressive

numbers (\_0001, \_0002, etc.) appended to the name, which identify the storage index. A file with suffix \_0000 is also produced, which contains the initial mesh topology.

In order to post-treat one of these split results with CAST3M, proceed as follows: choose the instant to be treated, say number 3 i.e. the third storage performed; then, produce a file by concatenating the 'zero' file (\_0000) and the file for the chosen instant; finally, read and post-process the resulting file with CAST3M. In this example:

```
cat mytest_0000.k20 mytest_0003.k20 >out.k20
```

```
K2000
```

```
opti rest form 'out.k20'; rest form; ...
```

### **Warning :**

Be careful: output files may become very large, because the total number of the time steps computed is often large. Therefore it is better to estimate that number from the stability step computed by the program. Then, the user can choose a reasonable number of writings on the output files.

The user seldom needs more than a dozen storage stations ('cases') to draw deformed structures and no more than fifty points to draw time functions.

## 11.8 POST-PROCESSING BY I-DEAS MASTER SERIES

### Object:

This is aimed at creating files for the post-processing of computation results by I-DEAS master series.

This model is part of the models developed by the CESI team (formerly at ENEL, Milano) in collaboration with JRC.

I-DEAS is a commercial software by SDRC.

### Syntax:

```
< "FICHIER" <FORMAT> "IDEA" ndidea    /CTIME/
                                ... < "POINT" /LECTURE/ > ...
                                ... < "ELEM"  /LECTURE/ > ...
    < "VARI" < "DEPL" "VITE" "FEXT" "ACCE" "MCXX" >
              < "CONT" "MESH" "MCVA" "MCVE" "MCMF" >
              < "FLVA" "ECOQ" >                >                >
```

#### "FORMAT"

If present, the file is formatted, otherwise it is unformatted.

#### "IDEA"

An output results file in the I-DEAS universal file format is produced (new version).

#### ndidea

File name in quotes of the universal output file for I-DEAS.

#### "POIN" /LECT/

Same meaning as for the other result files, see page G.70.

#### "ELEM" /LECT/

Same meaning as for the other result files, see page G.70.

#### "CTIM" .. "CONT"

Same meaning as for the other result files, see page G.70.

#### "VARI"

Introduces the list of variables to be stored.

#### "DEPL"

The I-DEAS universal file in output will contain the displacements.

"VITE"

The I-DEAS universal file in output will contain the velocities.

"FEXT"

The I-DEAS universal file in output will contain the external forces.

"ACCE"

The I-DEAS universal file in output will contain the accelerations.

"MCXX"

The I-DEAS universal file in output will contain the MC variables.

"CONT"

The I-DEAS universal file in output will contain the stresses.

"MESH"

The I-DEAS universal file in output will contain the mesh data (only node and element datasets).

"MCVA"

The I-DEAS universal file in output will contain the MC conserved variables: pressure, density, internal energy, maximum pressure, minimum pressure, temperature.

"MCVE"

The I-DEAS universal file in output will contain the MC velocities: x, y, z velocity components, velocity modulus, sound speed and Mach number.

"MCMF"

The I-DEAS universal file in output will contain the MC mass fractions.

"FLVA"

The I-DEAS universal file in output will contain the fluid finite elements variables (FLxx family with FLUT material): pressure, density, internal energy, max pressure, min pressure, sound speed.

"ECOQ"

The I-DEAS universal file in output will contain the COQI element ECR variables: hydrostatic pressure, Von Mises, plasticity flag, current yield stress.

**Comments:**

The general comments of page G.70 apply to the I-DEAS results file as well.

When using "IDEA" output format, the default options for the selection of the results are: whole geometry (i.e. all nodes/elements are treated if "POIN", "ELEM" are omitted), no variable (i.e. only the variables specified in the "VARI" option are stored).

Element output in I-deas universal file format is only available for FLxx, MCxx, CUB8, COQI, CQD3, CQD4 elements at the moment.



## 11.9 OUTPUT REGIONS

### Object:

This directive enables the printing of physical values within a given region.

A region is defined by the list of the elements which compose it. The region could correspond to a GIBI object.

### Syntax:

```
"REGION" ( 'nom region'
    $[ "RMAS" ; "VOLU" ; "BARY" ; "DIMX" ; "DIMN" ;
        "DMOY" ; "VEMX" ; "VEMN" ; "VMOY" ; "ACMX" ;
        "ACMN" ; "AMOY" ; "IMPU" ; "ECIN" ; "WINT" ;
        "WEXT" ; "PDV" ; "WINJ" ; "RESU" ; "IRES" ;
        "ECRG" ; "ECRM" ; "EMAS" ; "FLIR" ; "RISK" ;
        "EROD" ; "ENDO" ; "CLAS" ; "EPSM" ; "TOUT" ]$
    < "DIRX" rx "DIRY" ry "DIRZ" rz >
    |[ /LECTURE/ ;
        "POIN" /LECTURE/ ]| )
```

**nom region**

Name of the region given by the user (in apostrophes).

**RMAS**

Mass (components, computed via XMEL).

**VOLU**

Volume.

**BARY**

Center of gravity (barycentre) of the region.

**DIMX**

Maximum displacement (absolute) of the region (vector), only components 1 to 3.

**DIMN**

Minimum displacement (absolute) of the region (vector), only components 1 to 3.

**DMOY**

Average displacements (components).

**VEMX**

Maximum velocity (absolute) of the region (vector), only components 1 to 3.

**VEMN**

Minimum velocity (absolute) of the region (vector), only components 1 to 3.

**VMOY**

Average velocity (components).

**ACMX**

Maximum acceleration (absolute) of the region (vector), only components 1 to 3.

**ACMN**

Minimum acceleration (absolute) of the region (vector), only components 1 to 3.

**AMOY**

Average acceleration (components).

**IMPU**

Impulse (components).

**ECIN**

Kinetic energy (norm and components).

**WINT**

Internal energy.

**WEXT**

Work of external forces applied to the nodes of the region.

**PDV**

Work of pressure forces in ALE for a stand-alone domain.

**WINJ**

Injected energy (only for material EAU).

**RESU**

Resultant of the external forces applied at the nodes.

**IRES**

Impulse corresponding to the above resultant.

**ECRG**

For each component of ECR (in fact, for the first 10 components), sum of the values on the Gauss points of the region.

**ECRM**

Average of the ECR over the region.

**EMAS**

Mass (scalar, computed via the element's mass XM0).

**FLIR**

Resultant of the force due to LINK/LIAI applied at the nodes.

**RISK**

Global risk (average).

**EROD**

Number of eroded elements.

**ENDO**

Number of damaged elements (A damaged element contains a least a Gauss point which is not broken.) - (See A.30 : EROS ldam)

**CLAS**

Number of eroded classes and number of damaged classes - A class is eroded as soon as an element of this class is eroded - A class is damaged if the class is not eroded and if the class contains at least a damaged element.

**EPSM**

Average of the EPST (strain variables) over the region.

**TOUT**

All possible physical values are required.

**DIRX, DIRY, DIRZ**

Components of the direction vector specifying the user-defined frame. Definition of the local frame (x,y,z) with respect to the global frame (X,Y,Z):

- Local x-axis is collinear with the sliding direction specified by the user through the triplet DIRX, DIRY, and DIRZ.
- Local z-axis is orthogonal to x-axis and it is situated in the plane described by the axes x and Z. Positive projection on Z is used.
- Local y-axis completes the direct orthogonal axis system.

If the slider direction is vertical, the local x-axis is collinear with the sliding direction, the y-axis is collinear with Y-axis, and z-axis completes the direct orthogonal axis system.

Only the following quantities can be written in this frame: MASS, ECIN, AMOY, ACMX, ACMN, VMOY, VEMX, VEMN, DMOY, DIMX, DIMN, BARY, IMPU, QMVT, RESU, ECRG, FLIR

**LECTURE**

List of the elements composing the region.

POIN /LECTURE/

List of the nodes composing the region.

**Comments:**

The computation takes place within the elements.

It is possible to have elements which belong to several regions.

If the region is only known by its nodes (it has not been defined in the directive "GEOM"), the only possible balances are WEXT, RESU and IRES. In this case, it is mandatory to use the keyword "POIN" before the /LECTURE/ procedure, to avoid confusion between nodes and elements.

If the region is nothing else but the whole structure, the values of WINT and ECIN are the same as those printed in the energy balance.

The physical values of the regions are computed during the printing.

**Important: restart calculation (see page ED.10)**

There is no problem if the computed quantity does not depend on masses (WINT). If the physical values are dependant on the masses (ECIN BARY VMOY IMPU RMAS VOL PDV), the computation will be correct only if the masses are constant during the EUROPLEXUS computation. In fact, in order to lighten the file of the results (FICHIER ALICE), only the initial masses are copied out. There is no problem concerning a Langrangian computation. For an Eulerian or A.L.E computation, the masses change. Therefore, the physical values will not be correct.

All this happens during a restart; if the physical values are computed during a normal (non restart) EUROPLEXUS run, all the results are correct.

## 11.10 MEASUREMENTS AND MESH QUALITY

### Object:

This directive enables: 1) the printout of various types of simple **measurements** taken on the current geometrical mesh; 2) the verification of **mesh quality** (MQUA) on the initial configuration as well as at chosen later times during the simulation by choosing appropriate criteria; 3) the use of such mesh quality criteria to get rid of heavily distorted elements via the element **erosion** mechanism.

Normally, it is called from the input file and the requested measurements are then printed on the listing. However, the same directive (same syntax) is available also from the command line during an interactive execution (see pages A.25 and O.10). For this reason, the present directive **must** be terminated by the keyword **TERM**, as shown below in the syntax. In case of interactive use, the requested measurements are printed on the console window, not on the listing.

### Syntax:

```
MEASURE  $[ ELEM e ;
           NODE n ;
           OBJE /LECT/ ;
           EMIN /LECT/ ;
           EMAX /LECT/ ;
           DIST <POIN> /LEC1/ <POIN> /LEC2/ ;
           MQUA nmq <mesh quality assessment commands> ]$
TERM
```

#### ELEM e

Returns information about the chosen element **e**: its type, the associated nodes (with their coordinates), its size, its mass and the list of objects or groups to which it belongs.

#### NODE n

Returns information about the chosen node **n**: its coordinates, its mass, the list of elements to which it belongs and the list of objects or groups to which it belongs.

#### OBJE /LECT/

Returns information about the chosen object **/LECT/**: the associated elements, the associated nodes, its size and its mass.

#### EMIN /LECT/

Returns information (see **ELEM** above) about the “smallest” element among those belonging to the object specified in the following **/LECT/**. Use **LECT tous TERM** to get the smallest element in the whole mesh. As smallest element, we consider the one having the shortest (non-zero) intra-nodal distance among its nodes (and thus most likely the shortest element characteristic length as far as stability is concerned).

EMAX /LECT/

Returns information (see ELEM above) about the “largest” element among those belonging to the object specified in the following /LECT/. Use LECT tous TERM to get the largest element in the whole mesh. As largest element, we consider the one having the largest intra-nodal distance among its nodes (and thus most likely the longest element characteristic length as far as stability is concerned).

DIST <POIN> /LEC1/ <POIN> /LEC2/

Returns the minimum (intra-nodal) distance between the two objects defined in /LEC1/ and /LEC2/. Normally the two objects are interpreted as a set of elements. The nodes of such elements are automatically extracted and used to compute the minimum (intra-nodal) distance. However, one may want to specify an object composed only of points (nodes). This can be done by adding the optional POIN keyword just before the /LECT/ of the concerned object(s).

MQUA ...

Introduces the commands to activate mesh quality assessment. See below for a complete description of all such commands.

TERM

Indicates the termination of measurements. This keyword is necessary so that the present directive may be used also in interactive mode.

## Comments

An example of this directive (to achieve simple measurements) is as follows:

```
MEASURE
  ELEM 123
  NODE 74
  ELEM 1
  OBJE LECT toto TERM
  EMIN LECT tous TERM
  EMAX LECT 1 2 4 TERM
  DIST LECT toto TERM LECT tata TERM ! min distance between two
                                     ! objects made of elements
  DIST POIN LECT p1 TERM LECT tata TERM ! min dist. between a point
                                     ! (node) and an object made
                                     ! of elements
  DIST POIN LECT p1 TERM POIN LECT p2 TERM ! distance between two
                                     ! points (two nodes)

TERM
```

When used interactively, the code pauses for input after each sub-command, waiting for the next sub-command. To exit from the MEAS directive, give the final TERM command.

### 11.10.1 Mesh Quality assessment

#### Syntax:

```
MEAS ... MQUA nmq
( $[ ASPE ; SKEW ; WARP ; TAPE ;
    SYMM PX px PY py PZ pz
        NX nx NY ny NZ nz TOL tol ]$ < /LECT/ >
    < EROS eros > < PRIN > )
    < EVAL /CTIM/ >
```

#### MQUA nmq

Introduces the commands to activate mesh quality assessment. The **nmq** quantity is the number of mesh quality assessment criteria that will be defined next.

#### ASPE

Compute aspect ratio of the elements.

#### SKEW

Compute skewness of the elements.

#### WARP

Compute warping of the elements.

#### TAPE

Compute tapering of the elements.

#### SYMM

Compute symmetry of the chosen mesh with respect to a plane passing through a given point and of given normal. A value of 0.0 is assigned both to elements having a symmetric companion and to elements whose centroid lies on the symmetry plane. A value of 1.0 is assigned to elements not having a symmetric companion and whose centroid does not lie on the symmetry plane.

#### PX px

Defines the *x*-coordinate of the point.

#### PY py

Defines the *y*-coordinate of the point.

#### PZ pz

Defines the *z*-coordinate of the point.

#### NX nx

Defines the  $x$ -component of the normal. The normal needs not be unitary, since it is normalized to 1.0 internally.

NY ny

Defines the  $y$ -component of the normal.

NZ nz

Defines the  $z$ -component of the normal.

TOL tol

Defines the tolerance (absolute distance) to decide whether two points (nodes or element centroids) are coincident or whether a point (element centroid) lies on the symmetry plane.

/LECT/

Optional list of all elements for which the current quality criterion must be evaluated. By default (if omitted), the evaluation is done for all elements in the mesh (when applicable).

EROS eros

Optional keyword that produces the following effect: if the computed criterion exceeds the given value **eros** for an element, then the element is eroded (removed from the computation). If this keyword is omitted for a criterion, then no elements are eroded according to that criterion.

PRIN

Printout the evaluated criterion values on the listing each time they are evaluated. By default, i.e. without specifying the PRIN optional keyword, the evaluated values are not printed. Beware that the list of values may be long, since there is one value for each element in the mesh.

EVAL /CTIM/

Reading procedure (see page INT.57) of the chosen time steps or time instants at which the evaluation of the above defined mesh quality criteria should be performed. If omitted, the evaluation is performed *at every time step*. However, note that this may be very expensive so that in large applications it is strongly advised to perform the evaluation (and the optional element erosion) only at a certain chosen frequency in time.

## Comments

The parentheses ( ... ) in the above syntax signify that more than one criterion can be chosen at the same time, by simply repeating the parenthesized context. The total number of declared criteria **must** be equal to the number **nmq** declared just after the **MQUA** keyword.

The results of the chosen quality criteria evaluations become available for visualization at the time steps or time instants specified in the /CTIM/ directive.

Each time an **MQUA** directive is entered (either in the input data set or from the command line, in case of interactive execution), any pre-existing mesh quality assessment criteria are wiped out and are replaced by the newly declared criteria.

An example of this directive is as follows:



```
MEAS MQUA 3
  ASPE LECT plate TERM EROS 2.5
  WARP EROS 5.0 ! Erode if warping angle > 5 degrees
  SKEW
  EVAL TFRE 1.0E-3 NUPA LECT 5000 7500 TERM
TERM
```

This means that the following three quality criteria will be evaluated:

- Element aspect ratio, limitedly to the elements of the **plate** object. Such elements will be eroded if the evaluated criterion exceeds the value 2.5.
- Element warping, for the whole mesh. Elements will be eroded if their warping (angle) exceeds 5 degrees.
- Skewing. This criterion will be evaluated for all elements. However, it will generate no element erosion (it is only computed to be visualized).

The above mentioned evaluations (and the associated element erosions, if any) are performed every millisecond of physical time and, in addition, also at steps 5000 and 7500. At any of these chosen time instants the user will be able to visualize the computed quality criterion fields (one field for each active criterion).

## 11.11 SAVING FILE FOR SUCCESSIVE RESTART

### Object:

This keyword creates a saving file and, in conjunction with the keyword REPR (to be used in a subsequent run), allows splitting a computation in two or more parts.

This directive replaces the old (deprecated and obsolescent) directive SAUV (group A, see page SR.20).

The results are saved on a file (saving file) at times specified by the user. Each saving corresponds to a number or position on the file (1, 2, 3 etc.), from which a restart of the computation can be carried out in a successive run (see directive REPR on page SR.30).

### Syntax:

```
FICH  SAUV <ndsauv>  <PROT 'maclef'>  <LAST>  </CTIM/>
```

nbansav

Number of the saving file or name of the file in quotes. If completely omitted, the code will assume the default file name <basename>.sau where <basename> is the root of the input file name (i.e. without extension .epx).

PROT

Keyword entering a protection on the saving file.

'maclef'

Key of up to 8 characters, enclosed in apostrophes. In order to restart the computation from that file, the instruction REPR must contain the keyword PROT with an identical key.

LAST

This keyword indicates that the saving file should contain just one saving station, corresponding to the last saved time station in the present calculation. In other words, each new saving station replaces the former one, if any. This allows to obtain a saving file of the smallest possible size. However, restarting from an intermediate time is obviously not possible in this case: the only possibility to restart the calculation will be REPR ... POSI 1 (see page SR.30).

/CTIM/

The /CTIM/ procedure (see page INT.57) is used to specify the saving times via a step frequency (FREQ), a time frequency (TFRE), a list of steps (NUPA) or a list of times (TIME). If NUPA or TIME are used, do not forget to dimension accordingly using MTTI or MNTI, respectively, see page A.100. Note that the code always saves the last step of the calculation (if the run is terminated normally), irrespective of the frequency chosen. Therefore, if one is only interested in getting the possibility to continue the calculation further on, simply omit the /CTIM/ procedure.

**Comments:**

The keyword `PROT` is not compulsory. If it is not used, there is no protection (this is equivalent to a key of 8 blanks).

If a unit number is used for `nbansav`, the saving file and its number must have been defined before on the control cards.

A first saving station (position number 1) containing some header data is always produced at the initial time (step 0 of the calculation). Of course, it is normally meaningless to restart from this time station, unless the `LAST` keyword has been specified (see above), because it would be the same as starting the calculation anew from the initial time. On the contrary, if `LAST` has been specified, the only possibility for restart is to use the first time station which, in this case, will contain the data of the last saving performed (not the first one in general).

**Examples:**

Assume a calculation performs 4994 time steps to arrive at its final time. The following saving directives are accepted:

- `FICH SAUV 'myfile.sau' FREQ 1000` saves data for restart on the indicated file every 1000 steps. The following six saving stations are produced: 1 (step 0), 2 (step 1000), 3 (step 2000), 4 (step 3000), 5 (step 4000) and 6 (step 4994, i.e. the last step).
- `FICH SAUV FREQ 1000`: same as above but the saving file has the default name `<basename>.sau`.
- `FICH SAUV 'myfile.sau' LAST FREQ 1000` saves data for restart on the indicated file every 1000 steps, by always re-writing the previous saving. Only one saving station is thus present on the saving file (assuming that the run terminates normally): 1 (step 4994, i.e. the last step). If the run would fail, say, at step 2500, the saving file would also contain one station (step 2000).
- `FICH SAUV FREQ 10000`: the saving file has the default name `<basename>.sau`. Only one saving station is produced (assuming that the run terminates normally): 1 (step 4994, i.e. the last step). If the run would fail, say, at step 2500, no (useful) saving time station would be available (there is still one saving station, but it is at step 0).
- `FICH SAUV LAST`: the saving file has the default name `<basename>.sau`. Only one saving station is produced (assuming that the run terminates normally): 1 (step 4994, i.e. the last step). If the run would fail, say, at step 2500, no (useful) saving time station would be available (there is still one saving station, but it is at step 0).
- `FICH SAUV`: the saving file has the default name `<basename>.sau`. Two saving stations are produced (assuming that the run terminates normally): 1 (step 0) and 2 (step 4994, i.e. the last step). If the run would fail, say, at step 2500, no (useful) saving time station would be available (there is still one saving station, but it is at step 0).

## 12 GROUP H—OPTIONS

### Object:

These keywords give the additional options of the computation. They can be grouped as:

- options associated with time-steps;
- options associated with dampings;
- options for finite elements;
- output options;
- returning to default options;
- options for an advection-diffusion computation;
- options for ALE computations in structures;
- options for debugging purposes;
- options to declare FANTOME certain elements;
- options to define classes of elements within a list of elements;
- options related to the treatment of shocks and impacts;
- options for FSA (fluid-structure interactions of the ALE type);
- options for node-centered finite volumes;
- options for multiphase, multicomponent fluid flows;
- options for the automatic rezoning in ALE computations;
- options for cell-centred finite volumes;
- options for “LIAISONS”/LINKS (connections);
- options for graphical rendering;
- options for mesh-adaptive computations;
- options for strain rate filtering;
- options for parallel computing;
- computational options of the elastic gradient damage model.

### Syntax:

"OPTION"

**OPTION**

Announces that one or several options will be specified.

**Comments:**

The keyword "OPTION" may appear more than once in the EUROPLEXUS data.

The following sub-instructions may appear in any order.

The different options are described on the following pages.

## 12.1 OPTIONS RELATED TO THE TIME-STEP

### Object:

Additional options are given to provide optimum time stepping.

### Syntax:

```

< |[ "PAS" |[ "UTILISATEUR" ; "AUTOMATIQUE" ]| ;
    "PARTITION" $["PLIN" ; "PNOL"]$ <PLOG> ]| >
< "DTVAR"  dtvar      >
< |[ "NOTEST" ; "TEST" ]| >
< "STABILITE"          >    < "STEL" >
(< "NOCRITIC" < $ "UPTO" t ; "TRIG" $ > /LECTURE/ >)
< "CSTAB"      cstab    >
< "PASMINI"     pasmi    >
< "DTFORCE"     dtfor    >
< |[ "STEP" "IO" ; "STEP" "IOT" ; "STEP" "LIBR" ]| >
< "TION"        tionor   >
< "DTML"         >
< "DTBE"         kdtbe   >
< "DIVG"         divg     >
< "DTPDR"        dtdrop   >
< "CMDP" <"NPAS" npas > < "CPUT" cput > >

```

#### PAS UTILISATEUR

The time step is prescribed by the user (see also keyword **CALCUL**). Note that this option cannot be chosen in the case of impacts (the time increment may be limited by the program in case of an impact).

#### PAS AUTOMATIQUE

The time-step is determined by the program (see also keyword **CALCUL**). This is the default, i.e. if neither **PAS UTIL** nor **PAR** are specified the time step is automatically computed by the code.

#### PARTITION

The computation step is partitioned automatically in space (and the step also varies with time), according to the stability step of each element (see also keyword **CALCUL**). **PARTITION** is not available for MPI calculations.

#### PLIN

In the space partitioning procedure, dofs subjected to any links are treated according to the lowest level among the ones that are linked together. This works only with the **LINK** directive, while conditions imposed by the **LIAI** directive are not treated. The option has no effect in cases without space partitioning.

## PNOL

In the space partitioning procedure, dofs subjected to any liaisons or any links (but only of the permanent type) are put in the lowest partition level. This is the default, so this option should not be used, except for changing back from a previously issued OPTI PLIN. The option has no effect in cases without space partitioning.

## PLOG

In case of space partitioning, a special log file is written `<basename>.plog`. This file contains an output line for each sub-cycle, in contrast to the normal log file, which contains one line for each macro step. The extra information may be quite long but is sometimes useful for debugging. By default no such log file is written.

## dtvar

Maximum growth factor of the time step among two subsequent steps in PAS AUTO. Default is 2.0.

## NOTEST

The energy check and related information is not printed at each step, but only when general printouts are required (see ECRI).

## TEST

The energy check and related information is printed at each step. This is the default.

## STABILITE

The energy check and related informations is printed only if EUROPLEXUS reduces the time step.

## STEL

At each step for which a printout is produced, the stability steps  $\Delta t_{\text{stab}}$  for all elements are printed out. The *stability step* is the *critical step*  $\Delta t_{\text{crit}}$  estimated by the code (roughly the element length  $L$  divided by the speed of sound  $c$  in the element material) multiplied by the *safety coefficient*  $\phi$  (CSTA, by default 0.8):  $\Delta t_{\text{stab}} = \phi \Delta t_{\text{crit}} \approx \phi \frac{L}{c}$ .

## NOCRITIC

The elements defined in the following /LECTURE/ will not be considered in the calculation of the critical time step by EUROPLEXUS. In practice, they will be assigned a very large critical step. Optionally (see next keywords) this behaviour can be imposed only until a certain time or event. Note that the NOCR keyword (with its optional sub-keywords) can be repeated as many times as needed to set different criticality limits for the various elements in the mesh, if needed. Each element retains the last criticality limit that has been set for it (if any).

## UPTO t

The above mentioned elements are not considered in the calculation of the critical step only until time  $t$  is reached. Thereafter, they are treated just like any other elements.

## TRIG

The above mentioned elements are not considered in the calculation of the critical step only until a trigger is activated. The trigger refers to the **TRIG** keyword which activates mesh refinement in some adaptivity models, see **OPTI ADAP TRIG** on Page H.180. Thereafter, they are treated just like any other elements.

#### **cstab**

Safety coefficient assumed over the estimated stability (i.e., critical) time step for each element. Default value is 0.8. It is only effective for **PAS AUTO** or **PART**. See also the comments below.

#### **PASMINI**

The calculation will stop if the time increment becomes less than  $\text{dtmax} \times \text{pasmi}$ .

#### **DTFORCE**

The stability step of the more stringent elements is forced to assume the value **dtfor** by increasing their mass. This option is dangerous: see the comments below.

#### **STEP IO**

During the computation, the time step will be adjusted to exactly fit chosen times for output events such as printouts (see **ECRI**), storage of data for post-processing (**FICH ALIC** but not **FICH TPLO** nor **FICH ALIC TEMP** nor **FICH TABL!**), or storage of data for restart (**FICH SAUV**). Note that **TPLOT**, **ALICE TEMPS** and **TABL** data storages are not included (use **STEP IOT** instead). This choice is justified by the fact that **TPLOT**, **ALICE TEMPS** and **TABL** storage times are usually much more numerous than (normal) **ALICE** storages, but include only a limited number of nodes and elements. Note that this option has only effect in **PAS AUTO** or **PART**, but obviously it has no effect in **PAS UTIL**.

#### **STEP IOT**

Same as **STEP IO**, but now output events considered for time step adjustment include also **TPLOT**, **ALICE TEMPS** and **TABL** storages. Note that this option has only effect in **PAS AUTO** or **PART**, but obviously it has no effect in **PAS UTIL**.

#### **STEP LIBR**

During the computation, the step will be varied only according to stability limits. No adjusting to output times for printing, etc., will be performed. In this case, if the user chooses a given printout or storage time, the program will perform the action at the first step in which the time is equal or greater than the specified value. In general, the error on time is small since it is of the order of one time increment. This is the default (as opposed to **STEP IO**).

#### **tionor**

Important: to be effective, this option must be specified **before** the **ECRI** directive. This quantity represents the value of time units used for the normalization of selected times and time frequencies for printing and storage (in particular see the **ECRI** directive and its sub-directives for the different types of output files). It is only relevant to the **STEP IO** or **STEP IOT** options described above. The default value is 1 picosecond (1.D-12 s). Since at least 18 digits are available in an **INTEGER(8)**, the final time of a calculation can be up to 1.D6 s with the standard value of **tionor**. Be aware that the normalization process may



only take place if time values are less than `1.E18*tionor`. An error is produced otherwise. This precision should be largely sufficient in practical cases. In fact, this allows to specify a precision of e.g. `1.E-6` times the typical time step, for a computation with up to `1.E12` steps.

#### DTML

This option chooses a different rule from the standard one to estimate the critical time step of JRC's FLxx fluid elements. The standard rule for FLxx, originating from EUR-DYN, was quite complex and was documented in the report "Implementation of Compressible Fluid Models in PLEXIS-3C", Technical Note No. I.93.86. This rule was found to be inaccurate in some cases. The new rule activated with the present option uses the minimum intra-nodal distance as the characteristic length and the sound speed plus the maximum nodal  $(w - v)$  value (mesh velocity minus fluid velocity) as the characteristic speed. This is in accordance with the rule used in CEA's fluid elements. The DTML option can also be invoked to use the minimum intra-nodal distance for calculation of the stability of C27 elements in 3D (by default these elements use an estimation of the element's stretch and shear to compute the element's characteristic length).

#### DTBE

This option chooses a different rule from the standard one to estimate the critical time step for the POUT element. Three different values can be chosen: `kdtbe = 0` indicates the default version (CEA's formula); `kdtbe = 1` uses an optimized time step (formula for ED01 elements); `kdtbe = 2` considers only the length of the element and disregards the cross section. The default time step used for the POUT element seems to be very conservative. Larger time steps result using the formula for ED01 elements, which is as follows. If the element length  $L$  is larger than  $\sqrt{3}h$ , where  $h$  is the element thickness, then the normal expression is used:  $\Delta t = L/c$ , where  $c$  is the sound speed. Otherwise, the element length is corrected:  $L_{corr} = L^2/\sqrt{3}h$  and then  $\Delta t = L_{corr}/c$ .

#### divg

This options give the possibility to define a value that the energy balance can not exceed. The default is 0.

#### dtdrop

Define the coefficient `dtdrop`. A warning message is printed on the listing each time the stability is imposed by a finite element and the ratio  $\Delta t_2/\Delta t_1$  is smaller than `dtdrop`. The default is 0.3. Some special materials (such as e.g. the JWLS material) used to represent very violent explosions and wave propagations may abruptly reduce the time step in order to preserve stability. In such cases, it may be useful to re-define `dtdrop` to values smaller than the default (e.g., 0.005) in order to avoid too many warning messages on the listing.

#### npas

Define the number of time steps after which the existence of the command file "command.epx" is checked.

#### cput

Define the CPU time after which the existence of the command file "command.epx" is checked. More information about the command file can be found in [17](#).

**Comments:**

Options by default : PAS AUTOMATIQUE TEST STEP LIBR.

The calculation stops if the time step becomes too small. The limit value is proportional to `dtmax` (directive `CALCUL`). By default, `pasmi=0.001`, i.e. the calculation will stop whenever the time step becomes less than  $0.001 \times \text{dtmax}$ . This option is only active when the old syntax of the `CALCUL` directive is used. With the new syntax, `pasmin` is redundant because `DTMIN` directly gives the minimum step (see I.20).

The energy check deals with the energy balance. The value of the stability time step is also printed.

The option `PARTITION` is especially useful when the mesh contains a few very small elements among a large number of bigger ones. In this case, the small elements are paid more attention, without carrying out useless computations on the big ones. This option could be inefficient if used when all elements have nearly the same size, or if there are only a few large elements in the mesh.

Like all explicit programs, EUROPLEXUS requires a sufficiently small time increment in order to ensure the stability of calculations. By default, EUROPLEXUS uses the CFL time step (Courant-Friedrichs-Lévy condition), multiplied by a safety coefficient `CSTAB = 0.8`. However, for very fast phenomena this condition may be insufficient. It is then possible to ensure stability by assuming for `CSTAB` a value smaller than 0.8.

The option `DTFORCE` affects only Lagrangian elements. In an ALE calculation, only the Lagrangian elements (if any) will be considered, and the others will be ignored. Since the mass of elements is modified, it is necessary to check that such modifications do not affect too much the physics of the problem.

To this end, some indications are available on the listing:

- The mass by zones before and after modification;
- The list of the 20 most constraining elements, with the old and the new time step;
- A message `ATTENTION` also appears if the mass of a zone increases more than 10 %.

## 12.2 OPTIONS RELATED TO THE DAMPINGS

### Object:

To enter the dampings.

### Syntax :

```
|[  "QUASI" "STATIQUE"  fsys  beta <"FROM" t1> <"UPTO" t2> ;
    "AMORT" "LINE"      betal                                     ;
    "AMORT" "AXIA"      betal                                     ;
    "AMORT" "QUAD"      a2                                       ;
    $[ "HOURL" <"ASSE"> ; "VISC"  hvis ; "RIGI" hrig ; "NOHOURL" ]$ ]|
```

#### QUASI STATIQUE

Quasi static computation. A linear damping with a given cut-off frequency is applied.

#### fsys

Frequency  $f$  of the first system mode to be cut off.

#### beta

Reduced damping coefficient  $\beta$ .

#### t1

Initial time  $t_1$  at which the quasi-static option starts to operate. By default, this coincides with the initial time of the calculation. See comments below for the use of  $t_1$  and  $t_2$  to define a closed interval or two open intervals.

#### t2

Final time  $t_2$  until which the quasi-static option operates. By default, this coincides with the final time of the calculation. See comments below for the use of  $t_1$  and  $t_2$  to define a closed interval or two open intervals.

#### AMORT LINE

Computation with linear damping of high frequencies (artificial viscosity). This damping is advisable in FE calculations (CEA model: CAR1, CUBE etc.) involving a **liquid**, but it can also safely be added in calculations involving gases. The value to be used is between 0.05 and 0.20 in general. Note that this damping has no effect on calculations with cell-centred finite volumes (VFCC). In fact, the scheme with limiters used in that case is built in such a way that it does not need any damping. Note also that linear damping in FE models for fluids from JRC (FLxx elements) is activated by the keyword CL of the FLUT material, see Page C.520.

#### AMORT AXIA

Computation with linear damping of high frequencies, but only for the elements on the symmetry axis (for 2D axisymmetric problems only).

**betal**

Reduced damping coefficient  $\beta_l$  for linear damping (of type LINE or AXIA).

**AMORT QUAD**

Computation with quadratic damping (artificial viscosity). This damping is advisable in FE calculations (CEA model: CAR1, CUBE etc.) involving a **gas**, but it can also safely be added in FE calculations involving liquids. The value to be used is between 2.0 and 4.0 in general. Note that this damping has no effect on calculations with cell-centred finite volumes (VFCC). In fact, the scheme with limiters used in that case is built in such a way that it does not need any damping. Note also that quadratic damping in FE models for fluids from JRC (FLxx elements) is activated by the keyword CQ of the FLUT material, see Page C.520.

**a2**

Coefficient  $a_2$  for the quadratic damping (shock waves).

**HOOR ASSE**

Anti-hourglass damping for assumed-strain elements (IHOARG=3).

**VISC**

Anti-hourglass damping on viscous terms (IHOARG=1).

**hvis**

Reduced damping for anti-hourglass  $h_{vis}$  ( $h_{vis} = 0.5$  is suggested).

**RIGI**

Anti-hourglass damping based on artificial stiffness (IHOARG=2).

**hrig**

Reduced damping for anti-hourglass.

**NOHOARG**

Allows to eliminate the anti-hourglass damping.

**Comments :**

In the case of the QUASI STATIQUE option,  $\beta = 1$  corresponds to the critical damping for the frequency  $f$ . In fact, one adds an external force  $F_i^{QS}$  proportional to the mass  $M_i$  and to the particle velocity  $v_i$  for each degree of freedom  $i$ :

$$F_i^{QS} = -4\pi\beta f M_i v_i = -2\beta\omega M_i v_i$$

where  $\omega = 2\pi f$ .

In practice, only the product  $\beta f$  is relevant.

Linear damping of high frequencies is only possible for elements of type CAR1, CAR4, TRIA, TUBE, FUN2 and FUN3. This damping allows to eliminate the high-frequency oscillations

related to the finite element discretization. In order to obtain the critical damping of a free-free oscillation for each element, take  $\beta_l = 1$ .

When  $t_1$  is less than  $t_2$  (be these values specified or not) the quasi-static damping acts in the closed time interval  $t_1 \leq t \leq t_2$ , i.e. in the central part of the transient calculation. However, it is also possible to specify  $t_1$  greater than  $t_2$ : in this case the critical damping acts in the open time interval  $t \leq t_2$  (i.e. at the beginning of the calculation) and in the open time interval  $t_1 \leq t$  (i.e. at the end of the calculation). This second form of the directive may be useful when one wants to model a structure initially subjected only to gravity loads (with quasi static option so as to rapidly reach the initial static deformed configuration), followed by a dynamic event such as an explosion (without quasi static option), and finally by a stabilization phase (again with quasi static option) so as to rapidly compute the final static deformation. Thus this form of the directive allows to perform the complete analysis of the three phases in just one run of the code, instead of running three separate calculations (each one starting from the results of the previous one) via e.g. the directive `INIT ALIC` (see page E.140).

For the quadratic damping, it is suggested to take  $a_2 = 4$ .

Quadratic damping is only possible for elements of types CAR1, CAR4, CUBE, PRIS, TRIA and TUBE.

The present linear and quadratic damping models are distinct from the selective damping model (`AMOR`) described on page C.106, which applies to selected dofs and nodes of a zone specified by the user.

The anti-hourglass damping is currently available only for the elements CAR1 et CUBE. By default, an anti-hourglass damping with  $h_{\text{vis}} = 0.5$  is affected to a calculation. If the user wants to do a calculation without anti-hourglass damping, he must use the option `NOHOURG`.

### Warnings :

In case of restart, the `QUASI STATIQUE` damping remains active; to eliminate it, one must specify  $\beta = 0$ .

Linear damping should be used with care, since it may considerably perturbate the solution. It is advisable not to exceed the value  $\beta_l = 0.05$ . In case of axisymmetric linear damping, since the concerned elements are usually a few and with a small mass, one may go up to  $\beta_l = 0.5$ .

## 12.3 OPTIONS FOR FINITE ELEMENTS AND GEOMETRIC ISSUES

### Object:

To introduce optional parameters related to finite elements and geometric issues.

### Syntax:

```

<  "DECENT"  |[          "TOTAL" ; "CALC"          ;
                        "IMPOSE" "DCEN" de "DCMA" dm ]| >
<  "ROLIM"   rholim                                     >
<  "JAUMAN"                                     >
<  "CODG" < "REFE" zbar >    <SMAL>                                     >
<  "EDSS"                                     >
<  "LFUN"                                     >
<  "P2X2"                                     >
<  $ "NF34" ; "OF34" $                                     >
<  "MOMT" kmtran                                     >
<  "TOLC" tol c                                     >
<  "HGQ4" hgq4ro                                     >
<  "CLMT" < "FARF" farf> < " ABSI" absi>                                     >
<  "LMST"                                     >
<  "NTSM"                                     >

```

#### DECENT CALC

A.L.E. only. Upwinding computed by EUROPLEXUS according to the volume covered in one time step with respect to the total volume.

#### DECENT TOTAL

A.L.E. only. Total upwinding for the mass.

#### DECENT IMPOSE

A.L.E. only. Prescribed upwinding.

#### de

Upwinding concerning transport terms.

#### dm

Upwinding concerning mass fluxes.

#### ROLIM rholim

ALE/Eulerian only: if the donor element has a density less than `rholim`, the mass and energy fluxes are not considered for this element.

#### JAUMAN

Large strain computation with JAUMAN's stress tensor.

**CODG**

Introduces options for calculations with degenerated shell elements (CQDx).

**zbar**

Parameter defining the position of the reference surface for degenerated shell elements: -1 indicates the lower element surface, 0 the mean surface, +1 the upper surface. By default,  $zbar = 0$ .

**SMAL**

Specifies that a small strain model of membrane deformation has to be used for degenerated shell elements, so the thickness of these elements stays constant. By default, a large membrane deformation is assumed and the element thickness is varied accordingly. This option is only useful to compare a solution with an old run done by JRC's SHELL3D.

**EDSS**

Specifies that certain elements (ED01, FUN2, FUN3) will adopt a small strain, large displacements, large rotations formulation instead of the large strain formulation that is used by default.

**LFUN**

Specifies that certain elements (FUN2, FUN3) will adopt a fully linear, small strain model: element cross section stays constant and also length stays constant for the calculation of critical time step (which is therefore constant). This option should only be used for debugging purposes and for the study of time integration algorithms (to compare analytical and simplified numerical solutions).

**P2X2**

This option activates a spatial integration rule for pressure forces in CEA's fluid elements (CUBE, PRIS, TETR) which is equivalent to a 2x2x2 Gauss rule, and is therefore exact also for distorted geometry (e.g. non-planar faces). The standard rule uses a single-point scheme which is under-integrating the function in the presence of distortions. The resulting inaccuracy of pressure force computation leads to the effect that fluid nodes internal to the fluid domain and completely surrounded by a fluid at uniform pressure are not in perfect equilibrium when the surrounding mesh is irregular. Spurious resultant pressure forces cause spurious velocities in the fluid which are non-physical. Although these velocities were usually found to remain relatively small with respect to physical ones in typical applications (explosions etc.), it is generally preferable to avoid them altogether by using the present option, although it is of course slightly more computationally expensive. The standard rule (single-point) is left as a default for compatibility with old input files and applications.

**NF34**

Use new (2007) implementation for FL34 JRC's tetrahedral 4-node fluid element. The new implementation is described in reference [235]. From April 2014 this is the default, so it should be rarely necessary to specify this option.

**OF34**

Use old (before 2007) implementation for FL34 JRC's tetrahedral 4-node fluid element.

**MOMT**

This option allows choosing the degree of precision for the spatial integration rule used in the computation of momentum transport forces in Eulerian or ALE calculations using JRC's FL3x fluid elements. The **kmtran** parameter may assume the values 0 (no momentum transport forces at all), 1 (corresponding to single-point integration), 2 (for 2x2x2 spatial integration) or 3 (3x3x3 spatial integration). For distorted geometries only the 3x3x3 rule is exact. The default rule (as used in EURDYN) is the single-point one which is of course the most economical, but unfortunately may lead to spurious mechanisms (appearance of spurious fluid velocities) in some cases, typically when the geometry of the elements is irregular or distorted (e.g., non-planar faces). The mechanisms may rapidly grow and in some cases they completely destroy the numerical computation. In all practical cases investigated so far it was found that a 2x2x2 rule (MOMT 2) is accurate enough and sufficient to prevent the appearance of mechanisms. The cost of this is of the order of 20% to 30% overhead compared with the default, (MOMT 1) option. The MOMT 3 option is exact, but may cause a 100% overhead in computer time. Finally, note that the MOMT 0 option is only to be used for debugging purposes, since computations without momentum transport forces are of course largely inaccurate.

**TOLC**

This option allows to change the tolerance **tolc** that is used to automatically search for node correspondence, see page C.92. The default behaviour (no OPTI TOLC) is that two nodes are considered to match if their initial positions differ, along each one of the global coordinate axes, by less than 1.E-4 times the “mean” size of the mesh. This mean size is defined as the sum of the sizes of the mesh along each one of the global axes, divided by the space dimension. If **tolc** is explicitly specified, it is retained as the maximum distance between two coincident nodes along the global axes. In this case therefore, the above mentioned mean mesh size is not computed: **tolc** is used directly. Note that, in order to be effective, this option must be specified *before* the directives that might use it, in particular before the LIAI FSA directive.

**hgq4ro**

Adjusting coefficient for the anti-hourglass rotation stiffness of the Q4GR shell element. The default value of **hgq4ro** is 0.018.

**CLMT**

This keyword introduces options for the treatment of momentum transport forces in fluid Finite Elements (JRC formulation, i.e. FLxx family of elements). It applies to the CL22, CL2S, CL3I, CL3Q and CL3S element types, associated with either a FLUT material (for far-field conditions) or an IMPE ABSI material (for absorbing boundary conditions).

**FARF farf**

Use **FARF 1** to activate momentum transport forces in CLxx due to far-field conditions, or **FARF 0** to de-activate them. The default is 0, i.e. no momentum transport forces.

**ABSI absi**

Use **ABSI 1** to activate momentum transport forces in CLxx due to absorbing (IMPE ABSI) conditions, or **ABSI 0** to de-activate them. The default is 0, i.e. no momentum transport forces.

**LMST**



The LMST option (for Large Membrane STRains) is used to activate the update of the thickness of some shell elements (from CEA) due to large membrane strains. The affected elements are Q4GS and T3GS. Note, however, that the thickness update is activated only if such elements possess a non-linear material (i.e. *other* than LINE or GLRC). By default, the thickness of such elements is not updated even if large membrane strains occur. Note also that the thickness of other shell elements from CEA (namely Q4GR, QPPS, DST3, DKT3, T3MC) is also *never* updated and the present option will have no effect on such shell elements.

#### NTSM

The NTSM option deactivates the smoothing of the pressures of tetrahedrals. Tetrahedral elements show a pressure locking in particular for non-linear materials. A very strong locking can be identified in impact calculations. A smoothing of the pressures over the neighbour elements solves that problem. This option is activated by default. The keyword NTSM deactivates that option.

#### Comments:

A large strain calculation with the JAUMAN tensor is only possible at the moment with elements "CAR1", "CAR4" and "TRIA".

The upwinding is only effective for a computation with a non-Lagrangian formulation (keyword "ALE" or "EULER" in the type of problem to deal with , see page A.30).

By default, EUROPLEXUS uses the total upwinding ( $\mathbf{dm} = 1$  and  $\mathbf{de} = 0$ ).

## 12.4 OPTIONS FOR FLYING DEBRIS

### Object:

To introduce optional parameters related to the flying debris model, described on Page C.66.

### Syntax:

```
< "DEBR" <"NTRA" ntra> <STTR> <MAC1 ; MACN> >
```

#### DEBR

Starts the specification of debris-related options.

#### NTRA ntra

Number of points **ntra** for flying debris trajectories. The points are equi-spaced in time between the initial time and the final time of the calculation given in the **CALC** directive. The actual number of points will be **ntra + 1**, since also the initial position (initial time) is stored. The default value of **ntra** is 100 points (i.e., 101, if one counts also the initial point).

#### STTR

Store the flying debris trajectories on the ALIC file. This will allow visualizing the trajectories when reading back the results (**RESU**). If this option is not set, the trajectories are not stored in the ALIC file (because these data may be huge, if there are many particles), and in this case the trajectories can only be visualized during the main calculation (**not** when reading back the results).

#### MAC1

Associate only one marker (for the evaluation of macro debris risk) with each element. This is the default, so the keyword is only useful to erase the effect of the **MACN** keyword presented next.

#### MACN

Associate more than one marker (for the evaluation of macro debris risk) with each element. The actual number depends on the element type and on the **PLEV** specified in the **DEBR FILL** directive shown on Page C.66. This option is only useful for debugging purposes. It may be used to reproduce old solutions obtained with EPX versions until 2016, which adopted a different strategy for the markers.

## 12.5 OUTPUT OPTIONS

### Object:

These options enable the output format to be chosen.

### Syntax:

```

< $[ "NOPR" ;
      "PRIN" < "PMESH" > < "PCAST" > < "PCOMP" >
              < "PGRID" > < "PLOAD" > < "PLINK" >
              < "PRESU" > < "PLAW" > < "PMED" >
              < "PE2M" > ]$ >
< "DPMA" >
< $[ "NWAL" ; "WALI" ]$ >
< $[ "NWSA" ; "WSAU" ]$ >
< $[ "NWTP" ; "WTPL" ]$ >
< $[ "NWXF" ; "WXPL" ]$ >
< $[ "NWAT" ; "WATP" ]$ >
< $[ "NWK2" ; "WK20" ]$ >
< $[ "NWST" ; "WSTB" ]$ >
< $[ "NOEC" ; "ECHO" ]$ >
< "LOG" nlog >
< "K2FB" k2fibe >
< $[ "K2CH" ; "K2GP" ]$ >
< "K2MS" |[ "MANU" ; "READ" ; "SAUV" ]| >
< "DYMS" nobj*("OBJE" /LECT/) >
< "PRGR" >

```

#### NOPR/PRIN

This option allows to suppress or re-activate a part of the printouts of the following directives.

If one of the keywords PRIN/NOPR is followed by one or more parameters, only the corresponding parts of the listing are activated (or deactivated)

"PMESH" : mesh (nodal coordinates and elements topology)

"PCAST" : detail of the CASTEM objects

"PCOMP" : geometrical complements

"PGRID" : parameters of the ALE rezoning

"PLOAD" : details of the charges

"PLINK" : details of the liaisons/links

"PRESU" : details of the results files

"PLAW" : details of the material laws

"PMED" : detail of the MED objects

"PE2M" : table of MED element number associated to each EPX element

See also the comments below.

"DPMA"

Prints nodal and element masses with each general printout. This can be useful to check masses in problems where the mass varies, such as ALE calculations.

**NWAL**

No printout on the listing of information about each storage of data for ALICE (see "FICH ALIC").

**WALI**

A line of information containing the time, step number, etc. will be printed on the output listing at each storage of data on the ALICE file (see "FICH ALIC"). This is the default option.

**NWSA**

No printout on the listing of information about each storage of data for restart (see "SAUV").

**WSAU**

A line of information containing the time, step number, etc. will be printed on the output listing at each storage of data on the restart file (see "SAUV"). This is the default option.

**NWTP**

No printout on the listing of information about each storage of data for TPLOT (see "FICH TPLO"). This is the default option, since usually many storages are requested for TPLOT.

**WTPL**

A line of information containing the time, step number, etc. will be printed on the output listing at each storage of data on the TPLOT file (see "FICH TPLO").

**NWXP**

No printout on the listing of information about each storage of data for XPLOT (see "FICH XPLO").

**WXPL**

A line of information containing the time, step number, etc. will be printed on the output listing at each storage of data on the XPLOT file (see "FICH XPLO"). This is the default option.

**NWAT**

No printout on the listing of information about each storage of data for ALICE TEMPS (see "FICH ALIC TEMPS"). This is the default option, since usually many storages are requested for ALICE TEMPS.

**WATP**

A line of information containing the time, step number, etc. will be printed on the output listing at each storage of data on the ALICE TEMPS file (see "FICH ALIC TEMPS").

**NWK2**

No printout on the listing of information about each storage of data for K2000 (see "FICH K2000").

**WK20**

A line of information containing the time, step number, etc. will be printed on the output listing at each storage of data on the K2000 file (see "FICH K2000"). This is the default option.

**NWST**

No printout on the listing of information about each storage of data for SUPERTAB (see "FICH SPTAB").

**WSTB**

A line of information containing the time, step number, etc. will be printed on the output listing at each storage of data on the SUPERTAB file (see "FICH SPTAB"). This is the default option.

**NOEC/ECHO**

This option allows to suppress or re-activate input data echo in the EUROPLEXUS window.

**LOG**

Causes a one-line information to be written to standard error file each 'nlog' time steps. The information includes current step number, time, CPU time, critical step, critical element, energy check and mass check. This is useful e.g. to monitor the execution of very long and CPU-intensive runs. Usually, the standard error information will be redirected to a file, e.g. with the Unix command '2>file'. The columns of the log files (S standard calculation, P calculation using partitioning) are described in the table below.

	Description	S	P
STEP	Time step number (main step for Partitioning)	X	X
TIME	Time	X	X
CPU(S)	CPU time used	X	X
DTCRIT	Critical time step used	X	
ELCR	Element with the smallest time step	X	
DELMIN	Time step of the smallest substep		X
MINS	Minimum level factor		X
DE/E	Energy balance per element	X	X
DM/M(NOD)	Mass balance per node	X	X
DM/M(ELE)	Mass balance per element	X	X
DTMX	Maximum time step	X	
EL	Element of the maximum time step	X	
DELMAX	Time step of the main step		X
MAXS	Maximum level factor		X
VITMAX	Maximum velocity	X	X
NODE	Node of the maximum velocity	X	X
ISUBTO	Total number of substeps		X
MAXSTO	Total number of substeps		X
ELSTEP	Number of callings of element routines	X	X

**K2FB**

Indicates the index of the Gauss Point, along each fiber, for which variables are stored for subsequent K2000 postprocessing. For example, if there are 5 GPs along fibers in the shell elements used in a calculation, then `k2fibe = 1` indicates the GPs closest to one face of the structure, `k2fibe = 5` indicates the GPs closest to the opposite face of the structure, `k2fibe = 3` indicates the GPs on the midsurface of the structure, and so on. Note that this parameter has only effect for shell elements of types ED01, ED41, COQI and CQDx. The default value is `k2fibe = 1`.

#### K2CH

With this option, the output chameleons for K2000 will be defined for each element at the element nodes, rather than at the element barycenter (default) or at the Gauss points (K2GP option). Note, however, that the computation of values is crude: an average on all GPs is computed, and this value is affected to all nodes of the element (although the contributions to the same node from different elements may be different). The default (without the K2CH option) is to compute an average on all GPs and affect this value to the barycenter of the element.

#### K2GP

With this option, the output chameleons for K2000 will be defined for each element at the Gauss points, rather than at the element barycenter (default) or at the element nodes (K2CH option). The exact value is affected at each GPs of the element. In case of multilayer plates (CEA-plates: DKT3, Q4GS...) an average on the GPs in the thickness is computed, and each of these values is affected to the corresponding GP on the surface of the element. The default (without the K2GP option) is to compute an average on all GPs and affect this value to the barycenter of the element.

#### K2MS

With this option, the code will produce a file containing a series of GIBIANE instructions that, when processed by CASTEM2000, will produce the current mesh in CASTEM2000 format. This option is only useful when the mesh has been produced by a pre-processor different from CASTEM2000 (see also comments below).

#### MANU

The CASTEM2000 mesh generation commands will use the CASTEM2000 operator MANU. The name of the generated file is `pxtok200.dgibi` on the current directory

#### READ

The data for CASTEM2000 will be written on file `pxtok200.inp` on the current directory. These data are suitable to be read by CASTEM2000 via the READ operator (see also comments below).

#### SAUV

With this option, the code will produce a file containing the mesh in Cast3m's GIBI format, similar to what would be obtained by producing the mesh by Cast3m and saving it with Cast3m's SAUV command. The file produced has the fixed name `PXTOK200.SAUV`. The named element groups will be available but no relationships (dependencies) between the sub-objects will be created. To read back the file in Cast3m, use the commands: `OPTI REST FORM 'PXTOK200.SAUV'; REST FORM;`

#### DYMS

With this option, the code will produce an input file for LS-DYNA. For each of the `nobj` objects defined by the `OBJE` keyword (which must be repeated exactly `nobj` times), the nodes and elements are written in this file. No material and load definitions are exported.

#### PRGR

Print named element and node groups on the listing in a format that can be directly included in a .EPX file (on 72 columns and using `LECT ... PAS ... TERM` syntax). This printout is made *in addition* to the normal printout of named groups on the listing. To find the group of lines search for `COMP GROU` and for `COMP NGRO` in the listing. Note that in order to be effective, this option **must** be set **before** the definition of the named groups.

#### Comments:

The presence of `OPTI NOPR` immediately after the dimensioning in the input file minimizes the listing file. On the contrary, `OPTI PRIN` maximizes the listing file. It is possible to activate or deactivate the various printouts selectively. For example:

```
OPTI  NOPR  PMESH  PCAST  PLINK
```

will deactivate the printouts relative to the mesh, the CASTEM objects and the liaisons/links. This allows to avoid repeating the commands `NOPR` and `PRIN` within the input file.

In case of re-reading the results file (file `ALICE` or `ALICE TEMPS`) the option `NOPR` is taken by default. To have complete printouts, it is sufficient to add `OPTI PRIN` after the keyword `TERM` of directive `DIME`.

The `K2MS` option can be very useful in the case that an input file for EUROPLEXUS uses a mesh defined in a format different from CASTEM2000, but the user wants to do the post-processing of the calculation by CASTEM2000 (or to manipulate the mesh in CASTEM2000 before running the actual EPX calculation). This option will produce a file containing data that can be used by CASTEM2000 to generate the desired mesh.

Typically, in such cases one would perform the following steps:

1. - Run the EUROPLEXUS input file with the non-CASTEM mesh, including option `K2MS`. The calculation can be stopped at step 0 (use `VERI` or `CONV TEK` and then the stop interactive command). This will produce a file of data for CASTEM2000 in either file `pxtok200.dgibi` or file `pxtok200.inp` on the current directory.
2. - Run CASTEM2000 on the above mentioned file, to produce a mesh in CASTEM2000 format. See below for examples and details.
3. - Finally, run again EUROPLEXUS by specifying that the input geometrical data are from CASTEM2000 (CASTEM directive). Now, a CASTEM2000 post-processing file can be produced by EUROPLEXUS, because the input is indeed in CASTEM2000 format.

Note, however, that the CASTEM2000 mesh produced by this method will be somewhat special. The global mesh will be accessible as on bject named `MESH`. In addition, *but only if the `K2MS MANU` option is used*, then also all named element groups and all named node groups present in the original EPX input file will be available in the automatically generated CASTEM2000

mesh. However, no other *sub-objects* (in the sense of CASTEM2000 mesh generation building blocks) will be available.

When the K2MS MANU option is used, the file produced (`pxtok200.dgibi`) will contain a line for each node, of the form:

```
Pxxxxx = xcoor ycoor [zcoor];
```

where `xxxxx` is the node number (e.g., 00025 for node 25), `xcoor`, `ycoor` (and `zcoor` in 3D) are its coordinates.

For example:

```
P00332 = 1.000000000000D+01 1.000000000000D+01 ;
```

Then, for each element there will be a line of the form:

```
Eyyyyy = manu elem node1 node2 ... ;
```

where `yyyyy` is the element number, `elem` is the element type according to CASTEM2000 (e.g., QUA4 for 4-node quadrilaterals) and `node1`, `node2` etc. are its nodes. For example:

```
E00002=manu QUA4 P00004 P00006 P00005 P00003;
```

The global object will be called `MESH`. If you need to define sub-objects (in addition to the EPX named groups), use appropriate GIBIANE instructions.

A typical CASTEM2000 command file using `pxtok200.dgibi` is as follows:

```
(pxtok200.dgibi as produced by EUROPLEXUS) ...
```

```
mesh3 = mesh ELEM 'TRI3';
mesh4 = mesh ELEM 'QUA4';
...
opti sauv 'file';
sauv mesh;
```

In addition to file `pxtok200.dgibi`, another CASTEM2000 input file `pxrest.dgibi` is also automatically produced in this case. By running CASTEM2000 on `pxtok200.dgibi` first, the CASTEM2000 mesh is produced and saved in SAUV format. Then, by running CASTEM2000 on `pxrest.dgibi`, the CASTEM2000 mesh is read back (just for checking) from the SAUV file.

Unfortunately, it has been noted that CASTEM2000 changes the numbering of elements in a mesh generated in this way. The other method (using the READ option) can be used in cases this could cause trouble (which is typically the case if other input directives in the EUROPLEXUS input file use element numbers). Or, alternatively, try using the SORT operator instead of the SAUV operator to save the mesh, as detailed below.



When the K2MS READ option is used, the file produced (`pxtok200.inp`) contains a simple list of nodal coordinates and element topology (by zones). These data can be read by CASTEM2000 using the READ operator developed at JRC. No named element and node groups are translated into CASTEM2000 objects in this case.

To this end, use a command file of the form:

```
...
mesh = READ 'pxtok200.inp' MESH ELEM;
mesh3 = mesh ELEM 'TRI3';
mesh4 = mesh ELEM 'QUA4';
...
opti sauv 'file';
sauv mesh;
```

From the tests performed, it seems that CASTEM2000 maintained the element numbering in this case, but only up to version 9 of the SAUV operator included. For higher versions of the SAUV operator, numbering is generally changed.

In order to try to avoid renumbering, use the CASTEM operator SORT instead of SAUV to save the mesh. The SORT operator is more limited than SAUV (it may only save meshes, for example), but has the advantage that it apparently does not change mesh numbering, and its implementation is somewhat “frozen” in the code, unlike the SAUV operator which evolves constantly.

Recall that a mesh saved with SORT must be read in EUROPLEXUS by the GIBI directive, not by the CAST directive (see page A.30), and that SORT files are formatted by default.

The command file will be in this case of the form:

```
...
mesh = READ 'pxtok200.inp' MESH ELEM;
mesh3 = mesh ELEM 'TRI3';
mesh4 = mesh ELEM 'QUA4';
...
opti sort 'file';
sort mesh;
```

In EUROPLEXUS, the mesh will be read as follows:

```
...
GIBI 'file' mesh
...
```

## 12.6 RETURNING TO DEFAULT OPTIONS

**Object:**

To set the options relative to a standart computation back to their default values.

**Syntax:**

```
< "ZERO" >
```

**ZERO**

Discards any previous options, returning to default values.

**Comments:**

All the options which have been defined previously are discarded, and the options by default are assumed again.

## 12.7 OPTIONS FOR AN ADVECTION-DIFFUSION COMPUTATION

### Object:

To provide options for an advection-diffusion computation.

### Syntax:

```
<  "ADDF" < "GRAV" gravi > < "PSYS" psyst >
      < "ELEM" ielref > < "SORD" nsord >
      < "NGAU" ngau > < "ITER" nitef >
      < "ITEP" niter > < "TOLER" titer >
      < "ADTI" adtime > < "ERRO" errix >
      < "NIMA" nimax > >
```

#### gravi

Acceleration of gravity (default=0.0).

#### psyst

System pressure, used to remove the singularity of the pressure field solution matrix (default=0.0).

#### ielref

Index of element in which the pressure is equal to psyst. (default=1)

#### nsord

When 2, 3 or 4, a Taylor-Galerkin method is used of order 2, 3 or 4, respectively (default=2). When nsord=5, a Least-square, space-time method is used. When nsord=6, a Least-square, Crank-Nicolson method is used.

#### ngau

Number of Gauss points in each direction for the integration of advection terms, can be 1 or 2 (default=1).

#### nitef

Number of iterations in the factorization of the consistent mass matrix during the advection phase, can be 1 to 9. (default=3)

#### niter

Maximum number of iterations for the solution of the system of equations for the pressure phase. If set to null, a direct solution is performed (default=0).

#### titer

Convergence tolerance for the iterative solution of pressure phase equations (default=0.01).

**adtime**

Time step fraction.

**errix**

Tolerance of implicit resolution. Is only used with Least-square method (see nsord above).

**nimax**

Maximum number of iterations for implicit resolution. Is only used with Least-square method (see nsord above).

## 12.8 OPTIONS FOR ALE CALCULATIONS IN STRUCTURES

### Object:

To provide options for an ALE calculation in structures.

### Syntax:

```
< "ALES" |[ "KINT" kintm ; "UPWM" upwm ; "UPWS" upws ]| >
```

#### kintm

Integration type for momentum transport: 0 means 1x1 (not available for the moment!), 1 means 2x2 (exact for plane problems), 2 means 3x3 (exact for axisymmetric problems). Default is 1.

#### upwm

Upwind parameter for momentum transport, can be chosen between 0 and 1 (default is 1.0).

#### upws

Upwind parameter for stress transport, can be chosen between 0 and 1 (default is 1.0).

## 12.9 OPTIONS FOR DEBUGGING

### Object:

To provide options to help in debugging the program (for developers only).

### Syntax:

```
< $[ "DUMP" ; "NODU" ]$          >
<  "DPAS" /LECTURE/              >
<  "DPEL" /LECTURE/              >
<  "DPEM"                        >
<  "VIDA" /LECTURE/              >
<  "DPGR"                        >
<  "OLDS"                        >
<  "DPCA"                        >
<  "DPLE"                        >
<  "DPLM"                        >
<  "DPSD"                        >
<  "DPAR"                        >
<  "DPAX"                        >
```

#### "DUMP"

Prints dump of variables as long as they are initialised in the various routines before starting time integration. Of course, this option tends to produce extremely large output files and is only useful for very small test cases, for program development.

#### "NODU"

Turns off dumping option.

#### "DPAS"

The following list enumerates the integration time steps for which extensive information has to be dumped out. A maximum of 200 step indexes can be specified (this dimension is fixed).

#### "DPEL"

The following list enumerates the elements for which extensive information has to be dumped out. A maximum of 20 element indexes can be specified (this dimension is fixed).

#### "DPEM"

Prints (on the log file!) tables of available elements and materials in a format suitable for rapid inclusion in this user's manual.

#### "VIDA"

The following list indicates the indexes of the variables to be dumped (these can range from 1 to the total number of variables, see include MAPORGA), a value of 0 indicates that the contents of the commons has also to be dumped. Note that the commons are dumped at the moment when the directive 'OPTI VIDA LECT 0 TERM' is encountered in the input file, therefore it is suggested to place this directive just before the 'CALC' directive, which starts the time-marching calculation.

**DPGR**

Prints a table containing the list of all nodes with their grid motion attributes: L for Lagrangian, E for Eulerian, AA for ALE, manually rezoned, AM for ALE, automatically rezoned, AS for ALE, rezoned by "FSS ALE", AZ for ALE, rezoned by "MEAN". The dump is performed after complete processing of the input, immediately before starting the time loop. This allows to check possible changes applied by the program to conditions imposed by the user through the "GRILLE" directive. This option is only active for Eulerian or ALE calculations.

**"OLDS"**

Specifies that an old model for the VM23 material has to be used in place of the most recent model. The old model was slightly less accurate in elastoplastic cases and was used in the EURDYN programs. This option should only be used for debugging purposes, if a very precise comparison with an old EURDYN calculation is desired.

**"DPCA"**

Prints on the listing tables of element and material characteristics. For the elements, the NCEL variables are listed in tabular form, for the materials the MATALE and LGEP variables are listed.

**"DPLE"**

Prints on the listing a table of element characteristics in L<sup>A</sup>T<sub>E</sub>X input format. This may then e.g. be edited for inclusion in the present User's Manual.

**"DPLM"**

Prints on the listing a table of material characteristics in L<sup>A</sup>T<sub>E</sub>X input format. This may then e.g. be edited for inclusion in the present User's Manual.

**"DPSD"**

In multi-domain calculations, dumps out extra information on the listing file. Furthermore, for each sub-domain a separate log file is produced that reports, at every time station, a line collecting information relevant to the sub-domain. The name of such files is <base\_name>\_xxx.log, where xxx is the index of the sub-domain (e.g. 012 for the twelfth sub-domain), and base\_name is the base name of the test case (without the extension .epx). By examining these log files, one is able to follow precisely the time integration history of each sub-domain. At most 10 such log files are produced, therefore if the number of sub-domains is larger only the first 10 sub-domains will be dumped out.

**"DPAR"**

In calculations with space partitioning, dumps out extra information on the listing. All cycles, in addition to macro steps, are printed out.

**"DPAX"**

Dump out on the listing a list of all nodes on the axis of revolution i.e. nodes with  $x = 0$ . This option has only effect in 2D axisymmetric calculations, and must be issued **before** the GEOM directive.

### Comments:

Another useful debugging tool is the "ECHO" "VERI" directive (see page A.20) that causes, among other things, the memory allocated to each variable to be printed out.

Concerning the "DPSD" option, note that the per-domain log files are automatically opened under the Windows platform. On non-windows platforms (e.g. Unix), it may be necessary to explicitly open these files by including in the input file appropriate OPNF directives (see page A.28). Here is an example:

(on non-Windows platform)

```
OPNF FORMAT 51 '/disk1/fauvin/SD_001.LOG'
OPNF FORMAT 52 '/disk1/fauvin/SD_002.LOG'
. . .
OPTI DPSD
. . .
STRUCTURE 2
  DOMA LECT ZON1 TERM
  DOMA LECT ZON2 TERM
. . .
```

In this example there are 2 sub-domains. Note that the unit numbers to be used are 51, 52, etc. up to 60 (max. 10 sub-domains). The names associated with the files are arbitrary, and the files are formatted. On some platforms, full-path names only are accepted as in the above example.



## 12.10 PHANTOM OPTION (Element erosion by time)

### Object:

Elements are eroded when the time exceeds a given value.

### Syntax:

```
"FANTOME"      t_fant  /LECTURE/
```

t\_fan

Time starting from which the elements become eroded.

/LECTURE/

List of the concerned elements.

### Comments:

This option may appear at most once. However, it is possible to declare as many sequences t\_fant, /LECTURE/ as needed.

In order to use this option, do not forget to specify the **EROS** keyword in the problem type, see [GBA\\_0030](#). The value of **ldam** after **EROS** must be also given, but it has no effect on the present option.

## 12.11 CLASS (For a post-treatment with the directive REGION)

### Object:

This directive allows to create classes of elements within a list of elements. Each element of the list of elements may belong to one and only one class.

### Syntax:

```
"CLASSE"      /LECTURE/  
              nb_classes*(/LECTURE/)
```

```
/LECTURE/
```

List of elements.

```
nb_classes
```

Number of classes.

```
/LECTURE/
```

List of elements of each of the classes.

### Comments:

This option may appear at most once.

This option must be associated with the directive REGION defined on the list of elements to obtain informations on the classes (see G.100).

## 12.12 SHOCK AND IMPACT OPTIONS

### Object:

This option allows to define the energy restitution coefficients for the shocks and the impacts.

### Syntax:

```
"CHOC"      coechoc
```

`coechoc`

Energy restitution coefficient for shocks and impacts.

### Comments:

The restitution coefficient is between 0 (plastic shock) and 1 (perfectly elastic shock).

The default value (when the present option is not activated) is 0.5.

## 12.13 OPTIONS FOR FSA/FSR

### Object:

To provide options for fluid-structure interactions of the ALE type for an either deformable (FSA) or rigid (FSR) structure.

### Syntax:

```
< "FSA" "ALF0" alf0 >  
< $[ "NFSC" ; "FSCR" < "INCL" /LEC1/ > < "EXCL" /LEC2/ > ]$ >  
< "FSR" "MFSR" >
```

#### alf0

Maximum angle, in degrees, between two element faces for which a unique normal is computed. If the actual angle exceeds this value, then two distinct normals are generated. By default,  $\text{alf0} = 60$  degrees.

#### NFSC

Do NOT correct geometrically computed normals for the FSA and FSR fluid-structure interaction conditions. This is the default.

#### FSCR

After computing geometrically the normals for the FSA and FSR fluid-structure interaction conditions, apply a correction based on the direction of fictitious internal forces resulting from a uniform pressure field  $p = 1$ . This correction can be useful e.g. in 3D cases when the element faces are warped (non-planar), or when the integration of the element's internal forces is done with an integration rule that does not exactly match the estimation of the normal to the surface computed by purely geometrical considerations from the surface data.

#### INCL /LEC1/

An optional list of nodes to which the FSCR option is applied. By default, the option is applied to all FSA and FSR nodes.

#### EXCL /LEC2/

An optional list of nodes to which the FSCR option is **not** applied. By default, the option is applied to all FSA and FSR nodes.

#### MFSR

Allow a manually rezoned (i.e., moving) node to be declared FSR at the same time. These two conditions are normally incompatible and therefore an error message is normally issued and the code stops. However, there are cases when this is not an error (but only the user can judge on this, it cannot be done automatically). An example is a node on a rigid plane which at the same time must be moved by some manual rezoning to avoid mesh

entanglement. The node can be at the same time manually rezoned (along the plane) and FSR because the link coefficients stay constant even though the node moves. The option deactivates the error message: only one warning message is issued, for the first node concerned. Note that obviously, if this option is specified, it must be inserted in the input file *before* the `LIAI FSR` or the `LINK FSR` directive.

## Remarks

In some special cases it may be useful to exclude some `FSA` or `FSR` nodes from the `FSCR` correction. For example, in the transition zone of a pipeline mesh between a 3D representation and a 1D representation by means of the `TUYM` (deformable structure) or `TUBM` (rigid structure) junction: all fluid nodes in the external circumference of the 3D pipe mesh shall be declared `FSA` or `FSR`, but we want to make sure that no `FSCR` correction is applied to them (while it may be desirable for the other nodes). So we may explicitly exclude them by means of the `EXCL /LEC2/` directive.

## 12.14 OPTIONS FOR NODE-CENTERED FINITE VOLUMES

### Object:

To provide options for node-centered Finite Volumes (multicomponent fluid flows).

### Syntax:

```
< MC <ORDR ordr>  
    <NUFL $[ ROE ; VANL ; STWA ]$ >  
    <WBC>  
    <SYNC sync>  
>
```

#### ORDR

Introduces the order **ordr** of the numerical integration scheme. May be 1 (first order) or 2 (second order). By default, it is taken **ordr** = 2.

#### NUFL

Introduces the type of flux calculation in the bulk fluid; may be **ROE** (Roe flux), **VANL** (Van Leer flux) or **STWA** (Steger-Warming flux). It is only accepted in purely Eulerian calculations. Recall that the far-field flux type is chosen (independently from the bulk flux type) by directive **BDF0** in material **MCFF**. By default, it is taken **NUFL ROE** (Roe flux).

#### WBC

If specified, the boundary conditions are treated according to a weak formulation. It is only accepted in purely Eulerian calculations. In this case external forces at the boundaries are evaluated by imposing zero momentum flux across the solid boundaries, while in the default case (no **WBC** specified) these forces are evaluated by the method of Lagrange multipliers.

#### SYNC

Introduces the type of synchronization **sync** for the MC variables: 0 (the default) is the old procedure; 1 is the new procedure.

### Remarks

The “new” synchronization algorithm (**SYNC 1**), introduced in April 2010, should be used systematically for new calculations. The old algorithm is left only for compatibility with old input files.

## 12.15 OPTIONS FOR MULTIPHASE MULTICOMPONENT FLUIDS

### Object:

To provide options for multiphase multicomponent fluid flows.

### Syntax:

```
<  "FLMP"  < "EPS1"  eps1 > < "EPS2"  eps2 > < "EPS3"  eps3 >
      < "EPS4"  eps4 > < "NIMA"  nima > < "DUMP"  dump >  >

<  $[  "DPLG"      ;
      "VOFIRE" < "VSWP" > < "CORR" > < "RFCR" >
                        < "NOCR" > < "NORC" >
      < "SKIP"  /LECTURE/ >                ]$ >
```

#### eps1

Tolerance for the determination of number of effective components (a component is effectively present if its mass fraction is  $\geq \text{eps1mp}$ ). Default is 1.E-7.

#### eps2

Tolerance for the convergence of Newton-Raphson iterations. Default is 1.E-6.

#### eps3

Relative density variation to determine initial conditions in FLMPPR (case LIQ + GAS). Default is 1.E-5.

#### eps4

Tolerance to find the cut-off density for liquids in FLMPRP. Default is 1.E-12.

#### nima

Max. number of iterations in the above mentioned procedures

#### dump

Dump (1) or do not dump (0) informations on N-R iterations.

#### DPLG

Activates Despres-Lagoutiere anti-dissipative algorithm for multi-component flows on structured mesh (see comment below).

#### VOFIRE

Activates VOFIRE anti-dissipative algorithm for multi-component flows on unstructured mesh (see comment below).

**VSWP**

If present, exact advected volume is computed for each element face. If not, volume is approximated through the sweep formula.

**CORR**

Enables the use of CEA improved version of the VOFIRE algorithm.

**RFCR**

Enables the use of improved algorithm for mixture's density.

**NOCR**

Disables the use of CEA improved version of the VOFIRE algorithm.

**NORC**

Disables the use of improved algorithm for mixture's density.

**SKIP**

Deactivates VOFIRE for the given fluid elements.

**Comments:**

Despres-Lagoutiere anti-dissipative algorithm and its extension to unstructured meshes called VOFIRE are used to prevent numerical spreading of the mixing zone of physically non-miscible components. This is still a development in progress and is only available when multi-component material ADCR or SGMP is used for the fluid in the model. With the SGMP material, VOFIRE is only applicable with two constituents.

Improved algorithms for geometric reconstruction and computation of mixture's density on elements faces are currently disabled by default.



## 12.16 OPTIONS FOR AUTOMATIC REZONING IN ALE COMPUTATIONS

### Object:

To provide options for automatic rezoning algorithms in ALE computations.

### Syntax:

```
< REZO < SPLI |[ GIUL ; MODI ; BOTH ]| >
    < MVRE |[ NONE ; MODU <VFAC vfac>; MOPR <GAMO gam0> ]| >
    < MEAN |[ POSI ; DEPL ]| >
    < DIRE RMAX rmaxrz >
    < NSTE rznste > < CSHE cshear > < CSTR cstret >
    < YOUN rezyo NU reznu RHO rezro >
    $[ VFLU ; LIAI ]$ >
```

#### SPLI

Use the splitting algorithm specified next in order to split up the mesh elements around each node and to form the node's influence domain. The available possibilities are: GIUL for Giuliani's original splitting rule, MODI for the modified rule, or BOTH to use a superposition of both methods. The default value is GIUL. This parameter applies only to Giuliani's (AUTO) rezoning model and to the mean (MEAN) rezoning model. For the former, this parameter applies only to 2D quadrilateral ALE finite elements and finite volumes. For the latter, it applies to all elements (2D and 3D), but with a slightly different meaning: the GIUL option considers as neighbours of the node under consideration only the nodes that are connected to it by a face side; the other two options (MODI or BOTH) are equivalent and consider as neighbours all nodes belonging to neighbour elements.

#### MVRE

Use the mesh velocity restriction algorithm specified next in order to limit the 'raw' optimal mesh rezoning velocity computed by a rezoning algorithm. As shown in the preceding Sections, since all implemented algorithms are explicit, they are unstable unless some limitation is introduced. The available possibilities are: NONE for no restriction (as said, this is likely to be unstable), MODU for the modulus-based rule, or MOPR to use the standard modulus plus projection rule that was adopted in the original Giuliani algorithm. The default value is MOPR. This parameter applies to all rezoning methods described above.

#### VFAC

The velocity factor to be used in conjunction with the MVRE MODU option. By default it is 2.0. This parameter applies to all rezoning methods described above.

#### GAMO

The velocity factor to be used in conjunction with the MVRE MOPR option. By default it is 0.2. The obsolete specification of this parameter in the GRIL directive should be avoided from now on. This parameter applies to all rezoning methods described above.

**MEAN**

Use the mean algorithm variant specified next. The available possibilities are: POSI for an algorithm based on (current) nodal positions, or DEPL for an algorithm based on (current) nodal displacements. The default value is POSI. This parameter applies to all ALE element types.

**RMAX**

The maximum aspect ratio to be used in conjunction with the DIRE rezoning algorithm. By default it is 5.0. Note, however, that this parameter applies only to 2D quadrilateral ALE finite elements and finite volumes.

**NSTE**

The number of steps in which rezoning is applied (repartition parameter). By default it is 1.0. This parameter applies to all rezoning methods described above.

**CSHE**

The shear weight coefficient. By default it is 1.0. Note, however, that this parameter applies only to: a) any elements rezoned by Giuliani's method (AUTO); b) 2D triangles and quadrilaterals rezoned by the SPEC method; c) 2D quadrilaterals rezoned by the QUAD method; d) 2D quadrilaterals rezoned by the MECA method.

**CSTR**

The stretch weight coefficient. By default it is 1.0. Note, however, that this parameter applies only to: a) any elements rezoned by Giuliani's method (AUTO); b) 2D triangles and quadrilaterals rezoned by the SPEC method; c) 2D quadrilaterals rezoned by the QUAD method; d) 2D quadrilaterals rezoned by the MECA method.

**YOUN**

The fictitious material Young's modulus to be used in conjunction with the MECA rezoning algorithm. By default it is 1.0. Note, however, that this parameter applies only to 2D quadrilateral ALE finite elements and finite volumes.

**NU**

The fictitious material Poisson's coefficient to be used in conjunction with the MECA rezoning algorithm,. By default it is 0.0. Note, however, that this parameter applies only to 2D quadrilateral ALE finite elements and finite volumes.

**RHO**

The fictitious material density to be used in conjunction with the MECA rezoning algorithm. By default it is 1.0. Note, however, that this parameter applies only to 2D quadrilateral ALE finite elements and finite volumes.

**VFLU**

Choose the 'old' method of dealing with rezoning of nodes that are subjected to liaisons,. The imposed direction(s) are determined indirectly, from the fluid velocity components. As discussed, in 3D cases this method may be too restrictive and prevent the rezoning algorithm from fulfilling its tasks.

**LIAI**

Choose the 'new' method of dealing with rezoning of nodes that are subjected to liaisons. The imposed direction(s) are determined directly from inspection of the liaison coefficients.

## 12.17 OPTIONS FOR CELL-CENTRED FINITE VOLUMES

### Object:

To provide options for Cell-Centred Finite Volume (VFCC) computations.

### Syntax:

```
< VFCC <DUMP>
    <FCON fcon> <VISC visc>
    <ORDR ordr>
    $ <OTPS otps> ; ERK2 $
    <RECO reco> <GRAD grad>
    <LMAS lmas> <LQDM lqdm> <LENE lene> <LALP lalp>
    <LVEL lvel> <LPRE lpre> <LLAG llag>
    <KMAS kmas> <KQDM kqdm> <KENE kene> <KBAR kbar>
    <RVIT rvit>
    <CENE> <NTIL>
    <MO m0> <VINf vinf>
    <NCFS /LECT/>
    <FLSW flsw>
    <TGRA tgra>
    <PAS0 pas0>
>
```

### VFCC

Introduces the options for Cell-Centred Finite Volume computations.

### DUMP

Dumps out on listing the data structures FACE\_VFCC and SOLUTION\_VFCC (only for debugging).

### fcon

Solver for the calculation of numerical fluxes at interfaces between volumes. One of the following solvers can be chosen (**by default the code uses the HLLC solver**, number 6 in the following list):

1. Rusanov
2. Flux-centred with viscosity (see VISC below)
3. HLLE
4. Exact Riemann for perfect gas
5. Zha-Bilgen (Flux Vector Splitting)
6. HLLC **This is the default.**
7. Dominant Wave-Capturing
8. AUSM+ (Flux Vector Splitting)

9. Zha-Bilgen modified
10. LDFSS-2 (Flux Vector Splitting)
11. AUSM+ Low-Mach
12. AUSM+ -up- Low-Mach
13. HLLC Low-Mach

**visc**

Defines the viscosity for use with the Flux-centered solver (FCO<sub>N</sub> 2). **By default there is no viscosity.**

**ordr**

Order in space. Either first or second order is possible for 2D/3D VFCCs. Note that currently *for 1D-VFCCs only first order is implemented* and this keyword has no effect on any 1D-VFCCs possibly present in a computation. **The default is ORDR 2** which, however, corresponds to real second order in space only if a so-called *reconstruction* (see RECO below for an explanation) is chosen (RECO > 0). Since by default RECO is 0 (see below), **the default scheme** (obtained by specifying neither ORDR nor RECO) **is first-order** in practice. *An old (obsolescent) implementation of first order in space scheme is also available in the code, but is still accepted only for backward compatibility and should not be used in new calculations since it will be removed soon. This is activated by choosing ORDR 1 and no reconstruction (thus RECO is 0). However, be warned that the explicit choice of ORDR 1 is incompatible with use of the FLSW fluid-structure interaction model.* See the comments at the end of this page for examples of use of the ORDR and RECO keywords.

**otps**

Order in time. Only first or second order (Van Leer-Hancock predictor-corrector scheme) is possible for 2D/3D VFCCs. Note that currently *for 1D-VFCCs only first order is implemented* and this keyword has no effect on any 1D-VFCCs possibly present in a computation. **The default is first order in time.** In order to achieve second-order time integration in calculations with materials other than CDEM, use OTPS 2. For calculations with CDEM, use the special keyword ERK2, see below.

**ERK2**

This keyword (in alternative to OTPS), chooses a Runge-Kutta explicit second-order time integration scheme. This is the second-order time integration scheme to be preferably used for calculations with the CDEM material (instead of OTPS 2).

**reco**

Activates the so-called reconstruction of the variables at the inter-volume interfaces starting from the values at the centroids and from the (spatial) gradients at the centroids. Since the spatial gradients are only computed when second-order in space is activated (ORDR 2), reconstruction only makes sense in this case. **The default value is 0 (no reconstruction).** Option RECO 1 stays for Green-Gauss reconstruction of the conservative variables (density, momentum and total energy per unit volume). Option RECO 2 stays for Green-Gauss reconstruction of the primitive variables (density, velocity, internal energy per unit mass, mass fraction). Option RECO 3 is only available for the CDEM or DEMS materials and stays for Green-Gauss reconstruction of the primitive variables, which in this case involves the pressure instead of the internal energy per unit mass.

**grad**

Type of gradient used for the reconstruction.

- grad 1 refers to the Gauss-Green reconstruction (**the default one**). The gradient of a variable  $\phi$  at the center is computed according to the formula

$$\left(\vec{\nabla}\phi\right)_C = \frac{1}{\Omega} \sum_f S_f \phi_f \vec{n}_f$$

where  $\Omega$  is the cell volume,  $S_f$  and  $\vec{n}_f$  are the interface surface and normal.  $\phi_f$  is an estimation of the variable  $\phi$  at the center of the interface and is here obtained by a linear interpolation of the values at the two cell centers sharing this interface. It can be shown that the computed gradient is linear exact when the center of each interface is aligned with the cell centers sharing the interface itself. See for instance [908, formulae (1)-(6)] and details therein.

- grad 2 is based on the Modified Gauss-Green reconstruction of [908] and is linear-exact on irregular meshes, even if the center of each interface is not aligned with the cell centers sharing the interface itself.

The gradient of a variable  $\phi$  is computed according to the formula

$$\left(\vec{\nabla}\phi\right)_C = \frac{1}{\Omega} \sum_f S_f \left( \left(\vec{\nabla}\phi\right)_f \cdot \vec{n}_f \right) \left( \vec{X}_f - \vec{X}_C \right).$$

where  $\left(\vec{X}_f - \vec{X}_C\right)$  is the vector going from the cell center to the interface center.  $\left(\vec{\nabla}\phi\right)_f$  is an estimation of the gradient at the interface center and is here computed by imposing that

$$\left(\vec{\nabla}\phi\right)_f \cdot \left(\vec{X}_{Cf} - \vec{X}_C\right) = \phi_{Cf} - \phi_C$$

$Cf$  being the center of the neighbor sharing the interface, and

$$\left(\vec{\nabla}\phi\right)_f \cdot \vec{t} = \left(\vec{\nabla}\phi\right)_C \cdot \vec{t} \quad \text{if} \quad \vec{t} \cdot \left(\vec{X}_{Cf} - \vec{X}_C\right) = 0.$$

This generates, in each cell, a local implicit linear problem involving a square matrix (2x2 in 2D and 3x3 in 3D).

Before concluding we emphasize that the estimation of  $\left(\vec{\nabla}\phi\right)_f$  in [908] is different (it is the same as here only along the direction  $\left(\vec{X}_{Cf} - \vec{X}_C\right)$ ) and involves an iterative process which requires to store the gradient values at the interfaces and at the cells (see details therein).

**lmas, lqdm, lene, lalp, lvel, lpre, llag**

Limitation for the reconstruction ( $\text{RECO} > 0$ ) of the various quantities: **lmas** for the density, **lqdm** for the momentum, **lene** for the total energy per unit volume, **lalp** for the volume fraction (only for CDEM or DEMS material), **lvel** for the velocity, **lpre** for the pressure, **llag** for the Lagrangian variables, i.e. the mass fractions prior to chemical reaction (only for CDEM material). A limiter typically is a number between 0.0 and 1.0, which multiplies the value of the gradient in order to ensure that the reconstructed values at the interfaces do not violate some conditions. The value of the limiter is automatically computed by the code in each Finite Volume (and typically varies from volume to volume, and also in time). The available types of limiter are: 0 indicates no limitation (limiter

equal to 1.0), 1 indicates a first-order limitation (this corresponds to limiter equal to 0.0 and in practice vanifies the effects of the reconstruction), 2 indicates the limitation of Barth and Jespersen, and 3 indicates the limitation of Dubois. **By default the code assumes the limitation of Dubois.**

Summarizing. For RECO equal 1 or 2 one should specify `lmas`, `lqdm`, `lene` and `llag`. For RECO equal 3 one should specify `lmas`, `lvel`, `lpre` and `llag`.

`kmas`, `kqdm`, `kene`

Parameter for the limitation of Dubois for the density (LMAS 3), for the momentum (LQDM 3) or for the total energy per unit volume (LENE 3). This parameter should be between 0.0 and 1.0. **The default value is 0.5.**

`kbar`

Parameter for the limitation of Barth and Jespersen, all variables (e.g. LMAS 2). The value 0 indicates the standard one (**this is the default**), while 1 indicates a modified one which is more robust for the calculation of shock waves. The value `kbar` 1 produces the strongest possible limitation.

`rvit`

Type of reconstruction of the fluid velocity field at VFCC nodes, starting from the velocity field at the VFCC volume centres. This is used to compute the automatic rezoning (mesh velocity) of ALE VFCC fluid nodes and the motion of Lagrangian VFCC fluid nodes. A value of 0 indicates no reconstruction, 1 (**default**) indicates the **arithmetic mean of the neighboring volumes**, 2 is the mean weighted by the element volumes, 3 is the mean weighted by the element masses, 4 is the mean weighted by the inverse of the element volumes, 5 is the mean weighted according to Roe.

CENE

This option adds a correction of the gradients such that the internal energy is always positive. This affects only second-order in space calculations with RECO 1 or 2, but not 3.

`m0`

Cut-off value for the Mach number for use with low-Mach solvers (i.e. FCON 11, 12 or 13). For the other solvers, it is ignored. **By default it is 0.5.**

`vinf`

Reference velocity for use with low-Mach solvers (i.e. FCON 11, 12 or 13). For the other solvers, it is ignored. **By default it is 0.0.**

NTIL

No “tilt” in the calculation (i.e. suppress error message and subsequent stop if the internal energy becomes negative). **The default is to stop (tilt) if the internal energy becomes negative.** This option has no effect on calculations with the CDEM or DEMS materials.

NCFS

Announces that a (nodally) non-conforming fluid-structure interaction exists between a structure (typically meshed by shell elements) and a fluid meshed by VFCC. The following /LECT/ lists all fluid nodes (which must belong to the VFCC domain) which are located

along the non-conforming F-S interface. The code automatically searches the facing structural element, which must be “superposed” (within a small tolerance) to the fluid volume face (such an element *must* exist, else an error message is issued).

#### flsw

This option allows to choose the type of FLSW algorithm to be used for fluid-structure interaction modeling in conjunction with cell-centred finite volumes. The value 0 means that all numerical fluxes across interfaces near the structure are set to zero, except those related to momentum (which are the pressure forces). **The value 1 is the default** and means that all numerical fluxes across interfaces near the structure are computed by introducing fictitious “ghost” states corresponding to a rigid wall moving with the same speed as the structure.

#### tgra

By specifying **TGRA  $i$**  with  $i > 0$  one activates the non-regression test for the gradient limiter for the perfect gas in the CDEM model: the gradient of the  $i$ -th variable is stored in the ECR table. **By default ( $i = 0$ ) no gradient is stored.**

#### pas0

The initial time step is imposed to be **pas0**. **The default is 0.0**, which means that the code computes it **automatically**.

### Comments:

Below are some examples of use of the **ORDR** and **RECO** keywords. The effect of **RECO**  $> 1$  is similar to **RECO 1**, only the type of reconstruction is different. Recall also that currently the **ORDR** and **OTPS** keywords have no effect on any 1D-VFCC possibly present in a computation, since only first order in space and in time is currently implemented for 1D-VFCCs.

ORDR	RECO	Result
none	none	First-order in space (default)
none	0	First-order in space (default)
none	1	Second-order in space
2	none	First-order in space (default)
2	0	First-order in space (default)
2	1	Second-order in space
1	none	Older version of first-order in space: <b>FLSW not available!</b>
1	0	Older version of first-order in space: <b>FLSW not available!</b>
1	1	<b>Input error: this combination is not available!</b>



## 12.18 OPTIONS FOR CONNECTIONS ("LIAISONS"/LINKS)

### Object:

To provide options for the connections in general, for the LIAISON CONTACT directive and for the pinball impact model (either standard or generalized), see Section D.

### Syntax:

```

< LIAJ >
< CONT |[ CONS ; VARI ]| >
< GLIS < NORM |[ ELEM < ANGM >; NOEU ]| >
    < GAP |[ ELEM ; NOEU ]| > < DUMP >
    < $ ADAP ; NOAD $ > >
< PINS < DUMP > < STAT > < VIDE > < DTPB < CSPB cspb > > < UPDR >
    < EQVL > < EQVD > < EQVF > < NEQV > < ERCE >
    < $[ FACE ; FACI ]$ < FNOR > >
    < CNOR < $[ MIDP ; NCOL < RCEL < $[ MASL ; MAS2 ]$ > > ]$ > >
    < SNOR > < ASN > < NPSF >
    < $[ REB1 ; REB2 ; NORB ]$ >
    < $[ NOGR ; GRID <DGRI> <SORT>
        $[ HGRI hgri ; NMAX nmax ; DPIN dpin ]$
        < PACK ipac > ]$ >
    < ADJA <BODY> <SELF> > >
< GPNS < DUMP > < STAT > < DTPB < CSPB cspb > >
    <SLIM slim> <PHIR phir> <RCEL>
    < $[ REBO ; REB1 ; REB2 ; NORB ]$ > < REBC >
    < $[ NOGR ; GRID <DGRI> <SORT>
        $[ HGRI hgri ; NMAX nmax ; DPIN dpin ]$
    ]$ > >
< LNKS < STAT > < STAD > < DIAG > < DUMP > < VISU > <NOCU>
    < RIGI <DISP> <CMID> > >
< FLS <CUB8 c8> >

```

### LIAJ

This option causes all constraints on velocities to be imposed on the velocity at time  $n+1$ , rather than at time  $n + 3/2$ , which is the default (note that in this notation the current configuration is indicated by  $n + 1$ ). The first form was used for example in PLEXIS-3C. Therefore, this option is mainly useful in order to perform fine grain comparisons between results of PLEXIS-3C and EUROPLEXUS, for debugging purposes.

### CONT

Introduces options related to geometric bilateral restraints (LIAISON CONTACT, see Page D.40 and following ones).

### CONS

Constant coefficients will be used in the LIAI CONT directives of type SPHE, CYLI, CONE and TORE.

## VARI

Variable coefficients will be used in the LIAI CONT directives of type SPHE, CYLI, CONE and TORE. Remember to dimension adequately by the DIME VCON directive.

## GLIS

Introduces options to the LIAI or LINK GLIS (sliding surfaces) model.

## NORM

Options to control the face normal computation.

## ELEM

Exact face normals are computed.

## ANGM

Activate a blind spot detection for shell elements (with NORM ELEM only).

## NOEU

Nodal normals are first computed as weighted mean values from the faces surrounding each node. Face normals are then deduced by averaging the nodal normals at the centre of each face. This is the default method (see comment below).

## DUMP

Dump out on the listing the data structure related to GLIS each time it is created or updated (e.g. in case of adaptivity). This option should be only used for debugging purposes since it will produce huge output. Note that, in order to get the printout of the data structures as they are first created, the option must be set *before* the directive LINK GLIS.

## GAP

Options to control the way of considering the gap for contact between shell structures.

## ELEM

Gap is considered on the master side, which means that the master facet is translated by the gap value in its normal direction before contact detection.

## NOEU

Gap is considered on the slave side, which means that the slave node is translated by the opposite of the gap value in the direction normal to the master facet before contact detection. This is the default method (see comment below).

## ADAP

Combine sliding surfaces (GLIS) with adaptivity. This means that the sliding surfaces data structure is updated whenever the mesh is adaptively refined or un-refined. The calculation becomes more CPU-expensive but the contact is likely to be treated better. This feature is under development and should ultimately become the default, but for the moment we leave it as an option.

## NOAD

Do not combine sliding surfaces (GLIS) with adaptivity. This is currently the default. This means that the sliding surfaces data structure is not updated whenever the mesh is adaptively refined or un-refined. The calculation is less CPU-expensive but the contact is likely to be treated worse, since it “sees” only the *base* elements and nodes, and not the descendent ones in case of adaptivity. As already noted, **ADAP** (see above) should ultimately become the default, but for the moment we set **NOAD** as default in order not to break any older test cases.

#### PINS

Introduces options related to the **LIAI** or **LINK PINB** (pinball contact) model (see page D.480).

#### DUMP

Dumps out extensive pinballs information on the listing. Note that even further pinball-related dumps take place by activating the generic option **OPTI DUMP** in conjunction with the present pinball-specific option. Note also that this option should be activated *before* declaring the pinballs, i.e. before the **LINK PINB** directive, in order to get on the listing a detailed information on pinballs.

#### STAT

Dumps out statistics relative to pinballs on a special file **<basename>.pin**. At each time step are printed: the number of “raw” detected pinball contacts, the number of contacts remaining after the **CNOR** algorithm, the number after the **NCOL** algorithm, the number after the **RCEL** algorithm and the (final) number of contacts after the *a priori* rebound algorithm.

#### VIDE

Visualize all descendent pinballs generated by the hierarchic splitting process. This option should only be used for debugging purposes. When activated, all the descendent pinballs (of the highest level) generated during the splitting process are considered in contact, so that they may be visualized interactively e.g. by the **TRAC PINC** command (see pages A.25 and O.10). This allows to visualize the result of the splitting process. Beware that in complex cases a very large number of such pinballs may be generated. When the option is activated, pinball links are not generated, however, since the retained contacts are unphysical. In addition, the calculation is automatically stopped after time step 0, and the **PINS DUMP** (see above) option is automatically activated.

#### DTPB

Activate automatic limitation of the time increment  $\Delta t$  to account for contacts modelled by pinballs (irrespective of the specific model used, i.e. liaisons, coupled links, or uncoupled penalty). This option is ignored if the user pilots the time increment e.g. by specifying **PAS UTIL**. By default, i.e. without the present option, pinball contacts have no effect on time increments.

#### CSPB

Introduces the reading of the “stability” coefficient **cspb** to be used in conjunction with the **DTPB** option for the limitation of the time increment  $\Delta t$  due to pinballs. By default the code assumes **cspb=cstab** i.e. the same value as the stability coefficient used for the elements’ stability (see **OPTI CSTA** on page H.20). This quantity should be less than 1.0, like for **CSTA**.

## UPDR

Update the radius of parent (0-level) pinballs at every step. By default, the radius is computed only at the initial time. This option may be useful in problems with very large deformations.

## EQVL

The radius of parent pinballs (i.e. at the 0-level) is computed in such a way that the pinball volume equals the initial volume of the associated element. By default, the radius is computed so as to encompass all element nodes in the initial configuration.

## EQVD

Same as EQVL above, but concerning the descendent pinballs generated in hierarchic methods (and this at every level of the hierarchy). The sub-pinball radius is computed in such a way that its volume equals the initial volume of the associated element portion. By default, the radius is computed so as to encompass all element portion “nodes” in the current configuration.

## EQVF

Same as EQVD above, but affects only the proper descendent (i.e. of level  $L > 0$ ) pinballs generated in hierarchic methods **at the last (final) level** of the hierarchy. The parent (0-level) pinballs are not affected. The radius of a **final** proper sub-pinball is computed in such a way that its volume equals the initial volume of the associated element portion. By default, the radius is computed so as to encompass all element portion “nodes” in the current configuration. This option should be preferably used in most cases: the other options (EQVL, EQVD or NEQV) are in fact probably useful only in special cases, or for debugging purposes.

## NEQV

No equivalent volume calculations. The radius of parent pinballs is computed so as to encompass all element nodes in the initial configuration. The radius of any proper descendent pinballs is computed so as to encompass all element portion “nodes” in the current configuration. This is currently the default. It may be used to restore the default behaviour after one of the other options (EQVL, EQVD or NEQV) has been specified.

## FACE

The velocity constraint for a contact between parent pinballs is written at the centroids of the faces crossed by the line joining the pinball centers (it involves only the face nodes). By default, the velocity constraint is written at the pinball centers (which for 0-level pinballs corresponds with the element centroid) and thus involves all the nodes of the element. This option has no effect on contacts between sub-pinballs.

## FACI

The velocity constraint for a contact between parent pinballs is written at the intersections of the faces crossed by the line joining the pinball centers (it involves only the face nodes). By default, the velocity constraint is written at the pinball centers (which for 0-level pinballs corresponds with the element centroid) and thus involves all the nodes of the element. This option has no effect on contacts between sub-pinballs.

## FNOR

The velocity constraint for a contact between parent pinballs is written along a “mean” of the two face normals  $n = (n_A - n_B)$ . Note that this requires that either `OPTI PINS FACE` or `OPTI PINS FACI` be specified as well. By default, the velocity constraint is written along the direction of the line that joins the pinball centers.

#### CNOR

The velocity constraint for a contact between sub-pinballs is written along a “common” normal. One such normal is determined for each couple of contacting element faces. When multiple contacts between sub-pinballs occur (pinballs hierarchy at level  $> 0$ ) in case of flat (face to face) contact, this common normal is an approximation of the normal to the contacting faces.

#### MIDP

The velocity constraint for a contact between sub-pinballs is written at “midpoints” along the lines that join the retained contacting sub-pinballs. This option is part of the common normal algorithm and therefore it requires that the `CNOR` option be specified as well (see above). This option is incompatible with the `NCOL` option described below. Note that this option has effect only on constraints between sub-pinballs that are part of a “sequence” of two or more contacts between the same couple of ancestors. Single or “isolated” contacts between two ancestors (to which the concept of common normal does not apply) are not affected, and in such cases the constraint is written at the sub-pinball centers.

#### NCOL

Collapse onto the nearest node of the parent element the center of those descendent pinballs located at “corners”. In addition, for the remaining (non-corner) descendent pinballs, collapse their center onto the element side or face. Note that the above mentioned collapse is performed only as far as the application point of contact reaction forces is concerned, i.e. when writing down the constraints, and it does not affect the position (center, radius) of the descendent pinball itself. By this option the form of the resulting constraints is simpler because they involve less dofs, and the constraints are more independent from one another. This option has only effect on contacts between sub-pinballs (not for contact between parent pinballs) and is incompatible with the `MIDP` option described above. It requires the `CNOR` option.

#### RCEL

Eliminate repeated constraints for contacts between sub-pinballs that may result after collapse (see option `NCOL` above). This option may help removing a priori from the system repeated constraints that occur e.g. in flat contact between adjacent elements. It requires that the `NCOL` option (and thus also the `CNOR` option) be specified as well. Normally to obtain the maximum benefits a user would specify the three options `CNOR NCOL RCEL`.

#### MASL

Apply master/slave rule in order to further simplify constraints in case of multiple flat contact between bodies. Constraints of type NP (node-to-point) whose associated node belongs to the “hardest” one of the two contacting bodies are rejected. Body “hardness” is specified optionally in the `PINB BODY HARD` directive, see Page D.480. This option requires that `HARD` has actually been specified for both contacting bodies, and that the `RCEL` option described above has been specified as well. The result should be similar to the more traditional sliding lines (slave node / master surface) algorithm, and might lead to slight under-constraining (spurious penetration) in some cases (if this happens, try using the `MAS2` option below instead).

## MAS2

Apply master/slave rule in order to further simplify constraints in case of multiple flat contact between bodies. Multiple constraints of type NP (node-to-point) whose associated node belongs to the “hardest” one of the two contacting bodies are rejected. Body “hardness” is specified optionally in the PINB BODY HARD directive, see Page D.480. This option requires that HARD has actually been specified for both contacting bodies, and that the RCEL option described above has been specified as well. The result should be intermediate between a purely pinballs-based algorithm and the more traditional sliding lines (slave node / master surface) algorithm. It might lead to slight over-constraining (contact locking) in some cases (if this happens, try using the MASL option above instead).

## SNOR

When a *single* contact occurs between sub-pinballs belonging to the same couple of element faces, and only one of the two sub-pinballs is a face sub-pinball, then the used normal is the normal to that face. This option may be used alone or combined with the CNOR option (which acts only upon *multiple* contacts).

## ASN

The so-called “assembled surface normal” (ASN) algorithm of Belytschko and Law (1985) is used to compute a unique (normalized) normal to each external node of the mesh portion subjected to contact, and a unique (normalized) normal to each pinball (parent or descendent). The penetration direction between contacting pinballs is then computed using the ASNs of the two pinballs according to a set of rules. This ameliorates the treatment of flat contact, especially in conjunction with a penalty formulation to compute the contact forces. This option cannot be used together with (is an alternative to) options FNOR, CNOR (and its sub-options), or SNOR.

## NPSF

Add a scaling force factor for pseudo-nodal pinballs at an adaptivity level  $L > 1$ . If this option is specified, then the penalty force for newly created pseudo-nodal pinballs (i.e., pinballs associated with to a mass-less PMAT element attached at element nodes) are scaled down by multiplying the penalty force normally computed by a factor:

$$\phi = 1/(2^{L-1}) \quad (52)$$

where  $L$  is the hierarchy level in adaptivity of the node to which the pseudo-nodal pinball is attached (via the PMAT element). If the keyword is not specified, then  $\phi = 1.0$  and the penalty force is not scaled. If specified, the option has effect only on pseudo-nodal pinballs using the PENA (penalty) method, it has no effect on element-based pinballs nor on pseudo-nodal pinballs using the Lagrange multipliers method.

## REB1

The so-called *a priori* pinball contact rebound detection algorithm is used. This is the default contact rebound detection algorithm and therefore specifying this keyword is usually redundant. Rebound-related options are only used in the Lagrange Multipliers version of the pinball method. The penalty formulation does not use any special rebound treatment, so these options are ignored.

## REB2

The so-called *a posteriori* pinball contact rebound detection algorithm is used instead of the default *a priori* contact rebound detection algorithm. This option is only intended for internal code testing and verification, because the default algorithm is normally superior to the other one. Rebound-related options are only used in the Lagrange Multipliers version of the pinball method. The penalty formulation does not use any special rebound treatment, so these options are ignored.

**NORB**

Do not apply any pinball contact rebound detection algorithm. Rebound between pinballs is not treated. Rebound-related options are only used in the Lagrange Multipliers version of the pinball method. The penalty formulation does not use any special rebound treatment, so these options are ignored. This option is therefore useful only (for debugging purposes) with pinball contacts treated by Lagrange multipliers (see **LINK COUP PINB**).

**NOGR**

Do not use a grid of cells to speed up search of neighbours for contact detection. This is the default.

**GRID**

Use a grid of cells (as in bucket sorting) to speed up search of neighbours for contact detection. The grid encompasses all elements containing parent pinballs and is built up either automatically (if no further options are specified) in the way specified below, or according to one of the following criteria.

**DGRI**

Dump out the grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed.

**SORT**

Sort the list of contacts in growing order so they (should) become like in the case without grid. This option is only to be used for debugging, since it facilitates the comparison of results with and without grid.

**HGRI**

Specifies the size of the grid cell. Each cell has the same size in all spatial directions and is aligned with the global axes.

**NMAX**

Specifies the maximum number of cells along one of the global axes.

**DPIN**

Specifies the size of the grid cell as a multiple of the diameter of the largest parent pinball. For example, by setting **DPIN 4** the size of the cell is four times the diameter of the largest parent pinball. By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DPIN** are specified, the code takes **DPIN 1.1**. Normally, the cost of searching decreases as one takes smaller values of **DPIN**. However, the memory used tends to increase because there will be more cells. In large cases, a trade-off must be found but it is difficult to say a priori what is the optimal value for **DPIN**. Note that values of **DPIN** at or below 1.0 are **unsafe**. Some contacts may be overlooked (but this depends on the case). To be sure that **all** contacts are detected, use **DPIN** (slightly) larger than 1.0, say 1.001.

**PACK**

This optional keyword allows to specify a packing size **ipac** for the fast search grid. The search is then done by partially overlapping cubic (square in 2D) “macro cells” each containing **ipac** search cells along each spatial direction. See below for comments.

**ADJA**

This optional keyword introduces some options relative to the treatment of “adjacent” elements in the pinball contact method. Two elements are considered adjacent if they share at least one common node. Normally, contact between adjacent elements is *not* activated, since the pinballs associated with an element may slightly protrude from the element itself and therefore interpenetrate with the pinballs associated with the adjacent elements.

**SELF**

By specifying the **ADJA SELF** keyword, self contact between adjacent elements is activated.

**BODY**

By specifying the **ADJA BODY** keyword, contact between adjacent elements belonging to two different bodies is activated. This is the case, for example, of a zone of elements being assigned *both* element-based (possibly hierarchic) pinballs *and* nodal-based (via **PMAT**) pinballs, declared as two different **BODY** directives. This is an alternative way of achieving self-contact (to the use of only one set of **SELF** element-based pinballs) which might be advantageous for example in the case of a very thin shell impacting a rigid obstacle and forming very narrow plies, each formed by only one row of elements.

**GPNS**

Introduces options related to the **GPIN** (generalized pinball contact) model (see page D.490).

**DUMP**

Dumps out extensive GPINs information on the listing. Note that even further GPIN-related dumps take place by activating the generic option **OPTI DUMP** in conjunction with the present GPIN-specific option.

**STAT**

Dumps out statistics relative to GPINs on a special file **<basename>.gpn**. At each time step are printed: the number of “raw” detected GPIN contacts, the (final) number of contacts after the *a priori* rebound algorithm, the maximum penetration at the current step, the maximum penetration so far, the maximum penetration rate at the current step and the maximum penetration rate so far.

**DTPB**

Activate automatic limitation of the time increment  $\Delta t$  to account for contacts modelled by GPINs (irrespective of the specific model used, i.e. coupled links, or uncoupled penalty). This option is ignored if the user pilots the time increment e.g. by specifying **PAS UTIL**. By default, i.e. without the present option, GPIN contacts have no effect on time increments.

**CSPB**



Introduces the reading of the “stability” coefficient `cspb` to be used in conjunction with the DTPB option for the limitation of the time increment  $\Delta t$  due to GPINs. By default the code assumes `cspb=cstab` i.e. the same value as the stability coefficient used for the elements’ stability (see OPTI CSTA on page H.20). This quantity should be less than 1.0, like for CSTA.

#### SLIM

Limit value  $s_{\text{lim}}$  of the scalar product between two normals:  $s = \hat{\mathbf{n}}_1 \cdot \hat{\mathbf{n}}_2$  beyond which a single-sided structural GPIN should be treated as a two-sided GPIN. By default it is  $s_{\text{lim}} = -\sqrt{2}/2$ . See the report on generalized pinballs (GPIN) for more information.

#### PHIR

Value of the coefficient  $\phi$  which multiplied by the GPIN radius defines the validity zone for penetration into an L-GPIN. It should be  $0 \leq \phi \leq 1$ . By default the code currently assumes  $\phi = 0$  in order not to modify benchmarks evolved before the introduction of this parameter. However, in the future the default could be set to a more realistic value, e.g.  $\phi = 0.7$ . By setting PHIR 0 the entire length of the L-GPIN is valid for penetration, but this may produce P-L contacts instead of P-P contacts, and L-L contacts instead of P-L contacts (the effects on engineering results of such approximations should be small anyway). See the report on generalized pinballs (GPIN) for more information.

#### RCEL

Activate redundant constraints elimination. Unlike in the case of PINB, with GPIN this is done during the search of (raw) penetrations itself. For the moment, it is only implemented in 2D.

#### REB0

The so-called simplified *a priori* GPIN contact rebound detection algorithm is used. Rebound-related options are only used in the Lagrange Multipliers version of the generalized pinball method. The penalty formulation does not use any special rebound treatment, so these options are ignored. In this version of the *a priori* rebound algorithm the penetration rate is evaluated simply based upon the current velocities.

#### REB1

This is the default contact rebound detection algorithm for generalized pinballs and therefore specifying this keyword is usually redundant. This is a more elaborate version of the *a priori* GPIN rebound algorithm where the penetration rate is evaluated based upon the estimated “free” position of the GPINs at the next time step, in the absence of contact forces. It uses the accelerations (i.e. the forces and the masses) in addition to the velocities, similarly to what is done by default for standard pinballs (PINB). Rebound-related options are only used in the Lagrange Multipliers version of the generalized pinball method. The penalty formulation does not use any special rebound treatment, so these options are ignored.

#### REB2

The so-called *a posteriori* GPIN contact rebound detection algorithm is used instead of the default *a priori* contact rebound detection algorithm. This option is only intended for internal code testing and verification, because the default algorithm is normally superior to the other one. Rebound-related options are only used in the Lagrange Multipliers

version of the pinball method. The penalty formulation does not use any special rebound treatment, so these options are ignored.

**NORB**

Do not apply any GPIN contact rebound detection algorithm. Rebound between pinballs is not treated. Rebound-related options are only used in the Lagrange Multipliers version of the pinball method. The penalty formulation does not use any special rebound treatment, so these options are ignored. This option is therefore useful only (for debugging purposes) with pinball contacts treated by Lagrange multipliers (see **LINK COUP GPIN**).

**REBC**

This option activates a *combined pre-post* form of a priori rebound. In other words, rebound is checked both a priori (before solving the constraints) and a posteriori (i.e. by checking the sign of the Lagrange multiplier in the algorithm that treats the friction). The option affects only the two types of a priori rebound described above, i.e. either the simplified a priori rebound (**REB0**) or the full a priori rebound (**REB1**).

**NOGR**

Do not use a grid of cells to speed up search of neighbours for contact detection. This is the default.

**GRID**

Use a grid of cells (as in bucket sorting) to speed up search of neighbours for contact detection. The grid encompasses all GPINs and is built up either automatically (if no further options are specified) in the way specified below, or according to one of the following criteria.

**DGRI**

Dump out the grid of cells used for fast searching on the listing. For brevity, the print is done only the first time that the grid is computed.

**SORT**

Sort the list of contacts in growing order so they (should) become like in the case without grid. This option is only to be used for debugging, since it facilitates the comparison of results with and without grid.

**HGRI**

Specifies the size of the grid cell. Each cell has the same size in all spatial directions and is aligned with the global axes.

**NMAX**

Specifies the maximum number of cells along one of the global axes.

**DPIN**

Specifies the size of the grid cell as a multiple of the diameter of the largest GPIN. For example, by setting **DPIN 4** the size of the cell is four times the diameter of the largest GPIN. By default, i.e. if neither **HGRI**, nor **NMAX**, nor **DPIN** are specified, the code takes **DPIN 1.1**. Normally, the cost of searching decreases as one takes smaller values of **DPIN**.

However, the memory used tends to increase because there will be more cells. In large cases, a trade-off must be found but it is difficult to say a priori what is the optimal value for DPIN. Note that values of DPIN at or below 1.0 are **unsafe**. Some contacts may be overlooked (but this depends on the case). To be sure that **all** contacts are detected, use DPIN (slightly) larger than 1.0, say 1.001.

#### LNKS

This keyword introduces options which are specific of the links model. They are ignored by the “liaisons” model.

#### STAT

Dumps out statistics relative to *coupled* links (LINK COUP) on a special file <basename>.lks. At each time step are printed: the number of link groups (N\_GPS), the total number of links (N\_LKS), the total number of permanent links (N\_PLKS), the total number of non-permanent links (N\_NPLKS) and finally the number of links of each type (e.g., BLOQ, RELA etc.).

#### STAD

Print similar information to the statistics for coupled links, but relative to the decoupled links (LINK DECO), on file <basename>.lkd. Attention: the programming of this feature is under development. At the moment, statistics is available only for the following types of decoupled links: PINB.

#### DIAG

Dumps out additional diagnostics relative to current links (both permanent and non-permanent) on the listing, together with each normal printout (see directive ECRI. The information concerns the size of the links matrix, and its “fullness” (i.e. the relative number of non-zero entries). This information can be useful in view of the choice of the most appropriate solution strategy for the links problem.

#### DUMP

Dumps out all current links (both permanent and non-permanent) on the listing, together with each normal printout (see directive ECRI. The generated output can be huge, therefore this option should be used with great care (and for debugging purposes only).

#### VISU

Activate the possibility of visualizing the links in the built-in OpenGL graphical module.

#### NOCU

Do *not* check for unconnected nodes presence in links (coupled or decoupled). Unconnected nodes are nodes not belonging to any element. Having such nodes in links may be dangerous, especially in MPI calculations. By default, the code checks for the presence of such nodes and stops with an error message if any are found. In some particular cases (old models) such nodes are present but this is not harmful. In such cases, one may use this option (under his responsibility) to disable the check and to avoid the error (the error is converted into a warning). Note that, in order to be effective, the NOCU option **must be placed before** the LINK directive.

#### RIGI

Introduce options related to the treatment of links for rigid bodies (new JRC formulation).

**DISP**

Express the links for rigid bodies on the displacements rather than on the velocities.

**CMID**

Compute the links coefficients for rigid bodies at the new mid-step rather than at the current full step.

**FLS**

This keyword introduces options which are specific of the **FLSR** and **FLSW** fluid-structure interaction models.

**CUB8 c8**

Sets the error level for inverse mapping in 8-node cube elements. By default it is 0, meaning that any error is treated as a real error. By setting it to 1, a lack of convergence in the inverse mapping procedure is not considered an error, but simply that the point considered lies outside the 8-noded cube. By setting it to 2, both a lack of convergence and a zero determinant are considered not as errors, but as an indication that the point considered lies outside the 8-noded cube. These options should be set only in problematic cases (and until the inverse mapping for the CUB8 shape is reformulated in a more robust way). Note that this option has effects only on the **FLSR** and **FLSW** models, in the tracking of flying debris embedded in a fluid, and in contact by pinballs, but not on other model which use CUB8 inverse mapping. Note also that this option has the same effect **also** upon inverse mapping in **3-node triangles in 3D**, but with the following meaning of the **c8** parameter: 1 means that  $d_{\max} < \text{tol\_vol}$  is not considered as an error, while 2 means the above plus also  $\text{abs}(\text{err}) > \text{tol\_dis}$  is not an error. Finally, this option has the same effect **also** upon inverse mapping in **4-node quadrilaterals in 3D** (which may be warped), but with the following meaning of the **c8** parameter: 1 means that performing too many iterations is not considered as an error, 2 means the above plus also that a 0 determinant is not an error, and 3 means all the above plus also the fact that the point does not lie on the quadrilateral surface is not considered an error. If you activate this optional switch, it is probably safer to use the value 2 (or 3, in the case of 3D quadrilaterals) anyway.

**ERCE**

Instead of an error message for not possible inverse mapping to get the centroid shape, the concerned element is eroded. This might be helpful to continue calculations with highly distorted elements and pinballs.

**Comments:**

Be sure to consult also the interactive commands for the visualization of pinballs and of contacts, see Pages A.25 and O.10.

As far as nodal or elementary methods are concerned to compute the facet normals for contact detection and links generation, both may present advantages and drawbacks in different situations. Nodal approach produces smoother variations of the normal along the master side and may be useful for problems such as rolling bodies. However, in the case of strongly

folded structures (for example, self-contact crashed bodies), elementary approach ensures better detection of contact between folds and should be preferred.

Considering the gap on slave side is the original way that was implemented in EUROPLEXUS. It has shown recently to potentially produce instabilities for strongly folded structures. In this case, considering the gap on master side has proved to be much more robust. The former approach remains the default until the latter is fully tested and validated.

When a fast search by cells grid is specified for the macro pinballs (or for the GPINs) in contact (`PINS|GPNS GRID . . .`) and a large 3D problem is being solved with relatively few (but largely scattered) contacts, then one may easily generate an enormous number of cells and the memory required becomes prohibitive. In such cases, it may be convenient to do the search not as a unique scan but by several scans over contiguous “packs” (i.e. rectangular patches) of cells. Each pack or “macro cell” contains a number `ipac` of cells along each spatial direction. In addition, an extra cell is added along each boundary since the packs must be partially superposed for the algorithm to work. Thus in 2D the size of a pack will be  $(\text{ipac}+2 * \text{ipac}+2)$  and in 3D  $(\text{ipac}+2 * \text{ipac}+2 * \text{ipac}+2)$  cells. The search by packs is slightly slower than global search because of the increased number of operations and of the more complex algorithm, but the used memory might be much smaller (the user may reduce it by using a lower `ipac`).

## 12.19 OPTIONS FOR GRAPHICAL RENDERING

### Object:

To provide options for the graphical rendering (OpenGL).

### Syntax:

```
< REND < $[ FAST ; SAFE ]$ >
    < $[ NODU ; DUMP ]$ >
    < $[ NAVI ; NONA ]$ >
    <STAT>
    <FAC4 SPLI n TOLE eps>
    <SHAR <ANGL angl> <ABS>> >
```

#### REND

This keyword introduces the options related to rendering.

#### FAST

This option uses the fastest available algorithms for the in-software geometric calculations preliminary to geometric rendering operations (see **TRAC REND**). This is the default.

#### SAFE

This option uses straightforward (but inefficient) algorithms for the in-software geometric calculations preliminary to geometric rendering operations (see **TRAC REND**). It may be useful when one has doubts on the graphical results obtained with the fast version.

#### NODU

This option does not dump out data related to the in-software geometric calculations preliminary to geometric rendering operations (see **TRAC REND**). This is the default.

#### DUMP

This option dumps out on the listing data related to the in-software geometric calculations preliminary to geometric rendering operations (see **TRAC REND**). This may be useful for debugging purposes but it produces a big output file.

#### NAVI

This option declares that any changes in the following rendering operations will be due only to navigation (**NAVI**) around or inside a fixed (static) scene, so that use of **SAVE/REUS** becomes possible also in Lagrangian cases, see page O.0030. In this case the user is responsible for making sure that no geometrical data vary between a rendering and the next one(s): the mesh does not move, no elements are eroded, adaptivity does not modify the current mesh, etc. The option is useful in order to speed up preparation of an animation containing a navigation in a static scene containing Lagrangian nodes.

#### NONA

Disables the NAVI option set with a previous NAVI keyword so that the normal behaviour of the SAVE/REUS mechanism is restored, see page O.0030.

**STAT**

This option produces statistics on the allocations performed by the OpenGL graphics module on a special file <basename>.ogl. This is useful only for debugging purposes.

**FAC4**

This option introduces indications about how to render 4-node faces. By default each 4-node face is split into four triangles by generating an extra point at the face center. In this way the rendering of non-planar (warped) 4-node faces is best and does not depend upon face (or element) numbering. Also the representation of iso-values is best. However, a lot of memory is required. Memory can be saved, at the expense of a somewhat worse representation (and not completely numbering-independent), by treating them as a single quadrilateral.

**SPLI n**

The number of figures into which an almost-planar 4-node face is split. By default it is 4. It may be optionally set to 1. The value 2 was initially also foreseen, but it has still to be programmed and is not currently available.

**TOLE eps**

Tolerance  $\epsilon$  to decide whether a 4-node face is planar or not. The face is considered planar if the scalar product between the two unit normals to triangles 1-2-3 and 1-3-4 obtained from the face is greater than  $(1 - \epsilon)$ .

**SHAR**

Introduces options related to the visualization of sharp corners.

**ANGL**

Sets the minimum angle  $\alpha_0$  (between two 3D faces with a common side) beyond which the side is considered to be a sharp corner. By default, this angle is 60 degrees. Let  $n_1$  and  $n_2$  be unit normals to the two faces. Then the scalar product  $n_1 \cdot n_2 = \cos \alpha$  is equal to the cosine of  $\alpha$ , the angle between the normals (which is also the angle between the faces). Thus the corner is sharp if  $\cos \alpha < \cos 60^\circ$ , i.e. when  $\alpha < 60^\circ$ .

**ABS**

Consider the absolute value of the above scalar product instead of the signed value. This has the following effect: when two faces have a common side and opposite (or nearly opposite) normals, the side is *not* considered sharp (while by default it would be). This option may be useful in the presence of complex 3D shell structures, because it is not always easy (and sometimes even impossible) to orient them consistently. With this option many “spurious” sharp corners disappear. Thus with this option the rule becomes: the corner is sharp when  $|\alpha| < 60^\circ$ .

## 12.20 OPTIONS FOR MESH-ADAPTIVE COMPUTATIONS

### Object:

To provide options for mesh-adaptive computations.

### Syntax:

```
< ADAP < $[ NODU ; DUMP ]$ > <STAT> <CHEC> <RCON> <MAXL maxl> <NOPP>
    <RESE>
    < $[ PHAN CD cd <CV cv> ; DHAN < $[ DEPL ; VITE ]$ > ; WHAN ]$ >
    <PCLD $[ MODE imod ]$ $[ SMOO ]$ <DUMP> >
    <TRIG |[ CONT icon ; ECRO iecr ; EPST ieps ;
          DEPL idep ; VITE ivit ; ACCE iacc ; VCVI ivcv ]|
          TVAL tval /LECT/>
>
```

#### ADAP

This keyword introduces the options related to mesh-adaptive computations.

#### NODU

This option does not dump out data related to the mesh-adaptive computations. This is the default.

#### DUMP

This option dumps out on the listing data related to the mesh-adaptive computations. This may be useful for debugging purposes but it produces a big output file.

#### STAT

This option prints out on the listing some additional “statistical” data related to the mesh-adaptive computations. The increment in listing size is very small, but the calculation of these data requires some (small) computational effort, therefore they are not computed by default.

#### CHEC

This option performs some extra checks during mesh-adaptive computations. The CPU overhead is high, so the option should be used only for debugging. The checks are mainly of geometrical nature: consistency of neighbors and pseudo-neighbors, consistency of CCFV interfaces, etc. In case an inconsistency is detected, an extensive printout (dump) of the concerned data structure is made on the listing (which can become very big) and the code stops with an informative error message.

#### RCON

This option imposes a smooth refinement of the mesh, such that the difference in refinement level between two neighboring (or pseudo-neighboring) elements is at most 1.

#### MAXL



This option introduces an upper limit `max1` to the level of refinement of the adaptive mesh, for those types of adaptivity that do not allow to specify the maximum refinement level in their own syntax, e.g. `ADAP INDI` (see Page B.210). For the other types of adaptivity (e.g. `ADAP PCLD`), the maximum refinement level `MAXL` is specified directly in the corresponding `ADAP` directive, so the present option is still accepted with a warning message but it is ignored (it has no effect).

**NOPP**

Do not propagate `MAXCURV` and `ERRIND` to descendents upon elements split (only for debugging). By default they are propagated.

**RESE**

Upon un-splitting of a Q41L or Q42L element with a solid material (VM23 with linear elastic characteristics), recompute `SIG` and `ECR` from parent element nodal positions instead of doing averaging on child elements.

**PHAN**

Use penalty (decoupled) constraints on hanging nodes rather than Lagrange multipliers (fully coupled).

**CD cd**

Penalty coefficient on displacements.

**CV cv**

Penalty coefficient on velocities. This is zero by default.

**DHAN**

Use decoupled Lagrange-multiplier constraints on hanging nodes rather than fully coupled Lagrange multipliers.

**DEPL**

The decoupled Lagrange-multiplier constraints on hanging nodes are expressed on displacements. This is the default.

**VITE**

The decoupled Lagrange-multiplier constraints on hanging nodes are expressed on velocities rather than on displacements.

**WHAN**

Use “weak” decoupled constraints on hanging nodes rather than fully coupled Lagrange multipliers.

**imod**

Mode for mesh refinement using `PCLD` indicators. 1: refinement is homogeneous within one base cell (faster mesh adaptation, more cells, this is default), 2: refinement is heterogeneous with one base cell (slower mesh adaptation, fewer cells)

**SMOO**

Activates a smoothing step after mesh adaptation through PCLD criteria to avoid jumps of refinement levels between neighbor cells (option close to ADAP RCON option above for PCLD).

**DUMP**

Dumps out on the listing the PCLD internal data, but only in sequential calculations (it is ignored in MPI calculations). This option should only be used on small tests, for debugging purposes, since it will produce a huge output on the listing.

**TRIG**

Introduces a “trigger” which activates any forms of “automatic” mesh adaptivity present in the calculation only when a certain variable reaches a given value at a given location. The trigger affects following types of adaptivity models: WAVE, INDI, PCLD, THRS and FLSR/FLSW. The trigger has no effect on initial mesh adaptivity (INIT ADAP) or manually piloted adaptivity (ADAP SPLI/USPL interactive commands).

**CONT icon**

Set the trigger on stress component **icon**.

**ECRO iecr**

Set the trigger on hardening component **iecr**.

**EPST ieps**

Set the trigger on total strain component **ieps**.

**DEPL idep**

Set the trigger on displacement component **idep**. If one specifies 0 for **idep**, then the displacement norm of the first IDIM components is used.

**VITE ivit**

Set the trigger on velocity component **ivit**. If one specifies 0 for **ivit**, then the velocity norm of the first IDIM components is used.

**ACCE iacc**

Set the trigger on acceleration component **iacc**. If one specifies 0 for **iacc**, then the acceleration norm of the first IDIM components is used.

**VCVI ivcv**

Set the trigger on cell-centered velocity component **ivcv**. If one specifies 0 for **ivcv**, then the cell-centered velocity norm of the first IDIM components is used.

**TVAL tval**

Set the value which activates the trigger. The trigger is activated when the value of the monitored quantity exceeds **tval**. Once activated, the trigger remains active for the rest of the computation.

**/LECT/**

Specify the (single) element or the (single) node at which the specified variable is monitored.

## 12.21 STRAIN RATE FILTERING OPTION

### Object:

The strain rate filtering option allows to damp high frequency vibrations wich are not physical and therefore to obtain more physical strain rate values.

### Syntax:

"FVIT"            alpha

alpha

filter coefficient, must be of the order of the smallest element size.

### Comments:

This option is still under development and testing and should therefore be used with great care. this option is available only for isotropic Von Mises material depending on strain rate (VMIS DYNA).

The default value when the present option is not activated is 1. (no filtering).

## 12.22 OPTIONS FOR PARALLEL COMPUTING

### Object:

This section provides options for advanced parallel computing. This is still a work in progress and may be significantly modified in the future.

### Syntax:

```
"DOMD" < "MANU" /CTIM/ >  
      < "ADAP" >
```

#### MANU

Keyword used to enter a manual frequency for domain decomposition update

#### ADAP

Keyword used to trigger a domain decomposition each time the elements extension zones are exhausted (requires ADAP PCLD refinement indicators with OPTION ADAP PCLD MODE 2)

### Comments:

Using DOMD keyword toggles the update of the domain decomposition (MPI only) using either a given frequency (MANU keyword) or the exhaustion of elements zones (ADAP keyword). It allows to take into account strong changes in the topology of the models (large displacements, failure and fragmentation or mesh adaptation for instance), making a static domain decomposition less and less efficient as the simulation progresses. Using DOMD ADAP allows avoiding to overdimension the extension zones (see ADAP directive in group A), for significantly improved computation and memory performance.

## 12.23 OPTIONS FOR GRADIENT DAMAGE MODELS

### Object:

This section describes various numerical parameters for the gradient damage models **ENGR**, see [7.7.21](#). In particular, several options could be provided here for the parallel linear algebra library PETSc used to solve the structural scale damage evolution equation as a bound-constrained minimization problem.

### Syntax:

```
< "ENGR"  < "MONI" >
           < "DEBG" >
           < "PROJ" >
           < "SAIJ" >
           < "PREC" >
           < "INIT" >
           < "EDOT" >
>
```

#### MONI

Activate the PETSc monitor which allows the user to obtain setting and convergence information of the specified solver via an additional log file `*_petsc.log`. On the top of this file are summarized the global Hessian matrix information (number of rows, of non-zeros, etc.) and the solver setting (minimization method, tolerances, underlying linear solver, underlying preconditioner used, etc.). Then the log file prints at every time step following information: **STEP**, the current time step, **ITER**, number of CG iterations, **FVAL**, value of the objective functional (quadratic function), **RNOM**, norm of the residual vector, and **REASON**, the convergence information. At the end of this file some profiling information is given through PETSc's `log_summary` command.

#### DEBG

This option provides various debugging information concerning for example nodes partitioning with PETSc convention.

#### PROJ

By default we prescribe the use of **GPCG** solver for such constrained minimization problems. It performs several gradient projections to identify the active (constrained by the bounds) nodes, and several subsequent conjugate gradient iterations to solve a reduced unconstrained minimization problem for all free (non-active) nodes. *This method is extremely efficient.*

However for comparison we also provide this option **PROJ** to use instead the conjugate gradient method for the unconstrained problem and then an *a posteriori* projection on the admissible space to satisfy irreversibility condition. Note however, that this method **PROJ** makes sense only when the damage constitutive law **AT** chosen by specifying **LAW 2** is used.

**SAIJ**

When this option is used, only the upper triangular portion of the Hessian matrix is stored by the classical CSR format in PETSc. The memory use is reduced, however in terms of computational efficiency/cost nothing is gained through comparison with the full storage format.

**PREC**

This option sets the tolerance norm type of the underlying CG linear solver to be **PRECONDITIONED**, *i.e.* using the inner product defined by the preconditioner matrix. This options has virtually no influence on the computational efficiency through tests.

**INIT**

This option is concerned with the initial condition of damage to model for example an initial crack along some given nodes. When the option **INIT** is activated, all neighboring nodes of the previous ones are also prescribed by the damage value. In case of an initial crack, all the nodes of an element along this crack are thus totally damaged. (Tensile-type) wave propagation is hence prohibited across the crack.

**EDOT**

This option activates strain-rate effects in the damage criterion.

## 13 GROUP I—TRANSIENT CALCULATION DEFINITION

### Object:

The following directives define, run, verify (qualify) and stop the transient computation which has been defined with all directives given so far.

Furthermore, by means of a so-called “ED1D input deck” it is possible to perform a coupled 1-D/multi-D calculation, see also pages INT.80 and I.23.

### Syntax :

```
< "STRUCTURES" . . . >

< "INTERFACES" . . . >

< "XFEM" ... >

"CALCUL" . . .

< "ED1D" {Eurdyn-1D input deck} "ED1D END" >

< "PLAY" {interactive commands} "ENDPLAY" >

< "QUALIFICATION" . . . >

$ "SUITE" ; "FIN" $
```

These instructions are described in detail on the following pages.

## 13.1 STRUCTURES

### Object :

This directive enables the use of the domain decomposition method that has been recently implemented in EUROPLEXUS. Therefore, only some of the elements are currently available, see the list below.

This directive is optional. However, when used it **MUST APPEAR BEFORE** the “CALCUL” directive.

### Syntax :

```
"STRUCTURE" <"DTUN">
  |[ "AUTO" <"PMET"> <"ROB"> <"CINI"> ...
    ... <"WFIL" <ndwfil>> <"DACT" /LECDDL/> <"DPRE" ipre>
    ... <"REGU"> <"CART"> ;

    nbdo * (
      |[ "DOMA" /LECT/ <"IDEN" ndom> <["$DTMX" dtmx ; "DTFX" dtfx]$> ;
        "MODA" /LECT/ <"IDEN" ndom> ...
          ... "FICHIER_VIBRATIONS" <FORMAT> <ndfich> ...
          ... <"POST_TRAITEMENT" $["TOUS" ; "CHPO"]$ > ...
          ... <"NOFO"> ]|
      ) ]|
```

#### DTUN

Multiple time scales treatment (one per subdomain) is deactivated. Every subdomain has the same time scale (see comment below).

#### AUTO

MPI only. Automatic domain decomposition using available number of threads.

#### PMET

MPI only. ParMetis library is used to perform domain decomposition.

#### ROB

MPI only. Recursive Orthogonal Bisection algorithm is used to perform domain decomposition (see comment below).

#### CINI

MPI only. Automatic domain decomposition with ROB after a restart is performed using initial coordinates instead of current coordinates.

#### WFIL



MPI only. Use of an element weight file for automatic domain decomposition (see comment below).

**ndwfil**

MPI only. Number of the logical unit of the weight file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is .wgt.

**DACT**

MPI only. Selection of active directions (from 1 to 2 in 2D, from 1 to 3 in 3D) for automatic domain decomposition using ROB.

**ipre**

Number of the first cutting direction for automatic domain decomposition using ROB (see comment below).

**REGU**

MPI only. Activates a regularizing step for ROB domain decomposition to avoid quasi-orphans (i.e. elements sharing a majority of their faces with elements from another subdomain). The purpose is to optimize interfaces and to provide robustness for geometric operations associated with mesh adaptivity.

**NOOP**

MPI only. Activates an optimization step improving the splitting of elements located on a plane orthogonal to the actual cutting direction during ROB decomposition. Concerned grid cells are read through the **LECTURE** directive.

**nbdo**

Number of subdomains for fixed domain decomposition.

**/LECTURE/**

Indexes of the elements forming the current subdomain. Note that these must include also the indexes of the CLxx elements used to represent any *non-matching interfaces* belonging to the current subdomain (see directive **INTERFACE** below).

**DOMA**

This keyword introduces the definition of a subdomain.

**MODA**

This keyword introduces the definition of a subdomain represented by a modal basis.

**ndom**

Number to identify the subdomain when declaring the interfaces. If omitted, this number is the rank of the subdomain in the order of declaration of all the subdomains (including modal ones).

**dtmx**

Maximum time cycle imposed for the current subdomain (see comment below).

**dtfx**

Fixed (constant) time cycle imposed for the current subdomain (see comment below). This keyword is incompatible with the **dtmx** described above, and may be used only if the step is also user-driven (**OPTI PAS UTIL**). The given value **dtfx** should be an integer sub-multiple of the user-specified time step (typically **pasf**, see the **CALC** directive).

**ndfich**

Number of the logical unit of the file containing the modes and reduced matrices, or file name in quotes. If omitted, the program chooses a file name and unit number by default (see page A.27). The default extension is **.MSH** and the default unit number is 9, so that by default the modes are read from the same file that contains the CASTEM 2000 mesh (file **<base\_name>.msh**).

**FORMAT**

If this keyword is present, the file is formatted, otherwise it is unformatted.

**TOUS**

Keyword meaning that all CHAMELEMS and CHAMPOINTS will be calculated within the modal subdomain.

**CHPO**

Keyword meaning that only CHAMPOINTS will be calculated within the modal subdomain (see comment below).

**NOFO**

Keyword meaning that no external forces will be calculated within modal subdomain in order to save computation time.

**Comments :**

The elements currently available for calculations with domain decomposition are:

in 2D :

COQU, TRIA, BARR, MEMB, CL2D, CAR1, CAR4, COQC, Q92 , Q93 , COQI,  
ED01, CL2S, CL22, Q41L, Q42L, FUN2, T3VF, Q4VF

in 3D :

CUBE, COQ4, POUT, CL3D, BR3D, PR6, TETR, PRIS, PMAT, CL3T, CUB8,  
APPU, MECA, T3GS, FL38, DKT3, SHB8, FUN3, Q4G4, CL3Q, Q4GR, Q4GS,  
ASHB, T3MC, TEVF, PYVF, PRVF, CUVF

The code initially performs the following checks:

1. The union of the defined subdomains must cover the entire domain.
2. The intersection of any two subdomains must be empty.

In a calculation by domain decomposition, the following terminology applies:

**time cycle** Is the time increment associated with a specific subdomain. It varies in general from subdomain to subdomain.

**time step** Is a (macroscopic) time increment common to all subdomains (global quantity). At the end of each one of these time steps, all subdomains are “synchronized” by solving the equilibrium equations without any interpolation. For this reason, time steps are also called **sync steps**. Printout and storage of results and in general any interaction with the user is only available at sync steps.

**time stations** An integer counter that counts the union (not the sum) of time cycles and time steps. It is incremented by 1 each time the code computes at least one subdomain.

It is important to note that users normally have limited control over time cycles, which are managed internally by the code according to the characteristics of each subdomain. All time-related quantities, such as for example those of the **CALCUL** directive (see page I.20) or the chosen instants for data printing and storage (see the **ECRITURE** directive, page G.70), concern time steps (i.e. sync steps) as defined above, and not time cycles.

The user may explicitly **control the sync step** in the following ways:

- By choosing the option **OPTI PAS UTIL** and by specifying a **constant** sync step in the **CALCUL** directive by means of **PASFIX pfix**. In this case, the sync step is constant and equal to **pfix**.
- By choosing the option **OPTI PAS UTIL** and by specifying a list of sync steps in the **CALCUL** directive by means of **HIST /PROG/**. In this case, the sync step evolves according to the specified time sequence given in **/PROG/**.
- By choosing the (default) option **OPTI PAS AUTO** and by optionally specifying a factor (see **SDFA sdfac** keyword in the **CALCUL** directive). The code computes the maximum of the stability times steps of the various subdomains, and multiplies this value by **sdfac** to obtain the sync step (by default, **sdfac** is 1.0). This calculation is performed at the end of each sync step, so the sync step generally varies in time. Note also that in this case the sync steps may be automatically adapted to match the chosen printing and storage times by specifying the option **OPTI STEP IO** (default) or **OPTI STEP IOT**, see page H.20. Furthermore, in this case the maximum sync step may be limited by specifying the **DTMA dtmax** keyword of the **CALCUL** directive.

The user may explicitly **control the cycles** in the following ways:

- By choosing the option **OPTI CSTA** that specifies the safety factor over the stability value. This factor applies to all elements, and therefore it equally affects all subdomains (global quantity).
- By specifying, for each subdomain, a limiting value of the associated time cycle, see the **DTMX dtmx** keyword above.
- By specifying, for each subdomain, a constant time cycle, see the **DTFX dtfx** keyword above.

Note, however, that within each time step the time cycles vary in general from subdomain to subdomain, and vary in time for a given subdomain, even in the case that the user chooses `OPTI PAS UTIL` and a fixed time step, except in the case that a fixed cycle value `dtfx` value is explicitly specified.

When the `DTUN` keyword is used, subdomains are forced to follow one unique time scale. According to options set in the `OPTION` directive and to stability conditions on all subdomains, one single time-step is computed at each cycle and given to each subdomain. Subdomains are thus always synchronized.

Since the behaviour of the present domain decomposition model as regards time stepping depends upon the corresponding user option (i.e. upon `PAS AUTO` or `PAS UTIL`), it is advised to specify the `STRUCTURE` directive **after** any options that set the time stepping mode (but **before** the `CALCUL` directive, as already noted).

When option "POST" "CHPO" is activated, the CHAMELEMS are to be computed out of EUROPLEXUS from the CHAMPOINTS with a linear elastic constitutive law, which is the only valid within a modal subdomain.

Example 1 :

```
OPTION PAS AUTO STEP IO
...

STRUCTURE 3
  DOMA LECT zone1 TERM IDEN 91
  DOMA LECT zone2 TERM IDEN 92 DTMX 5e-6
  MODA LECT zone3 TERM IDEN 93 FICH FORM POST TOUS
      NOFO
CALCUL TINI 0.0   DTMAX 40e-5   NMAX 80000 TFIN 350e-3
```

The computational domain consists of three subdomains. The stability time cycle of the first subdomain is computed automatically by the code. The stability cycle of the second subdomain is the minimum between the computed value and 5E-6. The third subdomain is replaced by a modal basis, with modes and matrices given in file '`<base_name>.msh`' (i.e. the same file that contains the CASTEM 2000 mesh). Both the CHAMELEMS the CHAMPOINTS but no external forces are computed within this subdomain. The sync step is automatically computed by the code as the minimum value between 40E-5 (`DTMAX`) and 1.0 times the maximum stability step over all subdomains (recall that by default `sdfac` equals 1.0). Furthermore, the selected printing and storage times will be precisely matched by adapting the sync step (`OPTI STEP IO`). The three subdomains are identified as '91', '92' and '93' as far as interface declarations are concerned (see the `INTERFACES` directive next).

Example 2 :

```
OPTION PAS UTIL
...

STRUCTURE 3
  DOMA LECT zone1 TERM DTFX 1e-6
```

```
DOMA LECT zone2 TERM DTMX 5e-6
DOMA LECT zone3 TERM
CALCUL TINI 0.0 PASF 40e-5 NMAX 80000 TFIN 350e-3
```

The computational domain consists of three subdomains. The stability cycle of subdomain 1 is fixed to the constant value 1E-6. That of subdomain 2 is the minimum between its stability value and 5E-6. That of subdomain 3 is dictated only by local stability. The sync step is constant and has the value 40E-5. Printout and storage of results will occur at the sync steps whose times are greater than or equal to the chosen values. This is because the `OPTI STEP IO` or `OPTI STEP IOT` options may not be used in conjunction with `OPTI PAS UTIL`, i.e. the sync step (being constant) may not be adapted.

### MPI calculations

With fixed domain decomposition, the number of parallel threads **must** be equal to the number of declared subdomains.

For automatic domain decomposition, an external file can be entered to provide elementary weights, in order to optimize load balancing. Each line of the weight file is composed of 2 integers: first the number of the concerned elementary entity (finite element, finite volume, SPH particle...), second the weight associated to it.

Classically, EUROPLEXUS is used to generate the weight file (see page I.20). To generate the file from scratch, elementary numbers can be found in the listing file of a previous run with the same model. Every elementary entity, except CL elements and debris elements, must be given a weight.

If no weight file is used, all weights are set to 1.

Automatic decomposition using **Recursive Orthogonal Bisection** consists in successive recursive splittings of the domain along available space directions with circular permutations among them. Some directions may be deactivated, either by the user (`DACT` keyword) or automatically by the program if the bounds of the model along these directions are too small.

By default, the first splitting direction is the one along which the spatial extension of the model is maximal. However, one specific starting direction can be forced using `DPRE` keyword.

Classically, each splitting involved in the algorithm consists in creating from 1 part of the model 2 subparts of equal weight, yielding that the number of used threads must be a power of 2. However, the proposed algorithm allows to use any number of threads, by adjusting the number of levels of the recursive decomposition and the number of subparts per part created at the last level, 2 subparts per part being created at every other levels.

For example, with 12 threads, 3 levels will be considered, with 3 subparts per part at the last level ( $2 \times 2 \times 3 = 12$ ), whereas with 10 threads, 2 levels will be considered, with 5 subparts per part at the last level ( $2 \times 5 = 10$ ).

## 13.2 INTERFACES

### Object:

This directive allows to set options for the treatment of connections between subdomains. It also allows the explicit declaration of interfaces between couples of sub-domains. These interfaces *may correspond to matching or non-matching meshes*.

In the case of matching meshes, interface declaration is optional, provided the interface nodes are the same (i.e., have the same index) for the two sub-domains. If only the geometric points are identical (i.e., coordinates are the same), but each subdomain has its own nodes (with different indexes), a compatible interface has to be declared (see the `COMP` keyword below).

In the case of non-matching meshes, this directive is **mandatory**.

The `STRUCTURE` directive must appear **before** this directive.

### Syntax:

```
"INTERFACE" <"LINK"> < $[ "MULT" ; "NOMU" ]$ > <"NORE"> <"DUMP"> ...

... < nbinterf * (
    |[ "COMP" ; "MORTAR" ; "OPTIMAL" ]| <"TOLE" tole> ...

... "DOMAINE" ndom1 /LECTURE/  "DOMAINE" ndom2 /LECTURE/ ) >
```

### LINK

Interface connections are coupled with other kinematic links declared with the `LINK` directive (not the `LIAI` directive). This option is mandatory if some declared links concern more than just one subdomain.

Using this option causes the option `DTUN` of the `STRUCTURE` directive to be activated (see comment below).

Note that by default, i.e. by *not* specifying `LINK`, interface connections are treated independently from any other kinematic links, which implies that no kinematic link involving more than one subdomain can be declared.

### MULT

Every interface connections are treated by means of Lagrange multipliers. This is default.

### NOMU

Interface connections with matching meshes and coincident nodes are treated directly with no Lagrange multipliers (faster solution). Non-matching meshes or matching meshes with duplicated nodes are still treated by means of Lagrange multipliers.

The optional keyword **LINK** must be specified if **tt NOMU** is specified.

Using this option causes the option **DTUN** of the **STRUCTURE** directive to be activated (see comment below).

#### **NORE**

Do *not* re-localize the nodes and the dofs of the links on the S/D to which they belong. By default, they are re-localized.

The optional keyword **LINK** must be specified if **tt NORE** is specified.

#### **DUMP**

Produce extensive dump on the listing (for debugging).

#### **nbinterf**

Number of interfaces.

#### **COMP**

Keyword declaring an interface with matching meshes.

#### **MORTAR**

Keyword declaring an interface with non-matching meshes, treated by the mortar method (see comment below).

#### **OPTIMAL**

Keyword declaring an interface with non-matching meshes, treated by the optimal method.

#### **tole**

Tolerance given to find matching nodes (default=1.E-3).

#### **ndom1**

Identification number of the first sub-domain (see **STRUCTURE** directive).

#### **ndom2**

Identification number of the second sub-domain.

#### **/LECTURE/**

In the case of matching meshes, indexes of the **nodes** forming the sub-domains interfaces.

In the case of non-matching meshes, indexes of the interface **elements** forming the sub-domains interfaces (see comment below).

#### **Comments:**

Handling multiple time scales in a multi-domain framework requires a special treatment of the interface connections using Lagrange multipliers. This treatment is not available for generic kinematic links, so that multiple time scales option must be deactivated in order to couple interface connections to other kinematic links, which may thus involve more than one subdomain. This is done by automatically activating the **DTUN** option in the **STRUCTURE** directive. This is also the case when the treatment of connections between matching meshes with coincident nodes is accelerated by not using Lagrange multipliers. In those cases, multiple time scales in the model can still be taken into account by using the **PART** keyword in the **OPTION** directive.

When using the mortar method, the sub-domains whose mesh is used to discretize Lagrange multipliers has to be specified. It is the **second one** (**ndom2**) in the order of declaration of the sub-domains concerned by the interface.

When using interfaces with non-matching meshes, so-called **CLxx** elements (see pages INT.90 and INT.100) have to be affected to interface regions of each sub-domain. These elements **must** be given the “phantom” material (**MATE FANT**) with density equal to zero.

The treatment of non-matching meshes with 3D solid elements is restricted to hierarchical meshes. In this case, the mortar method and the optimal method are identical, and a mortar interface has to be declared, **with the second sub-domain corresponding to the finest mesh**.

The **mortar** method may be used with any element types in 2D, but only with shell element types in 3D. When using the **mortar** method with linear interfaces (2-noded element sides), there must be at least one geometrical point that has the same coordinates, within the tolerance **tole** defined above, in the two facing meshes. The node indexes (node numbers) of this point can be different in the two meshes. This is necessary because the interface model uses the point's coordinates internally in order to define a reference frame on the interface.

Example 1 :

```
STRUCTURE 3
  DOMA LECT zone1 inte12 inte13 TERM IDEN 91
  DOMA LECT zone2 inte21 inte23 TERM IDEN 92
  MODA LECT zone3 inte31 inte32 TERM IDEN 93 FICH FORM 'fich.mrd'
                                     POST TOUS NOFO
...
INTERFACE 3
  COMP TOLE 1.E-2
    DOMA 91 LECT inte12 TERM
    DOMA 92 LECT inte21 TERM
  MORTAR TOLE 1.E-2
    DOMA 92 LECT inte23 TERM
    DOMA 93 LECT inte32 TERM
  OPTIMAL
    DOMA 91 LECT inte13 TERM
    DOMA 93 LECT inte31 TERM
```

The computational domain consists of three sub-domains. Three interfaces are declared:

1. The first between sub-domain number 91 and sub-domain number 92 with matching meshes.



2. The second between sub-domain number 92 and sub-domain number 93 with non-matching meshes treated by the mortar method. The nodes of sub-domain number 93 are used to enforce kinematical continuity.
3. The third between sub-domain number 91 and sub-domain number 93 with non-matching meshes treated by the optimal method.

Note that `inte12`, `inte21` are nodes groups, whereas `inte23`, `inte32`, `inte13`, `inte31` are elements groups.

### 13.3 XFEM

#### Object :

This directive enables the use of the eXtended Finite Element Method (X-FEM). It uses a specific Level-set mesh to describe the crack in space.

The formulation of X-FEM is given by a standard part and an enriched one as follow:

$$\bar{\mathbf{U}} = \sum_{i \in \mathcal{N}} N_i(\mathbf{x}) \mathbf{U}_i + \sum_{j \in \mathcal{N}^e} N_j(\mathbf{x}) H(\mathbf{x}) \mathbf{U}_j^e \quad (53)$$

This formulation allows to take into account a displacement discontinuity in the mechanical mesh (a crack). And the characterization of propagation law makes the crack propagate through the mesh without remeshing at any time. But enriched element are used in order to describe the discontinuity with additionnal degrees of freedom. The corresponding available elements are XCAR and XCUB for 2D and 3D.

#### References

The model is described in reference [\[826\]](#), [\[830\]](#).

#### Syntax :

```
"XFEM"      "NI"      ni      "NJ"      nj      "DX"      dx      "DY"      dy
             "XZER"    xzer    "YZER"    yzer    "FISX"    fisx    "FISY"    fisy
             "FISC"    fisc    "PRBX"    prbx    "PRBY"    prby    "PRBC"    prbc
             "ORDR"    ordr    "KICR"    kicr    "RAYO"    rayo    "CHOI"    choi
             "CR"      cr
             <"NK"      nk      "DZ"      dz      "ZZER"    zzer    "FISZ"    fisz
             "PRBZ"    prbz    "RPLA"    rpla    "NCOU"    ncou >
/LECTURE/
```

**ni**

Number of subgrids in x direction.

**nj**

Number of subgrids in y direction.

**nk**

Number of subgrids in z direction (Default 1).

**dx**

Discretization of subgrids in x direction.

**dy**

Discretization of subgrids in y direction.

**dz**

Discretization of subgrids in z direction (Default 0.).

**xzer**

Position x0 of left bottom point of the mesh of the level-set.

**yzer**

Position z0 of left bottom point of the mesh of the level-set.

**zzer**

Position z0 of left bottom point of the mesh of the level-set (Default 0.).

**fisx**

Equation of the planned crack surface.

**fisy**

Equation of the planned crack surface.

**fisz**

Equation of the planned crack surface (Default 0.).

**fisc**

Equation of the planned crack surface.

**prbx**

Equation of the planned crack front surface.

**prby**

Equation of the planned crack front surface.

**prbz**

Equation of the planned crack front surface (Default 0.).

**prbc**

Equation of the planned crack front surface.

**ordr**

Paramater for level set algorithms (reinitialization, orthogonalization, propagation, extension).

**kicr**

Critical value for crack propagation.

**rayo**

Length to characterize average or integral "near crack tip".

**choi**

Parameter to choose: 1 for stress intensity factors (only 2D), and 2 for non-local stress near crack tip (2D and 3D).

**cr**

Rayleigh wave velocity (maximum crack velocity).

**rpla**

Length parameter to subcut element (choose around 3 times rayo).

**ncou**

Number of layer in thickness (1 in 2D, and Default 1).

**LECTURE**

List of the elements concerned (XCAR in 2D or XCUB in 3D).

The localization of the initial crack is given by 2 plans. The first one defines the surface of the crack by: (level-set  $\phi_1$ )

$$fisx \cdot X + fisy \cdot Y + fisz \cdot Z + fisc = 0 \quad (54)$$

The corresponding level-set is:

$$\phi_1(X, Y, Z) = \frac{fisx \cdot X + fisy \cdot Y + fisz \cdot Z + fisc}{\sqrt{fisx^2 + fisy^2 + fisz^2}} \quad (55)$$

And the second one defines the front of the crack: (level-set  $\phi_2$ )

$$prbx \cdot X + prby \cdot Y + prbz \cdot Z + prbc = 0 \quad (56)$$

The corresponding level-set is:

$$\phi_2(X, Y, Z) = \frac{prbx \cdot X + prby \cdot Y + prbz \cdot Z + prbc}{\sqrt{prbx^2 + prby^2 + prbz^2}} \quad (57)$$

The crack is the isozero  $\phi_1$  and the negative part of  $\phi_2$ . Both level-sets are exported in paraview output with the keyword XLVL. So the representation of the crack is possible in paraview by doing the negative part of PHI2 on isozero PHI1.

Example 1:

```
...
  ECRITURE
    ...
    FICHER FORMAT AVS PRVW FREQ 10
    VARI DEPL XLVL ECRO ECRC LECT 2 TERM
...
  XFEM
    NI 300    NJ 200    NK 100
```

```
DX 0.0005 DY 0.0005 DZ 0.00025
XZER 0.03 YZER 0.01 ZZER -0.004
FISX -0.2 FISY 1. FISZ 0. FISC -0.026
PRBX 5. PRBY 1. PRBZ 0. PRBC -0.416
ORDR 8 KICR 12.E6 RAYO 0.001 CHOI 2
CR 250. RPLA 0.003 NCOU 7
LECT MAILX
...
CALCUL TINI 0.0 DTMAX 40e-5 NMAX 80000 TFIN 350e-3
```

### Bibliography:

Belytschko T., Black T., "Elastic crack growth in finite elements with minimal remeshing. *International Journal for Numerical Methods in Engineering* (1999) **45**:601–620.

Moës N., Dolbow J., Belytschko T., "A finite element method for crack growth without remeshing", *International Journal for Numerical Methods in Engineering* (1999) **46**:131–150.

Moës N., Gravouil A., Belytschko T., "Non-planar 3D crack growth by the extended finite element and level sets - Part I: Mechanical model", *International Journal for Numerical Methods in Engineering* (2002) **53**:2549–2568.

Gravouil A., Moës N., Belytschko T., "Non-planar 3D crack growth by the extended finite element and level sets - Part II: Level set update", *International Journal for Numerical Methods in Engineering* (2002) **53**:2569–2586.

Menouillard T., Réthoré J., Combescure A. and Bung H., "Efficient explicit time stepping for the eXtended Finite Element Method (X-FEM)", *International Journal for Numerical Methods in Engineering* (2006) **68**:911–939.

Menouillard T., "Dynamique explicite pour la simulation numérique de propagation de fissure par la méthode des éléments finis étendus", Thèse de Doctorat INSA de Lyon 2007.

## 13.4 "CALCUL" DIRECTIVE

### Object:

This directive starts the time solution of a given problem. The keyword **CALCUL** is compulsory and should appear after the data sets A, B, C, D, E, F, G and H.

The user can specify the initial and final times of the computation, the value of or constraints on the time step and the maximum number of computation steps.

### Syntax:

```
"CALCUL"  "TINI" tini  "TEND" tend
          < "NMAX" nmax                                >
          < "DTMI" dtmin                               >
          < "DTMA" dtmax                                >
          < "TFAI" tfai                                 >
          < $[ "HIST" /PROG/ ; "PASFIX" pfix ]$ >
          < "PAS1" pasone                               >
          < "SDFA" sdfac                                >
          < "LBMS" nfreq                                >
          < "LBNS" nstep                                >
          < "LBMD" mxdev                                >
          < "LBST"                                       >
          < "LBPW" ndwfil                               >
          < "LBFT"                                       >
```

#### tini

Initial time of the computation. In case of a restart run, this value is ignored (it may actually be left out), since the actual initial time is set to the value read from the restart file.

#### tend

Final time of the computation (the keyword **TFIN** is also accepted as a synonym of **TEND**).

#### nmax

Maximum number of computation steps. Default is 1000000.

#### dtmin

Minimum value for the time step. Is only considered in a **PAS AUTO** or **PART** calculation. Default is 1.D-12 at JRC.

#### dtmax

Maximum value of the time step. Is only considered in a **PAS AUTO** or **PART** computation. Default is 1.D12.

**tfai**

Elements having a smaller stability time step than this value are eroded (failure). Note that the chosen value applies to *all* elements in the mesh. However, only those elements that possess an “erodible” material (see **EROS** directive on page A.30 for more information) are actually eroded. Whenever an element is eroded due to this criterion, the element characteristics at the moment of the erosion are written on the listing. This may allow to check *a posteriori* why the element’s stability dropped below the specified value (e.g., excessive distortion of the element).

**HIST**

Can only be used in **PAS UTIL** cases. The following **/PROG/** procedure defines the actual ‘time history’ of the computation, i.e. all the times for which the solution will be computed. In this way, the user assumes complete control over the time step. No check on minimum or maximum values are performed. Time steps are computed as the difference between two successive specified times. The initial time of the computation should not be specified in the **/PROG/**.

**prefix**

This is a short-cut to assign a user defined time step that remains fixed in time. Can only be used in **PAS UTIL**.

**pasone**

This option allows to specify the value of the time increment used during step 0 and is only useful for **PAS AUTO** cases. In particular, it is mandatory to use it in the case of an advection-diffusion calculation, because in such a case the program does not compute the first time increment automatically. Another useful case is in the presence of energy injection in MC (multicomponent fluid) calculations, in order to use a smaller initial  $\Delta t$  than that automatically computed by the code. In order to let the time increment grow slowly, use can be made of the **DTVA** option (see Options related to the time step).

**sdfac**

Factor for subdomains computations. This optional keyword is only relevant in multi-domain calculations (see **STRUCTURE** directive on page I.15) that use automatic calculation of the time step (**OPTI PAS AUTO**). In this case, the sync step common to all subdomains is automatically chosen as **sdfac** times the **largest** stability step of the various subdomains. By default, **sdfac** equals 1.0.

**nfreq**

MPI only. Number of time-steps between two load-balancing measuring periods (see comment below).

**nstep**

MPI only. Number of time-steps within a load-balancing measuring period (see comment below).

**mxdev**

MPI only. Maximum standard deviation for load-balancing quality estimation (see comment below)..

**LBST**

MPI only. Stop calculation if previous maximum standard deviation is exceeded for elementary tasks balance (see comment below).

**LBPW**

MPI only. Print elementary weight file at the end of each load-balancing measuring period (see comment below)..

**ndwfil**

MPI only. Number of the logical unit of the weight file or file name in quotes. If omitted, the program chooses a file name by default (see page A.27). The default extension is .wgt.

**LBFT**

MPI only. Enable filtering of measured computational costs before writing weight file (see comment below).

**Comments:**

The word **CALCUL** is compulsory and should appear only once.

If a user does not have an idea of the stability step for a given problem, he can run the program with the

If a user specifies **PAS UTILISATEUR**, he is responsible for the stability of the computation, because **EUROPLEXUS** does not check the stability in this case. This option may therefore lead to instabilities in the calculation, and should only be used in special cases.

The maximum value of the time step enables the time step to be limited in the case of the option **PAS AUTOMATIQUE** or **PARTITION**.

The computation stops when either the maximum number of steps or the final time is reached. A computation in **PAS AUTO** or **PART** also stops if the stability step becomes lower than the minimum value.

If **HIST** is used, remember to dimension adequately (see **TTHI** on page A.105).

Note that, in multi-domain computations, all time-related quantities in the **CALCUL** directive refer to sync steps and not to subdomains cycles (see directive **STRUCTURE** on page I.15).

**Load-balance measuring for MPI calculations**

This section is still under strong development.

Load-balance is a key point to achieve parallel performance. Load-balance measuring consists in measuring time taken by each thread to perform computational tasks within a given number of time steps.



The first measuring period starts at the first step of the simulation. After that, the number of time steps between the starting steps of two successive measuring periods is given using LBMS keyword. Measuring options are activated as soon as a positive integer is read after LBMS keyword.

The number of time steps within a measuring period is given using LBNS keyword, which should be smaller than the interval between two measuring periods to produce accurate results. Default value is 100.

Quality of load-balance is estimated through the value of the standard deviation of the quantity of interest. If the quantity is well distributed among the threads, standard deviation should be close to 0. 2 quantities are currently considered: first, the time needed to perform elementary computations, which is controlled by the quality of domain decomposition, second, the time needed to perform every computational tasks, including treatment of links. LBMD is used to enter a maximum authorized standard deviation for both quantities, generating a warning message if it is overcome at the end of a measuring period. The calculation can be forced to stop in the case of unauthorized standard deviation concerning elementary computations (LBST keyword), as it indicates that domain decomposition should be improved.

Using time measures during a period, the program is able to estimate the computational cost of elementary entities (finite elements, finite volumes, SPH particles...), as far as elementary operations only are concerned. A weight file can be written using LBPW keyword, to be used to improve automatic domain decomposition (see page I.15).

Measured weights can be filtered prior to file writing, to account for numerical noise in measures due to execution environment. Current filter consists in computing a global cost for groups of elements instead of individuals. Elements are grouped with respect to the couple of parameters (type of element, type of material). Group weight is then equally divided among concerned elements.

This can produce bad results and should be used with care. Raw weights and filtered weights can be visualized using PVTk output file.

## 13.5 ED1D INPUT DECK

### Object:

This directive allows to specify a so-called “EURDYN-1D input deck”, i.e. a set of input data to be read by the EURDYN-1D module (ED1D) that is now embedded within EUROPLEXUS. In this way it is possible to perform a coupled 1-D/multi-D calculation, as described on page I.80.

The ED1D input deck must be included within the normal EUROPLEXUS input file, immediately after the CALCUL directive and before any additional EUROPLEXUS directives (for example, QUALIFICATION).

The ED1D input deck must be immediately preceded by a line containing ED1D (capitals, starting in column 1, followed only by blanks if any) and be immediately followed by a line containing ED1D END (capitals, starting in column 1, followed only by blanks if any).

### Syntax:

```
. . .
"CALC" . . . (see CALCUL directive, page I.20)
*
*=====
ED1D
(as many ED1D data as needed to describe the 1D part of
the numerical model)
ED1D END
*=====
*
<"PLAY" . . . or "QUAL" . . . or "SUIT" or "FIN">
```

### Comments:

The keyword ED1D must appear as such and start in column 1. There must not be any other data on the same line.

The keywords ED1D END must appear as such and start in column 1. There must not be any other data on the same line.

The contents of the ED1D data deck proper (i.e. the lines contained between ED1D START and ED1D END) is described in the EURDYN-1D manual, listed in the References: ([33]).

By default, EUROPLEXUS reserves a memory of 50,000 REAL\*4 for the ED1D data. If necessary the size of this memory can be increased by the DIME ME1D keyword, see Page A.67.

## 13.6 PLAY (interactive commands)

### Object:

This directive allows to execute a set of “interactive” commands, i.e. any of the commands described on Pages A.25 and O.10, by reading them from a file (actually, from the regular EUROPLEXUS input file) rather than from the keyboard.

If present, this directive must immediately follow the **CALC** directive (and the optional **ED1D** . . . **ED1D END** directive, if present).

Normally, interactive commands are typed by the user at the keyboard. With the present directive, it is possible to store such commands in the regular EUROPLEXUS input file, with the advantage that a given “interactive” calculation may be repeated identically as many times as needed.

This feature is especially useful for the automatic execution of calculations with intermediate visualizations (**TRAC**) and for the automatic generation of animations (**AVI**).

After reading the **PLAY** directive, the code continues to read the following commands from the regular EUROPLEXUS input file, but interprets them as interactive commands (i.e. like if they were typed at the keyboard), until the termination sequence **ENDPLAY** is encountered. Then, normal input file reading (again from the regular EUROPLEXUS input file) is restored.

### Syntax:

```
. . .
"CALC" . . . (see CALCUL directive, page I.20)
*
<"ED1D" {Eurdyn-1D input deck} "ED1D END">
*
=====
PLAY
(as many 'interactive' commands (see Pages A.25 and O.10) as needed)
ENDPLAY
=====
*
<"QUAL" . . . or "SUIT" or "FIN">
```

### Comments:

The keyword **PLAY** must appear as such and start in column 1. There must not be any other data on the same line.

The keyword **ENDPLAY** must appear as such and start in column 1. There must not be any other data on the same line.

The available interactive commands are listed on page A.25.

## 13.7 QUALIFICATIONS

### Object :

This directive allows to verify (qualify) the results of a calculation by comparing them with given reference values.

The keyword **VALIDATION** is still accepted in place of **QUALIFICATION** for backward compatibility. However, new input files should always use the keyword **QUAL**.

The qualification is normally done at the final time of the transient calculation. However, when reading a results file it can also be done at an intermediate time by using the **SORT ARRE** commands, see comments below and Page ED.40.

Yet another possibility is that of doing a qualification *interactively*, i.e. on-the-fly (as the calculation is being performed interactively) at the *current step*. See the interactive **QUAL** command on page O.10. Since the syntax of the **QUAL** directive is a bit complicated to be typed interactively, this possibility is probably most useful from within a **PLAY ... ENDP** directive.

### Syntax :

```
"QUAL" <"AUTO"> (  |[  "COOR" ; "DEPL" ; "VITE" ; "ACCE" ; "FEXT" ;
                        "MASN" ; "ADFT" ; "MCPR" ; "MCRO" ; "MCTE" ;
                        "MCVI" ; "MCMF" ; "SIGN" ; "ECRN" ; "FINT" ;
                        "FLIA" ; "FDEC" ; "PFSI" ; "PFMI" ; "PFMA" ;
                        "CONT" ; "EPST" ; "ECRO" ; "ENEL" ; "RHO" ;
                        "MASE" ; "EPAI" ; "VCVI" ; "RISK" ; "DEBR" ;
                        "BILA" ; "WINT" ; "WEXT" ; "WCIN" ; "WECH" ;
                        "TIME" ; "COUR" icourb >                ]|

                        $[ |[ "COMP" icomp ; "NORM" ]| <"GAUS" igauss> ]$
                        ![ "REFE" valref      "TOLE" valtol ]!

                        /LECTURE/ )
```

### AUTO

Automatic qualification, see comments below. This keyword should be used only in very special cases.

### COOR ... PFMA

Name of the nodal variable to be checked: **COOR** (coordinate), **DEPL** (displacement), **VITE** (velocity), **ACCE** (acceleration), **FEXT** (external force), **MASN** (nodal mass), **ADFT** (advection-diffusion temperature), **MCPR** (finite volume pressure), **MCRO** (finite volume density), **MCTE** (finite volume temperature), **MCVI** (finite volume velocity), **MCMF** (finite volume mass fraction), **SIGN** (spectral element stress at nodes), **ECRN** (spectral element internal variable at nodes), **FINT** (internal force), **FLIA** (force due to LIAI/LINK COUP), **FDEC** (force due to LINK DECO), **PFSI** overpressure due to FSI, **PFMI** minimum FSI overpressure in time, **PFMA** maximum FSI overpressure in time.

**CONT ... DEBR**

Name of the element variable to be checked: **CONT** (stress), **EPST** (total deformation), **ECRO** (internal variable), **ENEL** (internal energy), **RHO** (density), **MASE** (element mass), **EPAI** (thickness), **VCVI** (FV centroid velocity). **RISK** (Human risk model. **COMP 1**: Death; **COMP 2**: Eardrum rupture; **COMP 3**: Max. overpressure; **COMP 4**: Impulse). **DEBR** (Flying debris model. **COMP 1**: Current flying area for an active debris particle, current impact area for an inactive debris particle (marker); **COMP 2**: Current activity flag for a debris particle: -1 for an unused debris, 0 for an inactive debris (marker), +1 for an active debris (particle)).

**COMP icomp**

Number of the component concerned (for nodes or elements only).

**NORM**

For vectorial quantities, such as velocities, this causes the code to compute the norm of the first IDIM components (length of the vector). For other types of quantities requesting the norm may be illegal and in this case it raises an error.

**igaus**

Number of the Gauss point (integration point) concerned (for elements only). By default **igauss=1**.

**BILA ... TIME**

Name of the global variable associated with the energy balance for the whole calculation, which allows to monitor the stability and the energy transfers: **BILA** energy balance, **WINT** internal energy, **WEXT** work of the external forces, **WCIN** kinetic energy, **WECH** energy exchanged with the external world, injected or lost, **TIME** physical time. For these parameters, the component and **/LECTURE/** are redundant.

**COUR icourb**

**icourb** is the number of the curve defined in **GRAPH** directive. For this parameter, the component and **/LECTURE/** are redundant.

**valref**

Reference value expected.

**valtol**

Relative tolerance. If a **negative** value is specified, the qualification is ignored, i.e. it is always considered valid. This feature should only be used during the code evolution process, when it is necessary to temporarily disable a certain qualification, in view of forthcoming changes in the code source which will have an influence on the results.

**LECTURE**

Number of the node or element concerned.

**Comments :**

Only one node or element must be concerned by the validation.

The specification of a Gauss point makes sense only for parameters related to elements, such as: **CONTR**, **EPST** or **ECRO**.

It is possible to check as many values as needed, by repeating the name of the variable. The calculation will be considered correct if **all** checks are correct.

By default the qualification is done at the final time of the calculation. When reading a results file, it is possible to perform the qualification at a different time (not the final one), by using the **SORT ARRE** command, see Page ED.40. This can be useful in case the “signal” (monitored quantity) to be checked has either a very small value or a very large time derivative at the final time, while it has a more “stable” value (plateau) at a previous time.

It is sometimes tedious to prepare an input data set for a new test case or benchmark calculation, especially when many quantities must be verified. In some cases, the reference values are not well known a priori from physical considerations, analytical solutions or experimental data. Therefore, sometimes these values are computed by the code itself.

Although this is in some sense a misuse of the **QUAL** directive (because it is no longer a true qualification of results!), it may be useful e.g. to verify non-regression of code results during the development phase.

In such cases, the **AUTO** directive may speed up the process of preparing the input file, in the following way. First, write the qualification directives for the desired quantities, but by setting arbitrary reference values (e.g. all zero). Specify the **AUTO** directive immediately after **QUAL** and run the code.

In this way, qualification is computed normally, but for each verified quantity the code writes on the listing an extra line, introduced by the sequence **AQ:**, and containing the qualification directive with the ‘correct’ reference value (i.e. the one found by the code).

By searching all such lines in the listing, cutting them and pasting into the input file, it is possible to obtain a ‘correct’ input file much more rapidly (and with less errors) than by typing in the correct reference values by hand. Do not forget to remove the **AUTO** keyword once done the job.

Example of using the command **COUR**:

```
GRAPH
  COURBE 52 'RR-P2' distance lecture 00 P2 term
  ...
  QUALIF COURBE 52  REFE 1.2          TOLE 1e-4
```

## Outputs :

The expected and obtained values are printed on the listing, together with the relative error, which is compared with the tolerance.

For each correct (respectively incorrect) check, the phrase: **==> VALIDATION : SUCCES** is printed on the listing (resp. **==> VALIDATION : ECHEC**).

At the end of the calculation, if all is fine the phrase: ==> LE CALCUL EST CORRECT ! is printed, else: ==> LE CALCUL EST FAUX !.



### 13.8 "SUITE" OR "FIN"

**Object :**

The word "SUITE" written immediately after the instruction "CALCUL" enables the next data set to be read, and the corresponding case to be processed, immediately after the first (or current) computation.

Using the word "SUITE" placed after each data set, the user can enter as many data sets as he wants.

The last set must end with the word "FIN".

**Syntax :**

```
$[ "SUITE" ; "FIN" ]$
```

**Comments :**

The word "SUITE" is the only word of the directive. It must immediately follow the instruction "CALCUL" .

The word "FIN" is compulsory at the end of the data.

## 14 GROUP ED—POST-TREATMENT BY EUROPLEXUS

### Object:

To post-treat a results file containing the EUROPLEXUS results from a previously executed transient calculation.

### Syntax :

1/ General syntax :

```

... title ...

<"ECHO">

<"OPNF" . . . >

"RESUL" . . .

<"DIME" . . . "TERM">

"SORT"      $  "ARRET"      . . .  $
              $              $
              $  "FICHIER"   . . .  $
              $              $
              $  "ECRITURE"  . . .  $
              $              $
              $  "GRAPHIQUES" . . .  $
              $              $
              $ ( "VISUALISER" . . . ) $

<"QUAL" . . . >

"FIN"
```

### Comments:

These directives are described in detail on the following pages, except for the **QUAL** directive, which has been already presented on page I.25.

The following page shows a full synopsis of the EUROPLEXUS post-treatment directives.

```

... title ...
<ECHO> <OPNF . . . >
RESU <FORM>
    |<SPLI> ALIC;ALIC TEMP;UNIV <CURR>;UNIV OBSO|
    |nban;'nom_fich'| <GARD> <PSCR>
<DIME <TIMP nimp> TERM>
<FONC ... , see page E.15>
SORT $ ARRE <TEMP time;NUPA npas;NSTO nsto> $
$ FICH <FORM> nfic $FREQ nfre;TFRE tfre $ $
$ $NUPA /LECT/;PASM pasm$ $
$ ECRI <COOR> <DEPL> <VITE> <ACCE> <FINT> <FEXT> $
$ <CONT> <EPST> <ECRO> /CTIM/ $
$ <NOPO;POIN /LECT/> <NOEL;ELEM /LECT/> $
$ <FICH <FORM> K200 ndca /CTIM/ POIN /LECT/ <CHAM>> $
$ GRAP AXTE coef 'nom_axe_0x' $
$ <MINM> <FENE tmin tmax> $
$ <PERF 'nom_fic'> <PERK 'nom_fic'> $
$ (COUR nuco <'nomcourbe'> $
$ $WINT;WEXT;WCIN;BILA;WSUM;DTMI;DTMA;MXSU$ <COMP ico> $
$ $COOR;DEPL;VITE;ACCE$ $
$ $FORC;ADFT;MCPR;MCRO$ $
$ $MCTE;MCMF;SIGN;ECRN$ $
$ $LFNO;LFNV;ILNO;DTNO$ $COMP ico;NORM$ NOEU /LECT/ $
$ $CONT;ECRO;EPST;ENEL$ $
$ $WAUX;LFEL;LFEV;DTEL$ COMP ico <GAUS igau> ELEM /LECT/ $
$ $VCVI$ $COMP ico;NORM$ ELEM /LECT/ $
$ $SOMM nbrs*(courbe_i coef_i)$ $
$ $PROD pcoef nbrp*(courbe_k) $ $
$ $INTE courbe_i $ $
$ $DIST /LECT/ $ $
$ $LIBR $ $
$ $MASS;VOLU;BARY;VMOY$ $
$ $IMPU;ECIN;EINT;EEXT$ $
$ $EPDV;EINJ;RESU;IRES$ $
$ $ECRG;DT1 $ <COMP ico;NORM> <REGI nure>) $
$ (LCOU nuco <'nomcourbe'> <FICH 'nom_fich'> $
$ $STEP;TCPU;DTCR;ELCR;DEE $ $
$ $DMMN;DMME;DTMX;ELMX;VMAX$ $
$ $NVMX;ELST;MEMO;MEMP$ <NMAX nmax>) $
$ (SCOU nuco <'nomcourbe'> <$T t;NPAS npas;NSTO nsto> $
$ $SAXE scoe 'nom_saxe' <INIT> /LECT/ $
$ $COOR;DEPL;VITE;ACCE$ $
$ $FORC;ADFT;MCPR;MCRO$ $
$ $MCTE;MCMF;SIGN;ECRN$ $
$ $LFNO;LFNV;ILNO;DTNO$ $COMP ico;NORM$ $
$ $CONT;ECRO;EPST;ENEL$ $
$ $WAUX;LFEL;LFEV;DTEL$ COMP ico <GAUS igau>) $
$ (RCOU nuco 'nomcourbe' FICH 'nom_fic' $
$ <RENA 'new_name'> <FACX fx> <FACY fy>) $
$ (DCOU nuco <'nomcourbe'> $npt*(x y);FONC ifon$) $
$ ($TRAC;XMGR$ $
$ $K200;LIST$ (nuco) <PS <TEXT>;MIF> AXES coef 'nom_axe_0y' $
$ <XAXE nxax coex 'nom_axe_0x'> $
$ <COLO (co)> <THIC (th)> <DASH (da)> $
$ <XZER> <YZER> <XGRD> <YGRD> <XLOG> <YLOG>) $
$ (VISU $T t;NPAS npas;NSTO nsto$ $
$ <PLAY> $
$ <sequel of interactive commands, see pages A.25, 0.10> $
$ <ENDPLAY>) $
<QUAL ... , see page I.25>
FIN

```

## 14.1 TITLE AND CHOICE OF RESULTS FILE

### Object:

The user gives a title and specifies the file (or files) from which the results to be edited will be read. The file(s) must have been produced during a previous execution of EUROPLEXUS (or during a previous phase of a composite execution, where the various phases are separated by the keyword **SUIT**).

Currently, results may be edited from any of the following file types:

- An ALICE file (either single or split);
- An ALICE TEMPS file;
- A file of type UNIVERSAL CURRENT;
- A file of type UNIVERSAL OBSOLETE.
- A file of type POCHHAMMER.

However, note that a file of type POCHHAMMER can only be read *in addition to* a file of the other types (usually an ALICE file). It cannot be read in by itself.

### Syntax :

```
/TITLE/
```

```
<"ECHO">
```

```
<"OPNF" < "FORMAT" > nfic 'nom.fic'>
```

```
"RESUL" (<"FORMAT">  
  |[ < "SPLI" > "ALIC" ; "ALIC" < "TEMP" > ; "POCH" ;  
    "UNIV" <"CURR"> ; "UNIV" "OBSO" ]|  
  $[ nban ; 'nom_fich' ]$  
  <"GARDE">  
  <"PSCR"> )
```

"ECHO"

Like for a normal calculation, this keyword indicates that the EUROPLEXUS input directives will be echoed in the execution window.

"OPNF"

This option may be used to open the chosen results file, like for a normal calculation. Refer to page A.28.

**"FORMAT"**

This keyword indicates that the chosen results file is a formatted file. By default, this file is unformatted.

**"SPLI"**

The chosen results file is a set of ALICE split files rather than a single file, produced by the directive `ECRI ... FICH SPLI ALIC ...`, see page G.70.

**"ALIC"**

The chosen results file is an ALICE file (this is the default).

**"ALIC TEMP"**

The chosen results file is an ALICE TEMPS file.

**"POCH"**

The chosen results file is a POCHHAMMER file (which is being read in addition to another results file).

**"UNIV CURR"**

The chosen results file is a file of type UNIVERSAL CURRENT. The keyword `CURR` may be omitted in this case since this is the default for a file of type UNIVERSAL.

**"UNIV OBSO"**

The chosen results file is a file of type UNIVERSAL OBSOLETE.

**nban**

Number of the logical unit on which the results file is stored.

**nom\_fich**

Name of the results file, enclosed in single quotes.

**"GARDE"**

This keyword allows to keep for the drawings the title read in the results file. Else, it is the title defined above.

**"PSCR"**

This keyword allows to produce the plots resulting from the `GRAP` directive in PostScript. Since 1995 it is the default, so that this keyword is redundant now.

**Comments:**

The word `RESULT` is compulsory.

When it is present, only an edition of results may be done and not a normal calculation.

## 14.2 DIMENSIONING

### Object :

Allocation of memory for the post-treatment of a results file by means of EUROPLEXUS.

If one limits itself to graphical output, EUROPLEXUS automatically allocates the necessary space, so it is no longer necessary to give dimensions. This directive must therefore be omitted in that case.

### Syntax:

```
"DIME"
```

```
<      "TIMP"      nimp  >
```

```
"TERM"
```

**nimp**

Number of time steps for which printing on the listing is requested (see option /CTIM/ of ECRI on page ED.50).

### 14.3 OUTPUTS

#### Object :

The following directive enables the types of output to be chosen.

#### Syntax:

```
"SORT"
$  "ARRET"  <"TEMPS" time ; "NUPAS" npas ; "NSTO" nsto>  $
$                                                    $
$  "FICHIER"      . . .                                     $
$                                                    $
$  "ECriture"     . . .                                     $
$                                                    $
$  "GRAPHIQUES"  . . .                                     $
$                                                    $
$ ( "VISUALISER" . . . )                                   $
```

#### ARRET

This directive allows to stop reading the results file at the time instant, at the time step or at the time station corresponding to values `time`, `npas` or `nsto`, respectively, specified in the directive, rather than reading the whole file. Note that a storage station is always produces at step 0 (beginning of the transient calculation): this storage station has the index `nsto=0`. This directive is only useful for the qualification of a calculation at intermediate times (and not at the final time as per default), since it may **not** be combined with the other directives `FICH`, `ECRI`, `GRAP` and `VISU`, as indicated in the syntax. For more details on the qualification, see directive `QUAL` on Page I.25.

#### FICHIER

To extract from the chosen results file a certain number of computation steps, and to store them in a new results file which will typically contain less information (less storage stations).

#### ECRI

To print out results on the EUROPLEXUS listing.

#### GRAP

To produce graphic outputs. The curves of certain variables are drawn with respect to time or are printed on file(s) in a variety of possible formats.

#### VISU

To produce (a subset of) the visualizations that are possible during direct execution of the code (see Pages A.25 and O.10). These include graphical rendering either interactively in a window or in batch mode on a file and production of animations. Not all visualization types and features are available, though (see below for details).

**Comments:**

The keyword **SORT** should appear only once in an input data sequence. Note, however, that the **VISU** sub-directive may be repeated as many times as needed inside the **SORT** directive.

The directives **ARRE**, **ECRI**, **GRAP**, **FICH** and **VISU** are mutually exclusive.

In the case that a graphical output is requested (**GRAP** ... **TRAC**), the produced file is in the PostScript format (a product of Adobe Co.).



## 14.4 CREATING A REDUCED RESULTS FILE

### Object:

To extract a certain number of computation steps from the chosen results file, in order to create a new results file which has the same structure as if it was created directly, but typically contains less information (less storage stations).

### Syntax:

```
"FICHIER" < "FORMAT" >  nfic    | [  "FREQ"      nfreq      ;
                                   "TFREQ"     tfreq      ;
                                   "NUPAS"     /LECTURE/   ;
                                   "PASMAX"    pasmax     ] |
```

#### FORMAT

This keyword indicates that the new file created will be formatted. By default, it is unformatted.

#### nfic

Logical number of the new file.

#### nfreq

All the results whose step number is a multiple of **nfreq** are extracted from the results file.

#### tfreq

Time interval between two extracted results.

#### /LECTURE/

List of the step numbers to be taken.

#### pasmax

Maximum number of the time step to be copied.

### Comments:

The options **FREQ**, **TFREQ**, **NUPAS** and **PASMAX** can be combined.

If the step number required is not stored in file **nfic**, EUROPLEXUS takes the step just above it.

The option **PASMAX** allows e.g. to “clean up” a results file that has become unusable due to a computation error. In fact, one may then create a new file containing the results of the steps from the beginning to a step **pasmax** prior to the encountered error.

**Warning:**

The logical unit number of the new file (**nfic**) must be different from that of the old one, **nban**, defined in the instruction **RESULT** (see page ED.20).

## 14.5 PRINTOUTS ON THE LISTING

### Object :

To print data extracted from a chosen results file onto the EUROPLEXUS listing file or to produce a CASTEM 2000 file for further post-processing by CASTEM 2000.

### Syntax :

```
"ECRITURE"
      < "COOR" > < "DEPL" > < "VITE" > < "ACCE" >
      < "FINT" > < "FEXT" > < "FLIA" >
      < "CONT" > < "EPST" > < "ECRO" >
      < "ENER" > < "MCVA" > < "MCVC" >
      < "MCVS" > < "FAIL" > < "VFCC" >
      /CTIM/

$[ "NOPOINT" ; "POINT" /LECTURE/ ]$
$[ "NOELEM" ; "ELEM" /LECTURE/ ]$

< "FICHIER" < FORMAT > "K2000" ndcast /CTIM/
      "POINT" /LECTURE/
      < "CHAMELEM" > >
```

#### "COOR"

Coordinates are printed on the EUROPLEXUS listing.

#### "DEPL"

Displacements are printed on the EUROPLEXUS listing.

#### "VITE"

Velocities are printed on the EUROPLEXUS listing.

#### "ACCE"

Accelerations are printed on the EUROPLEXUS listing.

#### "FINT"

Internal forces are printed on the EUROPLEXUS listing.

#### "FEXT"

Total external forces are printed on the EUROPLEXUS listing.

#### "CONT"

Stresses are printed on the EUROPLEXUS listing.

**"EPST"**

Total strains are printed on the EUROPLEXUS listing.

**"ECRO"**

Hardening parameters are printed on the EUROPLEXUS listing.

**"ENER"**

Energies are printed on the EUROPLEXUS listing.

**"MCVA"**

Printout of nodal quantities related to multicomponent fluids: pressure, density, temperature, sound speed and mass fractions. Note that this type of printout is incompatible with MCVC and MCVS.

**MCVC**

Printout of conserved variables (nodal quantities) related to multicomponent fluids: partial densities ( $\rho_i$ ) of the various components  $i$ , momentum ( $\rho \underline{u}$ ) (each spatial component separately), energy ( $\rho E$ ). Note that this type of printout is incompatible with MCVA.

**MCVS**

Printout of secondary variables (nodal quantities) related to multicomponent fluids: total density ( $\rho$ ), total pressure  $p$ , sound speed  $c$ , pressure derivative ( $\frac{\partial p}{\partial(\rho e)}$ ), absolute temperature ( $T$ ), pressure derivative ( $\frac{\partial p}{\partial(\rho_i)}$ ) for each component, mass fraction ( $\mu_i$ ) for each component. Note that this type of printout is incompatible with MCVA.

**"FAIL"**

Failure values are printed on the EUROPLEXUS listing.

**VFCC**

Printout at each selected output time of "element" quantities related to cell-centred Finite Volumes: various volume-related quantities and conserved variables.

**/CTIM/**

Reading procedure of the chosen time instants at which the results have to be printed on the listing. See page INT.57.

**"NOPOINT"**

Do not print any nodal variables. By default the chosen nodal variables are printed for all nodes stored in the results file.

**"POINT /LECTURE/"**

Print the chosen nodal variables only for the nodes defined in the /LECT/ (provided they are stored in the results file).

**"NOELEM"**

Do not print any element variables. By default the chosen element variables are printed for all elements stored in the results file.

"ELEM /LECTURE/"

Print the chosen element variables only for the elements defined in the /LECT/ (provided they are stored in the results file).

"FICH"

Produce a CASTEM 2000 results file from the EUROPLEXUS results file.

"FORMAT"

If this keyword is present, the CASTEM 2000 results file is formatted; else, it is unformatted (binary).

ndcast

Logical unit number of the CASTEM 2000 file; the results file is written with the standard SAUVER format of CASTEM 2000. It may be read by CASTEM 2000 by using the command RESTITUER. It is mandatory to specify the list of points for which results have to be included in the file, and if necessary also the word CHAMELEM.

/CTIM/

Reading procedure of the chosen time instants at which the results have to be stored. See page INT.57.

"POIN" /LECTURE/

List of the nodes for which the results are stored for a subsequent post-processing by CASTEM 2000. This directive is mandatory for a file of type "K2000".

"CHAMELEM"

This keyword causes the CHAMELEMS to be included in the CASTEM 2000 file. If it is omitted, the latter will only contain the selected CHAMPOINTS, on the nodes identified by the previous directive POINT.

### Comments :

The syntax is the same as for directive ECRI. For more details see page G.10 and following ones.

## 14.6 GRAPHIC OUTPUTS

### Object :

To produce drawings or lists on files (in a variety of formats) of different quantities in the form of curves with respect to time, or with respect to a curvilinear abscissa, or combined plots (e.g. sigma/epsilon graphs).

### Syntax:

```
"GRAP"  "AXTEMP"  coef  'nom_axe_0x'
        < "MINMAX" >
        < "PERFO"   'nom_fic'   >
        < "PERK"    'nom_fic'   >
        < "FENETRE" tmin tmax  >
        ( "COURBE"  . . . )
        ( "SCOURBE" . . . )
        ( "RCOURBE" . . . )
        ( "DCOURBE" . . . )
        ( "PCOURBE" . . . )
        ( "TRACE"   . . . )
        ( "XMGR"    . . . )
        ( "K2000"   . . . )
        ( "LISTE"   . . . )
        ( "FVAL"    . . . )
```

### coef

The time values are multiplied by **coef** (this e.g. enables the unit of measure to be changed).

### 'nom\_axe\_0x'

Name of the time axis (at most 16 characters), enclosed in apostrophes.

### MINMAX

Print on the EUROPLEXUS listing the minimum and the maximum values for each curve.

### PERFO

The value tables specified in the following **LISTE** directive will be output on an auxiliary file, whose name by default is **<base>.PUN**, where **<base>** is the base name of the current calculation. This directive allows to change the default name into the following **'nom\_fic'**.

### PERK

The value tables specified in the following **K2000** directive will be output on an auxiliary file, whose name by default is **<base>.PUK**, where **<base>** is the base name of the current calculation. This directive allows to change the default name into the following **'nom\_fic'**.

## FENETRE

Only the results in a given time interval (time window) are considered.

tmin

Minimum time (beginning of the time window).

tmax

Maximum time (end of the time window).

## COURBE

Define a curve representing the evolution **in time** of a certain variable in the current transient calculation. See below for the full details of this directive.

## SCOURBE

Define a curve representing the evolution **in space** of a certain variable in the current transient calculation. The space is a curvilinear abscissa ( $s$ ) defined by a sequence of nodes. The curve is by default built at the final time of the current calculation. To select a different time, use the **ARRET** directive described on page ED.40. See below for the full details of this directive.

## RCOURBE

Read in a curve representing the evolution **in time or in space** of a certain variable in a previously executed EUROPLEXUS calculation. The data are read in from a “punch” file produced by EUROPLEXUS via the **SORT LIST** directive, to be described below. In this way, results from different EUROPLEXUS runs may be compared on the same plot. See below for the full details of this directive.

## DCOURBE

Define a curve in the form of a table of  $(x, y)$  values. This allows e.g. to build a piecewise analytical solution to be compared with numerical results. It may even be used to input experimental results to be used as a reference solution. See below for the full details of this directive.

## PCOURBE

Define a *set of curves* for Pochhammer-Chree post-processing. See below for the full details of this directive.

## TRACE

Produce a graph containing one or more of the curves defined above, plotted either versus time, or versus space (curvilinear abscissa), or as a function of another curve (e.g.  $\sigma$ - $\epsilon$  type of plot). The graph is produced in the PostScript language on a file.

## XMGR

Same as **TRACE** but the graph data are stored on a file which may then be read by the XMGR program (a publicly available software) to produce the actual drawing.

## K2000

Same as TRACE but the graph data are stored on a file which may then be read by the CASTEM 2000 program to produce the actual drawing.

#### LISTE

Same as TRACE but the graph data are stored on a file which may then be read by a generic external tool to produce the actual drawing. The file format is very simple. This command also allows to store a curve in a certain EUROPLEXUS run and read it in (by the RCOURBE directive described above) in a subsequent EUROPLEXUS run, thus opening the way to the production of graphs containing comparisons of results from different EUROPLEXUS calculations, and even analytical curves or experimental data.

#### FVAL

Find values (abscissas)  $x$  of a curve for which the curve assumes a given value  $v$ , i.e. for which  $y(x) = v$ . Linear interpolation is used.

#### Comments:

- The time axis is the same for all drawings produced as a function of time.
- The time window is the same for all drawings produced as a function of time.

Example :

```
"GRAP"  "AXTEMP"  1000.  'TEMPS (MS)'  
"FENETRE"  0.    10E-3  MINMAX  
. . .
```



### 14.6.1 Post-processing in adaptivity

The post-processing of results in mesh adaptive computations may be somewhat different from the case of normal computations with a constant mesh connectivity. Some care is required in the interpretation of mesh adaptive results, especially as concerns time curves in selected nodes or elements.

The visualization of maps of values at a fixed time, e.g. a pressure field in the form of iso-values or a velocity field in the form of vectors, is similar to the case of non-adaptive calculations. The only thing to keep in mind is that, if a zone or part of the mesh must be selected for visualization, then the user should normally provide the list of the **base** elements. The code will then automatically replace these elements by the set of their active descendants at the chosen time. This mechanism is transparent to the user since object names and element group names always contain the indexes of the base elements concerned. Therefore, if no element indexes are explicitly given, the procedure from the user's viewpoint is exactly like in the case of non-adaptive computations.

The production of time curves in adaptive computations requires some more attention, since an element or node at which the results are to be extracted must be specified and this can be either a base item or a descendant item. Furthermore, the element or node can be active only during part of the time transient. The following rules are applied:

Rule 1: in an adaptive calculation, a **node-related** quantity

- represents the value belonging to the node itself, if the node is currently used. Note that in adaptivity nodes can be either used or unused. A used node is also active. An unused node is inactive. Base nodes are always used and therefore they are always active.
- is undefined (and is typically set to 0.0) if the node is currently not used.

Rule 2: in an adaptive calculation, an **element-related** quantity

- represents the value belonging to the element itself, if the element is currently active.
- represents the weighted average value of all its current active descendants, if the element is currently used but inactive. Note that, in particular, base elements (unlike base nodes) can become inactive, although they are always used.
- is undefined (and is typically set to 0.0) if the element is currently not used.

Rule 3: in an adaptive calculation, a **list of elements** or the name of an **object** or **group** made of elements

- represents the listed elements themselves, if such elements are currently active.
- represents the set of all current active descendants of the elements listed, if the elements are currently used but inactive.
- is illegal if the elements listed are currently not used.

### 14.6.2 Curve (Nodal Variables)

#### Object:

Definition of the variables **relative to nodes** to be drawn or listed.

#### Syntax :

```
"COURBE" nuco < 'nomcourbe' >

| [ "COOR" ; "DEPL" ; "VITE" ; "ACCE" ; "FINT" ; "FEXT" ;
    "FLIA" ; "ADFT" ; "MCPR" ; "MCRO" ; "MCTE" ; "MCMF" ;
    "MCUX" ; "MCUY" ; "MCUZ" ; "SIGN" ; "ECRN" ; "LFNO" ;
    "LFNV" ; "ILNO" ; "DTNO" ; "VITG" ; "NTLE" ; "MASN" ;
    "FDEC" ; "PFSI" ; "PFMI" ; "PFMA" ; "FORC" ] |

| [ "COMP" icomp ; "NORME" ] |

$[ "NOEU" /LECTURE/ ;
    "ZONE" /LECTURE/ ;
    "POSI" $[ x y <z> ;
        "FOLL" dx dy <dz> /LECT1/ ]$
    "OBJE" /LECT2/ ]$
```

#### nuco

Identifier of the curve (reference for TRAC, XMGR, K2000 or LISTE). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

#### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

#### COOR

Coordinate.

#### DEPL

Displacement.

#### VITE

Velocity.

#### ACCE

Acceleration.

FINT

Internal force.

FEXT

Total external force.

FLIA

External force due to coupled links (LINK COUP).

ADFT

Advection-diffusion temperature.

MCPR

Finite volume (MC) pressure.

MCRO

Finite volume (MC) density.

MCTE

Finite volume (MC) temperature.

MCMF

Finite volume (MC) component mass fraction.

MCUX

Finite volume (MC) fluid velocity along X computed from the conserved variable ( $u_x = (\rho u_x)/\rho$ ).

MCUY

Finite volume (MC) fluid velocity along Y computed from the conserved variable ( $u_y = (\rho u_y)/\rho$ ).

MCUZ

Finite volume (MC) fluid velocity along Z computed from the conserved variable ( $u_z = (\rho u_z)/\rho$ ).

SIGN

Spectral element stress.

ECRN

Spectral element internal variable.

LFNO

Logarithm in base 2 of the level factor associated with a node in the spatial time step partitioning algorithm.

LFNV

Logarithm in base 2 of the level factor associated with a node, including the neighbours in the spatial time step partitioning algorithm.

ILNO

Flag indicating whether a node is (1) or is not (0) subjected to a link condition, used in the spatial time step partitioning algorithm.

DTNO

Stability time step associated with a node, used in the spatial time step partitioning algorithm.

VITG

Grid velocity (ALE only).

NTLE

Node tree level (only in adaptivity).

MASN

Nodal mass.

FDEC

External force due to decoupled links (LINK DECO).

PFSI

Overpressure due to FSI in the nodes of CLxx elements associated with an **IMPE VISU** material (see Page C.885) and with either **COUP** or **DECO** specified. These CLxx elements, used only for results visualization purposes, must be attached to structural elements (typically shells) embedded in a fluid and subjected to either **FLSR** or **FLSW** model of FSI.

PFMI

Minimum FSI overpressure in time at the node (see **PFSI** above).

PFMA

Maximum FSI overpressure in time at the node (see **PFSI** above).

FORC

Total external force. This is an obsolescent alias of **FEXT**, still used in some old benchmarks. Use **FEXT** in new inputs.

COMP

Introduces the chosen component.

icom<sub>p</sub>

Component number. Default value is 1.

NORM

The norm of the considered vector (where applicable) is drawn.

**NOEU /LECTURE/**

Number of the node. The procedure /LECTURE/ allows if necessary to read a GIBI object, of which only the first node will be retained.

**ZONE /LECTURE/**

Set of nodal numbers defining a zone. The contributions of all these nodes are added together. This probably makes sense only for some types of variables (e.g. forces, masses etc.). This can be useful to plot e.g. the total (resultant) force acting on a set of nodes, or the total mass of such nodes. It is an alternative to the use of the **REGI** directive. The difference is that with **REGI** the region must be defined in the main calculation, and it cannot be defined when reading the results file (e.g. an Alice file). The present **ZONE** directive, on the contrary, can be defined “on the fly” when reading any results file (provided this file contains the results of all concerned nodes).

**POSI**

The nodal values should be extracted at the nearest node to the position specified next. Note that the nearest node may vary in time, either due to motion of the mesh or to mesh adaptivity. The position can either be specified by its coordinates (and in this case it is fixed in time), or by an offset to the position of a node in the mesh. In the latter case, if the specified node moves in time, then the position moves as well by “following” the specified node. This may be useful to track, say, the fluid velocity at a position slightly upstream of a deformable plate.

x y <z>

Coordinates of the position (fixed in time).

**FOLL /LECT1/**

The position should follow the node specified in the /LECT1/.

dx dy <dz>

Offset of the position with respect to the node.

**OBJE /LECT2/**

Object (list of **elements**) whose nodes are candidates for the search of the nearest node. If more than one node has the minimum distance from the position specified, then the first such node is retained. Note that although the variable to be drawn is relative to nodes, the object must be defined in term of (base) elements. The code then extracts automatically the nodes belonging to such elements (or to their active descendants, in case of adaptivity).

**Comments :**

The directive **COURBE** can be repeated as many times as desired, but each time with a different identifier.

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

The keyword `FORC` is accepted as a synonym of `FEXT` for backward compatibility, but is obsolescent and should not be used in new input files.

For an introduction to post-processing, and in particular to time curves production in mesh adaptive calculations, see page ED.65.

### 14.6.3 Curve (Element Variables)

#### Object:

Definition of the variables **relative to elements** to be drawn or listed.

#### Syntax:

```
"COURBE"  nuco  < 'nomcourbe' >

| [
  | [ "CONT" ; "ECRO" ; "EPST" ; "ENEL" ; "WAUX" ; "LFEL" ;
    "LFEV" ; "DTEL" ; "ELCE" ; "FAIL" ; "RISK" ; "CERR" ;
    "MAXC" ; "ERRI" ; "CLEN" ; "ILEN" ; "ETLE" ; "MASE" ] |

    "COMP"  icomp $[ "GAUS"  igaus ; "GAUZ"  igauz ]$ ;

  "VCVI"  |[ "COMP"  icomp ; "NORM" ]|
]|

$[ "ELEM" /LECTURE/ ;
  "ZONE" /LECTURE/ ;
  "POSI" $[ x y <z> ;
    "FOLL" dx dy <dz> /LECT1/ ]$
  "OBJE" /LECT2/          ]$
```

#### nuco

Identifier of the curve (reference for TRAC, XMGR, K2000 or LISTE). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

#### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

#### CONT

Stress tensor.

#### ECRO

Hardening quantity.

#### EPST

Total deformation tensor.

## ENEL

Internal energy.

## WAUX

Auxiliary energy terms for the element (see details below).

## LFEL

Logarithm in base 2 of the level factor associated with an element in the spatial time step partitioning algorithm.

## LFEV

Logarithm in base 2 of the level factor associated with an element including its neighbours in the spatial time step partitioning algorithm.

## DTEL

Stability time step  $\Delta t_{\text{stab}}$  associated with the element. The *stability step* is the *critical step*  $\Delta t_{\text{crit}}$  estimated by the code (roughly the element length  $L$  divided by the speed of sound  $c$  in the element material) multiplied by the *safety coefficient*  $\phi$  (CSTA, by default 0.8):  $\Delta t_{\text{stab}} = \phi \Delta t_{\text{crit}} \approx \phi \frac{L}{c}$ .

## ELCE

Coordinates of the barycentre of the element.

## FAIL

Failure level of the element.

## RISK

Risk level of the element (only if risk is activated). COMP must be given to define the kind of risk: COMP=1 chooses the risk of eardrum rupture, COMP=2 chooses the risk of death. Be aware that when reading results from an Alice file (produced by a previous calculation with risk activation), it is mandatory to (re-)specify the whole RISK directive (in particular as concerns the PROB ... and LUNG ... sub directives, see page A.30), because the risk is computed with the current values of the optional parameters.

## CERR

Constant used in element error indicator calculation (adaptivity), see the CERR input keyword of the ADAP directive on page B.210.

## MAXC

Maximum principal curvature of least-squares fitting function, used for element error indicator calculation (adaptivity).

## ERRI

Element error indicator (adaptivity).

## CLEN

Current characteristic element length used in element error indicator calculation (adaptivity).



**ILEN**

Optimal (indicated) characteristic element length resulting from error indicator calculations (adaptivity).

**ETLE**

Element tree level (adaptivity).

**MASE**

Element mass.

**VCVI**

Material or particle velocity (first idim components) in Finite Volumes Cell Centred model. Note that these vectors are not represented at the nodes but at the “elements” (i.e. Finite Volumes) centroids.

**COMP**

Introduces the component (unused for ENEL, LFEL, LFEV and DTEL).

**icomp**

Number of the component.

**GAUS**

Allows to choose a specific Gauss point index (only for the quantities CONT, EPST and ECRO).

**igauss**

Number of the Gauss point chosen. The special value 0 means that the average over all Gauss points in the element is taken. The default value is 1, i.e. if neither **GAUS** nor **GAUZ** is specified then the first Gauss Point of the specified element is taken. Note that this default is different from the default in rendering via OpenGL, where 0 (average over all Gauss Points) is assumed.

**GAUZ**

Allows to choose a specific “lamina” of the (shell) element. The value is the index of the lamina through the thickness (only for the quantities CONT, EPST and ECRO). In this case, the code takes the average value of all Gauss Points belonging to the specified lamina.

**igauz**

Number of Gauss point through the thickness (i.e. index of the chosen lamina).

**NORM**

The norm of the VCVI vector is drawn.

**ELEM /LECTURE/**

Number of the element. The procedure **/LECTURE/** allows if necessary to read a GIBI object, of which only the first element will be retained.

**ZONE /LECTURE/**

Set of element numbers defining a zone. The contributions of all these elements are added together. This probably makes sense only for some types of variables (e.g. masses). This can be useful to plot e.g. the total (resultant) mass of a set of elements. It is an alternative to the use of the **REGI** directive. The difference is that with **REGI** the region must be defined in the main calculation, and it cannot be defined when reading the results file (e.g. an Alice file). The present **ZONE** directive, on the contrary, can be defined “on the fly” when reading any results file (provided this file contains the results of all concerned elements).

#### POSI

The element values should be extracted at the nearest element (centroid) to the position specified next. Note that the nearest element may vary in time, either due to motion of the mesh or to mesh adaptivity. The position can either be specified by its coordinates (and in this case it is fixed in time), or by an offset to the position of a node in the mesh. In the latter case, if the specified node moves in time, then the position moves as well by “following” the specified node. This may be useful to track, say, the fluid pressure at a position slightly upstream of a deformable plate.

x y <z>

Coordinates of the position (fixed in time).

FOLL /LECT1/

The position should follow the node specified in the /LECT1/.

dx dy <dz>

Offset of the position with respect to the node.

OBJE /LECT2/

Object (list of **elements**) whose elements are candidates for the search of the nearest element. If more than one element has the minimum (centroid) distance from the position specified, then the first such element is retained. Note that the object must be defined in term of (base) elements. The code then extracts automatically the list of their active descendants, in case of adaptivity.

#### Comments:

The directive **COURBE** can be repeated as many times as desired, but each time with a different identifier.

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

If the keyword **GAUSS** is omitted, the first integration point is considered. If **GAUSS** is set to 0, the average over all integration points is used.

As concerns the auxiliary energy terms for the element (**WAUX**), the following components are available at the moment:

- WAUX 1  $\Rightarrow$  Energy dissipated by artificial viscosity (WARD)
- WAUX 2  $\Rightarrow$  Pressure work for fluids  $-PdV$  (W\_PDV)
- WAUX 3  $\Rightarrow$  Energy injected or lost at the walls (W\_INJ)
- WAUX 4  $\Rightarrow$  Kinetic energy for CCFV Elements on X axis
- WAUX 5  $\Rightarrow$  Kinetic energy for CCFV Elements on Y axis
- WAUX 6  $\Rightarrow$  Kinetic energy for CCFV Elements on Z axis
- WAUX 7  $\Rightarrow$  Total Energy for CCFV Elements
- WAUX 8  $\Rightarrow$  Used for saving initial injected energy
- WAUX 9  $\Rightarrow$  Total energy of the liquid phase (Only ADCR/ADCJ Model)
- WAUX 10  $\Rightarrow$  Total energy of the bubble (Only ADCR/ADCJ Model)
- WAUX 11  $\Rightarrow$  Total energy of the cover gas (Only ADCR/ADCJ Model)
- WAUX 12  $\Rightarrow$  Kinetic energy of the liquid phase (Only ADCR/ADCJ Model)
- WAUX 13  $\Rightarrow$  Kinetic energy of the bubble (Only ADCR/ADCJ Model)
- WAUX 14  $\Rightarrow$  Kinetic energy of the cover gas (Only ADCR/ADCJ Model)
- WAUX 15  $\Rightarrow$  Internal energy of the liquid phase (Only ADCR/ADCJ Model)
- WAUX 16  $\Rightarrow$  Internal energy of the bubble (Only ADCR/ADCJ Model)
- WAUX 17  $\Rightarrow$  Internal energy of the cover gas (Only ADCR/ADCJ Model)

Quantities 4 to 17 were added to account for auxiliary energies needed in FV cases. Contrary to FE, Total and kinetic energy FV contributions have to be computed at elements and not at nodes. Quantities 9 to 17 are only relevant for the ADCR/ADCJ Model, it will return 0 in other cases.

**Note:**

All printed energy quantities are equivalent in FE and FV cases, except for **W\_PDV** : In FE, the contribution of pressure work W\_PDV on the variation of kinetic energy is neglected in relation to the internal energy variation, which is relevant for smooth solutions. In FV, W\_PDV is computed as the variation of total energy, which is always the work of pressure forces for a stand-alone domain.

For an introduction to post-processing, and in particular to time curves production in mesh adaptive calculations, see page ED.65.

#### 14.6.4 Curve (Combinations)

##### Object:

Definition of **combinations** of the previously defined curves, to be drawn or listed.

##### Syntax:

```
"COURBE"  nuco  < 'nomcourbe' >

| [  "SOMME"      nbrs*( courbe_i  coef_i )      ;
     "PRODUIT"    pcoef  nbrp*( courbe_k )      ;
     "INTEGRALE"  courbe_i                      ;
     "DISTANCE"   /LECTURE/                     ;
     "LIBR"       ;                             ;
     "ADDC"  icou val      ;   "SUBC"  icou val      ;
     "MULC"  icou val      ;   "DIVC"  icou val      ;
     "EXPC"  icou val      ;   "CEXP"  icou val      ;
     "SHIFT" icou val      ;   "MOVE"  icou ival     ;
     "ADD"   icou jcou     ;   "SUB"   icou jcou     ;
     "MUL"   icou jcou     ;   "DIV"   icou jcou     ;
     "EXPF"  icou jcou     ;   "DUP"   icou jcou     ;
     "ABS"   icou         ;   "SEGN"  icou         ;
     "SQRT"  icou         ;   "INV"   icou         ;
     "EXP"   icou         ;   "LN"    icou         ;
     "LOG10" icou         ;   "SIN"   icou         ;
     "COS"   icou         ;   "ASIN"  icou         ;
     "ACOS"  icou         ;   "DIFF"  icou         ;
     "INT"   icou         ;   "AVER"  icou         ;
     "MAX"   icou         ;   "MIN"   icou         ;
     "MEAN"  nc*(icou)    ;   "SMAX"  nc*(icou)    ;
     "SMIN"  nc*(icou)    ;   "JOIN"  nc*(icou)    ;
     "FILT"  "MOYG"  icou nval                ] |
```

##### nuco

Identifier of the curve (reference for TRAC, XMGR, K2000 or LISTE). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

##### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

##### SOMME

The current curve results from the linear combination of **nbrs** curves, among those already defined.

---

$$\text{result} = \text{coef\_1} * \text{courbe\_1} \dots + \text{coef\_i} * \text{courbe\_i} + \dots$$
**PRODUIT**

The current curve results from the product of **nbrp** curves, among those already defined.

$$\text{result} = \text{pcoef} * \text{courbe\_1} \dots * \text{courbe\_k} * \dots$$
**INTEGRALE**

Each point of this curve is the value at time  $t$  of the integral between 0 and  $t$  of curve number **courbe\_i**, supposed already defined.

**DISTANCE**

The current curve results from the calculation of the distance between the two nodes specified by the following directive **/LECTURE/**.

**LIBR**

The variable concerned by this curve is computed by the subroutine **GRLIBR**, written by the user.

**ADDC**

Add to curve **icou** a constant value **val**.

**SUBC**

Subtract from curve **icou** a constant value **val**.

**MULC**

Multiply curve **icou** by a constant value **val**.

**DIVC**

Divide curve **icou** by a constant value **val**.

**EXPC**

Raise curve **icou** to a constant power **val**.

**CEXP**

Raise constant **val** to power values in curve **icou** (powers of a constant).

**SHIFT**

Translate of curve **icou** in its abscissa by a value **val**. Undefined values are set to zero. The abscissa of the generated curve is the same as that of curve **icou**.

**MOVE**

Translate of curve **icou** in its abscissa by a value **val**. The abscissa of the generated curve is no longer the same as that of curve **icou**, but it is shifted by the chosen amount **val**.

**ADD**

	Add curve <code>jcou</code> to curve <code>icou</code> .
SUB	
	Subtract curve <code>jcou</code> from curve <code>icou</code> .
MUL	
	Multiply curve <code>icou</code> by curve <code>jcou</code> .
DIV	
	Divide curve <code>icou</code> by curve <code>jcou</code> .
EXPF	
	Raise curve <code>icou</code> to power values contained in curve <code>jcou</code> .
DUP	
	Copy of curve <code>icou</code> having the abscissa of curve <code>jcou</code> . The result is set at zero in the non-overlapping abscissa zones.
ABS	
	Absolute value of curve <code>icou</code> .
SEGN	
	Sign (unit) function of curve <code>icou</code> .
SQRT	
	Square root of curve <code>icou</code> .
INV	
	Inverse of curve <code>icou</code> .
EXP	
	Exponential of curve <code>icou</code> .
LN	
	Natural logarithm of curve <code>icou</code> .
LOG10	
	Decimal logarithm of curve <code>icou</code> .
SIN	
	Sine of curve <code>icou</code> .
COS	
	Cosine of curve <code>icou</code> .
ASIN	
	Arc sine of curve <code>icou</code> .

**ACOS**

Arc cosine of curve `icou`.

**DIFF**

Derivative of curve `icou` with respect to its abscissa (usually time).

**INT**

Integral of curve `icou` with respect to its abscissa (usually time).

**AVER**

Average value of curve `icou`. This results in a single value, repeated over the whole abscissa.

**MAX**

Maximum value of a curve `icou`. This results in a single value, repeated over the whole abscissa.

**MIN**

Minimum value of a curve `icou`. This results in a single value, repeated over the whole abscissa.

**MEAN**

Arithmetic mean of a set of `nc` curves `icou`.

**SMAX**

Upper bound of a set of `nc` curves `icou`.

**SMIN**

Lower bound of a set of `nc` curves `icou`.

**JOIN**

Union of a set of `nc` curves `icou`. The values from each curve are merged together to form a new curve. This especially makes sense for “curves” consisting of just one point each, or for curves whose definition domains are disjoint.

**FILT MOYG**

Mobile average on `nval` consecutive values of curve `icou` .

**Comments:**

The directive **COURBE** can be repeated as many times as desired, but each time with a different identifier.

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

For **SOMME** and **PRODUIT**, the curves starting from which the sum (resp. product) is computed must have identifiers lower than that of the current curve and must have been already defined.

Commands **ADDC** to **SMIN** have been inspired from similar ones present in the **TPLOT** data management system, developed at **JRC** since the 1970's. For these commands, the curves identified by **icou**, **jcou**, etc., must have been already defined. Note also that for any of these commands that involve two or more curves **icou**, **jcou**, etc., with the notable exception of the **DUP** command, the abscissas (i.e. the discrete  $x$ -values) of all such curves must be identical, otherwise the combination may not be computed. Note also that, with respect to **TPLOT**, the meanings of **MIN**, **SMIN** and of **MAX**, **SMAX** have been interchanged. Moreover, **MIN** and **MAX** now produce a (uniform-valued) curve rather than the printout of a single value.

The subroutine **GRLIBR** allows to compute a quantity as a function of other quantities defined previously by a directive **COURBE**.



An example of such subroutine is:

### Programming example for GRLIBR:

```

      SUBROUTINE GRLIBR(TT,VAL,NT,NTEMAX)
C-----
C
C   CALCUL LIBRE DE GRANDEURS A TRACER EN FONCTION DU TEMPS
C
C-----
C           TT   = TABLEAU DES TEMPS (BANDE ALICE)
C           IT   = NUMERO DU PAS DE TEMPS
C           NT   = NOMBRE DE PAS DE TEMPS TOTAL (BANDE ALICE)
C           ICO  = NUMERO D'UNE COURBE
C   VAL(IT,ICO) = TABLEAU DES GRANDEURS DEFINIES PAR UNE COURBE
C           NTEMAX = NOMBRE MAXIMAL DE POINTS
C
      REAL TT, VAL
      DIMENSION TT(NTEMAX),VAL(NTEMAX,*)
C
C----  EXEMPLE : A = B * C
C      DO 10 IT=1,NT
C 10 VAL(IT,5)=VAL(IT,1)*VAL(IT,3)
C
C----  EXEMPLE d'INTEGRATION :
C      VAL(1,40)=0.
C      NT1=NT-1
C      DO 10 IT=1,NT1
C 10 VAL(IT+1,40)=VAL(IT,40)+0.5*(VAL(IT+1,22)+VAL(IT,22))
C      *      *(TT(IT+1)-TT(IT))
      RETURN
      END

```

Warning: the tables TT and VAL must be in simple precision (R\*4).

### 14.6.5 Curve (Regional Balances)

#### Object:

Definition of **quantities related to regions** to be drawn or listed.

#### Syntax:

```
"COURBE"  nuco  < 'nomcourbe' >
| [ "MASS" ; "VOLUME" ; "BARY" ; "VMOY" ; "VEMX" ; "VEMN" ; "DMOY" ;
    "DIMX" ; "DIMN" ; "AMOY" ; "ACMX" ; "ACMN" ; "IMPU" ; "ECIN" ;
    "EINT" ; "EEXT" ; "EPDV" ; "EINJ" ; "RESU" ; "IRES" ; "ECRG" ;
    "ECRM" ; "EMAS" ; "FLIR" ; "RRIS" ; "EPSM" ; "NERO" ; "NEND" ;
    "CERO" ; "CEND" ] |

$[ "COMP"  comp ; "NORM" ]$

"REGION"  nureg
```

#### nuco

Identifier of the curve (reference for TRAC, XMGR, K2000 or LISTE). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

#### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

#### MASS

Mass of the region (scalar, computed via XMEL).

#### VOLUME

Volume of the region (scalar).

#### BARY

Barycenter of the region (vector).

#### VMOY

Mean velocity of the region (vector).

#### VEMX

Maximum velocity (absolute) of the region (vector), only components 1 to 3.

VEMN

Minimum velocity (absolute) of the region (vector), only components 1 to 3.

DMOY

Mean displacement of the region (vector).

DIMX

Maximum displacement (absolute) of the region (vector), only components 1 to 3.

DIMN

Minimum displacement (absolute) of the region (vector), only components 1 to 3.

AMOY

Mean acceleration of the region (vector).

ACMX

Maximum acceleration (absolute) of the region (vector), only components 1 to 3.

ACMN

Minimum acceleration (absolute) of the region (vector), only components 1 to 3.

IMPU

Impulse (momentum) of the region (vector).

ECIN

Kinetic energy of the region (vector).

EINT

Internal energy of the region (scalar).

EEXT

Work of external forces applied to the region (scalar).

EPDV

Work of pressure forces (PdV) for the region (scalar).

EINJ

Energy injected in the region (scalar).

RESU

Resultant of the external forces applied to the region (vector).

IRES

Impulse due to external forces applied to the region (vector).

ECRG

Sum of the values of ECR on the Gauss points of the region (vector without norm).

ECRM

Average of the ECR over the region.

EMAS

Mass of the region (scalar, computed via the element masses XM0).

FLIR

Resultant of the force due to LINK/LIAI applied at the nodes.

RRIS

Average of the RISK over the region.

EPSM

Average of the EPST over the region.

NERO

Number of eroded elements over the region (See G.100).

NEND

Number of damaged elements over the region (See G.100).

CERO

Number of eroded classes over the region (See G.100).

CEND

Number of damaged classes over the region (See G.100).

COMP

Introduces the component.

icom<sub>p</sub>

Index of the component.

NORME

The norm of the chosen vector will be plotted.

nureg

Number of the concerned region in the order of definition.

**Comments:**

The directive **COURBE** can be repeated as many times as desired, but each time with a different identifier.

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

The directives **COMP** and **NORM** make sense only for vectors: they are possible with **BARY**, **DIMX**, **DIMN**, **VMOY**, **VEMX**, **VEMN**, **IMPU**, **ECIN**, **RESU**, and **IRES**. Furthermore, **NORM** does not make sense for **ECRG**, **ECRM**, **EPSM** and **RRIS** (only **COMP** is possible).

The directives **COMP** and **NORM** make no sense for scalars: **MASS**, **VOLU**, **EINT**, **EEXT**, **EPDV**, **EINJ**, **EMAS**, **NERO**, **NEND**, **CERO** and **CEND**.

If the keyword **COMP** is absent, it is the first component that is taken in the case of vectors.

The directive **EPDV** makes sense only for a stand-alone system, for example the fluid within a reservoir. In the remaining cases, it is suggested to use **EEXT**, which gives the work of the applied external forces.

**Note:**

In **FE**, the contribution of pressure work **W\_PDV** on the variation of kinetic energy is neglected in relation to the internal energy variation, which is relevant for smooth solutions. In **FV**, **W\_PDV** is computed as the variation of total energy, which is always the work of pressure forces for a stand-alone domain.

The directive **EINJ** is only valid for material **EAU**.

### 14.6.6 Curve (Global Quantities)

#### Object:

Definition of some **global quantities** to be drawn or listed, e.g. relative to energy balance or spatial time step partitioning.

#### Syntax:

```
"COURBE"  nuco  < 'nomcourbe' >

          | [      "WINT"                ;
                  "WEXT"                ;
                  "WCIN"                ;
                  "WTOT"                ;
                  "WIMP"                ;
                  "WSYS"                ;
                  "BILAN"               ;
                  "WSUM"  <COMP  icomp> ;
                  "DTMI"               ;
                  "DTMA"               ;
                  "MXSU"               ;
                  "DT1"                ;
                  "NSPL"               ;
                  "NUSP"               ;
                  "NSPT"               ;
                  "NUSE"               ;
                  "NACT"               ;
                  "NUSN"               ;
                  "LMAX"               ;
                  "LMIN"               ] |
```

#### nuco

Identifier of the curve (reference for TRAC, XMGR, K2000 or LISTE). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

#### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

#### WINT

Internal energy.

#### WEXT

	External work.
WCIN	
	Kinetic energy.
WTOT	
	Sum of all external energies (see comments below).
WIMP	
	Energy dissipated during contact/impact calculations.
WSYS	
	Total energy of the system (see comments below).
BILAN	
	Energy balance.
WSUM	
	Sum of the auxiliary energy terms, see below (vector).
DTMI	
	Minimum time increment in the time spatial step partitioning algorithm. This quantity is available only in calculations with partitioning (OPTI PART, see Page H.20).
DTMA	
	Maximum time increment in the time spatial step partitioning algorithm. This quantity is available only in calculations with partitioning (OPTI PART, see Page H.20).
MXSU	
	Logarithm in base 2 of the maximum depth of the time spatial step partitioning algorithm. This quantity is available only in calculations with partitioning (OPTI PART, see Page H.20).
DT1	
	Time integration step (scalar). This is the time increment that has led to the current time. However, at the initial time of the calculation (step 0, i.e. NPAS=0) this quantity does not make sense, so we take DT2 instead, i.e. the time increment that will lead to the following time.
NSPL	
	Number of elements which have been split during the current time step.
NUSP	
	Number of elements which have been unsplit during the current time step.
NSPT	
	Total number of elements which have been split during the calculation.

**NUST**

Total number of elements which have been unsplit during the calculation.

**NUSE**

Number of used elements (active or inactive) at the current time step.

**NACT**

Number of active elements at the current time step.

**NUSN**

Number of used (and also of active) nodes at the current time step.

**LMAX**

Maximum element level among all currently active elements at the current time step.

**LMIN**

Minimum element level among all currently active elements at the current time step.  
Level 0 (unused elements) is not considered in computing this quantity. Also currently used but inactive elements are excluded.

**i comp**

Index of the chosen component (only for vector quantities).

**Comments:**

The directive **COUR** can be repeated as many times as desired, but each time with a different identifier.

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

**WTOT** is the sum of all the “external” energies of the system: the work of external forces, the injected energy, the energy due to oil pyrolysis, etc. **WTOT** is used in the calculation of the energy balance.

**WIMP** is the energy dissipated due to contact-impact phenomena. This dissipation may come from the impact model used (soft impact, hard impact) in conjunction with the temporal discretization of the problem.

**WSYS** is the energy of the system, defined as:  $WSYS = WTOT + WIMP$ .  
In a closed system, **WSYS** must be conserved.

As concerns the global auxiliary energy terms (**WSUM**), the following components are available at the moment:

- **WSUM 1**  $\Rightarrow$  Energy dissipated by artificial viscosity (**WARD**)
- **WSUM 2**  $\Rightarrow$  Pressure work for fluids  $-PdV$  (**W\_PDV**)



- WSUM 3  $\Rightarrow$  Energy injected or lost at the walls (W\_INJ)
- WSUM 4  $\Rightarrow$  Kinetic energy for CCFV Elements on X axis
- WSUM 5  $\Rightarrow$  Kinetic energy for CCFV Elements on Y axis
- WSUM 6  $\Rightarrow$  Kinetic energy for CCFV Elements on Z axis
- WSUM 7  $\Rightarrow$  Total Energy for CCFV Elements
- WSUM 8  $\Rightarrow$  Used for saving initial injected energy
- WSUM 9  $\Rightarrow$  Total energy of the liquid phase (Only ADCR/ADCJ Model)
- WSUM 10  $\Rightarrow$  Total energy of the bubble (Only ADCR/ADCJ Model)
- WSUM 11  $\Rightarrow$  Total energy of the cover gas (Only ADCR/ADCJ Model)
- WSUM 12  $\Rightarrow$  Kinetic energy of the liquid phase (Only ADCR/ADCJ Model)
- WSUM 13  $\Rightarrow$  Kinetic energy of the bubble (Only ADCR/ADCJ Model)
- WSUM 14  $\Rightarrow$  Kinetic energy of the cover gas (Only ADCR/ADCJ Model)
- WSUM 15  $\Rightarrow$  Internal energy of the liquid phase (Only ADCR/ADCJ Model)
- WSUM 16  $\Rightarrow$  Internal energy of the bubble (Only ADCR/ADCJ Model)
- WSUM 17  $\Rightarrow$  Internal energy of the cover gas (Only ADCR/ADCJ Model)

Quantities 4 to 17 were added to account for auxiliary energies needed in FV cases. Contrary to FE, Total and kinetic energy FV contributions have to be computed at elements and not at nodes. Quantities 9 to 17 are only relevant for the ADCR/ADCJ Model, it will return 0 in other cases.

**Note:**

All printed energy quantities are equivalent in FE and FV cases, except for **W\_PDV** : In FE, the contribution of pressure work W\_PDV on the variation of kinetic energy is neglected in relation to the internal energy variation, which is relevant for smooth solutions. In FV, W\_PDV is computed as the variation of total energy, which is always the work of pressure forces for a stand-alone domain.

Note that the global quantities NSPL to LMIN in the above list are available only in calculations with adaptivity and with STAT option activated (OPTI ADAP STAT, see Page H.180).

### 14.6.7 Curve (Quantities from LOG file)

#### Object:

Definition of some quantities to be extracted from the LOG file, then drawn or listed.

#### Syntax:

```
LCOU  nuco  < 'nomcourbe' >
      < FICH 'nom_fich' >
      $ STEP ; TCPU ; DTCR ; ELCR ; DEE  ;
        DMMN ; DMME ; DTMX ; ELMX ; VMAX ;
        NVMX ; ELST ; MEMO ; MEMP $
      < NMAX nmax >
```

#### nuco

Identifier of the curve (reference for TRAC, XMGR, K2000 or LISTE). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

#### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

#### FICH 'nom\_fich'

Name of the log file from which data should be extracted. If omitted, the file `basename.log` is used, where `basename` is the base name of the current input file.

#### STEP

Step number.

#### TCPU

CPU time (in s).

#### DTCR

Critical time step.

#### ELCR

Critical element index.

#### DEE

Energy balance.

#### DMMN

Mass balance based on nodes.

**DMME**

Mass balance based on elements.

**DTMX**

Maximum elemental time step.

**ELMX**

Element index having the maximum elemental time step.

**VMAX**

Maximum velocity norm.

**NVMX**

Node (if  $> 0$ ) or Finite Volume (if  $< 0$ ) where the maximum velocity occurs.

**ELST**

Number of elements \* steps computed so far.

**MEMO**

Memory required.

**MEMP**

Memory peak so far.

**NMAX**

Maximum number of data points retained. If omitted, all data points present in the LOG file are retained.

### 14.6.8 Curve in space (Nodal Variables)

#### Object:

Definition of the variables **relative to nodes** to be drawn or listed **as a function of space** and not of time (as by default). The space is here represented by a curvilinear abscissa, which is built up starting by the definition of a sequence of nodes.

#### Syntax :

```
"SCOURBE"  nuco  < 'nomcourbe' >
            $[ "T" t ; "NPAS" npas ; "NSTO" nsto ]$
            "SAXE" scoe 'nom_saxe' <"INIT"> /LECTURE/

            |[ "COOR" ; "DEPL" ; "VITE" ; "ACCE" ; "FINT" ; "FEXT" ;
              "FLIA" ; "ADFT" ; "MCPR" ; "MCRO" ; "MCTE" ; "MCMF" ;
              "MCUX" ; "MCUY" ; "MCUZ" ; "SIGN" ; "ECRN" ; "LFNO" ;
              "LFNV" ; "ILNO" ; "DTNO" ; "VITG" ; "MASN" ]|

            |[ "COMP" icomp ; "NORME" ]|
```

#### nuco

Identifier of the curve (reference for TRAC, XMGR, K2000 or LISTE). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

#### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

#### t

Time of the desired storage station from which results have to be read in. If option STEP IO is active, then the code looks for the precise time *t* specified (within a small tolerance) among all stored time stations and, if no such time is found, an error message is issued. If option STEP LIBR is active, the code takes the first stored time station (if any) at a time equal to or greater than the specified time. Again, if no such time is found then an error message is issued.

#### npas

Time step number of the desired storage station from which results have to be read in.

#### nsto

Storage index number of the desired storage station from which results have to be read in.

## SAXE

Introduces the definition of the curvilinear abscissa to be used as  $x$ -axis for the curve.

## scoe

Multiplicative coefficient for the values of the curvilinear abscissa used as  $x$ -axis for the curve. By default, the abscissa is built up according to the distance between nodes, in the order they are defined in the following `/LECTURE/`.

## 'nom\_saxe'

Name of the curvilinear abscissa. This will appear on plots, etc.

## INIT

Build up curvilinear abscissa by using the *initial* nodal positions and not the *current* ones.

`/LECTURE/`

List of nodes defining the curvilinear abscissa. They are taken in the order given by the user (not re-ordered).

## COOR

Coordinate.

## DEPL

Displacement.

## VITE

Velocity.

## ACCE

Acceleration.

## FINT

Internal force.

## FEXT

Total external force.

## FLIA

External force due to liaisons (links).

## ADFT

Advection-diffusion temperature.

## MCPR

Finite volume (MC) pressure.

## MCRO

Finite volume (MC) density.

MCTE

Finite volume (MC) temperature.

MCMF

Finite volume (MC) component mass fraction.

MCUX

Finite volume (MC) fluid velocity along X computed from the conserved variable ( $u_x = (\rho u_x)/\rho$ ).

MCUY

Finite volume (MC) fluid velocity along Y computed from the conserved variable ( $u_y = (\rho u_y)/\rho$ ).

MCUZ

Finite volume (MC) fluid velocity along Z computed from the conserved variable ( $u_z = (\rho u_z)/\rho$ ).

SIGN

Spectral element stress.

ECRN

Spectral element internal variable.

LFNO

Logarithm in base 2 of the level factor associated with a node in the spatial time step partitioning algorithm.

LFNV

Logarithm in base 2 of the level factor associated with a node, including the neighbours in the spatial time step partitioning algorithm.

ILNO

Flag indicating whether a node is (1) or is not (0) subjected to a link condition, used in the spatial time step partitioning algorithm.

DTNO

Stability time step associated with a node, used in the spatial time step partitioning algorithm.

VITG

Grid velocity (ALE only).

MASN

Nodal mass.

**COMP**

Introduces the chosen component.

**icomp**

Component number.

**NORM**

The norm of the considered vector (where applicable) is drawn.

**Comments :**

The directive **SCOURBE** can be repeated as many times as desired, but each time with a different identifier. Identifiers should of course also be different from those of curves defined by the other curve-definition directives (**COURBE**, **RCOURBE**, **DCOURBE**).

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

If neither **T** nor **NPAS** nor **NSTO** are specified, then the last storage station is taken by default.

The keyword **FORC** is accepted as a synonym of **FEXT** for backward compatibility, but is obsolescent and should not be used in new input files.

### 14.6.9 Curve in space (Element Variables)

#### Object:

Definition of the variables **relative to elements** to be drawn or listed **as a function of space** and not of time (as by default). The space is here represented by a curvilinear abscissa, which is built up starting by the definition of a sequence of nodes.

#### Syntax :

```
"SCOURBE"  nuco  < 'nomcourbe' >
            $[ "T" t ; "NPAS" npas ; "NSTO" nsto ]$
            "SAXE" scoe 'nom_saxe' <"INIT"> /LECTURE/
            < "SUPP" /LECT_ELEM/ >

            |[ "CONT" ; "ECRO" ; "EPST" ; "ENEL" ; "WAUX" ; "LFEL" ;
              "LFEV" ; "DTEL" ; "CERR" ; "MAXC" ; "ERRI" ; "CLEN" ;
              "ILEN" ; "MASE" ]|

            "COMP" icomp |[ "GAUS" igauss ; "GAUZ" igausz ]|

            "VCVI" |[ "COMP" icomp ; "NORM" ]|
```

#### nuco

Identifier of the curve (reference for TRAC, XMGR, K2000 or LISTE). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

#### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

#### t

Time of the desired storage station from which results have to be read in. By default, the last storage station is taken.

#### npas

Time step number of the desired storage station from which results have to be read in. By default, the last storage station is taken.

#### nsto

Storage index number of the desired storage station from which results have to be read in. By default, the last storage station is taken.



**SAXE**

Introduces the definition of the curvilinear abscissa to be used as  $x$ -axis for the curve.

**scoe**

Multiplicative coefficient for the values of the curvilinear abscissa used as  $x$ -axis for the curve. By default, the abscissa is built up according to the distance between nodes, in the order they are defined in the following **/LECTURE/**.

**'nom\_saxe'**

Name of the curvilinear abscissa. This will appear on plots, etc.

**INIT**

Build up curvilinear abscissa by using the *initial* nodal positions and not the *current* ones.

**/LECTURE/**

List of nodes defining the curvilinear abscissa. They are taken in the order given by the user (not re-ordered).

**SUPP /LECT\_ELEM/**

The optional keyword **SUPP** allows to specify, via the following **/LECT\_ELEM/** directive, the geometrical support (list of the elements) to be considered for the projection onto nodes of the chosen element variable. By default, all elements of continuum, shell or beam type present in the mesh are considered. However, the default behaviour may lead to wrong results, for example in the case of shells whose nodes are merged with continuum fluid elements. If one traces, say, the pressure in the fluid, then also the (unrelated) value in the shell would be considered by default. To avoid the problem, specify **SUPP LECT fluid TERM**, where **fluid** is an object containing only the fluid elements. The **SUPP** directive should also be used in *adaptivity* for the definition of space curves (**SCOU**) involving elemental quantities, even in the absence of merged nodes. This allows to avoid ambiguities in the formation of the curvilinear abscissa starting from the base nodes, which are the only ones declared by the user.

**CONT**

Stress tensor.

**ECRO**

Hardening quantity.

**EPST**

Total deformation tensor.

**ENEL**

Internal energy.

**WAUX**

Auxiliary energy terms for the element (see details below).

## LFEL

Logarithm in base 2 of the level factor associated with an element in the spatial time step partitioning algorithm.

## LFEV

Logarithm in base 2 of the level factor associated with an element including its neighbours in the spatial time step partitioning algorithm.

## DTEL

Stability time step  $\Delta t_{\text{stab}}$  associated with the element. The *stability step* is the *critical step*  $\Delta t_{\text{crit}}$  estimated by the code (roughly the element length  $L$  divided by the speed of sound  $c$  in the element material) multiplied by the *safety coefficient*  $\phi$  (CSTA, by default 0.8):  $\Delta t_{\text{stab}} = \phi \Delta t_{\text{crit}} \approx \phi \frac{L}{c}$ .

## CERR

Constant used in element error indicator calculation (adaptivity), see the CERR input keyword of the ADAP directive on page B.210.

## MAXC

Maximum principal curvature of least-squares fitting function, used for element error indicator calculation (adaptivity).

## ERRI

Element error indicator (adaptivity).

## CLEN

Current characteristic element length used in element error indicator calculation (adaptivity).

## ILEN

Optimal (indicated) characteristic element length resulting from error indicator calculations (adaptivity).

## MASE

Element mass.

## VCVI

Material or particle velocity (first idim components) in Finite Volumes Cell Centred model. Note that these vectors are not represented at the nodes but at the “elements” (i.e. Finite Volumes) centroids.

## COMP

Introduces the component (unused for ENEL, LFEL, LFEV and DTEL).

## icomp

Number of the component.

## GAUSS

Introduces the Gauss point (only for the quantities CONT, EPST and ECRO).

igau

Number of Gauss point chosen.

GAUZ

Introduces the Gauss point through the thickness (only for the quantities CONT, EPST and ECRO).

igau

Number of Gauss point through the thickness.

NORM

The norm of the VCVI vector is drawn.

### Comments :

The directive SCOURBE can be repeated as many times as desired, but each time with a different identifier. Identifiers should of course also be different from those of curves defined by the other curve-definition directives (COURBE, RCOURBE, DCOURBE).

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

If the keyword GAUSS is omitted, the first integration point is considered. If GAUSS is set to 0, the average over all integration points is used.

As concerns the auxiliary energy terms for the element (WAUX), the following components are available at the moment:

- WAUX 1  $\Rightarrow$  Energy dissipated by artificial viscosity (WARD)
- WAUX 2  $\Rightarrow$  Pressure work for fluids  $-PdV$  (W\_PDV)
- WAUX 3  $\Rightarrow$  Energy injected or lost at the walls (W\_INJ)
- WAUX 4  $\Rightarrow$  Kinetic energy for CCFV Elements on X axis
- WAUX 5  $\Rightarrow$  Kinetic energy for CCFV Elements on Y axis
- WAUX 6  $\Rightarrow$  Kinetic energy for CCFV Elements on Z axis
- WAUX 7  $\Rightarrow$  Total Energy for CCFV Elements
- WAUX 8  $\Rightarrow$  Used for saving initial injected energy
- WAUX 9  $\Rightarrow$  Total energy of the liquid phase (Only ADCR/ADCJ Model)
- WAUX 10  $\Rightarrow$  Total energy of the bubble (Only ADCR/ADCJ Model)
- WAUX 11  $\Rightarrow$  Total energy of the cover gas (Only ADCR/ADCJ Model)

- WAUX 12  $\Rightarrow$  Kinetic energy of the liquid phase (Only ADCR/ADCJ Model)
- WAUX 13  $\Rightarrow$  Kinetic energy of the bubble (Only ADCR/ADCJ Model)
- WAUX 14  $\Rightarrow$  Kinetic energy of the cover gas (Only ADCR/ADCJ Model)
- WAUX 15  $\Rightarrow$  Internal energy of the liquid phase (Only ADCR/ADCJ Model)
- WAUX 16  $\Rightarrow$  Internal energy of the bubble (Only ADCR/ADCJ Model)
- WAUX 17  $\Rightarrow$  Internal energy of the cover gas (Only ADCR/ADCJ Model)

Quantities 4 to 17 were added to account for auxiliary energies needed in FV cases. Contrary to FE, Total and kinetic energy FV contributions have to be computed at elements and not at nodes. Quantities 9 to 17 are only relevant for the ADCR/ADCJ Model, it will return 0 in other cases.

**Note:**

All printed energy quantities are equivalent in FE and FV cases, except for **W\_PDV** : In FE, the contribution of pressure work W\_PDV on the variation of kinetic energy is neglected in relation to the internal energy variation, which is relevant for smooth solutions. In FV, W\_PDV is computed as the variation of total energy, which is always the work of pressure forces for a stand-alone domain.

### 14.6.10 Curve Read In from a File

#### Object:

Definition of curves to be read in from a file. The file must have been previously produced by EUROPLEXUS itself by means of the `SORT LIST` command, and is a file of type "PUNCH", see page ED.125.

#### Syntax :

```
"RCOURBE"  nuco      'nomcourbe'    FICH 'nom_fic'  
           <"RENAME" 'new_name'>  <"FACX"  fx>  <"FACY"  fy>
```

##### `nuco`

Identifier of the curve (reference for `TRAC`, `XMGR`, `K2000` or `LISTE`). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

##### `'nomcourbe'`

Name of the curve (reference for the user). This will appear on plots, etc. Unlike for the other curve definitions, this name is mandatory here and must exactly match the name by which the curve has been stored on the punch file during a previous EUROPLEXUS run.

##### `nom_fich`

Name of the punch file enclosed in apostrophes.

##### `RENA`

Allows to change the name of the curve if so desired.

##### `new_name`

New name of the curve enclosed in apostrophes.

##### `FACX`

Allows to change the  $x$ -scale of the curve if so desired.

##### `fx`

Multiplicative factor for the  $x$ -values.

##### `FACY`

Allows to change the  $y$ -scale of the curve if so desired.

##### `fy`

Multiplicative factor for the  $y$ -values.

**Comments :**

The directive `RCOURBE` can be repeated as many times as desired, but each time with a different identifier. Identifiers should of course also be different from those of curves defined by the other curve-definition directives (`COURBE`, `SCOURBE`, `DCOURBE`).

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

This directive allows to retrieve curves from different calculations and to compare them by plotting them on the same graph. The time scales and the number of points of the various curves are different in general. The program automatically takes this into account.

**Warning :**

A certain care should be taken concerning the units of measurement of curves stored and later retrieved for plotting. Note that curves are stored with exactly the  $x$ -values and the  $y$ -values as they would appear on a drawing. In particular, if the coefficients `AXTE coef`, see page ED.60 and `AXES coef`, see page ED.125, are not unitary, the stored values are multiplied by these coefficients.

When the data are subsequently read in by `RCOU`, the scaling is already included. So, plotting them by re-specifying again `AXTE coef` and/or `AXES coef` would probably not have the desired effect, since the coefficients would be applied twice! The results may be particularly confusing if the curves read from file are plotted together with “normal” curves (for which the coefficients are only applied once).

There is a simple way of avoiding this type of problem: when defining curves to be stored on file for subsequent plottings or comparisons, it is advisable to always specify `AXTE 1.0` and `AXES 1.0`. In this way all curves are saved with their “native” units of measurement. Any scale coefficients may be applied later, during the actual plotting phase.

In case of need, it is possible to assign a new name to a read-in curve by means of the `RENA` directive. This is the name that will appear on the plot legend. However, do not confuse this with the original name of the curve (`nomcourbe`) which must in any case exactly match the name stored in the file in order to select the desired curve.

A mechanism for changing the scales of a read-in curve both in  $x$  and in  $y$  is offered by means of the `FACX` and `FACY` directives. This is another way of overcoming the difficulties mentioned above concerning the scale factors. However, their use should be avoided whenever possible. The method outlined above of using unit factors at storage is cleaner and much preferable.

**Comments about the .pun file format:**

In case that the pun file should be used for external data the following format restrictions must be followed:

- The file starts with the word “VALUES”.

- The number of stored values is given between the character 7 and 12 of the first line.
- Characters 18:27 are the word COMPONENTS.
- Character 28 and 29 of the first line define the number of components.
- The next line consists from the name of the axe X (6:21) and axe Y (23:38).
- The next line contains the name of the curves (curve X 6:21 and curve Y 23:38). The last part is important since that is the name of the curve that must be defined in the RCOU command.
- After that, all values are defined by two fields of a length of 17 characters (2E17.6). Each couple must be written in a separate line.

### 14.6.11 Curve Defined By The User

#### Object:

Definition of curves in the form of tables containing a sequence of  $(x-y)$  values. These curves may represent a (piecewise) analytical solution, or even an experimental result, to be compared with numerical solutions by EUROPLEXUS.

The table containing the couples of values may be specified directly within the present directive, or refer to a function previously defined by the directive **FONCTION**, see page E.15, or represent an analytical solution to a perfect gas shock tube problem. The second possibility allows to plot and to visually check the functions which are defined and used in the calculation.

#### Syntax :

```
"DCOURBE"  nuco  < 'nomcourbe' >
            |[ npt*(x y) ;
              "FONC" ifonc ;
              "SHTU" "GAMM" gamm "ROM" rom "ROP" rop
                  "EINT" eint "LENM" lenm "LENP" lenp "TIME" time
                  "NRAR" nrar "VARI" vari ]|
```

#### nuco

Identifier of the curve (reference for **TRAC**, **XMGR**, **K2000** or **LISTE**). A (unique) integer number, freely chosen by the user, by which the curve may be successively referred to when needed.

#### 'nomcourbe'

Name of the curve (reference for the user). This will appear on plots, etc.

#### npt

Number of  $(x-y)$  couples defining the curve (i.e. number of points). In this case the values table is specified directly.

#### x

Value of the abscissa.

#### y

Corresponding value of the ordinate.

#### ifonc

Index of a function previously defined by the directive **FONCTION**, see page E.15.

#### SHTU



Introduces the parameters of the perfect gas shock tube problem for which the analytical solution (space curve of a chosen variable along the tube length) has to be generated. The high-pressure zone is assumed to be in the left part of the tube, of length `lenm`. The low-pressure zone is assumed to be in the right part of the tube, of length `lenp`. The initial specific energy (and hence the initial temperature) is the same in both parts. The initial density, and hence the initial pressure, is higher in the left part than in the right part of the tube.

`gamm`

Ratio  $\gamma$  between the specific heat  $C_p$  at constant pressure and the specific heat  $C_v$  at constant volume of the perfect gas.

`rom`

Initial density  $\rho_m$  in the left part of the tube (high-pressure zone).

`rop`

Initial density  $\rho_p$  in the right part of the tube (low-pressure zone).

`eint`

Initial specific energy  $i_0$  of the perfect gas.

`lenm`

Length  $l_m$  of the left part of the tube (high-pressure zone).

`lenp`

Length  $l_p$  of the right part of the tube (low-pressure zone).

`time`

Time  $t$  at which the analytical solution should be produced.

`nrar`

Number of spatial intervals  $n_r$  at which the analytical solution has to be computed in the rarefaction zone.

`vari`

Desired output variable for the analytical solution: 1 means pressure, 2 means density, 3 means specific internal energy, 4 means sound speed, 5 means velocity.

### Comments :

The directive `DCOURBE` can be repeated as many times as desired, but each time with a different identifier. Identifiers should of course also be different from those of curves defined by the other curve-definition directives (`COURBE`, `SCOURBE`, `RCOURBE`).

Curve identifiers may be freely chosen by the user, and the order in which they are given is irrelevant.

This directive allows to define arbitrary curves and to compare them with curves built up from EUROPLEXUS results by plotting them on the same graph. The time scales (or more generally the abscissas) and the number of points of the various curves are different in general. The program automatically takes this into account.

If a curve is specified by means of a function previously defined by the directive **FONCTION**, then:

- If the function is of type **TABL**, then the  $(x-y)$  values of the table function are directly used for the curve.
- If the function is of type **ROUT** (see user routine **TABANA**) or of type **LSQU** (least-squares fitting), then the  $x$ -values are not specified in the function. The code will try to use the stored time values for the current calculation as the  $x$ -values and to compute the corresponding  $y$ -values by calling the specified function.
- Other types of functions are not accepted at the moment and an error message is issued.

#### 14.6.12 Set of Pochhammer-Chree curves

##### Object:

Automatic generation of a *set of* curves for Pochhammer-Chree equation verification. The user must have previously read in results from a .POC file by means of the RESU directive, in addition to reading (global) results from an ordinary results file (typically an ALICE file).

##### Syntax :

```
"PCOURBE" "YOUN" youn "NU" nu "RHO" rho "R" r
          "NM" nm "IDOF" idof
          <"DHAR" dhar "TOL" tol "STEP" step "N1" n1 "AXTE" axte "FREQ" freq "M" m>
```

youn

Young's modulus of the bar material.

nu

Poisson's coefficient of the bar material.

rho

Density of the bar material.

r

Radius of the cylindrical bar.

nm

Number of the dispersive modes that will be calculated.

idof

Global dof along which the chosen variable is considered: 1 means radial direction, 2 means axial direction. A 2D axisymmetric calculation (with the bar axis directed vertically along the *y*-axis) is assumed.

dhar

Number of harmonics (or frequencies) that participate in the solution. In the case of the single harmonic excitation it should be 1. By default, 250 harmonics are taken.

tol

Relative error between an analytical and numerical solution. By default, it is 0.05.

step

Number of increments that will be used in the area of the relative error in order to identify a solution. By default, it is 200.

**n1**

Identifier (number) of the first generated curve. By default, it is 1.

**axte**

The name of the x-axis that will be used in the plotting of the curves. By default, it is 'RAD/WAVELENGTH'.

**freq**

The frequency of the excitation load (used only for the case of the single harmonic load).

**m**

The m ratio between the thickness of the shell and the mean radius of the hollow cylinder (in the case of hollow cylinder). By default, it is 0.

### Comments :

The directive **PCOURBE automatically** generates three sets of curves. The first set (ranging from **n1** to **n1 + nm - 1**) contains the analytical solutions, one for each chosen mode. These curves have the following names: **Mode\_1**, **Mode\_2** etc. The second set (ranging from **n1 + nm** to **n1 + 2\*nm - 1**) contains the numerical solutions, one for each chosen mode. These curves have the following names: **Nume\_1**, **Nume\_2** etc. The third set (ranging from **n1 + 2\*nm** to **n1 + 2\*nm + nlines\*dhar - 1**) contains the wavenumber spectrum for all the frequencies of interest for all the lines (parallel to the axis of the rod) of the calculation. The peaks on the wavenumber spectrum indicates the specific mode wavenumber for each frequency. These curves have the following name: **LINE\_1**, **LINE\_2** etc.

Note that any pre-existing curves with the same identifiers will be erased.

The phenomenon of dispersion is the reason why waves with different wavelengths will travel at different speed in the same material. The new module is dealing with the propagation of compressional waves in isotropic cylinders. It calculates the dispersion curves corresponding to each mode of propagation. The dispersion curves for each mode of propagation show the relationship between the phase velocity and the wavelength of a specific material. The procedure of defining those curves is described below in 7 steps

- An axial step function load is imposed in the circular face of the bar.
- Velocity versus time data are calculated and stored at equally spaced points along a predefined line in the axial direction of the bar.
- An FFT analysis is performed for each set of these time data in order to obtain the frequency spectrum for each point. This spectrum data are calculated and stored in order to be used in the next step.
- For each frequency, a history in the space domain across the predefined line is calculated and stored. This history can be calculated if the value of spectrum data is used for every point of the line for the desired frequency.

- By performing an FFT analysis on the space domain history across the predefined line, a wavenumber spectrum can be obtained for each frequency. These results corresponds to the third set of curves produced under **PCOURBE** directive.
- Each peak of the wavenumber spectrum corresponds to specific mode wavenumber for each frequency. The identification of the peaks for each frequency, leads to one point on the dispersion curves for each mode (for the modes that appeared in the desired frequency). Each peak indicates a wave number for the desired frequency and from this pair of values (wavenumber and frequency) the phase velocity and the wavelength of the mode can be defined. Lower modes peaks are located in higher wavenumbers.
- Finally the numerical results are compared with the analytical results of Pochhammer-Cree solution.

In the case where the **m** directive is defined, the hollow cylinder case is considered for the calculation of the analytical solution. The Mirksy-Herrman frequency equation is used in the case of the hollow cylinder. Also the user is encouraged to use the **hole** directive in order to define the inner radius of the hollow cylinder.

### 14.6.13 Drawings (TRACE)

#### Object:

This instruction is aimed at defining the drawings to be produced.

#### Syntax:

```
"TRACE" ( nuco ) $["PS" ; <"TEXT"> ; "MIF" ]$
          "AXES"      coef 'nom_axe_0y'
          <"XAXE" nxax coex 'nom_axe_0x'>
          <"COLO" (co)>
          <"THIC" (th)>
          <"DASH" (da)>
          < $[ "NOLI" ; "LINE" (li) ]$ >
          <"SYMB" <(sy)>> <"SYSC" sysc>
          <"NOXL" (nx)>
          <"NOYL" (ny)>
          <"XZER"> <"YZER">
          <"XGRD"> <"YGRD">
          <"XLOG"> <"YLOG">
          <"XMIN" xmin "XMAX" xmax $[ "DX" dx ; "NX" nx ]$>
          <"YMIN" ymin "YMAX" ymax $[ "DY" dy ; "NY" ny ]$>
```

#### nuco

Identifiers of the curves to be drawn (at most 12 curves).

#### PS

Draw on a PostScript file (this is the default).

#### TEXT

In addition to drawing on a PostScript file, also produce a list of the drawn data in tabular form (*x*-value, *y*-value) on a text file. The name of this file is <base>.txt, where <base> is the base name of the current calculation.

#### MIF

Draw on a MIF file. MIF is Adobe FrameMaker's language and may be suited to embed the graphics in a FrameMaker document. The drawing remains fully editable in FrameMaker (line style, colors, fonts etc.).

#### coef

Multiplying coefficient to change the units of the 0y axis.

'nom\_axe\_0y'

Name of the 0y axis (at most 16 characters).

**nxax**

Optional identifier of a curve to be used for the  $x$ -axis. By default, the drawing of the specified curves is done vs. time. However, by specifying the **XAXE** sub-directive, it is possible to produce a combined graph in which one or more quantities are plotted vs. another quantity rather than vs. time. For example, a  $\sigma$ - $\epsilon$  graph may be produced.

**coex**

Multiplying coefficient to change the units of the 0x axis.

**'nom\_axe\_0x'**

Name of the 0x axis (at most 16 characters).

**COLO**

Optional keyword that introduces the colors to be used for the various curves. If omitted, all curves are drawn in black.

**co**

Name of the color for the curve, (**not** enclosed in quotes). This must be repeated exactly as many times as there are curves in the drawing (see **nuco** above). The valid names are those of Cast3m, i.e. **bleu**, **roug**, **rose**, **vert**, **turq**, **jaun**, **blan** or **noir**.

**THIC**

Optional keyword that introduces the line thicknesses to be used for the various curves, in points. If omitted, all curves are drawn with a line thickness of 0.1 points.

**th**

Line thickness for the curve, in points. This must be repeated exactly as many times as there are curves in the drawing (see **nuco** above).

**DASH**

Optional keyword that introduces the dash patterns to be used for the various curves. If omitted, all curves are drawn as solid lines.

**da**

Code for the curve dash pattern. This must be repeated exactly as many times as there are curves in the drawing (see **nuco** above). Valid dash pattern codes are: 0 for a solid line, 1 for long dashes, 2 for medium dashes, 3 for short dashes, 4 for extra-short dashes, and 5 for long-short dashes.

**NOLI**

Do not draw any lines connecting points on (all) the curves.

**LINE**

Choose which curve(s) should be drawn as lines or not.

**li**

Code for the line connecting the curve points. This must be repeated exactly as many times as there are curves in the drawing (see **nuco** above). Valid line codes are: 0 means no line, 1 means line (with the chosen color, thickness and dash pattern, if any).

**SYMB**

Draw a symbol at each data point on each of the curves. The symbol is drawn *in addition* to the curve line. To remove the line (leaving only the symbols), use the **NOLI** or the **LINE** (**li**) keywords described above. To selectively choose which curves will get symbols, and/or the symbol used for each curve, specify the following (optional) sequence (**sy**). By default (no (**sy**) specified), symbol types 1 to 12 (see below) are used for curves 1 to 12.

**sy**

Code for the symbol drawn on each curve data point. If present, this must be repeated exactly as many times as there are curves in the drawing (see **nuco** above). Symbols are drawn with the same color and thickness as the associated curve. Valid symbol codes are: 0 no symbol, 1 plus, 2 cross, 3 square, 4 octagon, 5 triangle north, 6 triangle south, 7 triangle east, 8 triangle west, 9 hourglass, 10 hourglass horizontal, 11 diamond, 12 Y, 13 Z.

**SYSC**

Introduce a symbol scaling factor **sysc**. By default, the factor is 1.0.

**NOXL**

Optional keyword that introduces the definition of whether or not the various curves participate in the definition of the  $x$ -axis (automatic search of the limits and of the major and minor subdivisions). If omitted, all curves participate in the definition of the  $x$ -axis.

**nx**

Code for the curve participation in the definition of the  $x$ -axis. This must be repeated exactly as many times as there are curves in the drawing (see **nuco** above). Valid codes are: 0 means that the curve participates in the definition of the axis, 1 means that the curve is ignored in definition of the axis.

**NOYL**

Optional keyword that introduces the definition of whether or not the various curves participate in the definition of the  $y$ -axis (automatic search of the limits and of the major and minor subdivisions). If omitted, all curves participate in the definition of the  $y$ -axis.

**ny**

Code for the curve participation in the definition of the  $y$ -axis. This must be repeated exactly as many times as there are curves in the drawing (see **nuco** above). Valid codes are: 0 means that the curve participates in the definition of the axis, 1 means that the curve is ignored in definition of the axis.

**XZER**

Draw a vertical dotted line in correspondence of the abscissa  $x = 0$ .

**YZER**

Draw a horizontal dotted line in correspondence of the ordinate  $y = 0$ .



**XGRD**

Draw vertical grid lines at every major axis tick.

**YGRD**

Draw horizontal grid lines at every major axis tick.

**XLOG**

Use a logarithmic (10-base) scale for the  $x$ -axis instead of a linear scale. Obviously, all  $x$ -values must be strictly positive.

**YLOG**

Use a logarithmic (10-base) scale for the  $y$ -axis instead of a linear scale. Obviously, all  $y$ -values must be strictly positive.

**XMIN**

Use the specified lower limit for the  $x$ -axis instead of computing it automatically.

**XMAX**

Use the specified upper limit for the  $x$ -axis instead of computing it automatically.

**DX**

Use the specified scale increment for the  $x$ -axis instead of computing it automatically.

**NX**

Use the specified number of increments for the  $x$ -axis instead of computing it automatically.

**YMIN**

Use the specified lower limit for the  $y$ -axis instead of computing it automatically.

**YMAX**

Use the specified upper limit for the  $y$ -axis instead of computing it automatically.

**DY**

Use the specified scale increment for the  $y$ -axis instead of computing it automatically.

**NY**

Use the specified number of increments for the  $y$ -axis instead of computing it automatically.

**Comments:**

The instruction **TRAC** may be repeated as many times as desired.

It is possible to use the same curve (same identifier) for several drawings.

Normally the axes scales are computed automatically. However, the user may take full control of this process by specifying **XMAX** ... and / or **YMAX** .... When specifying a lower bound also the corresponding upper bound and either the increment or the number of increments must be specified as well.

Examples:

```
"TRAC"    1 4 2  "AXES"  1.  'PRESSION (PA) '
"TRAC"      1 2  "AXES" 1E-6 'PRESSION (MPA)'
"TRAC" 6 "AXES" 1E-6 'STRESS (MPA)' "XAXE" 5 1.0 'STRAIN'
```

#### 14.6.14 Output on file (XMGR)

##### Object:

Definition of the variables to be printed on the auxiliary files directly readable by the XMGR software (Copyright Paul J. Turner). See also the directive PERK on page ED.60, which allows to change the default name of the output file.

##### Syntax:

```
"XMGR"    (  nuco  )    "AXES"      coef 'nom_axe_0y'  
          <"XAXE" nxax coex 'nom_axe_0x'>
```

##### nuco

Identifiers of the curves to be printed (at most 12 curves).

##### coef

Multiplying coefficient to change the units of the 0y axis.

'nom\_axe\_0y'

Name of the 0y axis (at most 16 characters).

##### nxax

Optional identifier of a curve to be used for the  $x$ -axis. By default, the drawing of the specified curves is done vs. time. However, by specifying the XAXE sub-directive, it is possible to produce a combined graph in which one or more quantities are plotted vs. another quantity rather than vs. time. For example, a  $\sigma$ - $\epsilon$  graph may be produced.

##### coex

Multiplying coefficient to change the units of the 0x axis.

'nom\_axe\_0x'

Name of the 0x axis (at most 16 characters).

##### Comments:

The XMGR directive may be repeated as many times as needed.

The use of this directive is identical to that of directive TRACE. It is possible to combine them by using the same curves:

Example:

```
"TRACE"      1 4 2  "AXES"  1.  'PRESSION (Pa)'  
"XMGR"       4 2  "AXES"  1.  'PRESSION (Pa)'
```

The files created for XMGR have names of the form: <base\_xxx>.MGR, where <base> is the base name of the current calculation and xxx is a counter. A separate file is produced for each XMGR directive. If no base name is available, then the file name becomes TRACXMGR\_xxx.MGR.

It is possible to use the same curve (same identifier) for more than one list.

Examples:

```
"XMGR"      1 4 2  "AXES"  1.  'PRESSION (Pa)'  
"XMGR"      1 2  "AXES" 1E-6 'PRESSION (MPa)'  
"XMGR" 6 "AXES" 1E-6 'STRESS (MPa)' "XAXE" 5 1.0 'STRAIN'
```

### 14.6.15 Output on file (K2000)

**Object:**

Definition of the variables to be printed on an auxiliary file directly readable by the CASTEM 2000 software. See also the directive **PERF** on page ED.60, which allows to change the default name of the output file.

**Syntax:**

```
"K2000"    (  nuco  )    "AXES"      coef 'nom_axe_0y'  
          <"XAXE" nxax coex 'nom_axe_0x'>
```

**nuco**

Identifiers of the curves to be printed (at most 12 curves).

**coef**

Multiplying coefficient to change the units of the 0y axis.

'nom\_axe\_0y'

Name of the 0y axis (at most 16 characters).

**nxax**

Optional identifier of a curve to be used for the *x*-axis. By default, the drawing of the specified curves is done vs. time. However, by specifying the **XAXE** sub-directive, it is possible to produce a combined graph in which one or more quantities are plotted vs. another quantity rather than vs. time. For example, a  $\sigma$ - $\epsilon$  graph may be produced.

**coex**

Multiplying coefficient to change the units of the 0x axis.

'nom\_axe\_0x'

Name of the 0x axis (at most 16 characters).

**Comments:**

The K2000 directive may be repeated as many times as needed.

The use of this directive is identical to that of directive **TRACE**. It is possible to combine them by using the same curves:

Example:

```
"TRACE"      1 4 2  "AXES"  1.  'PRESSION (Pa) '  
"K2000"      4 2  "AXES"  1.  'PRESSION (Pa) '
```

The formatted file may be directly inserted in the input data for CASTEM 2000. The contained objects are of type "LISTREEL", and the names are "L\_TEMPS" for the time and "L\_number" for the curves (**number** is the curve identifier).

It is possible to use the same curve (same identifier) for more than one list.

Examples:

```
"K2000"      1 4 2  "AXES"  1.  'PRESSION (Pa) '  
"K2000"      1 2  "AXES" 1E-6 'PRESSION (MPa) '  
"K2000" 6 "AXES" 1E-6 'STRESS (MPa)' "XAXE" 5 1.0 'STRAIN'
```

### 14.6.16 Output on file (LIST)

#### Object:

Definition of the variables to be printed on an auxiliary file of type "PUNCH" (see also the directive PERF).

#### Syntax:

```
"LISTE"    (  nuco  )    "AXES"        coef 'nom_axe_0y'  
              <"XAXE" nxax coex 'nom_axe_0x'>
```

##### nuco

Identifiers of the curves to be printed (at most 12 curves).

##### coef

Multiplying coefficient to change the units of the 0y axis.

'nom\_axe\_0y'

Name of the 0y axis (at most 16 characters).

##### nxax

Optional identifier of a curve to be used for the  $x$ -axis. By default, the drawing of the specified curves is done vs. time. However, by specifying the XAXE sub-directive, it is possible to produce a combined graph in which one or more quantities are plotted vs. another quantity rather than vs. time. For example, a  $\sigma$ - $\epsilon$  graph may be produced.

##### coex

Multiplying coefficient to change the units of the 0x axis.

'nom\_axe\_0x'

Name of the 0x axis (at most 16 characters).

#### Comments:

The LISTE directive may be repeated as many times as needed.

The use of this directive is identical to that of directive TRACE. It is possible to combine them by using the same curves:

Example:

```
"TRACE"      1 4 2  "AXES"  1.  'PRESSION (Pa)'
"LISTE"       4 2  "AXES"  1.  'PRESSION (Pa)'
```

The tables come out as **nbco** blocks of NT lines with two numbers ( $x$ - $y$  values) each each. The first value is the abscissa (by default the time), and the second value is the corresponding ordinate ( $y$ -value). Each block therefore fully describes one curve. Blocks are given in the same order as they appear in directive **LISTE**.

To facilitate the subsequent reading of these tables, each block is preceded by three description lines:

- On the first line, after the word **VALEURS** there is **NT**, the number of lines of the block, that is also the number of  $x$ - $y$  couples. Then comes the word **COMPOSANTES**, followed by the number of  $y$ -value columns (this number is always 1).
- On the second line, which starts by **\***, are given the names of the **Ox** axis and of the **Oy** axis (string **nom\_axe\_Oy** of directive **AXES**).
- On the third line, which also starts by **\***, are given the names of the curves (**nomcourbe**) defined in **COURBE**.

It is possible to use the same curve (same identifier) for more than one list.

Examples:

```
"LISTE"      1 4 2  "AXES"  1.  'PRESSION (Pa)'
"LISTE"       1 2  "AXES" 1E-6 'PRESSION (MPa)'
"LISTE"  6 "AXES" 1E-6 'STRESS (MPa)' "XAXE" 5 1.0 'STRAIN'
```

### Warning :

A certain care should be taken concerning the units of measurement of curves stored and later retrieved for plotting. Note that curves are stored with exactly the  $x$ -values and the  $y$ -values as they would appear on a drawing. In particular, if the coefficients **AXTE coef**, see page ED.60 and **AXES coef**, see above, are not unitary, the stored values are multiplied by these coefficients.

When the data are subsequently read in by **RCOU**, the scaling is already included. So, plotting them by re-specifying again **AXTE coef** and/or **AXES coef** would probably not have the desired effect, since the coefficients would be applied twice! The results may be particularly confusing if the curves read from file are plotted together with “normal” curves (for which the coefficients are only applied once).

There is a simple way of avoiding this type of problem: when defining curves to be stored on file for subsequent plottings or comparisons, it is advisable to always specify **AXTE 1.0** and **AXES 1.0**. In this way all curves are saved with their “native” units of measurement. Any scale coefficients may be applied later, during the actual plotting phase.



**14.6.17 Find value on a curve (FVAL)****Object:**

Find values (abscissas)  $x$  of a curve for which the curve assumes a given value  $v$ , i.e. for which  $y(x) = v$ . Linear interpolation is used. All found values of  $x$  are printed on the listing.

If  $y(x_n) \leq v \leq y(x_{n+1})$ , or  $y(x_n) \geq v \geq y(x_{n+1})$ , then the value of  $x$  in the interval from  $x_n$  to  $x_{n+1}$  is interpolated linearly by the expression:

$$x = x_n + (x_{n+1} - x_n) \frac{v - y_n}{y_{n+1} - y_n} \quad (58)$$

**Syntax:**

```
"FVAL" nuco val
```

```
nuco
```

Identifier of the curves to be examined.

```
val
```

Value to be sought on the curve.

**Comments:**

Note that, like in the case of minimum and maximum values (MINM) of curves, the found values (if any) are printed on the listing only when the corresponding curve is drawn via the TRAC command.

Therefore, in order to get the desired values actually printed make sure to first use the FVAL directive for the desired curve(s) and then let the curve number appear in at least one TRAC directive. For example:

```
SORT GRAP ...
. . .
COUR 1 ...
COUR 3 ...
COUR 23 ...
. . .
FVAL 1 3.14 ! search value 3.14 on curve #1
FVAL 23 -1.0 ! search value -1.0 on curve #23
. . .
TRAC 23 ...
FIN
```

In the above example, the value search for  $-1.0$  in curve 23 is printed on the listing, but the search for value 3.14 in curve 1 is not printed.

## 14.7 VISUALIZATIONS

### Object:

To produce, by reading results stored in the results file, (a subset of) the visualizations that are possible during direct execution of the code (see Pages A.25 and O.10). These include graphical rendering interactively in a window or in batch mode on file and production of animations. Not all visualization types and features are available, though (see below for details).

### Syntax:

```
( "VISU" $ "T" t ; "NPAS" npas ; "NSTO" nsto $  
  <PLAY>  
  <sequel of interactive commands, see pages A.25 and O.10>  
  <ENDPLAY>  
)
```

**t**

Time of the desired (initial) storage station from which results have to be read in. Subsequent time stations may then be reached by suitable commands (e.g. **GO** and **FREQ**) in the **PLAY ... ENDPLAY** sequence.

**npas**

Time step number of the desired (initial) storage station from which results have to be read in. Subsequent time stations may then be reached by suitable commands (e.g. **GO** and **FREQ**) in the **PLAY ... ENDPLAY** sequence.

**nsto**

Storage index number of the desired (initial) storage station from which results have to be read in. Subsequent time stations may then be reached by suitable commands (e.g. **GO** and **FREQ**) in the **PLAY ... ENDPLAY** sequence.

**PLAY**

Introduces a sequel of “interactive” commands (see pages A.25 and O.10) that are read subsequently from the input file rather than from the keyboard.

**ENDP**

Terminates the sequel of “interactive” commands (see pages A.25 and O.10) that are read subsequently from the input file rather than from the keyboard.

### Comments:

As indicated by the parentheses in the above syntax, the **VISU** sub-directive may be repeated as many times as needed within the **SORT** directive (see Page ED.40). However, only one **SORT** directive is allowed within each input data set.

Repetition of the **VISU** sub-directive (without repeating **SORT**) may be useful e.g. to **step back** in the ALICE file, i.e. to go to a previously saved time step. To **step forth** in the ALICE file, simply use the **GO** and **FREQ** commands in the **PLAY ... ENDPLAY** sequence, as mentioned above.

The options **T**, **NPAS** and **NSTO** are mutually exclusive. Exactly one of them must be specified, in order to position the read cursor of the storage file at the initial storage position of interest. Following storage positions may then be accessed by the “interactive” commands if so desired (e.g. to produce an animation).

So-called “interactive” commands such as **TRAC** may then be issued from the keyboard. Alternatively, they may be embedded in the input file by enclosing them in the pair of keywords **PLAY ... ENDPLAY**.

The read cursor may be advanced by means of the **GO** command. In this case, however, the frequency **FREQ** counts the storage stations rather than the time steps. To terminate the execution of interactive commands (when typing them actually at the keyboard) and to return control to the input file, use the **ENDP** command.

### Warnings:

Note that not all the visualization features described in pages A.25 and O.10 for direct execution of the code are available when visualizing results from a results file. Most restrictions come from the fact that the results file (typically an ALICE file) does not contain all the information that is available during direct execution.

For example, the following features will not work:

- Visualization of thicknesses.
- Visualization of material-related data.
- Etc. etc.

Note also that, although the **RESU** directive allows to read data from several types of results files, not all of them are suitable for visualizations. For example, an **ALIC TEMP** results file typically contains only very limited information (just a few nodes and elements) and therefore it is suitable for the production of graphs (time curves) but not of visualizations involving the whole mesh.

## 15 GROUP O—INTERACTIVE COMMANDS

### Object

This Section describes all the interactive commands which can be issued during a foreground (interactive) execution of the code. To launch EUROPLEXUS in interactive mode, include the CONV keyword at the beginning of the input file, as described in Section 4.2 (Page A.25).

When interactive execution is chosen, EUROPLEXUS reads the input data-set as usual, performs step 0 to initialise the computation, then prompts the user for commands from the keyboard with the phrase: `COMMANDE ?`.

The user can then issue various commands and subcommands **typically from the keyboard** in order to pilot the computation. For example, he can ask the program to perform a certain number of steps, then to pause again for further commands. Each time the calculation is paused, the current computational model can be visualized (e.g. by means of the built-in OpenGL-based visualization module) and information concerning the computation (time step, CPU time, etc.) can be printed. Furthermore, the current time step can be varied by the user.

As an alternative to typing commands by hand from the keyboard, such commands may be included in the regular EUROPLEXUS input file by enclosing them into a special directive `PLAY ... ENDPLAY` as described in Section 13.6 (Page I.24) and in Section 14.7 (Page ED.140). For example, this may be useful when the “interactive” command have to be saved for later re-execution, or when the command sequence is particularly complex, e.g. for the production of an animated visualization sequence.

### COMMANDS

The following Sections list all the available “interactive” commands:

- Section 15.1 lists all primary interactive commands;
- Section 15.2 lists all the keystrokes and mouse events which are interpreted as commands in the built-in OpenGL-based graphical visualizer;
- Section 15.3 lists all `CALCUL` commands, used to pilot the current time increment;
- Section 15.4 lists all `TRACE` commands, used to visualize the current results;
- Section 15.5 lists all `MAVI` commands, used to produce an animation AVI file from a sequence of images previously produced by EUROPLEXUS itself;
- Section 15.6 lists all `GOTRAC` commands, used to activate a simple looping mechanism, useful e.g. in animation production;
- Section 15.7 lists all `CAMERA` commands, used to set the camera used for visualization;
- Section 15.8 lists all `SLERP` commands, used to set the motion of the camera used for visualization;
- Section 15.9 lists all `SCENE` commands, used to set all the details of a visualization;
- Section 15.10 lists all `TITLES` commands, used to insert titles in an animation.

## 15.1 Primary interactive commands

### Object

To pilot a calculation interactively. Note, however, that it is also possible to store such commands within the regular EUROPLEXUS input file (rather than typing them at the keyboard) and then to execute them by means of the **PLAY** directive, described on Page I.24. In this way, the unique functionalities offered by the “interactive” commands become available also for unattended code execution, allowing e.g. to automatize the production of graphics or animated sequences.

### Syntax

Here is the syntax of interactive commands and subcommands:

```

$ "?"                                     $
$ "GO"                                   $
$ "STOP"                                $
$ "INFO"                                 $
$ "FREQ" npas                           $
$ "TFRE" tfreq                           $
$ <$ HPIN ; NOHP $>                       $
$ "CALC" $ "?" $                         $
$      $ "AUTO" $                       $
$      $ "UTIL" $                       $
$      $ "DT" tstep $                   $
$      $ "R" $                           $
$ "TRAC" $ "?" $                         $ $
$      $ "NORM" $                       $ $
$      $ "ELEM" i1 i2 $                 $ $
$      $ "ZOOM" $ "?" $                 $ $
$      $      $ "POIN" xmin ymin xmax ymax $ $
$      $      $ "RETI" $                 $ $
$      $      $ "R" $                     $ $
$      $ "OEIL" xoeil yoeil zoeil $ $
$      $ "CULL" $                       $ $
$      $ "NOCU" $                       $ $
$      $ "NUME" $ "NOEU" $               $ $
$      $      $ "ELEM" $                 $ $
$      $ "NONU" $                       $ $
$      $ "R" $                           $ $
$      $ "CDEP" $                       $ $
$      $ "CDNO" $                       $ $
$      $ <$ "FAIL" ; "NFAI" ; "FANF" $> $ $
$      $ "OBJE" /LECT/ <SURF ; FSIN /LECT/> $ $
$      $ "NOOB" "NOGR" "OMEM" $         $ $
$      $ "PINB" $                       $ $
$      $ "PINC" $                       $ $

```

```

$      $ "DEFO"                                     $ $
$      $ "AMPD" ampd                               $ $
$      $ "VITE"                                     $ $
$      $ "VITG"                                     $ $
$      $ "AMPV" ampv                               $ $
$      $ "FEXT"                                     $ $
$      $ "FINT"                                     $ $
$      $ "AMPF" ampf                               $ $
$      $ "DASH" idsh                               $ $
$      $ "AVS" | "DEPL" "VITE" "FEXT" "ACCE" "MCXX" $ $
$      $      "VITG" "FINT" "CONT" "EPST" "ECRO"   $ $
$      $      "ECRC" /LECT/                         | $ $
$      $ "PS"                                       $ $
$      $ "MIF"                                       $ $
$      $ "POVR"                                       $ $
$      $ "P10"                                       $ $
$      $ < <"OFFS" <"SIZE" w h> <$ "RPOV" ; "BPOV"$> $ $
$      $      <"ZIP"> <"FICH" $ "BMP"              $ $
$      $      $ "PPM"                               $ $
$      $      $ "PPMA"                             $ $
$      $      $ "TGA"                               $ $
$      $      $ "EPS"                               $ $
$      $      $ "EPSB"                             $ $
$      $      $ "PRAY"                             $ $
$      $      $ "AVI" <pars> $ <'base'> >         $ $
$      $ <"SYMX"> <"SYMY"> <"EXTZ" nz dz>          $ $
$      $ <"AXIS" na ang> <"NOSY">                  $ $
$      $ <"SAVE"> <"REUS">                          "REND" > $ $
$ <Keystrokes and mouse events (OpenGL visualizer): see Page 0.15> $
$ "MAVI" <"DUMP"> <"FROM" 'base'> <"UZIP"> <"RZIP"> <"TO" 'to'> $
$      <"FIRS" firs> <"LAST" last> <"STEP" step"> <pars> "REND" $
$ "CAME" icam < "EYE" ex ey ez >                   $
$      < $ "Q" qr qx qy qz                          $ $
$      $ "VIEW" vz vy vz "RIGH" rx ry rz "UP" ux uy uz $ > $
$      < FOV fovy >                                 $
$ "LCAM"                                             $
$ "SLER" "CAM1" ic1 <"CAM2" ic2> <"NFRA" nfra>      $
$      <INTE /PROG/> <CENT cx cy cz>                 $
$ "LSLE"                                             $
$ "SCEN" <spars>                                     $
$ "TITL" <tpars>                                     $
$ "GOTR" <"LOOP" n> <trac_options> trac_terminator $
$ "R"                                                $
$ "TIME"                                             $
$ "BENS"                                             $
$ "NOBE"                                             $
$ "QMS"                                              $
$ "COPY"                                             $
$ "MEAS" <measurement commands> TERM              $
$ "ADAP" (<"RECU"> $ "SPLI" ; "USPL" $ $ iel ; "OBJE" /LECT/ $) $
$      TERM                                           $

```

```

$ "EROD" /LECT/                                     $
$ "QUAL" ... (see syntax on Page I.25)              $
$ "EFSI" "STRU" /LECTS/ "FLUI" /LECTF/             $
$          "VARI" $ "PRES" ; "DENS" ; "PMAX" ; "PMIN" $ "DIST" d $
$          <"SLOW">                                   $

```

?

Lists the available primary interactive commands.

GO

Advances the computation of npas time steps.

STOP

Stops the computation.

INFO

Prints information: current time and step number of the computation, time step increment, stability step and critical time step.

FREQ

In a direct calculation, this specifies the interval (in time steps) between two successive interruptions of the calculation. Initial value is 1. The computation will halt every **npas** steps (counted from step 0!) and prompt for commands. This command may be combined with **TFRE**, see below. Note that in a post-treatment calculation (**RESU** keyword, which reads previously stored data from a results file, typically an ALIC file) the meaning of **npas** is different: it indicates the storage stations and not the time step numbers. Furthermore, in this case the **FREQ** and **TFRE** commands may not be combined (i.e., either **npas** or **tfreq** *must* be 0).

npas

Computation interval in time steps (or in storage stations).

TFRE

In a direct calculation, this specifies the interval (in time) between two successive interruptions of the calculation. Initial value is 0.0. The computation will halt every **tfreq** time units (counted from the initial time!) and prompt for commands. This command may be combined with **FREQ**, see above. Note that in a post-treatment calculation (**RESU** keyword, which reads previously stored data from a results file, typically an ALIC file) the **FREQ** and **TFRE** commands may not be combined (i.e., either **npas** or **tfreq** *must* be 0).

tfreq

Computation interval in time units.

HPIN

Halt interactive execution whenever pinball contacts are established (passing from a situation of zero contacts to one or more contacts) or completely disappear (passing from one or more contacts to zero contacts), so that the user may e.g. visualize the contacts. This switch has a toggling behaviour (see comments and sample usage below).

**NOHP**

Do not halt interactive execution whenever pinball contacts appear or disappear. This is the default so normally it does not need to be specified explicitly. However, the keyword is useful to restore the default behaviour after the optional keyword **HPIN** has been specified.

**CALC**

Allows to change the current time step increment. See options below.

**TRAC**

Displays (on graphics screens) or plots (on plotting devices) the current, deformed mesh shape. By default the entire mesh is visualized. See options below.

**Keystrokes and mouse commands**

When using the OpenGL built-in visualizer interactively, some keystrokes and mouse events are interpreted as commands. A complete list of these commands is given in Page O.15.

**MAVI**

Make an animation file (.AVI) starting from a sequence of bitmap images. At the moment, this functionality is available only starting from bitmap files of type BMP. See syntax and options below.

**CAME**

Defines a camera for OpenGL rendering. See below for the various options.

**LCAM**

List all the cameras defined so far. Note that on EUROPLEXUS versions implemented on a non-OpenGL platform, this directive is simply ignored. This enhances portability of benchmark tests on the various platforms.

**SLER**

Defines a spherical linear interpolation (slerp) for OpenGL rendering. See below for the various options.

**LSLE**

List the currently valid slerp. Note that on EUROPLEXUS versions implemented on a non-OpenGL platform, this directive is simply ignored. This enhances portability of benchmark tests on the various platforms.

**SCEN**

Define the current scene parameters **<spars>** to be used for OpenGL rendering. See below for the various parameters.

**TITL**

Define the titles (**<tpars>**) to be used for the production of a titles frame (or AVI sequence) in off-screen OpenGL rendering. See below for the various parameters.

**GOTR**



Performs a **G0** followed by a **TRAC**. The sequence may be automatically repeated **n** times by using the **LOOP** sub-keyword. This command is useful, among other things, for the automatic preparation of image sequences or animations. See below for a complete description.

**R**

Repeat the last command issued (this works also if the preceding command was a **G0TR**).

**TIME**

Prints the current physical and CPU time.

**BENS**

Activates Benson plotter graphics output instead of screen output.

**NOBE**

Deactivates Benson plotter output; subsequent graphics are visualized on the screen.

**QMS**

Pilots a QMS laser printer directly connected to a graphics terminal (Tektronix emulation) to obtain a copy of the graphics appearing on the screen. This command was formerly used at JRC and is now obsolete.

**COPY**

Used at JRC to redirect Tektronix graphics output from the screen to a file, defined as logical unit number 17. The typical command sequence is "**COPY TRAC NORM**", by which the current mesh plot is added to the contents of the file connected to unit 17 (initially void). This can be later visualized again (under UNIX, by the '**cat file\_name**' command) and/or printed.

**MEAS ... TERM**

Introduces some measurement commands, see the full syntax on page G.105.

**ADAP ... TERM**

Introduces interactive adaptivity commands. These commands may be repeated any number of times. The end of this directive is marked by the **TERM** keyword.

**RECU**

Treat the element(s) specified next recursively, not literally. See the comments below for a detailed explanation of the use of the **RECU** optional keyword and for an example of its use.

**SPLI iel**

Split element **iel**.

**SPLI OBJE /LECT/**

Split the object defined by **/LECT/**.

**USPL iel**

Unsplit element **iel**.

USPL OBJE /LECT/

Unsplit the object defined by /LECT/.

EROD /LECT/

Erode immediately the object defined by /LECT/. The keyword EROS (see [GBA\\_0030](#)) must be present at the beginning of the input file (before DIME and GEOM) to activate the erosion mechanism.

QUAL ...

Qualify the results interactively (on the fly, i.e. at the current step). The complete syntax of the QUAL directive is given on page I.25. This is probably most useful within a PLAY ... ENDP directive.

EFSI ...

Introduces commands to Extract embedded FSI fields (EFSI). These are fluid fields (pressure, density etc.) that can be visualized on a structure used as geometrical support. They are useful in conjunction with FSI algorithms of the embedded type (FLSR, FLSW).

STRU /LECTS/

List of the (base) structural elements (typically shells) forming the geometrical support on which the extracted fluid field will be visualized.

FLUI /LECTF/

List of the (base) fluid elements from which the field will be extracted.

VARI ...

Selects the physical variable to be extracted: PRES is the pressure, DENS the density, PMAX the maximum pressure ever experienced, PMIN the minimum pressure ever experienced.

DIST d

Explicitly sets the distance  $d$  from each structural element at which the fluid data should be extracted. Positive values of  $d$  indicate that data should be extracted from the positive direction of the oriented normal to the structure, negative values of  $d$  indicate that data should be extracted from the negative (opposite) direction of the oriented normal to the structure. For the resulting visualization to be meaningful, it is **fundamental** that the structure (shells) be oriented consistently. In 3D, this can be verified by simply drawing the (geometry of the) structure in the OpenGL built-in visualizer: the front faces of the structure will appear as colored while the back faces will appear as empty (since the default is to fill the front faces but not the back faces).

SLOW

Option al keyword that activate the brute force search of candidate fluid elements for the extraction of the EFSI field, instead of the fast search used by default. To be used only for debugging purposes since the search becomes very slow in medium/large applications.

## Comments

An example of use of the HPIN and NOHP keywords is as follows. Suppose a user wants to visualize contact details in a calculation using pinballs. Normally it is difficult to exactly foresee when a contact will be established and/or it will disappear. Use the following interactive commands:

```
FREQ 1000000
HPIN
GO
```

In this way, the code will halt when the first pinball contact is detected and the user will have the possibility of visualizing the contact conditions. Then, type again:

```
GO
```

The execution will continue and will halt again when there are no more pinball contacts (toggling behaviour). This is useful because normally a contact remains for a number of successive time steps after it has first occurred. Then, type again:

```
GO
```

The calculation will halt when a new contact is detected, and so on.

Note that the HPIN keyword is automatically combined with the effects of FREQ, TFRE etc. The first of the specified conditions which occurs determines code halting.

To disable this behaviour and restore the normal behaviour, use the NOHP command.

### Example of interactive adaptivity

To pilot adaptivity interactively, proceed as follows (this is useful mainly for debugging purposes). Assume we have a base mesh of quadrilaterals with ten elements. We want to split elements 1 and 3 at step 1, generating descendent elements 11 to 18. Then at step 2 we want to further split element 15. Finally, at step 3 we want to unsplit element 1. At each step we want to dump out the whole adaptivity data structure for debugging, and also plot the adapted mesh.

We can do this either interactively or in batch mode (via the PLAY ... ENDPLAY command).

In the second case, the input would be as follows.

```
Title of test
CONV win
<normal input of a test case with adaptivity>
ECRI ... FREQ 1
OPTI ADAP DUMP ! to dump out adaptivity data structure at printouts
CALC ...
PLAY
  TRAC REND                ! draw base mesh at step 0
  ADAP SPLI 1 SPLI 2 TERM ! split elements 1 and 2
  GO                       ! compute step 1
  TRAC REND                ! draw adapted mesh at step 1
  ADAP SPLI 15 TERM        ! further split element 15
```

```

GO                ! compute step 2
TRAC REND         ! draw adapted mesh at step 2
ADAP USPL 1 TERM  ! unsplit element 1
GO                ! compute step 3
TRAC REND         ! draw adapted mesh at step 3
STOP              ! terminate calculation
ENDPLAY

```

Obviously, the same interactive commands can also be typed from the keyboard, if one removes the `PLAY ... ENDPLAY` block from the input file.

Note that the first time station at which adaptivity commands can be prescribed is step 1 (not step 0), because step 0 is always computed by the code before asking for interactive commands for the first time.

### Use of the optional `RECU` keyword in interactive adaptivity

The optional `RECU` keyword can be used in conjunction with the `SPLI` and `UNSPL` commands that perform interactive user-driven adaptivity.

If the `RECU` keyword is not specified, the following element number (`iel`) or numbers (`LECT`) are meant literally. It is the user's responsibility to ensure that each mentioned element can be actually split (being a leaf) or unsplit (being a branch whose children are all leaves).

If the `RECU` keyword is specified, the following element number (`iel`) or numbers (`LECT`) are recursively inspected for descendents:

- With `RECU SPLI` the active descendents of each mentioned element are split. If the mentioned element is a leaf (having no descendents) then the element itself is split.
- With `RECU USPL` we first build up the set of active proper descendents of each mentioned element. If the mentioned element is a leaf (having no descendents) then the element is skipped and we move on to the next element, if any. Next, for each element in the set we extract its parent, thus obtaining the set of parents. Multiple element indexes are eliminated from the set of parents. Finally, each element in the set of (unique) parents is unsplit.

By using the `RECU` keyword one can perform some particular types of splitting and unsplitting which are not possible (or more complicated) with the standard syntax. For example, assume that we want to refine a given object `obj` up to level 4. This can be achieved by the command:

```

PLAY
  ADAP RECU SPLI OBJE LECT obj TERM
      RECU SPLI OBJE LECT obj TERM
      RECU SPLI OBJE LECT obj TERM
  TERM
ENDPLAY

```

The `RECU` keyword could be omitted in the first of the `SPLI` commands above, if none of the elements of the `obj` has been previously split (i.e., if they are all base elements). Without the `RECU` keywords, in order to perform the second and third split operation in the above example one would have to identify (and explicitly list) the indexes of the (descendent) elements generated by the previous split operation.

## 15.2 Keystrokes and mouse events in the OpenGL graphical visualizer

### Object

When using the built-in OpenGL graphical visualizer interactively, some keystrokes and mouse events are interpreted as navigation commands.

### Keystrokes

A list of the available keystrokes is given in the following Table.

Key	Action	Amount	CTRL-	CTRL/SHIFT-	SHIFT-
0 (zero)	Reset default view	—	—	—	—
Up arrow	Rotate camera “up”	5°	1°	10°	30°
Down arrow	Rotate camera “down”	5°	1°	10°	30°
Left arrow	Rotate camera “left”	5°	1°	10°	30°
Right arrow	Rotate camera “right”	5°	1°	10°	30°
PgUp	Rotate camera anticlockwise	5°	1°	10°	30°
Ins	Rotate camera clockwise	5°	1°	10°	30°
b	Translate camera backwards	$R/2$	$R/10$	$R$	$3R$
f	Translate camera forwards	$R/2$	$R/10$	$R$	$3R$
i	Zoom camera in (without moving)	$\times 1.2$	$\times 1.1$	$\times 1.5$	$\times 2.0$
o	Zoom camera out (without moving)	$\times 1.2$	$\times 1.1$	$\times 1.5$	$\times 2.0$
r	Move camera rightwards (free mode only)	$R/2$	$R/10$	$R$	$3R$
l	Move camera leftwards (free mode only)	$R/2$	$R/10$	$R$	$3R$
u	Move camera upwards (free mode only)	$R/2$	$R/10$	$R$	$3R$
d	Move camera downwards (free mode only)	$R/2$	$R/10$	$R$	$3R$

Table 15: Available keystrokes.

The keystrokes are *not* case sensitive: for example, **b** has the same effect as **B**. Some keystrokes (**r**, **l**, **u** and **d**) have effect only when the camera is set in “free navigation” mode (not in “rotating” mode). In order to change the camera navigation mode interactively, press the right mouse button in order to bring up the interactive menu and then use the Geometry → Navigation sub-menu.

The description of motions refers to the “camera” model, i.e. to the ideal observer. If preferred, the user may think of the same motion as applied to the object that is being viewed, by just inverting the “sign” of the motion. For example, the “Left arrow” key rotates the observer to the left or, alternatively, the object to the right.

Each motion has pre-defined amounts: 5 degrees for rotations,  $1/2$  of the object radius for translations, 20% magnification/reduction for zooming. Smaller amounts may in some cases be obtained by pressing the Control key (**CTRL**), larger ones by the Control-Shift keys (**CTRL-SHIFT**), and even larger amounts by the Shift key (**SHIFT**) in conjunction with any of the above described keys. For example, the keys combination **CTRL-PgUp** turns the camera by 1 degree, **CTRL/SHIFT-PgUp** turns it by 10 degrees and **SHIFT-PgUp** turns it by 30 degrees.

### Mouse events

A simple and intuitive way of moving the model (or the observer) which is alternative to the keyboard commands described above is by means of the mouse.

With the navigation mode set in “rotating” camera mode, by pressing the left mouse button while the pointer is inside the graphical window, a sort of “virtual trackball” is activated. The object “follows” any subsequent motions of the mouse by rotating around its centerpoint in the corresponding direction.

The effects that may be obtained with the left mouse button are summarized below.

Action	Effect
Press left button	The object starts “following” the mouse cursor by rotating around its centerpoint
Release button while not moving	The object stops rotating, in the final position reached during the previous motion
Move button while pressed	The object follows the mouse motion
Release button while moving	The object continues to “spin” around its centerpoint along the last rotation axis that was active immediately before releasing the button (sort of continuous animation)
Release button while not moving	The object stops rotating, in the final position reached during the previous motion

Table 16: Available mouse events.

Some experimentation will make readily clear what the above somewhat complicated verbal descriptions mean.

A situation which may arise with inexperienced users is that, after using the mouse to rotate the body, it does not stop but it continues “forever” its rotation. This happens when the mouse button is released while still moving (though slowly) the mouse. To stop a rotating object, the following technique may be used: just give a single, quick “click” of the mouse in the window (i.e. press and then immediately release the button) by making sure that you do not move the mouse meanwhile.

### 15.3 CALCUL options

#### Object

To pilot current time step.

#### Syntax:

```
"CALC"    $ "?"          $  
           $ "AUTO"       $  
           $ "UTIL"       $  
           $ "DT" tstep $  
           $ "R"          $
```

?

Lists available subcommands

AUTO

Sets automatic time step calculation (see "OPTI PAS AUTO").

UTIL

Sets fixed time step (see "OPTI PAS UTIL")

DT

Set fixed time step to following value.

tstep

Time step value.

R

Return to primary commands.

## 15.4 TRACE options

### Object

To set options for successive mesh visualizations.

### Syntax

```

"TRAC" $ "?" $
$ "NORM" $
$ "ELEM" i1 i2 $
$ "ZOOM" $ "?" $
$ $ "POIN" xmin ymin xmax ymax $
$ $ "RETI" $
$ $ "R" $
$ "OEIL" xoeil yoeil zoeil $
$ "CULL" $
$ "NOCU" $
$ "NUME" $ "NOEU" $
$ $ "ELEM" $
$ "NONU" $
$ "R" $
$ "CDEP" $
$ "CDNO" $
$ <$ "FANF" ; "NFAI" ; "FAIL" $> $
$ "OBJE" /LECT/ <SURF ; FSIN /LECT/> $
$ <"NOEL"> "OBJN" /LECT/ $
$ "NOOB" "NOGR" "OMEM" "NOIS" $
$ "PINB" $
$ "PINC" $
$ "DEFO" $
$ "AMPD" ampd $
$ "VITE" $
$ "VITG" $
$ "AMPV" ampv $
$ "FEXT" $
$ "FINT" $
$ "AMPF" ampf $
$ "DASH" idsh $
$ "AVS" | "DEPL" "VITE" "FEXT" "ACCE" "MCXX" $
$ "VITG" "FINT" "CONT" "EPST" "ECRO" $
$ "ECRC" /LECT/ | $
$ "PS" $
$ "MIF" $
$ "POVR" $
$ "P10" $
$ < <"OFFS" <"SIZE" w h> <$ "RPOV" ; "BPOV"$> $
$ <"ZIP"> <"FICH" $ "BMP" $

```



```

$          $ "PPM"          $          $
$          $ "PPMA"         $          $
$          $ "TGA"          $          $
$          $ "EPS"          $          $
$          $ "EPSB"         $          $
$          $ "PRAY"         $          $
$          $ "AVI" <pars> $ <'base'> >$
$ <"SYMX"> <"SYMY"> <"EXTZ" nz dz>          $
$ <"SYXY"> <"SYYZ"> <"SYXZ">          $
$ < $ "AXIS" ; "AXOL" $ na ang> <"TOLS" tols> $
$ <"NOSY">          $
$ <"SAVE"> <"REUS">          $
$ <$ "ADAV" ; "ADAP" ; "NOAD" $> "REND" > $

```

?

Lists available subcommands.

NORM

Displays current mesh according to current options.

ELEM i1 i2

Chooses elements i1 to i2 for display. To select a non-contiguous set of elements use the OBJE directive, see below.

ZOOM

Activates zoom display mode.

OEIL x y z

Sets position of viewpoint (3D only) for parallel projection.

CULL

Activates backfacing polygon culling (3D only).

NOCU

Deactivates backface polygon culling (default).

NUME

Activates number visualization for nodes and/or elements.

NONU

Deactivates number visualization (default).

R

Returns to primary commands.

CDEP

Represents 3D degenerated shells of type CQDx with their physical thickness (the topological thickness of these elements is zero). All other element types are automatically hidden in the plot.

**CDNO**

Represents 3D degenerated shells of type CQDx with their topological (zero) thickness. This is the default.

**FANF**

Allows to choose for display both the failed and the non-failed elements. So, all the elements (including all DEBR elements possibly present, irrespective of their activity state), are visualized. This is the default, so this keyword is normally redundant.

**NFAI**

Allows to choose for display only the non-failed elements. By default, all elements (both failed and non-failed) are chosen for display. Note that failed elements have a tendency to assume strange forms due to excessive deformation. This is not a problem for the numerical simulation itself, since they are excluded from the calculation after failure. But with the present keyword they are (also) removed from the visualization, thus avoiding possible cluttering of the scene. Note also that for elements of type DEBR (flying debris particle), this keyword has a special meaning. It visualizes: i) the *active* debris particles, i.e. those resulting from the fragmentation of a previously failed element (which is not shown, since it is failed), and ii) any *markers*, i.e. any used but inactive particles, possibly present in the model.

**FAIL**

Allows to choose for display only the failed elements. By default, all elements (both failed and non-failed) are chosen for display. Note that failed elements have a tendency to assume strange forms due to excessive deformation. This is not a problem for the numerical simulation itself, since they are excluded from the calculation after failure. However, beware that they might somewhat clutter the scene when visualized. Note also that for elements of type DEBR (flying debris particle), this keyword has a special meaning: it visualizes only the *unused* debris particles. This means that both markers and active particles are excluded from the scene.

**OBJE**

Allows to choose non-consecutive elements for display. The list of elements is given in the following /LECT/ and may be in the form of one or more CASTEM2000 objects. This directive is alternative to the ELEM directive, which only allows to specify a range of consecutive elements. To draw a set of nodes instead of (or in addition to) a set of elements, see the OBJN directive below and the techniques described in the comments at the end of this Section (see “Drawing nodes alone or in conjunction with elements”).

**SURF**

Allows to choose only the external surface of the chosen object. This greatly reduces the amount of information to treat in the graphical module with respect to the full 3D case in large and complicated models (but of course it prevents the possibility of visualizing results in the internal parts of the model). This visualization mode makes sense only in 3D and requires the presence of continuum-like fluid elements. This option is only available for the OpenGL-based visualizer (TRAC ... REND) and, if specified, it must immediately follow the OBJE /LECT/ directive (so it is mutually exclusive with the FSIN keyword described below).

**FSIN**

Allows to visualize only the fluid-structure interface portions of the fluid part of the chosen object. These appear as fluid element faces “sticking” onto the matching structural parts, if any are specified as well. For optimal visualization in the OpenGL renderer, it is suggested to turn on backface rendering and to apply some shrinking, e.g. by the directive (see the `SCEN` directive below): `SCEN ... GEOM SHRI 0.98 ISOL FACE SBAC ...`. The following `/LECT/` lists the concerned fluid *nodes*, i.e. the fluid nodes that lie on the fluid-structure interface: in simple cases these are just the same nodes used in the `FSA` and/or `FSR` directives. A fluid face is drawn if and only if all its nodes belong to the given `/LECT/`. This visualization mode makes sense only in 3D and requires the presence of continuum-like fluid elements. This option is only available for the OpenGL-based visualizer (`TRAC ... REND`) and, if specified, it must immediately follow the `OBJE /LECT/` directive (so it is mutually exclusive with the `SURF` keyword described above).

#### NOEL

Do not show any elements in the visualization. This should probably be used in combination with the following `OBJN` directive.

#### OBJN

Choose a set of nodes (specified in the following `/LECT/`) for visualization, rather than a set of elements. If the following `/LECT/` contains elements (e.g. via a Cast3m object name), the nodes of the object (not the elements) are chosen for visualization (as a cloud of thick points). To visualize **only** a set of nodes, say `p1` and `fsan`, the command is `TRAC NOEL OBJN LECT p1 fsan TERM REND`. To visualize **both** a set of elements, say `flui`, and a set of nodes the command is `TRAC OBJE LECT flui TERM OBJN LECT p1 fsan TERM REND`. To visualize **all** elements with a set of nodes highlighted as thick points the command is `TRAC OBJN LECT p1 fsan TERM REND`.

Other techniques that can be used to visualize a set of nodes alone or in addition to a set of elements are described below in the comments at the end of this Section (see “Drawing nodes alone or in conjunction with elements”).

#### NOOB

Do not make available object names (either defined by `CAST3M` or by `I-DEAS`) in the graphical rendering module. This option may be useful to speed up the rendering operations since the number of defined objects is sometimes very large. By default, object names are made available in on-screen rendering, because the user may decide interactively to use them. In off-screen rendering, they are made available only if they are needed for the visualization of the specified scene.

#### NOGR

Do not make available element group names (defined by the `GROU` directive, see page C.61) nor node group names (defined by the `NGRO` directive, see page C.62) in the graphical rendering module. By default, group names are made available in on-screen rendering, because the user may decide interactively to use them. In off-screen rendering, they are made available only if they are needed for the visualization of the specified scene.

#### OMEM

Optimize memory during the graphical rendering, at the expense of some (or may be a lot) more CPU time. This optional keyword should only be activated in extreme cases where the size of the geometrical model is so large that the memory is not sufficient to render it (the graphics-related arrays are too big). In this way the code tries to save

memory by computing some big arrays “on the fly” rather than storing them in memory. An example is the representation of iso-surfaces in very large fluid volumes. This is only useful in off-screen rendering, since in on-screen rendering any manipulation of the mesh (e.g. rotation, zoom etc.) would be extremely slow. Furthermore, if there is memory shortage, then off-screen rendering is by far preferable since only the strictly necessary tables are allocated, in contrast to on-screen (interactive) rendering. Another way of saving some memory is to specify also the `NOOB` and `NOGR` keywords described above, if objects/groups are not needed.

#### NOIS

Do not make available the “true” element fields for visualization in iso form in the graphical rendering module. This option may be useful to speed up the rendering operations since the number of data for the true element fields (stresses, hardening components, etc.) is sometimes very large. By default, all element fields are made available in on-screen rendering when no `SCEN` is specified, so that the user can decide interactively to show any of them. In off-screen rendering (or in on-screen rendering with a specified `SCEN`), only the element field (if any) needed for the visualization of the specified scene is made available. Recall for completeness that all *nodal* fields are always made available for visualization under iso-value form, in on-screen rendering.

#### PINB

Draw the pinballs declared by the `LIAI PINB` directive. These are represented by circles. If this directive is combined with the `TRAC DEFO` directive, then the displacement amplification factor (`AMPD`) must be 1.0.

#### PINC

Draw the contacting (sub-)pinballs. These are represented by circles. A straight line joins the centers of each couple of contacting (sub-)pinballs. If this directive is combined with the `TRAC DEFO` directive, then the displacement amplification factor (`AMPD`) must be 1.0.

#### DEFO

This directive produces a plot of the initial, undeformed geometry of the model, superposed to the deformed one, which is plotted by default. The initial geometry is traced using a dashed line style (see directive `DASH` below) in order to distinguish it better from the current one. If this directive is combined with the `TRAC PINB` or `TRAC PINC` directive, then the amplification factor `AMPD` must be set at 1.0.

#### AMPD

Sets the amplification factor for displacements used to draw the deformed geometry. By default it is 1.0. If this directive is combined with the `TRAC PINB` or `TRAC PINC` directive, then the amplification factor must be 1.0.

#### VITE

This directive produces a plot of the material velocity vectors, superposed to the deformed mesh.

#### VITG

This directive produces a plot of the grid (mesh) velocity vectors, superposed to the deformed mesh.

**AMPV**

Sets the amplification factor for velocity vectors. By default it is 1.0.

**FEXT**

This directive produces a plot of the external force vectors, superposed to the deformed mesh.

**FINT**

This directive produces a plot of the internal force vectors, superposed to the deformed mesh.

**AMPF**

Sets the amplification factor for force vectors. By default it is 1.0.

**DASH**

Sets the line type for plotting the initial geometry, when "DEFO" is specified. There are 4 different styles, so idsh should be from 1 to 4. By default, idsh=3.

**AVS**

Produce a storage for AVS postprocessing. The variable(s) to be stored (each one on a separate file) are specified next. Before listing the variables, one may optionally specify a deformation factor (1.0 by default) via the DEFO directive and an object via the OBJE directive (by default the entire mesh is stored). Note that output results for AVS may also be produced (in "batch" modality) by means of the ECRI FICH AVS directive, see page G.70.

**DEPL**

Store displacements for AVS post-processing. In this case the geometry stored is the initial one and displacements are also stored as a nodal field. For all other variables, the stored geometry is the current (deformed) one.

**VITE**

Store particle velocity for AVS post-processing.

**FEXT**

Store external forces (including reactions) for AVS post-processing.

**ACCE**

Store particle acceleration for AVS post-processing.

**MCXX**

Store multicomponent fluid variables for AVS post-processing.

**VITG**

Store grid velocity for AVS post-processing.

**FINT**

Store internal forces for AVS post-processing.

CONT

Store stresses for AVS post-processing.

EPST

Store total strains (still to be implemented for AVS post-processing).

ECRO

Store hardening quantities for AVS post-processing. The relevant components are chosen by ECRC.

ECRC

Select ECR components to be chosen (/LECT/).

PS

Produce output on PostScript file instead of screen.

MIF

Produce output on MIF (FrameMaker) file instead of screen.

POVR

Produce output in the form of a POV-Ray (Persistence of Vision Ray tracer) file instead of screen. Only the geometry is stored.

P10

Produce PLOT-10 (Tektronix) output on file instead of screen (available also on MS-Windows).

<OFFS ...> REND

Start OpenGL rendering (currently available only under MS-Windows or Linux). The rendering process may take place either on-screen (the default) or off-screen (in a file), as specified by the OFFS sub-directive (see full description below) which, if present, must precede the REND keyword. Note that on EUROPLEXUS versions implemented on a non-OpenGL platform, the TRAC ... REND directive is simply (and entirely) ignored. This enhances portability of benchmark tests on the various platforms.

SYMX

Perform a symmetry with respect to the X-axis before rendering. (This option is still under development).

SYMY

Perform a symmetry with respect to the Y-axis before rendering. (This option is still under development).

EXTZ nz dz

Perform an extrusion with respect to the Z-plane before rendering. The extrusion amount is dz, subdivided into nz increments. (This option is still under development).

**SYXY**

Perform a symmetry with respect to the XY-plane before rendering. (This option is still under development).

**SYYZ**

Perform a symmetry with respect to the YZ-plane before rendering. (This option is still under development).

**SYXZ**

Perform a symmetry with respect to the XZ-plane before rendering. (This option is still under development).

**AXIS na ang**

Perform an axial symmetry around the Y-axis before rendering. The total angle of symmetry is **ang**, subdivided into **na** increments. If necessary, a small hole is generated along the axis of symmetry of the symmetrized mesh in order to simplify the generation of symmetrized elements. This command supersedes an older version of the command that is still available via the **AXOL** command, see below. This new strategy is compatible with the use of **SUPP**.

**AXOL na ang**

Same as **AXIS** described above, but by using an older strategy for axisymmetric symmetrization. Any quadrangular element with only one point on the axis of revolution produces two new elements instead of just one. This version of the procedure does not need to generate a small hole in the symmetrized mesh along the axis of symmetry, but it is likely to be incompatible with **SUPP**.

**TOLS tols**

Set tolerance used for nodes relative positions in symmetries. For example, when asking for symmetrization with respect to plane XY, all nodes must either have  $z \leq 0$  or  $z \geq 0$ . The absolute nodal positions are divided by the maximum size of the visualized object along the coordinate axes in order to obtain relative values. Default value of **tol**s is 1.E-5. One may try to slightly increase this value in case an error is printed by the code (this happens when some nodes lie “slightly” on the wrong half-space for symmetrization).

**NOSY**

Disable any symmetries previously defined by the **SYMX**, **SYMY**, **EXTZ**, **SYXY**, **SYYZ**, **SYXZ** and **AXIS** directives. Also, the symmetrization tolerance **TOLS** is reset to its default value (see above). Beware that when performing a visualization with symmetrization, the code *remembers* the symmetrization settings for the next visualizations. Therefore, the **NOSY** command can be used to reset all symmetrization parameters to their default (no symmetrization) and then use a different set of parameters, if needed.

**SAVE**

Save for the next rendering action(s) all geometrical quantities computed in the rendering process. Since the computation of these quantities is very CPU time consuming, this option may allow to considerably shorten the time required to produce animations composed by long sequences of frames in which the geometrical quantities stay constant and only the iso

field or vector field change from frame to frame. The default is not to save the computed quantities. A typical use of **SAVE/REUS** would be in an Eulerian calculation, to show the time evolution of pressure field and velocity vectors in the fluid domain. The chosen domain (fluid) is always the same in all frames (although possibly the viewpoint may change) and the nodes do not move since they are Eulerian. The same optimization may be obtained also in an ALE calculation, if one visualizes only a completely Eulerian sub-domain: typical is the case of a fluid-structure interaction calculation with the **FLSR** model, where the fluid domain is typically completely Eulerian. For safety, the code verifies that all nodes to be visualized be Eulerian. If this rule is not respected, the geometrical quantities are re-computed and so no (or little) optimization takes place. This verification may be skipped by activating the **OPTI REUD NAVI** option, see page H.170. With this option, the user declares that any changes in the following rendering operations will be due only to navigation (**NAVI**) around or inside a fixed (static) scene, so that use of **SAVE/REUS** becomes possible also in Lagrangian cases. In this case the user is responsible for making sure that no geometrical data vary between a rendering and the next one(s): the mesh does not move, no elements are eroded, adaptivity does not modify the current mesh, etc.

#### REUS

Re-use the geometrical quantities computed and saved in a preceding rendering process (by the **SAVE** option described above) rather than spend CPU time to recompute them. The default is to re-compute these quantities anew each time. Note that **REUS** is not compatible with symmetry commands, i.e. with the **SYMX**, **SYMY**, **EXTZ** and **AXIS** directives.

#### ADAV

Take into account adaptivity in the visualization. An adapted mesh is usually non-conforming: so-called hanging nodes are present along locally non-conforming element-to-element interfaces. This produces locally incorrect visualization of internal faces, sharp corners and free edges. When this option is set, the computational mesh is replaced by a newly built visualization mesh. This is a pseudo-conforming mesh built up and passed to the graphical visualizer in place of the computational mesh, in order to allow correct visualization of internal faces, sharp corners and free edges. The strategy used is similar to that of the **PL2T** option (see Page G.70), but the latter option applies to the ParaView output (**PVTK**), while the present one applies to the OpenGL-based graphical module. Note, however, that this strategy has some serious limitations. For example, it works for mesh visualization but not for vector fields or iso fields. Therefore, it should only be used for debugging purposes and the **ADAP** keyword should be used instead.

#### ADAP

The strategy activated by this keyword is similar to the one of the **ADAV** keyword described above, but the visualization mesh is built up and passed to the OpenGL module *in addition* to the computational mesh, and is used *exclusively* to compute and visualize the free edges and the sharp corners. This strategy has no known limitations and should be preferred to the **ADAV** keyword in practical applications.

#### NOAD

Do not take into account adaptivity in the visualization. This is currently the default, so using this keyword should be unnecessary. An adapted mesh

#### Comments



The **CULL** option performs a very basic hidden surface removal. All element faces (polygons) whose outward normal points away from the observer are eliminated from the plot. This results in hidden surface removal for very simple, basic shapes, but is imperfect for complex, arbitrary geometries.

At each required AVS storage (see AVS above), one file is produced for each variable, with the name **avs.<VARI>.N.inp**, where **<VARI>** stands for the variable (i.e., **DEPL**, **VITE** etc.), and **N** is an integer counter which is automatically incremented by one each time storage is requested. Such files may be postprocessed interactively by AVS while EUROPLEXUS is running.

At each required POV-Ray storage (see POVR above), one file is produced containing the current geometry of the model. The file names are **povray.GEOM.N.pov** where **N** is an integer counter starting at 0 and incremented by one each time storage is requested. Such files may be postprocessed interactively by POV-Ray while EUROPLEXUS is running.

## Symmetries

Note that these options are still under development.

The **SYMX**, **SYMY**, **EXTZ** and **AXIS** directives are only available in conjunction with OpenGL-based rendering (**REND**).

They may be combined, but the following combinations are invalid:

- **EXTZ** and **AXIS** are mutually exclusive.
- **SYMY** and **AXIS** are mutually exclusive.

Furthermore, the following restrictions apply:

- All nodes of the original mesh must have  $z = 0$ .
- **SYMX** requires that all nodes of the original mesh have  $y \geq 0$ .
- **SYMY** requires that all nodes of the original mesh have  $x \geq 0$ .
- **AXIS** requires that all nodes of the original mesh have  $x \geq 0$ .

## Off-screen rendering

The **REND** directive admits an optional **OFFS** sub-directive that allows to produce OpenGL rendered images off-screen and to prepare animations of the results. By default, when the **OFFS** keyword is specified alone, each frame of the scene is recorded in a bitmap file, having the name **base\_nnnn.<ext>**, where **base** is the base name of the run, **nnnn** is a four-digit integer counter that starts at 0001 and is incremented automatically by the program, and **<ext>** is the file extension which depends upon the chosen file type.

The following bitmap file types are currently supported:

- BMP indicates a MS Windows bitmap (binary) file, with the extension `.bmp`.
- PPM indicates a Portable Pixmap (binary) file, with the extension `.ppm`.
- PPMA indicates a Portable Pixmap (ASCII) file, with the extension `.ppm`.
- TGA indicates a Targa (binary) file, with the extension `.tga`.
- EPS indicates an Encapsulated PostScript file in color, with the extension `.eps`.
- EPSB indicates an Encapsulated PostScript file in black and white, with the extension `.eps`.
- PRAY indicates a POV-Ray scene description file, with the extension `.pov`. Note that this is *not* a bitmap file, but a text (Ascii) file which, interpreted by the POV-Ray program, can generate a bitmap rendering of the scene. POV-Ray can be launched directly from EPX by adding the optional RPOV or BPOV keyword, as explained below.

Such files are uncompressed by default and may therefore require a large disk space. To save space, the optional ZIP keyword may be specified, which automatically compresses the bitmap file after its generation.

In the case of a POV-Ray file, the optional RPOV keyword can be used to launch POV-Ray on the generated `.pov` file *immediately* after its creation (and before zipping it, if the ZIP keyword is specified as well). This will generate a `.bmp` file ready for use as an illustration or for preparation of an animation in EPX via the MAVI command. This of course requires that POV-Ray is installed on the platform where EPX is being run. Note that under Windows POV-Ray always opens a window even when it is launched from the command line (as it occurs with the RPOV or BPOV keywords). This may be slightly annoying, although the window is automatically closed when the execution of POV-Ray terminates.

As an alternative to RPOV keyword, one may use the BPOV keyword (for Batch POV-Ray execution), which will launch POV-Ray *only at the end* of the EPX job, and will process all the previously produced `.pov` files in a single execution of POV-Ray. This considerably speeds up the process of conversion from POV-Ray to bitmap files since POV-Ray is launched only once and processes all the files in a single execution. Also, only one POV-Ray window is created (and finally destroyed) on screen.

Note that in case of POV-Ray output the SAVE/REUS keywords described previously have a special meaning. The SAVE/REUS mechanism (combined with the OPTI REND NAVI option described on page H.170, if necessary) can be used in order to speed up and also to drastically reduce the disk storage needed when preparing a POV-Ray based animation. In such a case the geometrical data, which occupy a lot of space on disk, are written only once, when the first POV-Ray file is produced, in a “geometry” file named `<base>.geom.pov`. This file is then included in all following regular POV-Ray files `<base>_0001.pov`, `<base>_0002.pov`, etc., which will only contain the scene data that vary from frame to frame. The OPTI REND NAVI option is necessary if any of the nodes to be rendered with the SAVE/REUS mechanism are Lagrangian.

By specifying the optional keyword FICH, users may change the base name mentioned above: for example the sequence `TRAC OFFS FICH BMP 'toto' REND` would produce frame files `toto_0001.bmp`, `toto_0002.bmp`, etc.

In addition (or in alternative), the user may request the production of an AVI animated sequence from the single frames. For example, the directive `TRAC OFFS FICH AVI 'anim' REND`

would produce an animation file `anim_01.avi` but no BMP frame files. To produce both the BMP frames and the AVI file, specify both options. The default name of the animation file (i.e. if 'base' is omitted in the above syntax) is `base_nn.avi` where `base` is the base name of the run and `nn` is a two-digit integer counter that starts at 01 and is incremented automatically by the program.

By default, the size of off-screen generated (OFFS) bitmap and AVI files is of 500 pixels (width) by 500 pixels (height). To produce a different size, use the `SIZE w h` sub-directive where `w` is the width in pixels and `h` is the height in pixels. Both these quantities should be multiples of 4.

For on-screen generated images, the *initial* size as the window is popped up is of 500 pixels (width) by 500 pixels (height). This may then be changed interactively by the user.

The production of AVI files may be piloted by a sequence of parameters `<pars>` that are described below.

## ZOOM suboptions

### Syntax:

```
"ZOOM" $ "?" $
        $ "POIN" xmin ymin xmax ymax $
        $ "RETI" $
        $ "R" $
```

?

Lists available subsubcommands.

POIN

Sets zoom window using coordinates of lower left and upper right corner.

xmin ymin

Coordinates of lower left window corner.

xmax ymax

Coordinates of upper right window corner.

RETI

Activates crosshair cursor for the definition of zoom window. Position the cursor with direction keys on lower left corner, then type `<CR>`, repeat for upper right corner.

R

Return to primary commands.

## NUME suboptions

### Syntax:

```
"NUME" $ "NOEU" $
      $ "ELEM" $
```

NOEU

Activates node number display.

ELEM

Activates element number display.

## Direct AVI file generation

### Object

To generate directly an animated AVI file without using an external utility program that generates the AVI starting from a sequence of still frames (bitmaps).

A serious drawback for the use of this command, in particular during the (direct) calculation of a transient solution, is that the total number of frames in the animation (see **NFTO** below) must be set exactly. If for some reason the application stops before having written all the frames and having closed properly the AVI file, this file is unusable. The problem may be circumvented by producing bitmaps (frames) during the direct calculation, and then by making the AVI file separately starting from the frames sequence, as described in the **MAVI** directive below. In fact, in that case the program is able to determine the total number of frames automatically, if needed, because all the frames are available when the animation production is started. At the moment, the **MAVI** command is only available for bitmap images of type BMP.

Note that this functionality is based upon the Microsoft Video for Windows library and therefore it is currently available only on MS-Windows based platforms. If the following commands are issued on a different platform, they are simply ignored by EUROPLEXUS.

### Syntax:

```
TRAC   <OFFS <FICH AVI <pars> $ <'base'> > > REND
```

where **<pars>** represents the following syntax:

```
<CONT> <NOCL>
<NFTO nfto>
<FPS fps> <COMP comp> <KFRE kfre> <CQUA cqua>
```

CONT

The AVI scene is a continuation of the AVI file created with a previous **TRAC OFFS FICH AVI** command *during the same EUROPLEXUS run*. This optional keyword allows to build up a complex animation as a series of simple sequences (scenes), each one produced by a separate command. By default (i.e. in the absence of the **CONT** keyword) a new AVI file is started. Note that, if present, the keywords **CONT** and/or **NOCL** must immediately follow the keyword **AVI** and come before the other optional keywords.

**NOCL**

Do not close the AVI file after writing the current scene. This allows to add further scenes by subsequent commands. By default, i.e. in the absence of the **NOCL** keyword, the AVI file is closed after writing the current scene. Note that, if present, the keywords **CONT** and/or **NOCL** must immediately follow the keyword **AVI** and come before the other optional keywords.

**nfto**

The total number of frames forming the AVI file animated sequence. Note that, unlike the following ones, this parameter is mandatory but only when the scene being defined is the first one of a multi-scene AVI file, i.e. when the **CONT** keyword is not present but the **NOCL** keyword is specified. When both **CONT** and **NOCL** are omitted, then the AVI file contains just one scene (the current one) and the (total) number of frames needs not be specified, since it may be obtained as the value given for the currently valid **slerp** (see **SLER** below).

**nfto**

The total number of frames forming the AVI file animated sequence. Note that, unlike the following ones, this parameter is mandatory.

**fps**

Number of frames per second for the visualization of the AVI file. If omitted, the default value of 5 frames per second is used.

**comp**

Compression type of the produced AVI file. The value **-1** indicates that the Microsoft Video 1 codec has to be used. This codec is somewhat obsolete for realistic films, but perfectly adequate for the type of technical graphics produced by EUROPLEXUS, and has the advantage of being present on virtually any MS-Windows based computer. The value **1** produces a popup dialog box that allows the user to interactively choose (and somewhat configure) the desired codec from the list of those available on his platform. Obviously, this is adequate only for interactive execution of EUROPLEXUS (while the previous value **-1** is the normal choice for unattended AVI file creation, i.e. batch execution). If omitted, the default value of **0** (no compression) is used. Note, however, that without compression the produced AVI file size grows very rapidly since it is simply the sum of the size of its (uncompressed) frames. An advantage of this choice is that the AVI file may be compressed *a posteriori* by means of an external utility (e.g. Virtualdub).

**kfre**

Key frame frequency. This parameter is only used when the Microsoft Video 1 codec is chosen (see **comp** above). The default value is **0**, meaning that every frame is a key frame. This somewhat increases the file size but it simplifies navigation through it during playback.

**cqua**

Compression quality in %. This parameter is only used when the Microsoft Video 1 codec is chosen (see **comp** above). The default value is **100**, meaning full (loss-less) quality.

**Drawing nodes alone or in conjunction with elements**

Normally EUROPLEXUS draws an object composed of **elements**, and the nodes belonging to such elements are drawn as a consequence of elements visualization.

A user may sometimes want to visualize a set of **nodes** *independently* from the elements to which they belong, or *in addition* to the elements to which they belong (by highlighting the chosen nodes in some manner so that they stand out). Below are summarized some techniques to obtain such effects.

1. Drawing nodes only.

Assume that we have an object **mynodes** composed only of nodes. This may come from a mesh generator such as Cast3m, it can be the result of an NGRO nodes group definition, or be simply a list of node indexes. To visualize just these nodes, the command is:

```
TRAC NOOB OBJN LECT mynodes TERM REND
```

Note that the view will be set up automatically to fit only the drawn nodes. This technique can be used either in interactive or in batch mode.

2. Drawing nodes in addition to elements in interactive mode.

A possible approach is the following: first draw the whole mesh (or a sub-mesh of elements containing all the nodes that should be visualized):

```
TRAC REND
```

Then unselect all elements by right-clicking on Objects → Hide All.

Finally, select the group of nodes **mynodes** by right-clicking on Objects → Select Groups. Note that the **mynodes** group has 0 elements and  $N > 0$  nodes. In this way, only the selected nodes will remain visible (no elements).

The appearance of the nodes can be set by right-clicking on the Geometry → Points menu. Note that with this technique the view will be set up automatically to fit the initially drawn mesh, and not only the drawn nodes. Sometimes this may be inappropriate.

3. Drawing nodes in addition to elements in batch mode.

A possible approach is the following:

```
PLAY
CAME ... ! set up the chosen camera, e.g. encompassing
        ! only the chosen nodes
SCEN OBJE USLM LECT tous TERM      ! unselect all elements
      SELP LECT mynodes TERM ! select the wanted nodes
      GEOM NAVI FREE
      POIN SPHE 2 ! for example ...
      COLO PAPE
SLER CAM1 1 NFRA 1
TRAC REND ! to show on screen. use OFFS to draw offscreen
ENDPLAY
```

In this way, only the selected nodes will be visible (no elements).

Note that with this technique the view is set up by the user via the **CAME** directive. This gives complete freedom, but is more laborious.

## 15.5 AVI file generation from a sequence of bitmaps (MAVI)

### Object

To generate an animated AVI file starting from a sequence of still frames (bitmaps). At the moment, the only type of bitmap images that are recognized by the MAVI command are BMP images.

Note that this functionality is based upon the Microsoft Video for Windows library and therefore it is currently available only on MS-Windows based platforms. If the following commands are issued on a different platform, they are simply ignored by EUROPLEXUS.

### Syntax:

```
MAVI <DUMP> <FROM 'base'> <UZIP> <RZIP> <TO 'to'>  
      <FIRS first> <LAST last> <STEP step> <pars> REND
```

#### DUMP

Dump out verbose information about the bitmaps that are being read in during the AVI file generation (only for debugging).

#### 'base'

Base name of the sequence of bitmap images, in quotes. If omitted, the base name of the test case is used. For example, by specifying FROM 'toto' the program looks for files of the form `toto_0001.bmp`, `toto_0002.bmp` etc. in the current directory, if uncompressed bitmaps (as by default) are used. If compressed bitmaps are used (see the next keyword UZIP), then the expected file names are `toto_0001.bmp.gz`, `toto_0002.bmp.gz` etc. in the current directory.

#### UZIP

Unzip (decompress) the bitmap files before using them to produce the AVI file. The bitmap files have either been compressed by hand, or they have been produced by the TRAC OFFS ZIP ... directive as explained above. By default, uncompressed bitmaps are expected.

#### RZIP

Re-zip (re-compress) the bitmap files after using them to produce the AVI file. This saves a lot of disk space. By default, the bitmap files are left uncompressed after use.

#### 'to'

Base name of the AVI file to be produced, in quotes. If omitted, the base name of the test case is used. For example, by specifying TO 'tata' the program generates AVI file(s) of the form `tata_01.avi`, `tata_02.avi` etc. in the current directory.

#### FIRS

Index of the first bitmap file (frame) to be used for the animation. By default, the first found file in alphabetical order is used.

**LAST**

Index of the last bitmap file (frame) to be used for the animation. By default, the last found file in alphabetical order is used, so the total number of frames in the animation is in this case determined automatically by the program.

**STEP**

Increment in the index of the bitmap file (frame) to be used for the animation. By default, all files are used, in alphabetical order.

**REND**

This keyword terminates the **MAVI** sequence and triggers its execution. If omitted, no animation file is produced!

In the above **MAVI** directive, the sequence **<pars>** represents the following syntax (similar to the one already described above for the direct AVI file creation):

**<FPS fps> <COMP comp> <KFRE kfre> <CQUA cqua>**

**fps**

Number of frames per second for the visualization of the AVI file. If omitted, the default value of 5 frames per second is used.

**comp**

Compression type of the produced AVI file. The value **-1** indicates that the Microsoft Video 1 codec has to be used. This codec is somewhat obsolete for realistic films, but perfectly adequate for the type of technical graphics produced by EUROPLEXUS, and has the advantage of being present on virtually any MS-Windows based computer. The value **1** produces a popup dialog box that allows the user to interactively choose (and somewhat configure) the desired codec from the list of those available on his platform. Obviously, this is adequate only for interactive execution of EUROPLEXUS (while the previous value **-1** is the normal choice for unattended AVI file creation, i.e. batch execution). If omitted, the default value of **0** (no compression) is used. Note, however, that without compression the produced AVI file size grows very rapidly since it is simply the sum of the size of its (uncompressed) frames. An advantage of this choice is that the AVI file may be compressed *a posteriori* by means of an external utility (e.g. Virtualdub).

**kfre**

Key frame frequency. This parameter is only used when the Microsoft Video 1 codec is chosen (see **comp** above). The default value is **0**, meaning that every frame is a key frame. This somewhat increases the file size but it simplifies navigation through it during playback.

**cqua**

Compression quality in %. This parameter is only used when the Microsoft Video 1 codec is chosen (see **comp** above). The default value is **100**, meaning full (loss-less) quality.



## 15.6 GOTRAC: a simple looping mechanism

### Object

To perform a **GO** in order to advance the solution to the next desired time step or time value, directly followed by a **TRAC** operation to display the results. This sequence may be automatically repeated a given number of times, if so desired.

### Syntax:

```
"GOTR" <"LOOP" n> <trac_options> trac_terminator
```

**n**

An integer used to specify the number of times the **GOTRAC** sequence has to be repeated. By default, the sequence is executed just once.

**trac\_options**

Any valid sequence of sub-commands of the **TRAC** command, see above.

**trac\_terminator**

A valid terminator of the **TRAC** command, which actually produces the drawing or visualization. The possible values are **NORM** for vector-graphics based (on-screen or on file) drawing or **REND** for OpenGL-based rendering. See the above description of the **TRAC** command for further details.

## 15.7 CAMERA parameters and options

### Object

To define a camera for OpenGL rendering. Repeat this command any number of times to define as many cameras as needed (with different identifiers `icam`, see below). The orientation of the camera in space may be defined in two alternative ways: either via a quaternion, or via a triplet of versors defining a right-handed reference frame.

Note that on EUROPLEXUS versions implemented on a non-OpenGL platform, this directive is simply (and entirely) ignored. This enhances portability of benchmark tests on the various platforms.

### Syntax:

```
CAME icam < EYE ex ey ez >
          < $ Q qr qx qy qz          $
          $ VIEW vz vy vz RIGH rx ry rz UP ux uy uz    $ >
          < FOV fovy >
```

### `icam`

An integer used to identify the camera later on. It must be a positive number and it is mandatory (no default value is provided). Typically, use 1, 2, 3, etc. Best efficiency is obtained by starting the definition of cameras with the highest index. By repeating the definition of an existing camera (same index), the old one is replaced by the new one and is no longer available.

### EYE

Position of the camera in space, i.e. position of the observer's eye. If omitted, the program assumes the position (0,0,1).

### Q

Quaternion defining the orientation of the camera in space. Here `qr` is the real part while `qx`, `qy`, `qz` are the components of the imaginary part. Its norm must be unitary, so that the quaternion represents a rigid-body rotation in space, with respect to a default orientation. This default orientation is assumed such that the *x*-axis points to the right of the picture, the *y*-axis points upwards and the negative *z*-axis points "inside". The default value of this parameter is the identity quaternion (1,0,0,0).

### VIEW RIGH UP

Triplet of unit versors that may be used, in alternative to the quaternion form described above, to define the orientation in space of the camera. The **VIEW** vector points from the camera position (**EYE**) to the observed object. The **RIGHT** vector defines the right-hand orientation (horizontally in the picture) and the **UP** vector defines the upright direction (vertically in the picture). These vectors must be unitary in length and be mutually orthogonal so as to define a left-handed reference frame. The vector product of **RIGH** times **VIEW** must equal **UP** (and cyclic permutations thereof). The default values are: (0,0,-1) for **VIEW**, (1,0,0) for **RIGH** and (0,1,0) for **UP**.

**FOV**

Angle representing the field of view of the camera, in degrees. Smaller angles produce a zoom-in effect while larger ones produce a zoom-out effect. The default value is 60 degrees.

## 15.8 SLERP parameters and options

### Object

To define a slerp (spherical linear interpolation) of camera positions for OpenGL rendering.

Note that on EUROPLEXUS versions implemented on a non-OpenGL platform, this directive is simply (and entirely) ignored. This enhances portability of benchmark tests on the various platforms.

### Syntax:

```
SLER CAM1 ic1 <CAM2 ic2> <NFRA nfra>  
      <INTE /PROG/> <CENT cx cy cz>
```

#### CAM1

Identifier of the first (initial) camera for the slerp. This camera must of course have been previously defined by the **CAME** directive. This value is mandatory, and thus no default value is provided.

#### CAM2

Identifier of the second (final) camera for the slerp. This camera must of course have been previously defined by the **CAME** directive. This value may be omitted, and in that case 0 is assumed. This means that the scene is still: the first camera defined above is used for the whole sequence.

#### NFRA

Number of frames of the slerp sequence. If omitted, 1 is assumed: the scene consists of a single frame, produced by **CAM1**. If greater than 1, then there are two cases: if **CAM2** is not defined, then all frames are produced with the first camera (**CAM1**), i.e. the sequence is still. If **CAM2 is defined**, then the camera is interpolated between **CAM1** and **CAM2**. Note, however, that in the case of linear interpolation (missing **INTE**, see below) the first interpolated value is **not** **CAM1** but the first non-zero value going from **CAM1** to **CAM2**. The last interpolated camera, however, coincides with **CAM2**. This convention allows to chain successive sequences one after the other without obtaining double (repeated) frames at the intermediate camera values.

#### INTE

Interpolation values for the calculation of parameters for the intermediate frames. If omitted, linear equidistant values are used. Linear interpolation is applied to the camera FOV while slerp interpolation is used for the camera orientation (**Q** or **VIEW**, **RIGH**, **UP**). The camera **EYE** is interpolated as described below (see **CENT**).

#### CENT

Centre of rotation for the interpolation of the camera eye positions. If omitted, or if its position coincides with the eye position for the first camera (**CAM1**), the eye position is interpolated linearly between the initial and final (if relevant) specified positions. Thus, the observer moves along a straight line (while at the same time possibly rotating around the eye). When present and different from the eye position for the first camera (**CAM1**), it represents the centre of a circle along which the camera eye moves. The circle passes through the initial and final camera eye positions. Therefore, the given point must be equidistant from these two points (but of course not aligned between them, so that the three points define a unique plane).

## 15.9 SCENE options

### Object

To define a set of parameters (globally indicated above as **<spars>**) for the definition of the characteristics of the current scene, to be used during OpenGL rendering.

These parameters parallel as closely as possible the menu items that are available for interactive OpenGL visualization. For more details on the parameters and options, see the reference manual of the interactive OpenGL renderer.

Once defined by a **SCEN** directive, a set of scene parameters (the **current scene**) remains active for any following **TRAC** directive(s), until a new set of parameters is defined by a new **SCEN** directive.

If no **SCEN** directive is given, some reasonable default values are assumed.

In order to restore the default scene values during a calculation *after* a scene has been defined, use an empty **SCEN** directive, i.e. **SCEN** followed by no other sub-keywords or parameters.

Note that on EUROPLEXUS versions implemented on a non-OpenGL platform, this directive is simply (and entirely) ignored. This enhances portability of benchmark tests on the various platforms.

### Syntax:

```
SCEN
  <OBJE ( <SELM /LECT/> <USLM /LECT/>
        <SELG /LECT/> <USLG /LECT/>
        <SELP /LECT/> <USLP /LECT/> )
        <SELV | FLSR ; FLSW ; HANG ; BHAN | >
        <DHAS $ OUTL ; CGLA ; BGLA ; GGLA ; GLAS ; FADE ffac $> >
  <GEOM <NAVI FREE>
    <NPTO npto>
    <PROJ ORTH>
    <REFE <FRAM> <BBOX> <CENT> >
    <FACE <HFRO> <SBAC> <SINT> <HBIS> <SHOW /LECT/> >
    <LINE <HEOU> <SSHA> <SFRE> <SPER> <SISO> <ANTI> <SBOU> <SIOU> >
    <POIN $ DOT dsiz ; SPHE ssiz ; SPHP <FACT fact> $>
    <SHRI sh <GROU> <NOUT> <ISOL> <HFAC> <PINS> <NODE> >
    <PINB <PARE> <CDES> <CPOI> <NORM> <JOIN> <NASN> <PASN> <DASN>
    <$ SOLI ; TRAN $> >
    <GPIN <DOMA> <SPHE> <CONE> <PRIS> <HEXA> <CDOM>
    <PENE> <PDOT> <CPOI> <NORM> <JOIN> <PASN>
    <$ ALen afac ; ASFA asfa $>
    <$ NLEN nfac ; NSFA nsfa $> >
    <INIT $ ASIS ; CGLA ; WIRE ; OUTL $ >
    <DEBR $ TRAJ ; TRCO $ >
```

```

<FLSR <DOMA> <SPHE> <CONE> <PRIS> <HEXA> <NORM> <COUP> <BLOQ> >
<FLSW <DOMA> <SPHE> <CONE> <PRIS> <HEXA> <NORM> <COUP> <BLOQ> >
<PCLD <POIN> <SEGM> <CLOU /LECT/> <COPO> <COSE>>
<LNKS <SHOW <$ ALL ; (link_type)>>
      <HIDE <$ ALL ; (link_type)>>
      <$ LENG fac ; SFAC sfac $>
      <JOIN> >
<VECT $ SCAV ; COLO ; SCCO $
      <vec_field> <SUPP /LECT/>
      <SCAL $ A6 ; A14 ; USER /PROG/ $>
      <$ LENG fac ; SFAC sfac $>
      <COSC $ COLS ; GRAY ; ICOL ; IGRA $ >
      <SIVE> >
<ISO $ LINE ; FILL ; FILI ; FELE <$ AVE ; MAX ; MIN ; AMAX $>;
      SMOO ; SMLI ; SMEL <$ AVE ; MAX ; MIN ; AMAX $>;
      SURF ; SULI $
      <SHIN> <FADE ffac>
      <iso_field> <SUPP /LECT/> <GAUS igauss | GAUZ igauss>
      <SCAL $ A1 ; A6 ; A14 ; USER /PROG/ $>
      <COSC $ COLS ; GRAY ; ICOL ; IGRA $ > >
<TEXT <NODE> <ELEM> <OBJE> <$VSCA ; NVSC$> <$ISCA ; NISC$>
      <HINF> <CAME> <DEBU> <PCON> >
<COLO ( SELE $ RED ; GREE ; BLUE ; CYAN ; MAGE ;
      YELL ; BLAC ; WHIT ; GR05 ; GR10 ;
      GR15 ; GR20 ; GR25 ; GR30 ; GR35 ;
      GR40 ; GR45 ; GR50 ; GR55 ; GR60 ;
      GR65 ; GR70 ; GR75 ; GR80 ; GR85 ;
      GR90 ; GR95 $
      APPL $ BGRN ; CENT ; BBOX ; IFAC ; ELOU ;
      SHAR ; FRED ; PERP ; VECT ; ISOE ;
      ISOL ; POIN ; NNUM ; ENUM ; ONAM ;
      TEXT ; INWI ; INOU ; TRAJ ; ISOD $ )
      <$ PAPE ; SCRIN $> >
<LIMA <ON>
      <LIGX ligx> <LIGY ligy> <LIGZ ligz>
      <LAMB $ LOW ; MEDI ; HIGH >
      <LDIF $ LOW ; MEDI ; HIGH >
      <LSPE $ LOW ; MEDI ; HIGH >
      <LSHI $ LOW ; MEDI ; HIGH >
      <LMAM $ LOW ; MEDI ; HIGH >
      ( SELE $ BRAS ; BRON ; PBRO ; CHRO ; COPP ;
      PCOP ; GOLD ; GOL2 ; PGOL ; PEWT ;
      SILV ; PSIL ; EMER ; JADE ; OBSI ;
      PEAR ; RUBY ; TURQ ; BLAP ; CYAP ;
      GREP ; REDP ; WHIP ; YELP ; BLAR ;
      BLR2 ; CYAR ; GRER ; REDR ; WHIR ;
      YELR $
      APPL $ MESH ; SELO ; /LECT/ $ ) >
<POVR <GLOB <GAMM gamm> <MTRC mtrc> >
      <DEFA <FINI <AMBI ambi> <DIFF diff> > >
      <LIGH ( <light_definition> ) >

```

```
<TXTR ( $ SELE 'texture_identifier' ; DEFI <texture_definition> $  
      APPL $ MESH ; SELO ; /LECT/ $ ) > >
```



### 15.9.1 Objects Menu Parameters

#### OBJE

Introduces the parameters relative to the **Objects** menu.

#### SELM

Choice of the elements (objects of mesh type) to be visualized.

#### USLM

Choice of the elements (objects of mesh type) to be hidden.

#### SELG

Choice of the groups (of elements) to be visualized.

#### USLG

Choice of the groups (of elements) to be hidden.

#### SELP

Choice of the points (objects of points type) to be visualized.

#### USLP

Choice of the points (objects of points type) to be hidden.

#### SELV

Select some “variable” objects, i.e. objects (of elements or of points type) whose composition varies in time rather than being topologically constant. At the moment, only the keywords **FLSR**, **FLSW**, **HANG** and **BHAN** are available. This directive is useful only in the generation of graphics in ‘batch’ mode. In fact, when using the graphics interactively one may access the same objects from the **SELG** or **SELP** menus. For example, the fluid nodes currently subjected to **FLSR** coupling conditions are available interactively in a special node group named “\_FLSR”.

#### FLSR

Select for visualization the fluid nodes currently subjected to **FLSR** coupling conditions. These nodes are drawn according to the selected drawing mode for points (see **GEOM POIN** directive).

#### FLSW

Select for visualization the fluid elements currently subjected to **FLSW** coupling conditions.

#### HANG

Select for visualization the currently hanging nodes in adaptivity. These nodes are drawn according to the selected drawing mode for points (see **GEOM POIN** directive).

#### BHAN

Select for visualization the currently boundary-hanging nodes in adaptivity. These nodes are drawn according to the selected drawing mode for points (see **GEOM POIN** directive).

**DHAS**

Allows to choose the way to draw the hidden (non-visualized) portions of the mesh. If omitted, they are ‘drawn’ as hidden (i.e., not drawn at all).

**OUTL**

Draw hidden mesh portions as element outlines (wireframe representation).

**CGLA**

Draw hidden mesh portions as colored glass.

**BGLA**

Draw hidden mesh portions as blue glass.

**GGLA**

Draw hidden mesh portions as green glass.

**GLAS**

Draw hidden mesh portions as colored glass (variation of **CGLA** that produces better results in some circumstances).

**FADE**

Draw hidden mesh portions as fading out objects.

**ffac**

Fading out factor (between 1.0 i.e. fully visible and 0.0 i.e. fully hidden).

### 15.9.2 Geometry Menu Parameters

#### GEOM

Introduces the parameters relative to the **Geometry** menu.

#### NAVI

Introduces the parameters relative to the *Navigation* sub-menu.

#### FREE

Choose the free camera navigation mode. By default, the rotating camera navigation mode is used.

#### NPTO

Introduces the parameters relative to the *Near Plane Tolerance* sub-menu.

#### npto

Near plane tolerance. By default, a value of 1.E-4 is used. During navigation inside bodies, it may be useful to increase this value (e.g. to 1.E-2).

#### PROJ

Introduces the parameters relative to the *Projection* sub-menu.

#### ORTH

Choose the orthogonal projection. By default, the perspective projection is used.

#### REFE

Introduces the parameters relative to the *References* sub-menu.

#### FRAM

Show the global reference frame.

#### BBOX

Show the bounding box.

#### CENT

Show the centre.

#### FACE

Introduces the parameters relative to the *Faces* sub-menu.

#### HFRO

Hide the front faces.

#### SBAC

Show the back faces.

#### SINT

Show the internal faces.

HBIS

Hide the back iso surfaces.

SHOW /LECT/

Force visualization of front and back faces belonging to the elements specified in the following /LECT/. This may be useful e. g. in a calculation with both a fluid (volumetric mesh) and a structure when the user wants to display both the structure and some iso-surfaces in the fluid. When ISO SURF or ISO SULI is selected, the code automatically disables the view of front faces (as a global setting), therefore the structure would not be drawn. By specifying SHOW LECT **stru** TERM the structural faces (both front and back faces) are forced to be drawn, thus obtaining the desired effect.

LINE

Introduces the parameters relative to the *Lines* sub-menu.

HEOU

Hide element outlines.

SSHA

Show sharp corners.

SFRE

Show free edges.

SPER

Show perpendicular contours.

SISO

Show iso surface outlines.

ANTI

Antialias lines.

SBOU

Show backface outlines (even when backfaces are not shown).

SIOU

Show internal face outlines (even when internal faces are not shown). This is a way to show the internal part of the mesh in wireframe representation.

POIN

Introduces the parameters relative to the *Points* sub-menu.

DOT

Render points as dots of size **psiz**. By default, points are rendered as dots of size 2.

**SPHE**

Render points as spheres of size **ssiz**.

**SPHP**

Render points as spheres of “physical” size (it must be possible to determine this size from some physical parameter associated with the point, e.g. the radius of a material particle).

**FACT**

Optional factor by which the physical radius of each sphere is multiplied for visualization purposes. By default it is 1.0.

**SHRI**

Introduces the parameters relative to the *Shrinkage* sub-menu.

**sh**

Shrink by a factor **sh**. This parameter is mandatory and must immediately follow the **SHRI** keyword.

**GROU**

Shrinkage will occur by element groups (which must have been defined), rather than element-by-element. If a group-related centerpoint has been specified in the **GROU** directive (see page C.61), then it is used for the shrinkage operation, otherwise shrinkage occurs around the average (unweighted) of the center points of the elements contained in the group. If a group-related shrink factor has been specified (see page C.61), it overrides the scene’s generic **sh** factor. Finally, if a group-related shift has been specified (see page C.61), it is applied as well during rendering.

**NOU**

Do not shrink outlines (they are shrunk by default if some shrinkage is activated).

**ISOL**

Shrink isolines.

**HFAC**

Shrink hidden faces.

**PINS**

Shrink pinballs and gpinballs.

**NODE**

Shrink node numbers.

**PINB**

Introduces the parameters relative to the *Pinballs* sub-menu.

**PARE**

	Show parent pinballs.
CDES	
	Show contacting descendents.
CPOI	
	Show contact points.
NORM	
	Show contact normals.
JOIN	
	Show contact joints.
NASN	
	Show nodal ASNs (assembled surface normals).
PASN	
	Show pinball ASNs (assembled surface normals) for the parent pinballs.
DASN	
	Show pinball ASNs (assembled surface normals) for the contacting descendent pinballs.
SOLI	
	Show pinballs as solid spheres.
TRAN	
	Show pinballs as semi-transparent spheres (this is the default).
GPIN	
	Introduces the parameters relative to the <i>Gpinballs</i> sub-menu.
DOMA	
	Render <i>all</i> the GPIN domains types (i.e. spheres, cones, prisms and hexahedra).
SPHE	
	Render the spherical GPIN domains.
CONE	
	Render the conical GPIN domains.
PRIS	
	Render the prisms GPIN domains.
HEXA	
	Render the hexahedra GPIN domains.

**CDOM**

Show the contacting GPIN domains.

**PENE**

Show the penetrations. By convention, and for representation clarity, each penetration is represented by two (equal and mutually opposite) arrows. The length of each arrow is equal to the amount of penetration in the geometric scale of the drawing. The arrows are located externally to the segment 12 that joins the two contacting points and in contact with each extremity of the segment. If the penetration is positive (real), then the arrows point towards each other and have their heads on the extremities of the segment. If the penetration is negative (fictitious), then the arrows point away from each other and have their tails on the extremities of the contact segment.

**PDOT**

Show the penetration rates. We use the same type of representation (by two mutually opposite arrows) as for the penetrations (see above). Each arrow representing the penetration rate has the same length as the penetration. In fact, it is not possible to take it equal (or proportional) to the penetration rate, since the range of the rate is arbitrary, unlike that of the penetration. The two arrows are pointing towards each other if the penetration rate is positive, otherwise they are pointing away from each other. Be aware that only the direction, and not the amplitude, of the arrows makes sense in this case!

**CPOI**

Show contact points. The “first” contact point (of each pcontact) is drawn in red, the “second” is drawn in green.

**NORM**

Show contact normals.

**JOIN**

Show contact joints.

**PASN**

Show gpinball ASNs (assembled surface normals) for the gpinballs.

**ALEN**

Introduces the parameters relative to the *ASN length* sub-menu.

**afac**

Scaling factor with respect to the default length of ASN vectors, which is 10% of the geometric model size.

**ASFA**

May be used in alternative to the **ALEN** directive to set the absolute length or the maximum length of the drawn ASN vectors. This is useful e.g. in animations when the size of the geometric model may vary considerably during a transient calculation.

**asfa**

Draw ASN vectors as arrows of uniform length (non-scaled), **asfa** represents the length of the drawn vectors.

**NLEN**

Introduces the parameters relative to the *Contact normal Length* sub-menu.

**nfac**

Scaling factor with respect to the default length of contact normal vectors, which is 10% of the geometric model size.

**NSFA**

May be use in alternative to the **NLEN** directive to set the absolute length or the maximum length of the drawn Contact normal vectors. This is useful e.g. in animations when the size of the geometric model may vary considerably during a transient calculation.

**nsfa**

Draw contact normal vectors as arrows of uniform length (non-scaled), **nsfa** represents the length of the drawn vectors.

**INIT**

Render the initial geometry of the model besides the current one.

**ASIS**

Render the initial geometry in the same way as the current one.

**CGLA**

Render the initial geometry as colored glass.

**WIRE**

Render the initial geometry as wireframe.

**OUTL**

Render the initial geometry as outline.

**DEBR**

Render the flying debris besides the current geometry.

**TRAJ**

Render the flying debris trajectories.

**TRCO**

Render the flying debris trajectories in shades of color. The color is related to the local debris velocity.

**FLSR**

Introduce rendering of quantities related to FLSR domains besides the current geometry.

**DOMA**



Render *all* the FLSR domains themselves (i.e. spheres, cones, prisms and hexahedra).

SPHE

Render the spherical FLSR domains.

CONE

Render the conical FLSR domains.

PRIS

Render the prisms FLSR domains.

HEXA

Render the hexahedra FLSR domains.

NORM

Render the FLSR domain normal(s). First normals are rendered in blue, second normals (if any) in green and third normals (if any) in red.

COUP

Render the FLSR couplings.

BLOQ

Render the FLSR blocked MC fluxes.

FLSW

Introduce rendering of quantities related to FLSW domains besides the current geometry.

DOMA

Render *all* the FLSW domains themselves (i.e. spheres, cones, prisms and hexahedra).

SPHE

Render the spherical FLSW domains.

CONE

Render the conical FLSW domains.

PRIS

Render the prisms FLSW domains.

HEXA

Render the hexahedra FLSW domains.

NORM

Render the FLSW domain normal(s). First normals are rendered in blue, second normals (if any) in green and third normals (if any) in red.

COUP

Render the FLSW couplings.

**BLOQ**

Render the FLSW blocked MC fluxes.

**PCLD**

Render the point clouds according to the following sub-keywords.

**POIN**

Render the cloud points. If this keyword is specified, all the points are rendered by default, irrespective of their cloud index. If you want to restrict the visualized clouds, use the **CLOU** keyword described next.

**SEGM**

Render the cloud segments. If this keyword is specified, all the segments are rendered by default, irrespective of their cloud index. If you want to restrict the visualized clouds, use the **CLOU** keyword described next.

**CLOU /LECT/**

Render only the points/segments belonging to the cloud indexes (e. g. 1, 2, etc.) listed in the following **/LECT/**.

**COP0**

Color the cloud points according to their cloud index: red for the first cloud, green for the second cloud etc. By default cloud points are shown in black or white depending on the choice of the default color scheme (for paper or for screen, respectively).

**COSE**

Color the cloud segments according to their cloud index: red for the first cloud, green for the second cloud etc. By default cloud segments are shown in black or white depending on the choice of the default color scheme (for paper or for screen, respectively).

**LNKS**

Render the links. All links are rendered by default, i.e. if no further keywords such as **SHOW** or **HIDE** are specified after the **LNKS** keyword.

**SHOW**

Set all links as hidden and then introduce selection of which links have to be shown.

**ALL**

All links are shown (this is the default).

**link\_type**

Specify one or more link types. Only the links of the selected type(s) will be shown. The list of available link types is given below (see also module **M\_LIAISORGA**).

**HIDE**

Set all links as shown and then introduce selection of which links have to be hidden.

ALL

All links are hidden.

**link\_type**

Specify one or more link types. The links of the selected type(s) will be hidden. The list of available link types is given below (see also module M.LIAISORGA).

LENG

Introduces the parameters relative to the *Length* sub-menu.

fac

Scaling factor with respect to the default length of links vectors, which is 10% of the geometric model size.

SFAC

May be use in alternative to the **LENG** directive to set the absolute length or the maximum length of the drawn links vectors. This is useful e.g. in animations when the size of the geometric model may vary considerably during a transient calculation.

sfac

Since links vectors are of uniform length (non-scaled), **sfac** represents the length of the drawn vectors.

JOIN

Show link joints (lines connecting all the nodes linked by a link).

## Link types

The available link types are:

RELA,COQM,FS ,BLOQ,NAVI,SOLI,BIFU,IMPA,  
CONT,ACCE,VITE,DEPL,DRIT,FLST,FSA ,FSTG,  
COLL,FLSR,GLIS,IMPA,FSR ,TUYM,TUYA,SH3D,  
MENS,PINB,MAP2,MAP3,MAP4,MAP5,MAP6,MAP7,  
FESE,SH3D,MOY4,MOY5,FLSW,PELM,GPIN,EDEF,  
SPEF,TBLO,TMEN,FLSS,FLSX,MEC1,MEC2,MEC3,  
MEC4,MEC5,RIC ,RIIL,RIIS,ADHE,HANG

### 15.9.3 Vectors Menu Parameters

#### VECT

Introduces the parameters relative to the **Vectors** menu.

#### SCAV

Show scaled vectors.

#### COLO

Show colored vectors.

#### SCCO

Show scaled colored vectors.

#### SUPP

Define, by means of the following */LECT/*, the list of the nodes that form the geometric support of the vector field. By default, the support is the entire mesh.

#### <vec\_field>

Choose the vector field to be represented (see below).

#### SCAL

Introduces the parameters relative to the *Scale* sub-menu.

#### A6

Use an automatic scale with 6 values. By default, an automatic scale with 14 values is used.

#### A14

Use an automatic scale with 14 values. This is the default.

#### USER

Use the fixed, user-specified scale given by */PROG/*.

#### LENG

Introduces the parameters relative to the *Length* sub-menu.

#### fac

Scaling factor with respect to the default vector length, which is 10% of the geometric model size.

#### SFAC

May be use in alternative to the **LENG** directive to set the absolute length or the maximum length of the drawn vectors. This is useful e.g. in animations when the size of the geometric model may vary considerably during a transient calculation.

#### sfac

The meaning of this value depends upon the chosen vector type representation. For scaled vectors (colored or not) **sfac** is the scale factor by which the physical vector norm is multiplied to obtain the length of the drawn vector. Thus, the length of a drawn vector may be associated with a physical value of the represented quantity, independently from the geometric model size and its variations, and on the chosen scale. For colored vectors of uniform length (non-scaled) **sfac** represents the length of the drawn vectors.

**COSC**

Introduces the color scheme to be used for the visualization of vectors.

**COLS**

Use colors (this is the default).

**GRAY**

Use a scale of grays.

**ICOL**

Use colors but invert the colors set (blue indicates the maximum value insted of minimum value).

**IGRA**

Use a scale of grays but invert the colors set (dark gray indicates the maximum value insted of minimum value).

**SIVE**

Show internal vectors in 3D models. By default, vectors are only traced on the visible faces, i.e. typically just on the envelope of 3D models.

#### 15.9.4 Choice of a vector field

##### Object

To select the vector field to be represented by the **VECT** directive described above.

##### Syntax:

```
VECT . . . <FIEL $ VITE ; VITG ; ACCE ; DEPL ; FINT ; FEXT ; FLIA ;  
          MASS ; VCVI ; FDEC $> . . .
```

##### VITE

Material or particle velocity (first **idim** components). This is the vector field represented by default, if the **FIEL** directive is omitted.

##### VITG

Mesh velocity in an ALE calculation (first **idim** components).

##### ACCE

Material or particle acceleration (first **idim** components).

##### DEPL

Displacement (first **idim** components).

##### FINT

Internal force (first **idim** components).

##### FEXT

Total external force (first **idim** components).

##### FLIA

External force due to liaisons (coupled links) (first **idim** components).

##### MASS

Nodal mass (first **idim** components).

##### VCVI

Material or particle velocity (first **idim** components) in Finite Volumes Cell Centred model. Note that these vectors are not represented at the nodes but at the “elements” (i.e., Finite Volumes) centroids.

##### FDEC

External force due to decoupled links (first **idim** components).

### 15.9.5 Iso Menu Parameters

#### ISO

Introduces the parameters relative to the **Iso** menu.

#### LINE

Show iso lines.

#### FILL

Show iso fill.

#### FILI

Show iso fill lines.

#### FELE

Show iso fill elements. By default, the average over all Gauss points is shown for each element. Other possible types of representations are the maximum value, the minimum value or the absolute maximum value over all Gauss points for each element. See the following optional keywords.

#### AVE

Choose the average value over all Gauss points of the element to fill the element. This is the default so this keyword may be omitted for brevity.

#### MAX

Choose the maximum value over all Gauss points of the element to fill the element.

#### MIN

Choose the minimum value over all Gauss points of the element to fill the element.

#### AMAX

Choose the absolute maximum value over all Gauss points of the element to fill the element.

#### SMO0

Show iso smooth.

#### SMLI

Show iso smooth lines.

#### SMEL

Show iso smooth elements. By default, the average over all Gauss points is shown for each element. Other possible types of representations are the maximum value, the minimum value or the absolute maximum value over all Gauss points for each element. See the following optional keywords.

#### AVE

Choose the average value over all Gauss points of the element to fill the element. This is the default so this keyword may be omitted for brevity.

**MAX**

Choose the maximum value over all Gauss points of the element to fill the element.

**MIN**

Choose the minimum value over all Gauss points of the element to fill the element.

**AMAX**

Choose the absolute maximum value over all Gauss points of the element to fill the element.

**SURF**

Show iso surfaces.

**SULI**

Show iso surfaces lines.

**SHIN**

Render iso surfaces as shiny surfaces. By default, iso surfaces are rendered as dull surfaces.

**FADE**

Draw iso surfaces as fading out objects (this is only applicable to **SURF** or **SULI**. Useful to see “through” the iso surfaces, e.g. in case of pressure waves in a blast.

**ffac**

Fading out factor (between 1.0 i.e. fully visible and 0.0 i.e. fully hidden).

**SUPP**

Define, by means of the following **/LECT/**, the list of the elements that form the geometric support of the iso field. By default, the support is the entire mesh.

**GAUS**

Allows to choose a specific Gauss point index (only for the quantities **CONT**, **EPST** and **ECRO**).

**igaus**

Number of the Gauss point chosen. The special value 0 means that the average over all Gauss points in the element is taken. This is the default, i.e. if neither **GAUS** nor **GAUZ** is specified. Note that this default is different from the default in curve plotting (Page ED.80) where 1 is assumed, i.e. the first Gauss Point is plotted.

**GAUZ**

Allows to choose a specific “lamina” of the (shell) element. The value is the index of the lamina through the thickness (only for the quantities **CONT**, **EPST** and **ECRO**). In this case, the code takes the average value of all Gauss Points belonging to the specified lamina.



igauz

Number of Gauss point through the thickness (i.e. index of the chosen lamina).

<iso\_field>

Choose the iso field to be represented (see below).

SCAL

Introduces the parameters relative to the *Scale* sub-menu.

A1

Use an automatic scale with just one value. This value is the average of the data extremes. By default, an automatic scale with 6 values is used.

A6

Use an automatic scale with 6 values. This is the default.

A14

Use an automatic scale with 14 values. By default, an automatic scale with 6 values is used.

USER

Use the fixed, user-specified scale given by /PROG/.

COSC

Introduces the color scheme to be used for the visualization of iso values.

COLS

Use colors (this is the default).

GRAY

Use a scale of grays.

ICOL

Use colors but invert the colors set (blue indicates the maximum value insted of minimum value).

IGRA

Use a scale of grays but invert the colors set (dark gray indicates the maximum value insted of minimum value).

### 15.9.6 Choice of an iso field

#### Object

To select the iso field to be represented by the ISO directive described above.

#### Syntax:

```
ISO . . . <FIEL $ CONT icon ; EPST ieeps; ECRO iecr ; DTEL ;
          VITE <icom> ; VITG <icom> ; ACCE <icom> ;
          DEPL <icom> ; FINT <icom> ; FEXT <icom> ;
          MASS <icom> ; FLIA <icom> ; FDEC <icom> ;
          MCPR ; MCRO ; MCTE ; MCCS ; MCMF icom ;
          MCP1 ; MCP2 ; PFSI ; PFMI ; PFMA ; EFSI ;
          SIGN isig ; ECRN iecr ;
          ADFT ;
          FAIL ;
          RISK irsk ;
          LFEL ; LFEV ;
          LFNO ; LFNV ; ILNO ; DTNO ;
          VCVI <icom> ;
          CERR ; MAXC ; ERRI ; CLEN ; ILEN          $> . . .
```

CONT icon

The icon-th component of the stress tensor.

EPST ieeps

The ieeps-th component of the cumulated strain.

ECRO iecr

The iecr-th component of the hardening parameters.

DTEL

Stability time step  $\Delta t_{\text{stab}}$  associated with the element. The *stability step* is the *critical step*  $\Delta t_{\text{crit}}$  estimated by the code (roughly the element length  $L$  divided by the speed of sound  $c$  in the element material) multiplied by the *safety coefficient*  $\phi$  (CSTA, by default 0.8):  $\Delta t_{\text{stab}} = \phi \Delta t_{\text{crit}} \approx \phi \frac{L}{c}$ .

VITE <icom>

Material or particle velocity: icom-th component if specified, else norm of the first idim components. This is the iso field represented by default, if the FIEL directive is omitted.

VITG <icom>

Mesh velocity in an ALE calculation: icom-th component if specified, else norm of the first idim components.

ACCE <icom>

Material or particle acceleration: `icom`-th component if specified, else norm of the first `idim` components.

DEPL `<icom>`

Displacement: `icom`-th component if specified, else norm of the first `idim` components.

FINT `<icom>`

Internal force: `icom`-th component if specified, else norm of the first `idim` components.

FEXT `<icom>`

External force: `icom`-th component if specified, else norm of the first `idim` components.

MASS `<icom>`

Nodal mass: `icom`-th component if specified, else norm of the first `idim` components.

FLIA `<icom>`

Liaison (coupled links) force: `icom`-th component if specified, else norm of the first `idim` components.

FDEC `<icom>`

Decoupled links force: `icom`-th component if specified, else norm of the first `idim` components.

MCPR

Finite volumes pressure (defined at nodes).

MCRO

Finite volumes density (defined at nodes).

MCTE

Finite volumes temperature (defined at nodes).

MCCS

Finite volumes sound speed (defined at nodes).

MCMF `icom`

Finite volumes mass fraction of the `icom`-th component. (defined at nodes).

MCP1

Finite volumes minimum pressure during the transient (defined at nodes).

MCP2

Finite volumes maximum pressure during the transient (defined at nodes).

PFSI

Overpressure due to FSI in the nodes of CLxx elements associated with an IMPE VISU material (see Page C.885) and with either COUP or DECO specified. These CLxx elements, used only for results visualization purposes, must be attached to structural elements (typically shells) embedded in a fluid and subjected to either FLSR or FLSW model of FSI.

**PFMI**

Minimum FSI overpressure in time at the node (see PFSI above).

**PFMA**

Maximum FSI overpressure in time at the node (see PFSI above).

**EFSI**

Extracted FSI fluid quantity to be visualized on the structure used as geometrical support (see EFSI sub-directive in the Primary interactive commands section above).

**SIGN isig**

Stress (isig-th component) in spectral elements (defined at nodes).

**ECRN iecr**

Hardening parameter (iecr-th component) in spectral elements (defined at nodes).

**ADFT**

Advection-diffusion temperature (defined at nodes).

**FAIL**

Failure level of the element which has been reached: 0 means virgin element, 1 means completely failed element and an intermediate values indicates a partially failed element.

**RISK irsk**

Risk due to the effects of an explosion. Risk values go from 0 (no risk) to 1 (full risk). Risk is estimated in the fluid field, and at the moment, it is only computed in JRC's FLxx elements and the cell centred finite volumes (VFCC). To activate this calculation, it is necessary to specify the RISK keyword in the calculation type (see page A.30). The irsk parameter indicates the "component" (i.e. the type) of risk considered: 1 means eardrum rupture risk, 2 means death risk. Be aware that when reading results from an Alice file (produced by a previous calculation with risk activation), it is mandatory to (re-)specify the whole RISK directive (in particular as concerns the PROB ... and LUNG ... subdirectives, see page A.30), because the risk is computed with the current values of the optional parameters.

**LFEL**

Logarithm in base 2 of the level factor associated with elements in the spatial time step partitioning algorithm.

**LFEV**

Logarithm in base 2 of the level factor associated with elements including the neighbours in the spatial time step partitioning algorithm.

**LFNO**

Logarithm in base 2 of the level factor associated with nodes in the spatial time step partitioning algorithm (defined at nodes).

**LFNV**

Logarithm in base 2 of the level factor associated with nodes including the neighbours in the spatial time step partitioning algorithm (defined at nodes).

**ILNO**

Flag indicating whether a node is (1) or is not (0) subjected to a link condition, used in the spatial time step partitioning algorithm (defined at nodes).

**DTNO**

Stability time step associated with nodes, used in the spatial time step partitioning algorithm (defined at nodes).

**VCVI** <icom>

Material or particle velocity in Finite Volumes Cell Centred model: **icom**-th component if specified, else norm of the first **idim** components.

**CERR**

Constant used in element error indicator calculation (adaptivity), see the **CERR** input keyword of the **ADAP** directive on page B.210.

**MAXC**

Maximum principal curvature of least-squares fitting function, used for element error indicator calculation (adaptivity).

**ERRI**

Element error indicator (adaptivity),

**CLEN**

Current characteristic element length used in error indicator calculations.

**ILEN**

Optimal (indicated) characteristic element length resulting from error indicator calculations.

The **MCPR**, **MCRO**, **MCTE**, **MCVI** and **MCMF** keywords are available only in calculations with finite volumes.

The **SIGN** and **ECRN** keywords are available only in calculations with spectral elements.

The **ADFT** keyword is available only in calculations with advection-diffusion.

The **FAIL** keyword is available only in calculations with the element erosion algorithm, see Page A.30.

The LFEL, LFEV, LFNO, LFNV, ILNO and DTNO keywords are available only in calculations with spatial time step partitioning (OPTI PART, see Page H.20).

The CERR, MAXC, ERRI, CLEN, ILEN keywords are available only in calculations with “true” adaptivity (see the ADAP directive on page B.210).

### 15.9.7 Text Menu Parameters

#### TEXT

Introduces the parameters relative to the **Text** menu.

#### NODE

Show node numbers.

#### ELEM

Show element numbers.

#### OBJE

Show object names.

#### VSCA

Show vectors scale. Note that the scale is automatically shown when vectors are visualized.

#### NVSC

Do not show vectors scale. This can be used to disable the automatic visualization of the vectors scale when vectors are rendered.

#### ISCA

Show iso scale. Note that the scale is automatically shown when isovalues are visualized.

#### NISC

Do not show iso scale. This can be used to disable the automatic visualization of the iso scale when isovalues are rendered.

#### HINF

Hide info.

#### CAME

Show camera values.

#### DEBU

Show debug info.

#### PCON

Show pinball contact indexes (in yellow) and gpinball contact indexes (in magenta).

### 15.9.8 Colors Menu Parameters

#### COLO

Introduces the parameters relative to the **Colors** menu. This menu allows to choose the colors of many graphical elements of the rendered scene, such as the background, the element outlines etc. To apply special colors to the model itself, or to parts of it, see the parameters relative to the **Lights/Mats** menu below. To apply a color, first it is selected by means of the **SELE** keyword, then it is applied to the desired graphical element by means of the **APPL** keyword. This sequence may be repeated as many times as necessary.

#### SELE

Introduces the selection of a color by means of the *Select color* sub-menu. The available colors are listed in the following Table. For greys, the number indicates the luminosity, i.e. GR05 is almost black, while GR95 is almost white.

Name	Color	Name	Color	Name	Color
RED	Red	GREE	Green	BLUE	Blue
CYAN	Cyan	MAGE	Magenta	YELL	Yellow
BLAC	Black	WHIT	White	GR05	Grey 05%
GR10	Grey 10%	GR15	Grey 15%	GR20	Grey 20%
GR25	Grey 25%	GR30	Grey 30%	GR35	Grey 35%
GR40	Grey 40%	GR45	Grey 45%	GR50	Grey 50%
GR55	Grey 55%	GR60	Grey 60%	GR65	Grey 65%
GR70	Grey 70%	GR75	Grey 75%	GR80	Grey 80%
GR85	Grey 85%	GR90	Grey 90%	GR95	Grey 95%

#### APPL

Introduces the application of the selected color by means of the *Apply it to* sub-menu. The available items to which a color may be applied are listed in the following Table.

Name	Item	Name	Item	Name	Item
BGRN	Background	CENT	Centre	BBOX	Bounding box
IFAC	Internal faces	ELOU	Element outlines	SHAR	Sharp corners
FRED	Free edges	PERP	Perpendicular contours	VECT	Vectors
ISOE	Iso surface edges	ISOL	Iso surface outlines	POIN	Points
NNUM	Node numbers	ENUM	Element numbers	ONAM	Object names
TEXT	Text	INWI	Initial wireframe	INOU	Initial outline
TRAJ	Debris trajectories	ISOD	Iso default color		

#### ISOD

Introduces the default color for isovalues. When iso-values are drawn and the user has selected only part of the mesh by the **SUPP** directive, the non-selected parts of the mesh are drawn in this color. The default value of this color is GR50 (i.e. 50% grey).

#### PAPE

Select colors suited for paper. This is the default. The scene background is white and the element outlines are black.

#### SCRN



Select colors suited for screen. The background becomes black and element outlines become white, in contrast with the **PAPE** (default) option where the background is white and the element outlines are black.

### 15.9.9 Lights and Materials Menu Parameters

#### LIMA

Introduces the parameters relative to the **Lights/Mats** menu. This menu allows to switch the light on, to choose some parameters relative to the lighting model and to apply special materials (colors) to the model, or to parts of it. To apply a material, first it is selected by means of the **SELE** keyword, then it is applied to the desired object by means of the **APPL** keyword. This sequence may be repeated as many times as necessary.

#### ON

Switches the light on. The light must be on to see the special material effects properly.

#### LIGX

Introduces the  $x$ -position of the light **ligx**. By default, **ligx=-1**, i.e. the light comes from the left.

#### LIGY

Introduces the  $y$ -position of the light **ligy**. By default, **ligy=1**, i.e. the light comes from the top.

#### LIGZ

Introduces the  $z$ -position of the light **ligz**. By default, **ligz=1**, i.e. the light comes from the front.

#### LAMB

Introduces the ambient light, which may be **LOW**, **MEDIUM** or **HIGH**. By default, the ambient light is **MEDIUM** for dull surfaces, **LOW** for shiny surfaces. The values for dull or shiny surfaces are somewhat different.

#### LDIF

Introduces the diffuse light, which may be **LOW**, **MEDIUM** or **HIGH**. By default, the diffuse light is **HIGH** for dull surfaces, **HIGH** for shiny surfaces. The values for dull or shiny surfaces are somewhat different.

#### LSPE

Introduces the specular light, which may be **LOW**, **MEDIUM** or **HIGH**. By default, the specular light is **LOW** for dull surfaces, **HIGH** for shiny surfaces. The values for dull or shiny surfaces are somewhat different.

#### LSHI

Introduces the light shininess, which may be **LOW**, **MEDIUM** or **HIGH**. By default, the light shininess is **LOW** for dull surfaces. This quantity is unused for shiny surfaces.

#### LMAM

Introduces the model ambient light, which may be **LOW**, **MEDIUM** or **HIGH**. By default, the model ambient light is **MEDIUM** for shiny surfaces. This quantity is unused for dull surfaces.

#### SELE

Introduces the selection of a material by means of the *Select material* sub-menu. The available materials are listed in the following Table.

Name	Material	Name	Material	Name	Material
BRAS	Brass	BRON	Bronze	PBR0	Polished bronze
CHRO	Chrome	COPP	Copper	PCOP	Polished copper
GOLD	Gold	GOL2	Gold 2	PGOL	Polished gold
PEWT	Pewter	SILV	Silver	PSIL	Polished silver
EMER	Emerald	JADE	Jade	OBSI	Obsidian
PEAR	Pearl	RUBY	Ruby	TURQ	Turquoise
BLAP	Black plastic	CYAP	Cyan plastic	GREP	Green plastic
REDP	Red plastic	WHIP	White plastic	YELP	Yellow plastic
BLAR	Black rubber	BLR2	Black rubber 2	CYAR	Cyan rubber
GRER	Green rubber	REDR	Red rubber	WHIR	White rubber
YELR	Yellow rubber				

#### APPL

Introduces the application of the selected material by means of the *Apply it to* sub-menu.

#### MESH

Apply the selected material to the whole mesh.

#### SELO

Apply the selected material to the currently selected objects.

#### /LECT/

Apply the selected material to the specified objects.

### 15.9.10 POV-Ray Menu Parameters

Notice: the present set of commands is experimental and still under development. Some of the described commands might not be implemented yet.

#### POVR

Introduces the parameters relative to the **POV-Ray** menu. These contain settings to be passed to the POV-Ray ray-tracing software via the generated `.pov` scene description file(s), when choosing a POV-Ray type of output.

#### GLOB

Introduces some global parameters to be set in POV-Ray's `global_settings` statement.

#### GAMM

This is POV-Ray's `assumed_gamma` parameter. By default it is set to 2.2, which is the advised value for Intel-based computed (according to Lohmueller).

#### MTRC

This is POV-Ray's `max_trace_level` parameter. By default it is set to 5.

#### DEFA FINI

Introduces some global POV-Ray parameters to be set in an initial `#default finish` statement.

#### AMBI

Ambient component of POV-Ray's default `finish`. By default it is set to 0.5.

#### DIFF

Diffuse component of POV-Ray's default `finish`. By default it is set to 0.5.

#### LIGH

Introduces the definition of light sources (if any) to be used in POV-Ray's scene. If no specific POV-Ray light sources are declared, then EPX's light source is tentatively translated to POV-Ray syntax. See *Defining POV-Ray lights* below for the syntax to define POV-Ray lights (`<light_definition>`).

#### TXTR

Introduces the definition of surface textures (if any) to be used in POV-Ray's scene. If no specific POV-Ray surface textures are declared, then EPX's (uniform) color for each element is tentatively translated to POV-Ray syntax.

#### SELE

Select a pre-defined POV-Ray texture by its identifier, i.e. by specifying the texture's name in quotes. For example: `SELE 'White_Marble'`, where the identifier `White_Marble` is defined in POV-Ray's `textures.inc` include file. This file is included automatically in the header of the generated `.pov` files. Note that POV-Ray identifiers are case-sensitive and must be encoded exactly, by respecting the letter case.

**DEFI**

Define a (new) POV-Ray texture by specifying its components, see `<texture_definition>` below. Only so-called plain textures may be defined via this input mechanism (not the more complex patterned or layered textures which are also available in POV-Ray). Typical components of a texture are a **pigment**, a **normal** and a **finish**. All components are optional and default values are used if they are not specified. In addition, a modifier such as **scale**, **rotate** and **translate** can be applied to a texture. See *Defining POV-Ray textures* below for the syntax to define POV-Ray textures (`<texture_definition>`).

**APPL**

Introduces the application of the selected or defined texture by means of the *Apply it to* sub-menu.

**MESH**

Apply the texture to the whole mesh.

**SELO**

Apply the texture to the currently selected objects.

**/LECT/**

Apply the texture to the specified objects.

**Defining POV-Ray lights**

A POV-Ray light may be defined by the syntax given below.

## Defining POV-Ray textures

A (plain) POV-Ray texture may be defined by the syntax given below, by specifying the associated pigment, normal and finish (all of which are optional) and an optional scaling, rotation and translation.

### Syntax:

```
DEFI < PIGM $ SELE 'pigment_identifier' ;
      COLO $ 'color_identifier' ;
      <R r> <G g> <B b> <T t> $ $ >
    < NORM ...                               >
    < FINI ...                               >
    < SCAL scal >
    < ROTA <RX rx> <RY ry> <RZ rz>          >
    < TRAN <TX tx> <TY Ty> <TZ tz>          >
```

#### DEFI

Introduces the definition of a plain POV-Ray texture.

#### PIGM

Introduces the definition of the texture's pigment.

#### SELE 'pigment\_identifier'

Select a pre-defined POV-Ray pigment by its identifier, i.e. by specifying the pigment's name in quotes.

#### COLO

Introduces the definition of the pigment's color.

#### 'color\_identifier'

Select a pre-defined POV-Ray color by its identifier, i.e. by specifying the color's name in quotes. The colors defined in POV-Ray's include file `colors.inc` are shown below and are listed in the following Table.

Color	R	G	B
Red	1.000000	0.000000	0.000000
Green	0.000000	1.000000	0.000000
Blue	0.000000	0.000000	1.000000
Yellow	1.000000	1.000000	0.000000
Cyan	0.000000	1.000000	1.000000
Magenta	1.000000	0.000000	1.000000
Clear	1.000000	1.000000	1.000000
White	1.000000	1.000000	1.000000
Black	0.000000	0.000000	0.000000

Table 17: Definition of POV-Ray's default colors (from `colors.inc`).

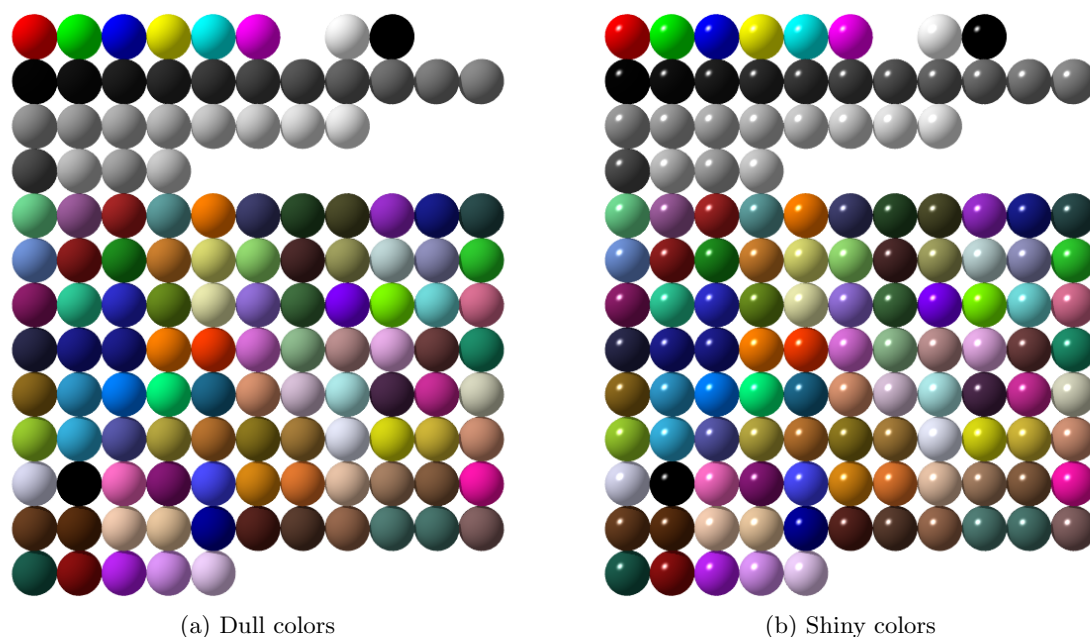


Figure 6: POV-Ray's default colors with a dull or shiny finish.

R

Color's red component in the RGB color system ( $0 \leq R \leq 1$ ). Default is 0.

G

Color's green component in the RGB color system ( $0 \leq G \leq 1$ ). Default is 0.

B

Color's blue component in the RGB color system ( $0 \leq B \leq 1$ ). Default is 0.

T

Color's transparency ( $0 \leq T \leq 1$ ). Default is 0 (fully opaque), 1 means completely transparent (invisible).

NORM

Introduces the definition of the texture's normal.

FINI

Introduces the definition of the texture's finish.

SCAL

Introduces the definition of the texture's scaling. Default scaling is 1.0.

ROTA

Introduces the definition of the texture's rotation.

RX



Texture's rotation angle in degrees around the *global X*-axis. Defaults is 0. Note that POV-Ray's rotation follow the left-hand rule.

RY

Texture's rotation angle in degrees around the *global Y*-axis. Defaults is 0. Note that POV-Ray's rotation follow the left-hand rule.

RZ

Texture's rotation angle in degrees around the *global Z*-axis. Defaults is 0. Note that POV-Ray's rotation follow the left-hand rule and that the *Z*-axis points towards the screen interior (left-handed reference) in POV-Ray.

TRAN

Introduces the definition of the texture's translation.

TX

Texture's translation along the *global X*-axis. Defaults is 0.

TY

Texture's translation along the *global Y*-axis. Defaults is 0.

TZ

Texture's translation along the *global Z*-axis. Defaults is 0. Note that the *Z*-axis points towards the screen interior (left-handed reference) in POV-Ray.

## 15.10 TITLES options

### Object

To define a set of parameters (globally indicated above as **<tpars>**) for the definition of a frame or AVI sequence containing titles, to be used during OpenGL rendering.

Once defined by a **TITL** directive, the first following **TRAC** directive will produce a titles frame (or AVI sequence) instead of the normal rendering of geometrical objects.

The color of the background and of the text used for the titles may be set by means of the **SCEN** directive described above (background color and text color, respectively).

Note that on EUROPLEXUS versions implemented on a non-OpenGL platform, this directive is simply (and entirely) ignored. This enhances portability of benchmark tests on the various platforms.

### Syntax:

**TITL**

**<TIT1 'text1'>**

**<TIT2 'text2'>**

**<TIT3 'text3'>**

**TIT1**

Introduces the text of the first title (**text1**), enclosed in quotes. This text appears centered in the upper part of the frame or sequence.

**TIT2**

Introduces the text of the second title (**text2**), enclosed in quotes. This text appears centered in the central part of the frame or sequence, and is therefore the “main” title.

**TIT3**

Introduces the text of the third title (**text3**), enclosed in quotes. This text appears centered in the lower part of the frame or sequence.

### Comments:

If none of the titles is given, the **TITL** directive deactivates the production of title frames (i.e. the next **TRAC** directive will produce regular geometry rendering).

Any omitted title is not represented in the titles frame.

## 16 GROUP V—The built-in OpenGL Graphical Visualizer

### Object

This Section describes the built-in OpenGL-based graphical visualizer. Note that *at the moment this tool is only available in the MS-Windows version* of EPX. Under Linux, an implementation of the visualizer may or may not be available depending on the version, but in any case the functionality is somewhat restricted with respect to the MS-Windows version. For example, the direct production of animations in the form of AVI files is not available under Linux because some of the necessary graphical libraries are not available.

The graphical visualizer can be used in two modes:

- **Interactive mode.** The tool allows to obtain a graphical representation of the computational model (mesh) and of computed results, e.g. in the form of vectors or iso-values. Such visualizations can be obtained at any moment during the numerical simulation by making use of the interactive code piloting commands described in Section O.10, in particular the **FREQ**, **TFRE** and **GO** commands in order to advance the solution up to the desired time or time step. In this mode, the user interacts with the visualizer by means of a set of menus, keystrokes or mouse events.
- **Batch mode.** All visualizations that can be obtained interactively can also be obtained in “batch” (non-interactive) mode, by means of the commands described in Sections O.60 (**CAME** for piloting the camera), O.70 (**SLER** for camera interpolation) and O.80 (**SCEN** to set the scen parameters and options). The commands should be enclosed in a **PLAY ... ENDPLAY** environment, as described in Section I.24. This is very handy in order to produce automatically a set of visualizations (raster images and/or animations) of the computed results, since the graphical commands may be embedded in the normal .EPX input file and executed in unattended mode whenever necessary.

It should always be kept in mind that the scope of the built-in graphical visualizer is not to rival the full-fledged powerful post-processors that can be used with EPX. For industrial use the ParaView software (or the Salome platform) are the obvious choice. However, during the development of the EPX code it may be handy to have access to a rapidly-programmable built-in visualizer that can be readily customized in order to obtain the exact desired graphical representation of some new features or model being developed, so as to help in the development and testing/debugging processes. Also, occasional users or students dealing with small academic examples may find it useful to exploit the embedded graphical visualizer, since it avoids the need of installing any additional software besides EPX itself.

An introduction to (an early version of) the graphical visualizer is given in reference [208]. A list of all menus and commands is also provided below with short explanations of each functionality.

## 16.1 Preparing to use the built-in graphical visualizer

In order to have access to interactive graphics capabilities, EPX must (at least formally) be run in “interactive’ mode. This is accomplished by inserting in the input file (which has an extension .EPX) the following directive:

```
CONV WIN
```

where the keyword **CONV** stays for “conversationnel” (the French word for interactive) and the following one designates the type of graphical platform.

Historically, various platforms have been supported such as **TEKT** (Tektronix-like or PLOT-10 terminal) and **WIN** (MS-Windows graphics, originally implemented via QuickWin). For the purpose of the current discussion on 3D rendering, the choice is rather irrelevant since the OpenGL-based module operates separately and independently from the previous old-fashioned graphical routines, so users may want to just use **WIN** as shown above. The above directive must be inserted immediately after the problem title in the input file, i.e. before the “problem dimensioning” section of the file. For full details, see page A.25.

## 16.2 Interactive code execution

When interactive execution is chosen, EPX reads the input data-set as usual, performs step 0 to initialise the computation, then prompts the user for commands from the keyboard with the phrase:

COMMANDE ?

The user can then issue various commands and subcommands from the keyboard in order to pilot the computation. For example, he can ask the program to perform a certain number of steps, then to halt again for further commands. Each time the calculation is halted, the current mesh can be visualized and information concerning the computation (time step, CPU time, etc.) can be printed. To activate the 3D rendering, just type the command:

`trac rend`

(note that both input directives and interactive commands are case-insensitive in EPX). This should open a graphical OpenGL window visualizing the mesh in the current configuration. The default initial position of the “observer” is along the positive  $z$ -axis, and it looks towards the geometric centre of the model.

As long as the graphical window stays open, all input is “grabbed” by it. In particular, text typed at the keyboard is interpreted as graphic commands and does not go to the “normal” EPX console window as usual. To return control to the main EPX application, and to continue the calculation, close the graphical window by clicking on the “cross” in the top right corner of the border (or by using the pop-up Menu, see below). In the current implementation, just one graphical window may be open at any given time.

The initial size of the graphical window is 500 by 500 pixels. The window may be resized, and the shown object will resize accordingly.

By default, the viewing model is such that the observer always points its view towards the geometrical centre of the object. This is called the “rotating camera” model. Upon motion, the user may imagine that either the observer rotates around the model, or that the observer is fixed and the model is rotated, whatever seems most natural to him/her.

Rotation and other changes in the view parameters may be obtained in two ways: with keyboard command or by means of menu commands. We first list all the available keyboard commands, then we describe the menus.

### 16.3 Keyboard commands

The complete list of available keyboard command (at the moment of writing) is given in the following Table (see also Section O.15).

Key	Action	Amount	CTRL-	CTRL/SHIFT-	SHIFT-
0 (zero)	Reset default view	—	—	—	—
Up arrow	Rotate camera “up”	5°	1°	10°	30°
Down arrow	Rotate camera “down”	5°	1°	10°	30°
Left arrow	Rotate camera “left”	5°	1°	10°	30°
Right arrow	Rotate camera “right”	5°	1°	10°	30°
PgUp	Rotate camera anticlockwise	5°	1°	10°	30°
Ins	Rotate camera clockwise	5°	1°	10°	30°
b	Translate camera backwards	$R/2$	$R/10$	$R$	$3R$
f	Translate camera forwards	$R/2$	$R/10$	$R$	$3R$
i	Zoom camera in (without moving)	$\times 1.2$	$\times 1.1$	$\times 1.5$	$\times 2.0$
o	Zoom camera out (without moving)	$\times 1.2$	$\times 1.1$	$\times 1.5$	$\times 2.0$
r	Move camera rightwards (free mode only)	$R/2$	$R/10$	$R$	$3R$
l	Move camera leftwards (free mode only)	$R/2$	$R/10$	$R$	$3R$
u	Move camera upwards (free mode only)	$R/2$	$R/10$	$R$	$3R$
d	Move camera downwards (free mode only)	$R/2$	$R/10$	$R$	$3R$

Table 18: Available keystrokes.

The keystrokes are *not* case sensitive: for example, **b** has the same effect as **B**. Some keystrokes (**r**, **l**, **u** and **d**) have effect only when the camera is set in “free navigation” mode (not in “rotating” mode). In order to change the camera navigation mode interactively, press the right mouse button in order to bring up the interactive menu and then use the Geometry → Navigation sub-menu.

The description of motions refers to the “camera” model, i.e. to the ideal observer. If preferred, the user may think of the same motion as applied to the object that is being viewed, by just inverting the “sign” of the motion. For example, the “Left arrow” key rotates the observer to the left or, alternatively, the object to the right.

Each motion has pre-defined amounts: 5 degrees for rotations,  $1/2$  of the object radius for translations, 20% magnification/reduction for zooming. Smaller amounts may in some cases be obtained by pressing the Control key (CTRL), larger ones by the Control-Shift keys (CTRL-SHIFT), and even larger amounts by the Shift key (SHIFT) in conjunction with any of the above described keys. For example, the keys combination CTRL-PgUp turns the camera by 1 degree, CTRL/SHIFT-PgUp turns it by 10 degrees and SHIFT-PgUp turns it by 30 degrees.

## 16.4 Mouse-driven motions

A simple and intuitive way of moving the model (or the observer) which is alternative to the keyboard commands described above is by means of the mouse (see also Section O.15).

With the navigation mode set in “rotating” camera mode, by pressing the left mouse button while the pointer is inside the graphical window, a sort of “virtual trackball” is activated. The object “follows” any subsequent motions of the mouse by rotating around its centerpoint in the corresponding direction.

The effects that may be obtained with the left mouse button are summarized below.

Action	Effect
Press left button	The object starts “following” the mouse cursor by rotating around its centerpoint
Release button while not moving	The object stops rotating, in the final position reached during the previous motion
Move button while pressed	The object follows the mouse motion
Release button while moving	The object continues to “spin” around its centerpoint along the last rotation axis that was active immediately before releasing the button (sort of continuous animation)
Release button while not moving	The object stops rotating, in the final position reached during the previous motion

Table 19: Available mouse events.

Some experimentation will make readily clear what the above somewhat complicated verbal descriptions mean.

A situation which may arise with inexperienced users is that, after using the mouse to rotate the body, it does not stop but it continues “forever” its rotation. This happens when the mouse button is released while still moving (though slowly) the mouse. To stop a rotating object, the following technique may be used: just give a single, quick “click” of the mouse in the window (i.e. press and then immediately release the button) by making sure that you do not move the mouse meanwhile.

## 16.5 The Main menu

The bulk of the user interface of the built-in graphics module is represented by a pop-up menu system that is activated by pressing the right mouse button while the pointer is inside the graphical window.

This stems from one of the fundamental choices that have been done during the design of the module, i.e. that of ensuring as wide as possible portability over different platforms. To achieve portability, use is made of the standard package GLUT (OpenGL Utility Toolkit). The only GUI features offered by this package is a hierarchy of pop-up menus. Other features that are common in many graphical environments such as a menu bar, buttons, labels, dials etc. are not supported. Although this is a serious limitation, it is believed that a quite usable interface has been realized, thanks to a careful choice of the arrangement of commands in the menus.

Each menu entry may represent either a command, or the “root” of a lower-level (or sub-) menu. In the latter case, a right-pointing arrow appears after the menu label.

### Syntax:

```
Objects      --> ...
Geometry     --> ...
Vectors      --> ...
Isovalues    --> ...
Text         --> ...
Colors       --> ...
Lights/Mats  --> ...
Win/Copy     --> ...
Quit
```

In the following, menus and sub-menus are highlighted in bold for clarity, e.g. **Objects**, while commands are emphasized, e.g. *Quit*. The entries of the main menu are:

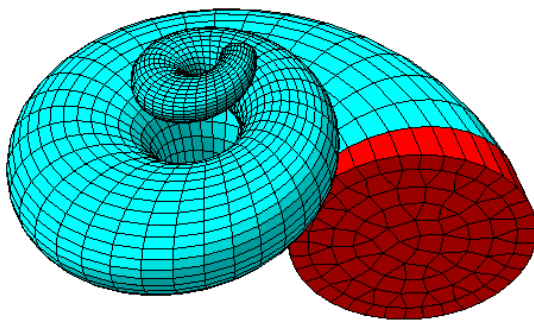
- **Objects**. Allows to select or unselect parts of the mesh.
- **Geometry**. Allows to choose details of the geometrical representation of the mesh: surfaces, lines, points, etc.
- **Vectors**. Allows to present some results in the form of vectors (arrows) on the mesh.
- **Isovalues**. Allows to present some results in the form of iso-lines or iso-surfaces on the mesh.
- **Text**. Allows to choose some text to be displayed such as node numbers, element numbers or object names.
- **Colors**. Allows to choose the colors used to represent various parts of the scene.
- **Lights/Mats**. Allows to add lighting and materials to the geometrical representation of the mesh.
- **Win/Copy**. Allows to set the graphical window size to some pre-determined values and to produce a bitmap copy of the graphical window in various raster formats. Note that the graphical window can also be resized by hand, maximized, minimized etc. with the normal window resizing techniques.



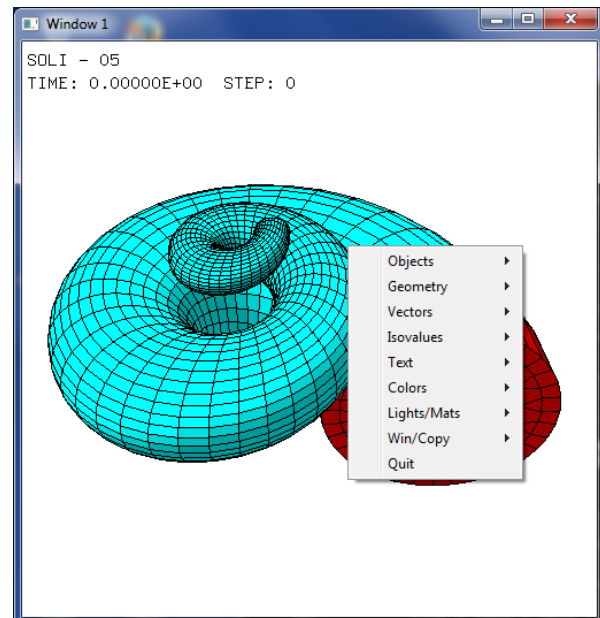
- *Quit*. Quits gracefully the graphical visualizer and returns control to the EPX program proper. It is equivalent to clicking on the “cross” in the upper right border of the window, but more stable. Caution should be taken not to close the graphical window while a menu is displayed, else the application may crash (blame GLUT for that).

The contents and functionality of each sub-menu is shortly described and illustrated in the following Sections.

SOLI - 05  
TIME: 0.00000E+00 STEP: 0



(a) Standard view



(b) Right-Click shows the Main menu

Figure 7: The interactive graphical window.

## 16.6 The Objects menu

### Object:

This menu allows to choose the objects to be visualized. Basically, objects are named lists of nodes and elements. They are optionally defined in the data structure that interfaces the OpenGL visualization module with the EPX program proper. If no objects are specified in the data interface, then the whole mesh is visualized and the user cannot select or unselect parts of the mesh. If some objects are specified, then the whole mesh is still visualized *by default*, but the user may choose to select or unselect some of these objects. The available sub-menus are:

### Syntax:

```

Objects --> Show all
           Hide all
           Select meshes  --> <list of mesh objects>
           Unselect meshes --> <list of mesh objects>
           Select groups  --> <list of group objects>
           Unselect groups --> <list of group objects>
           Select points  --> <list of points objects>
           Unselect points --> <list of points objects>
           Draw hidden as --> Hidden
                           Outlines
                           Colored glass
                           Blue glass
                           Green glass
                           Glass
                           Fading

```

- *Show all* (the default). This may be used explicitly to return to full mesh visualization after a partial visualization.
- *Hide all*. This deselects all objects (nothing is visualized). This allows to clear up (re-initialize) the object selection process.
- **Select meshes**. This opens a sub-menu containing a list of all the objects of mesh type that are available for selection. An object of mesh type is one that is composed by elements, besides nodes. To select an object, just click on the corresponding menu entry. Note that this menu appears only when there are objects of mesh type.
- **Unselect meshes**. This opens a sub-menu containing a list of all the objects of mesh type that are available for un-selection. An object of mesh type is one that is composed by elements, besides nodes. To un-select an object, just click on the corresponding menu entry. Note that this menu appears only when there are objects of mesh type.
- **Select groups**. This opens a sub-menu containing a list of all the “named element groups” and/or “named node groups” that are available for selection. A user can define such named groups directly in EPX by means of the `COMP GROU` and `COMP NGRO` directives, see page C.61 and C.62, respectively. To select a group, just click on the corresponding menu entry. Note that this menu appears only when there are named element or node groups.

- **Unselect groups.** This opens a sub-menu containing a list of all the “named element groups” and/or “named node groups” that are available for un-selection. A user can define such named groups directly in EPX by means of the `COMP GROU` and `COMP NGRO` directives, see page C.61 and C.62, respectively. To un-select a group, just click on the corresponding menu entry. Note that this menu appears only when there are named element or node groups.
- **Select points.** This opens a sub-menu containing a list of all the objects of points type that are available for selection. An object of points type is one that is composed just by nodes (no elements). To select an object, just click on the corresponding menu entry. Note that this menu appears only when there are objects of points type.
- **Unselect points.** This opens a sub-menu containing a list of all the objects of points type that are available for un-selection. An object of points type is one that is composed just by nodes (no elements). To un-select an object, just click on the corresponding menu entry. Note that this menu appears only when there are objects of points type.
- **Draw hidden as.** This opens a sub-menu that allows to specify how the hidden objects (of mesh type), i.e. those not currently selected, are to be drawn.

Note that individual objects may not be de-selected once they have been selected. This is due to the fact that objects are not necessarily disjoint. To deselect an object among a group of selected objects, first deselect all objects (Hide all) and then select the other members of the group.

### 16.6.1 The Draw hidden as menu

#### Syntax:

```
Objects --> Draw hidden as --> Hidden
                                Outlines
                                Colored glass
                                Blue glass
                                Green glass
                                Glass
                                Fading
```

This sub-menu allows to specify how the hidden objects (of mesh type), i.e. those not currently selected, are to be drawn.

- *Hidden* (the default). Unselected parts of the mesh are hidden, i.e. not drawn.
- *Outlines*. Only the elements outlines (not the faces) are drawn, i.e. a wireframe representation of the mesh is drawn.
- *Colored glass*. The element faces are drawn as partially transparent with a glass-like appearance that maintains the original color of the element.
- *Blue glass*. The element faces are drawn as partially transparent with a bluish glass-like appearance.
- *Green glass*. The element faces are drawn as partially transparent with a greenish glass-like appearance.

- *Glass.* The element faces are drawn as partially transparent with a glass-like appearance.
- *Fading.* The element faces are drawn as fading away.

To draw just the edges (sharp corners, free edges, perpendicular contours) of the hidden parts of the mesh use the corresponding switches in the **Geometry** → **Lines** menu, described in Section V.70.

## 16.7 The Geometry menu

### Object:

The Geometry menu allows to choose the way in which the geometrical appearance of the model is represented. By default, the body is rendered as a solid (opaque) surface composed by elements (subdivided into facets) with the border (outline) of each element highlighted.

Note incidentally that each command of the type *Show xxx* is of the toggle type. By activating the command, the menu entry changes into *Hide xxx* (the next time the menu is shown), and reciprocally. The available sub-menus are described hereafter.

### Syntax:

```

Geometry --> Navigation          --> ...
           Near Plane Tolerance --> ...
           Projection            --> ...
           References            --> ...
           Faces                 --> ...
           Lines                 --> ...
           Points                --> ...
           Shrinkage             --> ...
           Pinballs              --> ...
           Initial geometry      --> ...
           Flying debris         --> ...
           FLSR domains          --> ...
           FLSW domains         --> ...
           Gpinballs             --> ...
           Links (coupled)       --> ...

```

- **Navigation.** This allows to specify details of the navigation model.
- **Near plane tolerance.** This allows to choose the tolerance used for the near clipping plane.
- **Projection.** This allows to choose the type of geometrical projection.
- **References.** This allows to visualize the reference frame, the bounding box and other visual hints together with the geometrical model.
- **Faces.** This allows to choose how to visualize the various faces of the model.
- **Lines.** This allows to choose how to visualize the various lines of the model.
- **Points.** This allows to choose how to visualize the various points of the model.
- **Shrinkage.** This allows to activate element shrinkage: each element is drawn as “shrunk” by a chosen amount around its centroid, so that the different elements of the mesh can be seen individually.
- **Pinballs.** This sub-menu appears only if there are pinballs in the computational model, and it allows to visualize the pinballs used for contact.

- **Initial geometry.** This sub-menu appears only if the user has given the **DEFO** optional keyword in the **TRAC** directive used to open the graphical window (**TRAC ... DEFO ... REND**). It allows to visualize the initial (undeformed) configuration of the computational model, in addition to the current (deformed) configuration which is drawn by default.
- **Flying debris.** This sub-menu appears only if there are flying debris in the computational model, and it allows to visualize the debris in the form of particles.
- **FLSR domains.** This sub-menu appears only if the **FLSR** model of FSI is used in the computational model. It allows to visualize the geometrical domains used in order to detect fluid-structure interaction by the **FLSR** model.
- **FLSW domains.** This sub-menu appears only if the **FLSW** model of FSI is used in the computational model. It allows to visualize the geometrical domains used in order to detect fluid-structure interaction by the **FLSW** model.
- **Gpinballs.** This sub-menu appears only if there are generalized pinballs (see the **GPIN** directive) in the computational model, and it allows to visualize the generalized pinballs used for contact.
- **Links (coupled).** This sub-menu appears only if there are coupled links in the computational model and if the user has activated the **OPTI LINK VISU** option. It allows to visualize the link directions and to visually connect the linked nodes.

The various sub-menus are described next.

### 16.7.1 The Navigation menu

#### Syntax:

```
Geometry --> Navigation --> Rotating camera
                               Free camera
                               Write camera
                               Update cameras list
                               Select camera      --> <list of camera files>
                               Focus on          --> ...
```

This sub-menu allows to specify details of the navigation model.

- *Rotating camera* (the default). In this navigation mode, the computational model is bound to rotate around its centroid. In other words, the observer's view vector always points towards the model centroid. This is the "safest" navigation mode because some parts of the mesh will always be visible, irrespective of the navigation commands that the User may give.
- *Free camera.* In this navigation mode, the camera motion is completely free. The User must be aware that, if the view vector points away from the model, nothing will be shown in the graphical window.
- *Write camera.* This command writes on the current directory a small text file **epx\_cam.n.txt**, where **n** is an integer starting at 1 in each **EPX** session and being increased each time the command is issued. The file contains the commands that would be needed in **EPX** to set the viewing camera at the exact position currently displayed in the graphical window

(see the **CAME** directive, which can be quite complex). This feature is very handy in order to “store” the camera position(s) most suited to represent a given computational model. The user may then include the text in the EPX command file in order to obtain again the desired view.

- *Update cameras list.* This command reads from the current directory any text files `epx_cam_n.txt` present, and stores the corresponding cameras in a list. The user may then select one of the available cameras from the list.
- **Select camera.** This command opens a sub-menu containing the list of available cameras stored in the current directory (the camera files are typically produced previously by the *Write camera* command). The list must have been previously built by the **Update cameras list** command.
- **Focus on.** This command causes the focus, i.e. the point onto which the observer’s view vector is pointing, to be changed from the default (which is the centroid of the mesh). The focus can be set on a named part of the mesh such as an object, group or point.

### The Focus on menu

#### Syntax:

```
Geometry --> Navigation --> Focus on --> All
                                         Meshes --> <list of mesh objects>
                                         Groups --> <list of group objects>
                                         Points --> <list of points objects>
```

This sub-menu allows to specify the part of the geometrical model on which focus should be set. It can be useful in case the User wants to concentrate attention on small details of the mesh which would be difficult to localize in the default view of the model, which is focused on the (whole) model’s centroid.

- *All* (the default). Focus goes on the centroid of the whole mesh. This command can be useful to re-set the default focus after the focus has been previously moved to a specific part of the model.
- **Meshes.** This opens a sub-menu containing a list of all the objects of mesh type that are available for selection. An object of mesh type is one that is composed by elements, besides nodes. To select an object, just click on the corresponding menu entry. The focus will be set on the centroid of the chosen object. Note that this menu appears only when there are objects of mesh type.
- **Groups.** This opens a sub-menu containing a list of all the “named element groups” that are available for selection. A named element group is one that is composed by elements, besides nodes. To select a group, just click on the corresponding menu entry. The focus will be set on the centroid of the chosen group. Note that this menu appears only when there are named element groups in the model.
- **Points.** This opens a sub-menu containing a list of all the “named node groups” that are available for selection. A named node group is one that is composed by nodes (no elements). To select a group, just click on the corresponding menu entry. The focus will be set on the centroid of the chosen group. Note that this menu appears only when there are named node groups in the model.

### 16.7.2 The Near plane tolerance menu

#### Syntax:

```
Geometry --> Near plane tolerance --> 1.E-2
                                         1.E-3
                                         1.E-4
                                         1.E-5
```

This sub-menu allows to choose the tolerance for the near clipping plane among one of the values proposed. Changing the value may sometimes help in ameliorating some views.

### 16.7.3 The Projection menu

#### Syntax:

```
Geometry --> Projection --> Perspective
                              Orthogonal
```

This sub-menu allows to choose the type of geometrical projection used in the rendering process.

- *Perspective* (the default). A perspective projection is adopted. This is the most realistic projection.
- *Orthogonal*. An orthogonal projection is adopted, in which parallel lines always remain parallel. This type of projection can be sometimes useful to check some features of the geometrical model, but it is less natural than the Perspective projection. Note also that zooming is not available under Orthogonal projection.

### 16.7.4 The References menu

#### Syntax:

```
Geometry --> References --> Show reference frame
                              Show bounding box
                              Show center
```

The References menu may be used to visualize some geometrical elements that help understanding the positioning of the object in space.

- *Show reference frame*. This activates the visualization of the global reference frame in which all coordinates are expressed, represented by three colored axes: red for the  $x$ -axis, green for the  $y$ -axis and blue for the  $z$ -axis.
- *Show bounding box*. This visualizes the bounding box of the body, i.e. a parallelepiped aligned with the global reference planes which exactly contains the body.
- *Show center*. This visualizes the geometric center of the body, in the form of a small cube. All rotations of the model described in Section V.30 occur around this point. The center can be changed by the **Focus on** menu.



### 16.7.5 The Faces menu

#### Syntax:

```
Geometry --> Faces --> Hide front faces
                        Show back faces
                        Show internal faces
                        Hide back iso surfaces
```

The Faces menu allows to choose the way in which the element faces are rendered.

- *Hide front faces.* This disables external face filling and thus produces a wireframe representation of the body. By default, front faces are rendered as filled surfaces.
- *Show back faces.* This enables back face filling. By default, back faces are not rendered (they appear as fully transparent surfaces).
- *Show internal faces.* This has as an effect that not only the external surface of the body, but also the internal element faces, are shown. To actually see the difference with respect to the standard representation, for example a wireframe representation may be chosen, as explained above. This type of representation may be useful to highlight the internals of the body (e.g. by making the surface partially transparent, and in some particular iso-values or vectors representations). **Hide back iso surfaces.** This enables culling of back-facing iso surfaces. By default, these surfaces are not culled, unlike the other surfaces (element faces, for example).

### 16.7.6 The Lines menu

#### Syntax:

```
Geometry --> Lines --> Hide element outlines
                        Show sharp corners
                        Show free edges
                        Show perp contours
                        Show iso surface outlines
                        Antialias lines
                        Show backface outlines
                        Show internal outlines
```

The Lines menu allows to choose the way in which various types of lines are rendered.

- *Hide element outlines.* This represents the surface without highlighting the subdivision into finite elements. Note, however, that if in addition front face filling has been disabled by the **Faces** → *Hide front faces* command, then the body will totally disappear, which is probably not the desired effect.
- *Show sharp corners.* This highlights any “sharp corners” in the model. A sharp corner is a segment common to two external faces whose normals form an angle greater than a predefined value (60° by default). Sharp corners are only computed in 3D.

- *Show free edges.* This highlights any “free edges” in the model. A free edge is a segment on the border of an external face that has no adjacent faces. In 2D it is just the perimeter of the model. In 3D it may be useful to visualize the borders of flat surfaces (shells, for example).
- *Show perp contours.* This highlights any “perpendicular contours” in the model. A perpendicular contour is a segment common to two external faces, of which one is visible and the other is not visible in the current view. This depends on the current view and therefore it must be recomputed as the model rotates, so it is quite expensive.
- *Show iso surface outlines.* This enables visualization of the outlines of the elementary polygons (triangles and quadrilaterals) that form the iso surfaces.
- *Antialias lines.* This activates line anti-aliasing (smoothing) for the representation of lines such as element outlines, sharp corners, free edges, and isolines. This reduces the jagged appearance of these lines, but at the expense of using thicker lines, so the effect appears of dubious utility.
- *Show backface outlines.* This activates rendering of the outlines of backfaces.
- *Show internal outlines.* This activates rendering of the outlines of internal faces. If in addition the front faces are hidden by the **Faces** → *Hide front faces* command, then the body is rendered as a wireframe including the “internal” wires.

### 16.7.7 The Points menu

#### Syntax:

```
Geometry --> Points --> Dots 1
                        Dots 2
                        Dots 4
                        Dots 8
                        Spheres 1
                        Spheres 2
                        Spheres 4
                        Spheres 8
                        Spheres physical
```

The Points menu allows to choose the way in which various types of points are rendered. These include material points but also SPH particles, for example. Two basic representations are possible: by means of dots or by means of spheres. A dot is just a square block of color while a sphere is a full-fledged sphere rendered as a mesh of facets, with their own outlines (if element outlines are activated in the scene). Dot sizes vary from 1 to 8 in pixel units. Sphere sizes also vary from 1 to 8, but in an arbitrary unit which corresponds to 1/128 of the model size. By default, points are rendered as dots of size 4. There is also the possibility to represent spheres by their “physical” size: in this case the radius of the sphere corresponds to the physical value it has in the model. For example, in the case of SPH particles, this is the value assigned in the CBIL RAYO directive.

- *Dots 1.* This represents the dots as square dots of size 1 (the smallest possible).
- *Dots 2.* This represents the dots as square dots of size 2.

- *Dots 4*. This represents the dots as square dots of size 4.
- *Dots 8*. This represents the dots as square dots of size 8 (the largest possible).
- *Spheres 1*. This represents the dots as spheres of size 1 (the smallest possible).
- *Spheres 2*. This represents the dots as spheres of size 2.
- *Spheres 4*. This represents the dots as spheres of size 4.
- *Spheres 8*. This represents the dots as spheres of size 8 (the largest possible).
- *Spheres physical*. This represents the dots as spheres of size equal to the diameter of the dot. In the case of material points this is the diameter which has been assigned to the corresponding elements by the COMP EPAI directive. In the case of SPH particles, this is the value assigned in the CBIL RAYO directive.

### 16.7.8 The Shrinkage menu

#### Syntax:

```

Geometry --> Shrinkage --> No shrinkage
                               Shrink by groups
                               Shrink 99%
                               Shrink 80%
                               Shrink 60%
                               Shrink 40%
                               Shrink outlines
                               Shrink isolines
                               Shrink hidden faces
                               Shrink pinballs
                               Shrink node numbers

```

The Shrinkage menu allows to apply some “shrinkage” to each element rendered. The element is “shrunk” by the chosen amount with respect to its centroid. This produces an effect like if the mesh had “exploded” and allows to distinguish the various elements forming the mesh. This may be useful to see the “internals” of the mesh or to distinguish between superposed elements (e.g. a flat shell structure attached to a continuum fluid element). A toggle button *Shrink outlines* allows to shrink the element’s outline like the element face, or not. Another toggle button *Shrink isolines* does the same for isolines. Finally, a toggle button *Shrink hidden faces* shrinks the faces of hidden objects (in case they are shown as translucent).

- *No shrinkage*. This is the default. No shrinkage is applied. The command may be useful to restore the default appearance after some shrinkage has been previously set.
- *Shrink by groups*. The chosen shrinkage amount (1.0 by default) is applied to each group of elements separately (i.e. with respect to the centroid of the group) rather than to each element separately. This might be useful to obtain an “exploded” view of a mesh subdivided into its (macroscopic) components, rather than in all its single elements.
- *Shrink 99%*. The shrinkage factor is set to 99%. This value is too close to 1.0 to produce a real “exploded” view of the mesh. However, it may be useful in some cases to obtain a clearer view of the mesh, since the element faces are drawn just slightly displaced towards

the element centroid. For example, in case of two superposed faces, say a continuum element face and a flat element face such as a CLXX, of different colors, the color in which the resulting face points are rendered is sort of random because the two face points have the same coordinates. Setting a small shrinkage amount can avoid this problem, showing the face in the color that the user would expect (i.e. the CLXX color, if the body is looked at from the “outside” of the continuum element volume). (Note incidentally that this particular type of representation may require to turn on the rendering of internal faces since the CLXX face and the continuum face may have opposite normals and may therefore be considered as internal faces.)

- *Shrink 80%*. The shrinkage factor is set to 80%.
- *Shrink 60%*. The shrinkage factor is set to 60%.
- *Shrink 40%*. The shrinkage factor is set to 40%.
- *Shrink outlines*. Shrink the element outlines in addition to the element faces. This becomes the default, as soon as some shrinkage is applied (so that the menu entry changes to *Dont shrink outlines*. In order *not* to shrink the element outlines (only the element faces), select *Dont shrink outlines*.
- *Shrink isolines*. Shrink the isolines (iso-value lines) in addition to the element faces. The default is *not* to shrink the isolines, even when some shrinking is selected.
- *Shrink hidden faces*. Shrink the “hidden” element faces (i.e. the faces of hidden objects being rendered as translucent, see the **Objects** menu) in addition to the visible element faces.
- *Shrink pinballs*. Shrink the pinballs used to detect contact in addition to the element faces.
- *Shrink node numbers*. Shrink the node numbers in addition to the element faces.

### 16.7.9 The Pinballs menu

#### Syntax:

```

Geometry --> Pinballs --> Show parents
                        Show nodal ASNs
                        Show parent ASNs
                        Show contacting descendents
                        Show contact points
                        Show contact normals
                        Show contact joints
                        Show descendent ASNs
                        Show pinballs as solid

```

The Pinballs menu allows to visualize pinballs used in EPX to model contact between bodies. This menu appears only when pinballs are present in the current calculation. Pinballs are visually represented by semi-transparent spheres, and contacts can be highlighted in the form of red lines joining the centres of each pair of contacting pinballs. In addition, various types of normals can be visualized.

- *Show parents.* Visualize the parent (zero-level) pinballs (whether or not they are in contact with one another). A parent pinball is usually embedded in each element which may undergo contact by the PINB directive. This command can be useful to visually check where the contact-detecting pinballs have been placed in the model.
- *Show nodal ASNs.* Visualize the Assembled Surface Normals (ASNs) at nodes. These are only available if the ASN model is enabled.
- *Show parent ASNs.* Visualize the Assembled Surface Normals (ASNs) at parent pinballs. These are only available if the ASN model is enabled.
- *Show contacting descendents.* Visualize the descendent pinballs which are found to be in contact. This is useful to visually check at which locations of the model the pinball method is actually detecting contact.
- *Show contact points.* Highlight the centres of the (descendent) pinballs which are found to be in contact. These correspond approximately to the points which are found to be in contact, on the current contacting surfaces.
- *Show contact normals.* At each contacting descendent pinball centre, show the contact normal, i.e. the direction along which interpenetration of the pinballs is being detected. This is also the direction along which the contact force acts (in the absence of friction).
- *Show contact joints.* Join the centres of each couple of contacting descendent pinballs by a red line, symbolizing the contact condition.
- *Show descendent ASNs.* Visualize the Assembled Surface Normals (ASNs) at contacting descendent pinballs. These are only available if the ASN model is enabled.
- *Show pinballs as solid.* Visualize the pinballs as solid spheres. By default, pinballs are rendered as translucent spheres.

#### 16.7.10 The Initial geometry menu

##### Syntax:

```

Geometry --> Initial geometry --> No initial geometry
                                     As current geometry
                                     Colored glass
                                     Wireframe
                                     Outline

```

The Initial geometry menu allows to visualize the initial (undeformed) configuration of the computational model in a variety of ways, in addition to the current (deformed) configuration. This menu appears only if the user has specified the DEFO optional keyword in the TRAC directive. An additional AMPD `ampd` keyword allows to choose an optional magnification factor `ampd` of the displacements (by default, `ampd` is 1.0).

- *No initial geometry.* Do not visualize the initial geometry (only the current one). This is the default. However, this command may be used to disable the visualization of the initial geometry, previously set by one of the following commands.
- *As current geometry.* Visualize the initial geometry with exactly the same appearance as the currently chosen appearance for the current geometry.

- *Colored glass.* Visualize the initial geometry as colored glass. Each initial-configuration element is rendered as partially translucent, with the same color as the current-configuration element.
- *Wireframe.* Visualize the initial geometry as wireframe. Only the element outlines are drawn (not the faces) for the initial-configuration elements.
- *Outline.* Visualize the initial geometry as an outline (free edges) only.

### 16.7.11 The Flying debris menu

#### Syntax:

```
Geometry --> Flying debris --> No trajectories
                                   Trajectories
                                   Colored trajectories
```

The Flying debris menu allows to visualize the trajectories of flying debris particles. The flying debris particles themselves are drawn as material points, with the corresponding options and settings (see the **Points** menu described above). The Flying debris menu appears only if there are flying debris (see **DEBR**) in the model and if the user has chosen to save the trajectories (which can lead to huge output).

- *No trajectories.* Do not visualize the flying debris trajectories. This is the default. However, this command may be used to disable the visualization of the trajectories, previously set by one of the following commands.
- *Trajectories.* Visualize the flying debris trajectories as thick lines of uniform color.
- *Colored trajectories.* Visualize the flying debris trajectories as thick lines of variable color. The color is “proportional” to the modulus of the particle velocity at each point of a trajectory.

### 16.7.12 The FLSR domains menu

#### Syntax:

```
Geometry --> FLSR domains --> Show all domains
                                   Show spheres
                                   Show cones
                                   Show prisms
                                   Show hexahedra
                                   Show normals
                                   Show couplings
                                   Show blocked fluxes
```

The FLSR domains menu allows to visualize the Fluid-Structure Interaction (FSI) domains used by the FLSR model (see **FLSR**) in order to detect the interacting entities. In addition, one may visualize the normal directions used for the interaction relationships, the couplings and the blocked numerical fluxes.

- *Show all domains.* Show all the FLSR domains (spheres, cones etc.). Sometimes, this may produce a somewhat cluttered picture. Therefore, commands are available to select the domains type by type.
- *Show spheres.* Show the spherical domains at the (structure) nodes.
- *Show cones.* Show the truncated cone domains along the (structure) edges. Each cone connects the spheres at the two nodes of the edge.
- *Show prisms.* Show the prism domains along the (structure) triangular faces (3D only).
- *Show hexahedra.* Show the hexahedra domains along the (structure) quadrilateral faces (3D only).
- *Show normals.* Show the normal(s) associated with each domain.
- *Show couplings.* Show the coupling by means of thick red lines connecting each couple of interacting entities.
- *Show blocked fluxes.* Show the blocked numerical by means of thick green lines connecting the centroids of couples of neighboring fluid elements between which the numerical fluxes are blocked.

### 16.7.13 The FLSW domains menu

#### Syntax:

```

Geometry --> FLSW domains --> Show all domains
                                Show spheres
                                Show cones
                                Show prisms
                                Show hexahedra
                                Show normals
                                Show couplings
                                Show blocked fluxes

```

The FLSW domains menu allows to visualize the Fluid-Structure Interaction (FSI) domains used by the FLSW model (see **FLSW**) in order to detect the interacting entities. In addition, one may visualize the normal directions used for the interaction relationships, the couplings and the blocked numerical fluxes. This menu is similar to the FLSR domains menu but it applies to Cell-Centred Finite Volume (VFCC) discretizations rather than to Finite-Element (FE) discretizations of the fluid sub-domain.

- *Show all domains.* Show all the FLSW domains (spheres, cones etc.). Sometimes, this may produce a somewhat cluttered picture. Therefore, commands are available to select the domains type by type.
- *Show spheres.* Show the spherical domains at the (structure) nodes.
- *Show cones.* Show the truncated cone domains along the (structure) edges. Each cone connects the spheres at the two nodes of the edge.
- *Show prisms.* Show the prism domains along the (structure) triangular faces (3D only).

- *Show hexahedra.* Show the hexahedra domains along the (structure) quadrilateral faces (3D only).
- *Show normals.* Show the normal(s) associated with each domain.
- *Show couplings.* Show the coupling by means of thick red lines connecting each couple of interacting entities.
- *Show blocked fluxes.* Show the blocked numerical by means of thick green lines connecting the centroids of couples of neighboring fluid volumes between which the numerical fluxes are blocked.

#### 16.7.14 The Gpinballs menu

##### Syntax:

```

Geometry --> Gpinballs --> Show all domains
                             Show spheres
                             Show cones
                             Show prisms
                             Show hexahedra
                             Show contacting domains
                             Show domain ASNs
                             Show penetrations
                             Show penetration rates
                             Show contact points
                             Show contact normals
                             Show contact joints
                             ASN length           --> ...
                             Contact normal length --> ...

```

*Caution: The GPIN model is still under development, so some of these features may have limited functionality.*

The Gpinballs menu allows to visualize the generalized pinballs (GPINs) used in EPX to model contact between bodies. This menu appears only when generalized pinballs (see GPIN) are present in the current calculation. Gpinballs are visually represented by semi-transparent spheres, cones, prisms and hexahedra. The penetrations and penetration rates are optionally visualized. Contacts can be highlighted in the form of red lines joining the contact points of each pair of contacting gpinballs. In addition, the contact normals and the contact points can be visualized.

- *Show all domains.* Show all the GPIN domains (spheres, cones etc.). Sometimes, this may produce a somewhat cluttered picture. Therefore, commands are available to select the domains type by type.
- *Show spheres.* Show the spherical domains at the nodes.
- *Show cones.* Show the truncated cone domains along the edges. Each cone connects the spheres at the two nodes of the edge.
- *Show prisms.* Show the prism domains along the triangular faces (3D only).
- *Show hexahedra.* Show the hexahedra domains along the quadrilateral faces (3D only).



- *Show contacting domains.* Show the contacting domains.
- *Show domain ASNs.* Visualize the Assembled Surface Normals (ASNs) at the gpinballs. Note that the GPINs model is not hierarchic (in contrast with the classical pinball model PINB), so that in a sense all GPINs are parent GPINs.
- *Show penetrations.* Visualize the penetrations between GPINs.
- *Show penetration rates.* Visualize the penetration rates between GPINs.
- *Show contact points.* Highlight the centres of the gpinballs which are found to be in contact. These correspond approximately to the points which are found to be in contact, on the current contacting surfaces.
- *Show contact normals.* At each contacting gpinball centre, show the contact normal, i.e. the direction along which interpenetration of the gpinballs is being detected. This is also the direction along which the contact force acts (in the absence of friction).
- *Show contact joints.* Join the centres of each couple of contacting gpinballs by a red line, symbolizing the contact condition.
- **ASN length.** Open a sub-menu which allows to adjust the length of the ASN arrows (see below).
- **Contact normal length.** Open a sub-menu which allows to adjust the length of the contact normal arrows (see below).

### The ASN length menu

#### Syntax:

```
Geometry --> Gpinballs --> ASN length --> Double
                                           Half
                                           Default
```

This sub-menu allows to adjust the length of the ASN arrows. The *Double* and *Half* commands can be issued repeatedly in order to achieve (almost) any desired length of the arrows.

- *Double.* Double the length of the ASN arrows which, by default, amounts to 1/10 of the model diameter.
- **Half.** Halve the length of the ASN arrows.
- **Default.** Restore the default length of the ASN arrows.

### The Contact normal length menu

#### Syntax:

```
Geometry --> Gpinballs --> Contact normal length --> Double
                                                         Half
                                                         Default
```

This sub-menu allows to adjust the length of the contact normal arrows. The *Double* and *Half* commands can be issued repeatedly in order to achieve (almost) any desired length of the arrows.

- *Double*. Double the length of the contact normal arrows which, by default, amounts to 1/10 of the model diameter.
- **Half**. Halve the length of the contact normal arrows.
- **Default**. Restore the default length of the contact normal arrows.

### 16.7.15 The Links (coupled) menu

#### Syntax:

```
Geometry --> Links (coupled) --> Show all links
                                   Length      --> ...
                                   Show link joints
                                   Show        --> <list of links by type>
                                   Hide        --> <list of links by type>
```

The Links (coupled) menu allows to visualize the coupled links present in the current model. This menu is only available if there are coupled links (see LINK COUP) and if the user has activated the OPTI LNKS VISU option in the input data file. A link is represented by a set of arrows, attached to each of the nodes that are connected by the link itself. The length of the arrows is arbitrary (by default, 1/10 of the model diameter) and can be adjusted by the **Length** sub-menu. The direction of the arrow is the direction along which the link constraint is acting.

- *Show all links*. Show all the coupled links. In some cases the figure can be cluttered. It is possible to select only certain types of links by the following sub-menus.
- **Length**. Open a sub-menu which allows to adjust the length of the link arrows (see below).
- *Show link joints*. In addition to the arrows, draw thick red lines joining each couple of linked nodes.
- **Show**. Open a sub-menu which allows to select which types of links should be visualized. The menu contains a list of all types of links currently present in the model (the number of such links is shown next to the type).
- **Hide**. Open a sub-menu which allows to select which types of links should be hidden. The menu contains a list of all types of links currently present in the model (the number of such links is shown next to the type).

### The Length menu

#### Syntax:

```
Geometry --> Links (coupled) --> Length --> Double
                                           Half
                                           Default
```

This sub-menu allows to adjust the length of the links arrows. The *Double* and *Half* commands can be issued repeatedly in order to achieve (almost) any desired length of the arrows.

- *Double*. Double the length of the links arrows which, by default, amounts to 1/10 of the model diameter.
- **Half**. Halve the length of the links arrows.
- **Default**. Restore the default length of the links arrows.

## 16.8 The Vectors menu

### Object:

The Vectors menu allows to visualize a vectorial field on the body geometry in the form of vectors (arrows). The vector field can be expressed at nodes (most commonly) or at the element centroids (e.g. in the case of Cell-Centred Finite Volumes).

### Syntax:

```

Vectors --> No vectors
           Scaled vectors
           Colored vectors
           Scaled colored vectors
           Field             --> <list of available fields>
           Scale             --> ...
           Length            --> ...
           Color scheme      --> ...
           Options           --> ...

```

- *No vectors*. This is the default. It may be used explicitly to switch off vectors representation when no longer required.
- *Scaled vectors*. This activates vectors rendering with a uniform (green) color. The length of the arrow is proportional to the local intensity (norm) of the vector field.
- *Colored vectors*. This activates vectors rendering with a uniform length. The color is proportional to the local intensity (norm) of the vector field.
- *Scaled colored vectors*. This activates vectors rendering with both the length and the color proportional to the local intensity (norm) of the vector field.
- **Field**. This opens a sub-menu which allows choosing the (vector) field to be visualized. By default, the velocity field is rendered. A list of the vector fields available in the present calculation is automatically built up. The user can select any field by clicking on the corresponding entry in the sub-menu.
- **Scale**. This opens a sub-menu which allows choosing the scale for the vector field rendering.
- **Length**. This opens a sub-menu which allows to adjust the length of the arrows representing the vector field (without changing the scale steps).
- **Color scheme**. This opens a sub-menu which allows to select the colors scheme for the vector field rendering.
- **Options**. This opens a sub-menu which allows to select some generic options for the vector field rendering.

The various sub-menus are described next.

### 16.8.1 The Scale menu

#### Syntax:

```
Vectors --> Scale --> Auto 6
                        Auto 14
                        User
```

This sub-menu allows to choose the scale for the vector field.

- *Auto 6.* A 6-color (linear) scale is chosen which automatically encompasses the range of vector intensities (lengths) of the field to be rendered.
- *Auto 14.* This is the default. A 14-color (linear) scale is chosen which automatically encompasses the range of vector intensities (lengths) of the field to be rendered.
- *User.* The user-specified scale is selected. This scale must have been specified by the user in the **SCEN** directive before giving the **TRAC REND** command (else this entry does not appear in the menu). In this case, this scale is used by default so it should not be necessary to use this command explicitly. However, if the user wants to try interactively an automatic scale (with either 6 or 14 colors), the present command may be useful to return to the user-defined scale.

Note that from January 2016 the 14-color scale has become the default for colored vectors rendering, and not the 6-color scale. Furthermore, the vectors scale is automatically popped up as soon as some vectors representation is activated, and popped down when no vectors are shown.

### 16.8.2 The Length menu

#### Syntax:

```
Vectors --> Length --> Double
                        Half
                        Default
```

This sub-menu allows to modify the length of the vector arrows in the current drawing, *without* modifying the vectors scale. The commands may be used repeatedly in order to obtain (almost) any desired length of the arrows.

- *Double.* This command doubles the length of the arrows without modifying the vectors scale in use nor the vectors colors (if any).
- *Half.* This command halves the length of the arrows without modifying the vectors scale in use nor the vectors colors (if any).
- *Default.* This command restores the default length of the arrows without modifying the vectors scale in use nor the vectors colors (if any).

### 16.8.3 The Color scheme menu

#### Syntax:

```
Vectors --> Color scheme --> Colors
                                Grayscale
                                Inverted colors
                                Inverted grayscale
```

This sub-menu allows to choose the color scheme of the vector arrows, if any (i.e., unless the vectors are rendered in the *Scaled vectors* mode).

- *Colors*. This is the default. Vectors are drawn using a scale of 6 or 14 colors depending on the chosen scale option.
- *Grayscale*. This replaces the colors by a scale of grey tones.
- *Inverted colors*. The color scale is used but the red color corresponds to the lowest value rather than to the highest value.
- *Inverted grayscale*. The grayscale scale is used but the darkest gray color corresponds to the lowest value rather than to the highest value.

### 16.8.4 The Options menu

#### Syntax:

```
Vectors --> Options --> Show internal vectors
```

This sub-menu allows to choose some options related to the visualization of vector fields.

- *Show internal vectors*. This visualizes the vector arrows also at “internal” nodes (or element centroids) rather than only at nodes on the surface of the body, in 3D models. Beware that, by default, in 3D models only the vectors at nodes located on the surface of the body are drawn.

## 16.9 The Isovalues menu

### Object:

The Isovalues menu allows to visualize a vectorial field on the body geometry in the form of vectors (arrows). The vector field can be expressed at nodes (most commonly) or at the element centroids (e.g. in the case of Cell-Centred Finite Volumes).

The Isovalues menu allows to represent a scalar field on the body geometry, in the form of iso values. The scalar field can be expressed either at the element centroids or at the nodes. Various types of representation are possible.

### Syntax:

```
Isovalues --> No iso
                Iso lines
                Iso fill
                Iso fill lines
                Iso fill elements --> <list of available representations>
                Iso smooth
                Iso smooth lines
                Iso smooth elements --> <list of available representations>
                Iso surfaces
                Iso surfaces lines
                Shiny iso surfaces
                Field --> <list of available fields>
                Component --> <list of available components>
                Scale --> ...
                Color scheme --> ...
```

- *No iso.* This is the default. It may be used explicitly to switch off isovalues representation when no longer required.
- *Iso lines.* This activates iso lines (think of “level curves” on a normal 2D geographical map) representation of the scalar field.
- *Iso fill.* This activates iso values representation by filling the spaces between the ideal iso lines (see previous options) by a uniform color.
- *Iso fill lines.* This is the combination of the two previous iso value representation modes (color filling plus lines).
- **Iso fill elements.** This corresponds to filling each element by a uniform color corresponding to the local value of the scalar field. The result is an arlequin-like representation of the field. This opens a sub-menu. The local value of the scalar field can be either the average over all Gauss points of the element, or the maximum, or the minimum, see the sub-menu that lists the various possible types of representations.
- *Iso smooth.* This activates iso values representation by first interpolating the field value at the nodes (if necessary) and then filling the element faces by a graded color filling using the values at the nodes.

- *Iso smooth lines.* This is a combination of the smooth rendering mode described above and of iso lines.
- *Iso smooth elements.* This is similar to the arlequin mode described above (each element is filled by a uniform color), but the color is interpolated continuously on the colors scale (like in the smooth color filling mode) rather than choosing the nearest fixed scale color. This opens a sub-menu. The local value of the scalar field can be either the average over all Gauss points of the element, or the maximum, or the minimum, see the sub-menu that lists the various possible types of representations.
- *Iso surfaces.* This activates iso surfaces (think of “level surfaces”) representation of the scalar field. It makes sense only in 3D models, Rendering of the model outer faces is automatically disabled so that one can see the iso surfaces in the interior of the body
- *Iso surfaces lines.* This is a combination of the iso surfaces rendering mode described above and of iso lines.
- *Shiny iso surfaces.* This renders the iso surfaces with a “shiny” appearance. For this to actually work, the light must be switched on, see the **Lights/Mats** menu below. This switch is a toggle, When activated, it changes to *Dull iso surfaces*, which allows to return to the default visualization of iso surfaces.
- **Field.** This opens a sub-menu which allows choosing the (scalar) field to be visualized. By default, the stress field (component 1) is rendered. A list of the scalar fields available in the present calculation is automatically built up. The user can select any field by clicking on the corresponding entry in the sub-menu. Note that the list does not include only the classical fields at element Gauss Points such as stress, total strain or hardening parameters, but also the nodal fields (velocity, force, etc.) which are typically considered as vector fields (see the **Vectors** menu in a previous Section. For the latter to be representable as iso-values, one must of course select just one component (or the norm of the vector field), see next sub-menu.
- **Component.** This opens a sub-menu which allows choosing the component of the field to be visualized. By default, the first component of the currently selected field is rendered. A list of the scalar field components available in the present calculation for the currently selected field is automatically built up, and the norm of all components is also added. The user can select any component (or the norm) by clicking on the corresponding entry in the sub-menu.
- **Scale.** This opens a sub-menu which allows choosing the scale for the isovalue field rendering.
- **Color scheme.** This opens a sub-menu which allows to select the colors scheme for the isovalue field rendering.

The various sub-menus are described next.

### 16.9.1 The Fill elements representations menu

**Syntax:**



```

Isovalues --> Iso fill elements --> Average over the GPs
                                     Maximum over the GPs
                                     Minimum over the GPs
                                     ABS MAX over the GPs

```

This sub-menu allows to choose the type of representation for the “Iso fill elements” or the “Iso smooth elements” isovalues field.

- *Average over the GPs.* The code builds up the average of the iso-field over all GPs of each element and then uses that (scalar) value to decide the (uniform) color by which each element is filled.
- *Maximum over the GPs.* The code takes the maximum of the iso-field over all GPs of each element and then uses that (scalar) value to decide the (uniform) color by which each element is filled.
- *Minimum over the GPs.* The code takes the minimum of the iso-field over all GPs of each element and then uses that (scalar) value to decide the (uniform) color by which each element is filled.
- *ABS MAX over the GPs.* The code takes the maximum in absolute value of the iso-field over all GPs of each element and then uses that (scalar) value to decide the (uniform) color by which each element is filled.

### 16.9.2 The Scale menu

#### Syntax:

```

Isovalues --> Scale --> Auto 6
                       Auto 14
                       User
                       Auto 1

```

This sub-menu allows to choose the scale for the isovalues field.

- *Auto 6.* A 6-color (linear) scale is chosen which automatically encompasses the range of intensities of the field and component to be rendered.
- *Auto 14.* This is the default. A 14-color (linear) scale is chosen whichg automatically encompasses the range of intensities of the field and component to be rendered.
- *User.* The user-specified scale is selected. This scale must have been specified by the user in the SCEN directive before giving the TRAC REND command. In this case, this scale is used by default so it is not necessary to use this command explicitly. However, if the user wants to try interactively an automatic scale (with either 6 or 14 colors), the present command may be useful to return to the user-defined scale.
- *Auto 1.* A single-value scale is chosen. This allows to divide the model into two zones: one where the field values are below and one where they are above the chosen value. By default the automatically chosen value is the mean value between the minimum and the maximum values of the chosen field/component. The user can precisely set this value by choosing a user scale with just one value (instead of 6 or 14 values).

Note that from January 2016 the 14-color scale has become the default for colored iso rendering, and not the 6-color scale. Furthermore, the iso scale is automatically popped up as soon as some iso representation is activated, and popped down when no iso are shown.

### 16.9.3 The Color scheme menu

#### Syntax:

```
Isovalues --> Color scheme --> Colors
                                   Grayscale
                                   Inverted colors
                                   Inverted grayscale
```

This sub-menu allows to choose the color scheme of the iso values representation.

- *Colors*. This is the default. A scale of 6 or 14 colors is used depending on the chosen scale option.
- *Grayscale*. This replaces the colors by a scale of grey tones.
- *Inverted colors*. The color scale is used but the red color corresponds to the lowest value rather than to the highest value.
- *Inverted grayscale*. The grayscale scale is used but the darkest gray color corresponds to the lowest value rather than to the highest value.

## 16.10 The Text menu

### Object:

The Text menu may be used to visualize some textual information, such as node numbers, object names etc., in addition or in alternative to the problem title, time value and time step, which appear by default in the top left corner of the graphical window.

### Syntax:

```
Text --> Show node numbers
         Show element numbers
         Show object names
         Show vectors scale
         Show iso scale
         Hide info
         Show camera values
         Show debug info
         Show pinball contacts
```

- *Show node numbers.* This activates the visualization of node numbers. Note that the text is represented in 3D such as any other geometrical element, therefore it may be obscured by other geometrical entities, such as element faces. To read the numbers, it may be useful to turn the body, or to switch off the face filling as described above in the **Geometry** menu.
- *Show element numbers.* This activates the visualization of element numbers. If faces are filled, then the element number is repeated at the center of each one of its faces (to maximize the probability that it becomes visible), else it is represented just once, at the geometric center of the element.
- *Show object names.* This activates the visualization of object names. Geometrical objects are inherited from pre-processors or mesh generators (such as Cast3m). They can also be generated in EPX as named element groups.
- *Show vectors scale.* This visualizes a legend for the vectors representation. This text is “fixed” in the sense that it does not rotate with the body, and should be always visible “on top” of the geometrical object.
- *Show iso scale.* This visualizes a legend for the iso-values representation (fixed text).
- *Hide info.* This hides the visualization of the problem title, time value and time step, which appear by default in the top left corner of the graphical window.
- *Show camera values.* This activates the visualization of the current value of the camera parameters (fixed text).
- *Show debug info.* This activates the visualization of debug-related information.
- *Show pinball contacts.* This activates the visualization of pinball contacts numbering. The text is only visualized if **Geometry** → **Pinballs** → *Show contact points* or **Geometry** → **Pinballs** → *Show contact joints* is selected.

## 16.11 The Colors menu

### Object:

The Colors menu allows to change the colors that are assigned by default to those components of the scene which are rendered in a fixed color. This is accomplished in two steps:

1. First, a color is selected from the **Colors → 1) Select color** menu. The available choices are listed below.
2. Then, the component to which the color must be applied is selected from the **Colors → 2) Apply it to** menu. The available choices are listed below.

### Syntax:

```
Colors --> 1) Select color --> <list of available colors>
           2) Apply it to --> <list of colorized scene components>
           Default for screen
           Default for paper
```

- **Select color.** This activates a sub-menu which lists the available colors. To select a color, click on the corresponding entry.
- **Apply it to.** This activates a sub-menu which lists the colorized scene components, i.e. the scene items which are drawn in a pre-determined color. To select a component, click on the corresponding entry. The previously selected color is immediately applied to the selected component.
- *Default for screen.* This switch may be used to select a color scheme suitable for an on-screen representation (black background, white lines etc.). Note, however, that this color scheme is probably not suited for obtaining hard copies of the scene to be inserted in documents.
- *Default for paper.* This is the default (even for on-screen visualization). The switch may be used to restore the default values of colors, which are suitable for an on-paper representation (white background, black lines etc.), after the color scheme has been previously altered.

Note that from January 2016 the paper-suited color scheme has become the default scheme even for on-screen rendering (while the screen scheme was the default previously). This is because when hard-copies of the graphical window are produced, the paper color scheme is probably more suitable for inclusion of figures into paper (or even on-line) documentation.

The various sub-menus are described next.

### 16.11.1 The 1) Select color menu

```
Colors --> 1) Select color --> Red
                                   Green
                                   Blue
                                   Cyan
                                   Magenta
                                   Yellow
```

Black  
White  
Grey05  
Grey10  
Grey15  
Grey20  
Grey25  
Grey30  
Grey35  
Grey40  
Grey45  
Grey50  
Grey55  
Grey60  
Grey65  
Grey70  
Grey75  
Grey80  
Grey85  
Grey90  
Grey95

The color names from **Red** to **White** are self-explaining. As concerns the greys, they range from nearly black (**Grey05**) to nearly white (**Grey95**).

#### 16.11.2 The 2) Apply it to menu

Colors --> 2) Apply it to --> Background  
Center  
Bounding box  
Internal faces  
Element outlines  
Sharp corners  
Free edges  
Perp contours  
Vectors  
Iso surface edges  
Iso surface outlines  
Points  
Node numbers  
Element numbers  
Object names  
Text  
Initial wireframe  
Initial outline  
Debris trajectories  
Iso default color

- **Background.** Apply the previously chosen color to the scene background.
- **Center.** Apply the previously chosen color to the model center. (See the **Geometry** menu.)

- **Bounding box.** Apply the previously chosen color to the model bounding box. (See the **Geometry** menu.)
- **Internal faces.** Apply the previously chosen color to the internal faces. (See the **Geometry** menu.)
- **Element outlines.** Apply the previously chosen color to the element outlines. (See the **Geometry** menu.)
- **Sharp corners.** Apply the previously chosen color to the sharp corners. (See the **Geometry** menu.)
- **Free edges.** Apply the previously chosen color to the free edges. (See the **Geometry** menu.)
- **Perp contours.** Apply the previously chosen color to the perpendicular contours. (See the **Geometry** menu.)
- **Vectors.** Apply the previously chosen color to vectors drawn as scaled vectors (not as colored or as scaled colored vectors). (See the **Vectors** menu.)
- **Iso surface edges.** Apply the previously chosen color to the edges of iso surfaces (3D only). (See the **Isovalues** menu.)
- **Iso surface outlines.** Apply the previously chosen color to the outlines of iso surfaces (3D only). (See the **Isovalues** menu.)
- **Points.** Apply the previously chosen color to the points (material points etc.). (See the **Geometry** menu.)
- **Node numbers.** Apply the previously chosen color to the node numbers. (See the **Text** menu.)
- **Element numbers.** Apply the previously chosen color to the element numbers. (See the **Text** menu.)
- **Object names.** Apply the previously chosen color to the object names. (See the **Text** menu.)
- **Text.** Apply the previously chosen color to the displayed text. (See the **Text** menu.)
- **Initial wireframe.** Apply the previously chosen color to the wireframe of the initial geometry, if any. (See the **Geometry** menu.)
- **Initial outline.** Apply the previously chosen color to the outline of the initial geometry, if any. (See the **Geometry** menu.)
- **Debris trajectories.** Apply the previously chosen color to the debris trajectories rendered as a uniform color, if any. (See the **Geometry** menu.)
- **Iso default color.** Apply the previously chosen color as filling color in iso-maps for any parts of the model which are excluded from the iso-values representation. (See keyword ISO ... SUPP in the SCEN directive on Page O.80.)

## 16.12 The Lights/Mats menu

### Object:

The Lights/Mats menu allows to switch on or off the light and to assign an appearance (“material”) to the whole mesh or to selected portions of it (objects). Lighting affects rendering of the scene and adds a touch of realism. The choice of materials takes place in two steps:

1. First, a material is selected from the **Lights/Mats** → **1) Select material** menu. The available choices are listed below.
2. Then, the object to which the material must be applied is selected from the **Lights/Mats** → **2) Apply it to** menu. The available choices are listed below.

### Syntax:

```
Lights/Mats --> Switch light on
                  Light X           --> ...
                  Light Y           --> ...
                  Light Z           --> ...
                  Light ambient     --> ...
                  Light diffuse     --> ...
                  Light specular    --> ...
                  Light shininess   --> ...
                  Light model ambient --> ...
                  Reset lights
                  1) Select material --> <list of available materials>
                  2) Apply it to    --> ...
                  Reset materials
```

- *Switch light on.* This switches the light on, i.e. it activates the lighting model.
- **Light X.** This opens a sub-menu which allows to define the position of the directional lighting source along the X axis. This light source always points towards the centroid of the model.
- **Light Y.** This opens a sub-menu which allows to define the position of the directional lighting source along the Y axis. This light source always points towards the centroid of the model.
- **Light Z.** This opens a sub-menu which allows to define the position of the directional lighting source along the Z axis. This light source always points towards the centroid of the model.
- **Light ambient.** This opens a sub-menu which allows to choose the intensity of the ambient light.
- **Light diffuse.** This opens a sub-menu which allows to choose the intensity of the diffuse light.
- **Light specular.** This opens a sub-menu which allows to choose the intensity of the specularity of surfaces subjected to lighting.

- **Light shininess.** This opens a sub-menu which allows to choose the intensity of the shininess of surfaces subjected to lighting.
- **Light model ambient.** This opens a sub-menu which allows to choose the intensity of the model ambient light.
- *Reset lights.* This resets the lights to their default values.
- **1) Select material.** This opens a sub-menu which allows to choose a material among a list of available materials.
- **Apply it to.** This opens a sub-menu which allows to select to which parts of the model to associate the previously chosen material as far as the lighting model is concerned.
- *Reset materials.* This resets the materials to their default values.

The various sub-menus are described next.

### 16.12.1 The Light X menu

Lights/Mats --> Light X --> Left  
Centre  
Right

- *Left.* Set the directional lighting source to come from the “left” (i.e. from the negative part of the X axis).
- *Centre.* Set the directional lighting source to come from the “centre” (i.e. from the central part of the X axis).
- *Right.* Set the directional lighting source to come from the “right” (i.e. from the positive part of the X axis).

### 16.12.2 The Light Y menu

Lights/Mats --> Light Y --> Bottom  
Centre  
Top

- *Bottom.* Set the directional lighting source to come from the “bottom” (i.e. from the negative part of the Y axis).
- *Centre.* Set the directional lighting source to come from the “centre” (i.e. from the central part of the Y axis).
- *Right.* Set the directional lighting source to come from the “top” (i.e. from the positive part of the Y axis).

### 16.12.3 The Light Z menu

Lights/Mats --> Light Z --> Front  
Centre  
Back

- *Front.* Set the directional lighting source to come from the “front” (i.e. from the positive part of the Z axis).



- *Centre*. Set the directional lighting source to come from the “centre” (i.e. from the central part of the Z axis).
- *Back*. Set the directional lighting source to come from the “back” (i.e. from the negative part of the Z axis).

#### 16.12.4 The Light ambient menu

Lights/Mats --> Light ambient --> Low  
Medium  
High

- *Low*. Set the ambient light to a low level of intensity.
- *Medium*. Set the ambient light to a medium level of intensity.
- *High*. Set the ambient light to a high level of intensity.

#### 16.12.5 The Light diffuse menu

Lights/Mats --> Light diffuse --> Low  
Medium  
High

- *Low*. Set the diffuse light to a low level of intensity.
- *Medium*. Set the diffuse light to a medium level of intensity.
- *High*. Set the diffuse light to a high level of intensity.

#### 16.12.6 The Light specular menu

Lights/Mats --> Light specular --> Low  
Medium  
High

- *Low*. Set the light specularity to a low level of intensity.
- *Medium*. Set the light specularity to a medium level of intensity.
- *High*. Set the light specularity to a high level of intensity.

#### 16.12.7 The Light shininess menu

Lights/Mats --> Light shininess --> Low  
Medium  
High

- *Low*. Set the light shininess to a low level of intensity.
- *Medium*. Set the light shininess to a medium level of intensity.
- *High*. Set the light shininess to a high level of intensity.

### 16.12.8 The Light model ambient menu

Lights/Mats --> Light model ambient --> Low  
Medium  
High

- *Low*. Set the model ambient light to a low level of intensity.
- *Medium*. Set the model ambient light to a medium level of intensity.
- *High*. Set the model ambient light to a high level of intensity.

### 16.12.9 The 1) Select material menu

Lights/Mats --> 1) Select material --> Brass  
Bronze  
Polished bronze  
Chrome  
Copper  
Polished copper  
Gold  
Gold2  
Polished gold  
Pewter  
Silver  
Polished silver  
Emerald  
Jade  
Obsidian  
Pearl  
Ruby  
Turquoise  
Black plastic  
Cyan plastic  
Green plastic  
Red plastic  
White plastic  
Yellow plastic  
Black rubber  
Black rubber2  
Cyan rubber  
Green rubber  
Red rubber  
White rubber  
Yellow rubber

The material names are self-explaining. Of course these are not real materials but are only used to set the surface appearance for the rendering process.

### 16.12.10 The 2) Apply it to menu

Colors --> 2) Apply it to --> Whole mesh  
Selected objects

- **Whole mesh.** Apply the previously chosen material to the whole mesh (all elements).
- **Selected objects.** Apply the previously material to the currently selected objects only. (See the **Objects** → **Select objects** menu.) The mechanism of objects selection, followed by material selection and finally by material application allows to assign different materials interactively to the various components of the mesh.

## 16.13 The Win/Copy menu

### Object:

The Win/Copy menu allows to resize the window to precise predefined sizes and to produce a copy on file of the current scene. The copy is not a simple dump of the graphical window, but a separate rendering in memory, which is successively dumped on a file in the well-known .BMP format (or in a number of other available formats, see below). Such images are then easily inserted in publications or may be used to produce animated sequences. Several options are available, as described below.

Note also that, of course, it is also possible to resize the graphical window interactively at any moment by means of the usual Windows techniques. However, it is generally difficult in that way to achieve a precise size of the graphical window. The default size of the graphical window is  $500 \times 500$  pixels.

### Syntax:

```
Win/Copy --> Window 320*240
              Window 600*600
              Window 640*480
              Window 800*600
              Window 1024*768
              Window 1200*1200
              Window 1280*1024
              Bitmap format    --> <available bitmap formats>
              Copy as is
              Copy public. (1600*1200)
              Copy poster (1800*1800)
```

- *Window 320\*240*. This sets the graphical window size to  $320 \times 240$  pixels.
- *Window 600\*600*. This sets the graphical window size to  $600 \times 600$  pixels.
- *Window 640\*480*. This sets the graphical window size to  $640 \times 480$  pixels.
- *Window 800\*600*. This sets the graphical window size to  $800 \times 600$  pixels.
- *Window 1024\*768*. This sets the graphical window size to  $1024 \times 768$  pixels.
- *Window 1200\*1200*. This sets the graphical window size to  $1200 \times 1200$  pixels.
- *Window 1280\*1024*. This sets the graphical window size to  $1280 \times 1024$  pixels.
- **Bitmap format** This opens a sub-menu which allows to choose the format in which a copy of the rendered scene will be saved.
- *Copy as is* Produce a copy of the current scene in the current window size on a file on the current directory, in the currently chosen format (.BMP format by default). A small approximation of the image size may come by the fact that the pixel size is rounded to the nearest multiple of 4, to avoid problems in the successive utilization of the file (e.g. when including it in a document or using it to produce an animation). For example, an image of  $600 \times 600$  pixels has a size of about 1 MB (uncompressed) in the BMP format.

- *Copy public. (1600\*1200)* Produce a publication-ready copy of the current scene with an image size of  $1600 \times 1200$  pixels on a file on the current directory, in the currently chosen format (.BMP format by default). The graphical window is *not* resized. Each image has a size of about 5.6 MB (uncompressed) in the BMP format.
- *Copy poster (1800\*1800)* Produce a poster-ready copy of the current scene with an image size of  $1800 \times 1800$  pixels on a file on the current directory, in the currently chosen format (.BMP format by default). The graphical window is *not* resized. Each image has a size of about 9.5 MB (uncompressed) in the BMP format.

It is important to note that when planning to use the bitmap for producing an animation all the “frames” must be of the same size. To obtain bitmaps of sizes other than “publication” (1600\*1200) or “poster” (1800\*1800), first resize the window either by hand or, preferably, by one of the above Window `www*hhh` commands, and then use the *Copy as is* command.

The various sub-menus are described next.

### 16.13.1 The Bitmap format menu

```
Win/Copy --> Bitmap format --> BMP
                                   PPM (binary)
                                   PPM (Ascii)
                                   TGA
                                   EPS (color)
                                   EPS (b&w)
                                   POV-Ray
```

- *BMP*. This is the default. The format is known as device-independent bitmap format and is widely used on MS-Windows platforms. The image copy will be produced in the BMP format. This command might be useful to restore the default format after other format(s) were previously selected.
- *PPM (binary)*. The image copy will be produced in the PPM binary format. This format is known as the Portable Pixmap format (binary version), and is popular in the Unix world.
- *PPM (Ascii)*. The image copy will be produced in the PPM Ascii (pure text) format. This file is larger than the binary one, but probably much more portable.
- *TGA*. The image copy will be produced in the TGA (Targa) format.
- *EPS (color)*. The image copy will be produced in the Encapsulated Postscript (EPS) format in color. This is a purely ASCII file, and therefore very portable. It may be used e.g. to insert illustrations in  $\text{\LaTeX}$  documents.
- *EPS (b&w)*. The image copy will be produced in the Encapsulated Postscript (EPS) format in black and white (i.e., by using shades of grey). The grey levels are obtained from the color version by means of the so-called NTSC color conversion algorithm:

$$\lambda = 0.30R + 0.59G + 0.11B$$

where  $R$ ,  $G$ ,  $B$  are the red, green and blue components of the pixel color (each one ranging from 0 to 1) and  $\lambda$  is the resulting luminosity value (a grey level from 0 to 1).

- *POV-Ray*. This is not a bitmap format, but an ASCII file containing scene description commands. An input file for the POV-Ray ray-tracer will be generated (.pov).

## 17 GROUP SR—SAVING AND RESTART

The following directives allow to save the data during a computation and to the successive restart of the computation.

### 17.1 SAVING

#### 17.1.1 DEFINING SAVING IN THE INPUT CODE

##### Object:

To produce a saving file for successive restart, the directive `ECRI FICH SAUV` has to be inserted in group G of the EUROPLEXUS input data (as part of the `ECRI` directive). See page G.110 for details on the `SAUV` directive.

#### 17.1.2 SAVING VIA COMMAND FILE

The code checks the existence of a file called "command.epx" regularly. The intervals can be defined via "OPTI" "CMDF" "NPAS" nn "CPUT" nn. When the file exists, it is read and the containing command are performed. The following commands are possible:

##### Syntax:

```
< "SAVE">  
< "STOP">  
< "PVTK">  
  "END"
```

##### SAVE

The model is saved at the current time step.

##### STOP

The calculation is stopped after the current time step.

##### END

End of command file.

##### PVTK

A PVTK output is written for the current time step.

## 17.2 RESTART

### Object:

The REPR keyword, to be inserted in group A, enables to restart a computation which was previously saved. Not available in MPI.

### Syntax:

```
... title ...

"REPRISE"  nbanrep    "POSI" numer    < "PROT"  'maclef' >

... Instructions of the groups C,D,E,F,G that are modified ...
... (in particular, if a further saving is desired, repeat
    the ECRI FICH SAUV directive :)                               ...

< ECRI ... FICH  SAUV <ndsauv>  <PROT 'maclef'>  <LAST>  </CTIM/> >

... Instructions of the group H that are modified ...

"CALCUL"   . . .

"FIN"
```

The REPR instruction is described in detail below, Page SR.30.

### Warning:

During a restart the structure of the data is very different. In general, only the instructions of the groups C,D,E,F,G,H which are modified have to be repeated. However, note that the directive ECRI FICH SAUV must be repeated even if it is not changed. All other instructions are useless. Especially, the geometry must NEVER be repeated.

During a restart run, the directive FICH ALIC reads the file produced during the previous run and adds the new results like if a single calculation would have been performed.

### Comments:

The various possibilities are described in the following pages.

During the first run, the directive ECRI FICH SAUV FREQ n causes data to be saved once every n time steps, on the default saving file.

During a restart run, the directive `REPR 'myfile.sau' j` reads the previously created saving file and re-starts the calculation from the `j`-th saved data station.

If saving is requested during a restart run, be aware that a saving is not performed at the restart time itself.

During a restart run, all frequencies `nf` or `tf` in directives of the type:

```
- ECRI FICH SAUV iu FREQ nf
- ECRI FREQ nf
- ECRI TFRE tf
- ECRI FICH ALIC ju FREQ nf
- ECRI FICH ALIC ju TFRE tf
- ...
```

are considered starting from the initial time (or the initial step) of the first run, not of the restart run.

During a restart, if a directive similar to `CALC ... NMAX nsteps ...` is used, be aware that `nsteps` is the total number of time steps, not the number of steps during the restart run.

### Files:

The listing file is always produced anew during a restart, beginning at the restart time.

The saving file is specified by the instruction `ECRI FICH SAUV`.

The restart one is specified by the directive `REPRISE`.

For postprocessing files, see the description of `REPRISE` below.



### 17.3 DIRECTIVE "SAUVE" (OBSOLETE FORM)

This is the obsolete form of the **SAUVER** directive, which produces a saving file for subsequent restart of the calculation. It is only included here for compatibility with old input files. For new input files, please use the **ECRI ... FICH SAUV** directive, described on page G.110.

#### Object:

This keyword creates a saving file and, in conjunction with the keyword **REPR** (to be used in a subsequent run), allows splitting a computation in two or more parts.

The results are saved on a file (saving file) at times specified by the user. Each saving corresponds to a number or position on the file (1, 2, 3 etc.), from which a restart of the computation can be carried out (see directive **REPR** on page SR.30).

#### Syntax:

```
SAUV nbansav < FREQ > ifreq < DER > < PROT 'maclef' >
```

#### nbansav

Number of the saving file or name of the file in quotes. If completely omitted, the code will assume the default file name **<basename>.sau** where **<basename>** is the root of the input file name (i.e. without extension **.epx**). However, note that in this case the following keyword **FREQ** becomes mandatory to introduce the frequency.

#### ifreq

Frequency of the savings, in time steps. The results are saved each **ifreq** computation steps. Note that the code always saves the last step of the calculation (if the run is terminated normally), irrespective of the frequency chosen. Therefore, if one is only interested in getting the possibility to continue the calculation further on, a very large frequency may be chosen, larger than the total number of steps expected in the present run.

#### DER

This keyword indicates that the saving file should contain just one saving station, corresponding to the last saved time station in the present calculation. In other words, each new saving station replaces the former one, if any. This allows to obtain a saving file of the smallest possible size. However, restarting from an intermediate time is obviously not possible in this case: the only possibility to restart the calculation will be **REPR ... POSI 1** (see page SR.30).

#### PROT

Keyword entering a protection on the saving file.

#### 'maclef'

Key of up to 8 characters, enclosed in apostrophes. In order to restart the computation from that file, the instruction REPR must contain the keyword PROT with an identical key.

### Comments:

The keyword **FREQ** introduces a fixed frequency in time steps and has been maintained for backward compatibility (also, if **FREQ** is missing, and a number is read instead, the program assumes that it indicates a step frequency **ifreq**). This form of the directive (e.g. **SAUV 30 1000**) is still accepted for backwards compatibility only but is deprecated.

The keyword **PROT** is not compulsory. If it is not used, there is no protection (this is equivalent to a key of 8 blanks). If specified, the keyword **PROT** must be at the end of the **SAUV** directive (i.e., after **DER**, if any).

If a unit number is used for **nbansav**, the saving file and its number must have been defined before on the control cards.

A first saving station (position number 1) containing some header data is always produced at the initial time (step 0 of the calculation). Of course, it is normally meaningless to restart from this time station, unless the **DER** keyword has been specified (see above), because it would be the same as starting the calculation anew from the initial time. On the contrary, if **DER** has been specified, the only possibility for restart is to use the first time station which, in this case, will contain the data of the last saving performed (not the first one in general).

### Examples:

Assume a calculation performs 4994 time steps to arrive at its final time. The following saving directives are accepted:

- **SAUV 'myfile.sau' FREQ 1000** saves data for restart on the indicated file every 1000 steps. The following six saving stations are produced: 1 (step 0), 2 (step 1000), 3 (step 2000), 4 (step 3000), 5 (step 4000) and 6 (step 4994, i.e. the last step).
- **SAUV FREQ 1000**: same as above but the saving file has the default name **<basename>.sau**.
- **SAUV 'myfile.sau' FREQ 1000 DER** saves data for restart on the indicated file every 1000 steps, by always re-writing the previous saving. Only one saving station is thus present on the saving file (assuming that the run terminates normally): 1 (step 4994, i.e. the last step). If the run would fail, say, at step 2500, the saving file would also contain one station (step 2000).
- **SAUV FREQ 10000**: the saving file has the default name **<basename>.sau**. Only one saving station is produced (assuming that the run terminates normally): 1 (step 4994, i.e. the last step). If the run would fail, say, at step 2500, no (useful) saving time station would be available (there is still one saving station, but it is at step 0).

The following saving directives are still accepted for backwards compatibility but are strongly deprecated:

- **SAUV 30 1000** saves data for restart on unit 30. Under Windows, this produces a file named **fort.30** on the current directory. The saving frequency is 1000, so the following six saving stations are produced: 1 (step 0), 2 (step 1000), 3 (step 2000), 4 (step 3000), 5 (step 4000) and 6 (step 4994, i.e. the last step).

## 17.4 DIRECTIVE "REPRISE"

### Object:

This option enables a computation to be restarted from a time which was previously saved on a 'saving' file. This file now becomes the restart file and contains, besides the computed values at the saved time stations, a certain amount of data of the preceding computation which **must not** be defined again in the restart input file (see page SR.40).

### Syntax:

```
REPR  nbanrep  POSI  numer  < PROT  'maclef'  >
```

#### nbanrep

Number of the restart unit, or name of the restart file in quotes.

#### numer

Number of the saving station which determines the restart (1, 2, 3 etc.). The restart will occur from the **numer**-th saved time station. If the keyword **LAST** was used for writing the restart file, **numer** must be 1.

#### PROT

Compulsory keyword if there is a protection on the saving file.

#### 'maclef'

Key protecting the saving file.

### Comments:

If a unit number is used for **nbanrep**, the restart file and its unit number must have been defined first in the control cards. The restart file corresponds to the former saving file (i.e., it has the same file name).

During a restart, the user can save data for a further restart. In this case, the logical numbers of the saving and restart files must be different.

As far as data storage for postprocessing purposes (FICH ALIC, TPLOT, XPLOT, K2000 etc.) is concerned, two strategies may be followed during a restart. The first one consists in producing a single results file that after the restart corresponds to the file which would have been produced by a single run. To obtain this behaviour, during the restart use the same storage file which was defined in the previous run. The program will read the previous results file, position the pointer at the correct time (restart time) and then continue writing the data on the file.

The second possibility is that of splitting the results file in several pieces, one for the first run, another for the first restart, and so on. This may be useful e.g. in very large computations, to keep the file size acceptable. To obtain this behaviour, simply change the name of the results file each time you restart the calculations. The program will then produce a new data set containing also the necessary header information (e.g., geometry, etc.).

In case of the ParaView pvtk output the files are already split into several .vtu-files. They are combined later on in ParaView using the file .pvtk. For a restart the vtu files must not be present but the pvtk file. This file will be read by the routine and the new result files will be added.

Examples of both techniques are available in the example files.

## 17.5 DATA NECESSARY FOR RESTARTS

### Warning:

The data structure for restarts is different from that of a normal run.

All the EUROPLEXUS data relative to the preceeding computation are written on the restart file.

By explicitly re-defining a directive, the former one is cancelled and replaced. The user can also add some new directives.

### The following directives must be repeated:

- Title card;
- Dimension (or keyword **TERM**);
- Directives **CALCUL** and **FIN**.

### The following directive is NOT repeated:

- Geometry (with mesh).

### The following directives may be defined again:

- Dimensions;
- Masses;
- Thicknesses;
- Materials;
- Connections;
- Loads;
- Printouts;
- Options.

### Comments:

**DIMENSIONS:** they can be larger (if for example loads are added), or smaller (if too much space had been provided) than the dimension of the preceeding computation. If it is not modified, the word **TERM** at least must be used.

**MATERIALS:** the density of the material and the stress-strain law can be changed, but the value of the initial stresses and strains (final value of the saving computation) must be compatible with the new law for all elements.

**CONNECTIONS:** the user can for example add or suppress a blocked displacement; in this case, do not forget to define again the whole instruction. If he wants to cancel all the connections of the preceeding computation, only the keyword **LIAISON** is necessary, without any other sub-instruction.

**LOADS:** the loads may be completely re-defined. Sometimes this is necessary, if the final time defined in the arrays becomes lower than the final time of the instruction **CALCUL**.

**COMPUTATION:** the start time corresponds to the time of restart. If it does not, EUROPLEXUS uses the time written on the restart file. The number of time steps takes the preceeding computation into account. If the user stops after 1000 steps, but wants to continue for a further 500 steps, the total number of time steps will be 1500.

## 18 GROUP RM—CHANGE OF TOPOLOGY ("REMAILAGE")

### Object:

This directive enables "remeshing", i.e. changing the topology of a part of the mesh previously defined by directive "GEOM".

Note that this is very different from mesh "rezoning", i.e. prescribing the motion of a fluid grid in ALE computations. In fact, rezoning operations just move the nodes, without changing the topology (number of nodes and elements, composition of the elements).

For remeshing, two options are possible:

- automatic;
- manual.

With the option "AUTOMATIQUE", the user has to supply only the boundary points of the zone submitted to remeshing, then EUROPLEXUS builds a new and more regular mesh based on the boundary.

With the option "MANUEL", the user must enumerate all the elements which are to be eliminated and supply the new mesh.

At the present time, only 3-noded triangles ("TRIA") can be taken into account.

### Syntax:

```
"REMAILAGE"
$  "AUTOMATIQUE"  . . .  $
$  "MANUEL"       . . .  $
```

### Comments:

The suppressed elements must belong to the same zone (defined by the instruction "GEOMETRIE").



## 18.1 AUTOMATIC REMESHING

### Object:

The new sub-mesh is created by EUROPLEXUS. Therefore, only the boundary points are necessary. As a matter of fact, these points should be common to the old and the new mesh.

### Syntax:

```
"AUTOMATIQUE"  "CONTOUR"  /LECTURE/  
                "AJOUTER"  npts*( xi , yi )  
                < "ELEM"  ideb ifin >  < "ZONE"  nzone >  
                < "MAIL"  < nmail > >  < "STOP" >
```

/LECTURE/

Numbers of the existing points making up the boundary.

npts

Number of points to be added.

xi , yi

Coordinates of the added points (abscissa then ordinate for each of the npts points).

ideb,ifin

The numbers of the elements of the zone concerned are lying between ideb and ifin.

nzone

The elements of the zone to be remeshed belong to the zone number nzone.

nmail

Number of the logical unit (by default nmail = 7), where the data set available for the procedure "LECTURE" of "COCO" is stored.

### Comments:

If the boundary point is an existing node, it is defined by its number.

If it is an added point, it is defined by a negative number. In this case, at the end to the procedure /LECTURE/, the user must enter the coordinates of these points in the same order as they are mentioned in the procedure, by the means of the instruction "AJOUTER".

The directives "ELEM" and "ZONE" enable the elements submitted for remeshing to be limited.

By means of the directive "MAIL", the user can obtain a data set for "COCO" in order to plot the results of the remeshing.

The key-word "STOP" stops EUROPLEXUS when the mesh is created and enables the user to check his mesh.

## 18.2 MANUAL REMESHING

### Object:

This option enables a new partial mesh to be explicitly entered.

### Syntax :

```
"MANUEL"   ize   nelim nelaj npaj
            "ANCIEN" /LECTURE/
            "NOUVEAU"

            . . .   COCO data set . . .

            "IDENTIFIER" npts*( numl numg )
```

#### ize

Number of the zone of elements which must be eliminated.

#### nelim

Total number of the elements to be eliminated.

#### nelaj

Total number of the elements to be added and belonging to the new sub-mesh.

#### npaj

Total number of points belonging to the new sub-mesh.

#### ANCIEN

Keyword defining the old sub-mesh.

#### LECTURE

List of the elements belonging to the sub-mesh to be eliminated.

#### NOUVEAU

Keyword introducing the "COCO" data set of the new sub-mesh.

#### npts

Number of pairs (local number, global number) to be identified.

#### numl

Local number of the node in the sub-mesh

numg

Number of that same node in the global mesh.

**Comments:**

The "COCO" data set with its title is composed of the coordinates of the new sub-mesh (in format 6E12.5) and of the numbers of the mesh elements (in format 18I4).

The key-word "IDENTIFIER" enables the integration of the new sub-mesh into the global mesh to be defined.

### 18.3 CHECKING THE REMESHING

**Object:**

To edit (verify) the result of a remeshing operation.

**Syntax:**

```
< "PERFO" nuperf > < "TRACE" nutrac > < "SAUVE" nusauv >
```

During a restart the user can save data for another restart.

**nuperf**

Number of the logical unit which will contain the "COCO" data necessary to plot a new sub-mesh.

**nutrac**

Necessary to write the results on the file "TRACE" number nutrac.

**nusauv**

To save the results on the file nusauv, so as to enable a further restart.

**Comments:**

The user is advised to use these commands, because the numbers of certain elements may have changed after a remeshing. That is why it is necessary to save that time step ( considered by EUROPLEXUS as step zero). Do not forget to check that the new mesh is correct before the computation continues by means of a restart ("REPRISE").

## 19 GROUP EX—EXAMPLES

On the following pages the user may find some examples of input files. The examples are purely indicative of the 'flavour' of the program. Much more examples could be found as benchmarks of the EUROPLEXUS consortium.

In addition, the user can find many actual EUROPLEXUS examples (including input, pre-treatment and post-treatment files) in some of the publications listed in the bibliography at the end of the present manual.

Unless it is specified otherwise, the mesh is created by means of GIBI. The GIBI data is given so as to define the objects without any ambiguity.

## 19.1 BENDING OF A BEAM

This example is part of the EUROPLEXUS benchmarks and has the name `bm_manex_01`.

### Object:

This is a 2D elastic computation.

A beam is subjected to an uniform stress (pressure).

### Geometry and meshing :

$L = 24.0$  mm : half length of the beam;

$e = 1.0$  mm : thickness;

The mesh is entered in free format;

There are 12 "COQU" shell elements.

### Physical properties:

$\rho = 8000$  kg/m<sup>3</sup> : density;

$\nu = 0.3$  : Poisson's ratio;

$E = 200$  GPa : Young's modulus.

### Boundary conditions:

- The pressure increases from 0 to 2 MPa in 0.1 millisecond, then remains constant.

Clamped boundary and symmetry conditions are applied at the centre.

### Computation:

The step is automatic and the computation ends at 0.01 s.

In order to visualize the results more easily, the computation is followed by the drawing of the time dependant displacement of the centre of the beam.

## List of the input file:

```

VIBRATION      !Title - must be given!
!ECHO          !Output in the console
DPLA           !Two-dimensional plane strain computation
!
! Geometry
!
GEOM LIBR POIN 13 COQU 12 TERM          !Input of the geometry in free format
  0 0  2 0  4 0  6 0  8 0 10 0 12 0
14 0 16 0 18 0 20 0 22 0 24 0
1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9 10 10 11 11 12 12 13
COMP EPAI 1. LECT TOUS TERM            !Thickness of the shell elements
!
! Materials
!
MATE LINE RO 8E-9 YOUN 2E5 NU 0.3 LECT TOUS TERM
!
! Boundary Conditions
!
LINK COUP BLOQ 23 LECT  1 TERM
              13 LECT 13 TERM
CHAR 1 FACT 2 PRES COQU -2. LECT TOUS TERM !Loading with a pressure
      TABL 3 0 0 1E-4 1 1 1              !Time depended variable for the pressure
!
! Outputs
!
ECRI VITE CONT ECRO TFRE 1.E-3          !Output into the listing
      POIN LECT 1 TERM
      ELEM LECT 1 TERM
      FICH ALIC TFRE 1E-4                !Output into the alice file
      FICH PVTk TFRE 1E-4                !Output into the ParaView file
      VARI ECRO CONT DEPL
!
! Options
!
OPTI NOTE
      LOG 1                             !log file is written for each time step
                                           !should not be done for MPI!
!
! Calculation Parameters
!
CALC TINI 0 TEND 10E-3                  !End of calculation at 10e-3 s
=====
SUIT
Post-Processing                          !Title
*
RESU ALIC GARD PSCR
SORT GRAP
*
AXTE 1E3 'TIME (MS)'
*

```



COUR 1 'COQU' DEPLA COMP 2 NOEU LECT 13 TERM

DESS 1 1 AXES 1. 'DEFLECTION (MM)'

\*=====

FIN

## 19.2 IMPACT ON A CIRCULAR PLATE

This example is part of the EUROPLEXUS benchmarks and has the name `bm_manex_02`.

### Object:

This is a 2D plastic computation.

A clamped plate is submitted to the impact of a rigid missile.

### Geometry and meshing :

$D = 465$  mm : diameter of the plate

$e = 6$  mm : thickness

The missile is a flat nose cylinder, 90 mm in diameter. It falls straight in the middle of the plate.

The user wants to follow the displacements of point P1 to P4 whose location has been imposed.

### GIBI data:

```
TITRE 'IMPACT SUR DES PLAQUES CIRCULAIRES' ;
OPTIO DIME 2 ELEM SEG2 ;
DPROJ=90; RPROJ=DPROJ / 2;
DENS 7;
CENTR= 0 0 ; BORD=RPROJ 0 ; BORD2=DPROJ 0 ; ENCAS=233.5 0 ;
  P1=60 0 ;   P2=80 0   ;   P3=120 0   ;   P4=160 0   ;
  P0= 0 0 ;
LIG1=CENTR DROIT BORD ; LIG1=LIG1 COUL ROUG ;
LIG2=BORD D P1 D P2 D BORD2 D P3 D P4 D ENCAS ;
LIG2=LIG2 COUL VERT ;
PLAQ=LIG1 ET LIG2 ;
PROJ=MANUEL POI1 P0;
MESH=PROJ ET PLAQ;
SORTIE MESH;
TASS MESH;
OPTI sauv form 'bm_manex_02.msh';
sauv form MESH;
FIN;
```

### Physical properties:

$\rho = 7800$  kg/m<sup>3</sup> : density;

$\nu = 0.3$  : Poisson's ratio;

$E = 230$  GPa : Young's modulus;

$Y = 188$  MPa : elastic limit.

**Boundary conditions:**

The missile has a mass of 257 kg, it falls at a speed of 11.38 m/s (be careful, the axisymmetric computation concerns ONE radian).

- Clamped boundary and symmetry conditions.

**Computation:**

The step is automatic and the computation concerns the first 12 milliseconds.

In order to visualize the results more easily, the computation is followed by three drawings:

- impulse of the missile
- displacements of the missile and the nodes submitted to the impact
- displacements of the center and the remarkable points

**List of the input file:**

```

--- IMPACT SUR DES PLAQUES CIRCULAIRES (E=6 D=90 M=257 VI=11.38)
ECHO
CAST MESH
AXIS
GEOM COQU PLAQ PMAT PROJ TERM
COMP
      EPAI    6 LECT PLAQ TERM
MATE VMIS ISOT RO 7.8E-9 YOUN 230E3 NU .3 ELAS 188
      TRAC 13  188  .0817E-2
              261  .2000E-2
              288  .3000E-2
              318  .4900E-2
              339  .7500E-2
              354  1.07E-2
              377  1.94E-2
              423  4.75E-2
              497  9.54E-2
              585 18.20E-2
              649 26.20E-2
              693 33.70E-2
              710 37.20E-2
      LECT PLAQ TERM
      MASS 40.903E-3      LECT PROJ TERM
INIT VITE 2 -11380 LECT PROJ TERM
LIAI BLOQ 123 LECT ENCAS TERM
      13 LECT CENTR TERM
      IMPACT DDL 2 COTE -1 PROJ  LECT PROJ TERM
              CIBLE LECT LIG1 TERM

```

```
ECRI VITE CONT ECRO TFRE 1.E-3          !Output into the listing
      POIN LECT 1 TERM
      ELEM LECT 1 TERM
      FICH ALIC TFRE 1E-3                !Output into the alice file
      FICH ALIT TFRE 1E-4 POIN LECT 11 16 21 23 25 31 32 TERM
OPTION NOTEST
CALCUL TINI 0 TEND 12.01E-3
SUIT
IMPACT SUR DES PLAQUES CIRCULAIRES (E=6 D=90 M=257 VI=11.38)
RESU ALIC GARD PSCR
SORT GRAP
AXTEMPS 1000 'TEMPS (MS)'
COURBE 1 'IMPULSION' ECROU COMP 1 ELEM 31
      DESSIN 1 1          AXES 1. 'IMPUL. (N*S)'
COURBE 2 'D-PROJ'      DEPLA COMP 2 NOEU 32
COURBE 3 'D-CENTRE'    DEPLA COMP 2 NOEU 31
COURBE 4 'D-BORD'      DEPLA COMP 2 NOEU 25
COURBE 5 'D-60 (J1)'   DEPLA COMP 2 NOEU 23
COURBE 6 'D-80 (D1)'   DEPLA COMP 2 NOEU 21
COURBE 7 'D-120 (D2)'  DEPLA COMP 2 NOEU 16
COURBE 8 'D-160 (D3)'  DEPLA COMP 2 NOEU 11
      DESSIN 3 2 3 4          AXES 1. 'DEPLA (MM)'
      DESSIN 6 3 4 5 6 7 8    AXES 1. 'DEPLA (MM)'
FIN
```

### 19.3 EXPLOSION IN A TANK

#### Object:

This is a 2-D computation in A.L.E.

A cylindrical tank is filled with water. At its center a micro charge of T.N.T. explodes. the development of a gas bubble (supposed perfect), and deformations of the cylinder may be observed.

#### Geometry and meshing:

Only one quarter of the tank is meshed. Water and gas are modelled by quadrilaterals and the cylinder by thin shell elements. There are also elements of fluid-structure interactions.

$R = 0.19$  m : radius of the cylinder;

$H = 0.19$  m : half height;

$r = 0.299$  m : initial radius of the bubble.

GIBI data:

```
TITRE 'MAILLAGE MANON';
OPTION DIME 2 ELEM QUA4;
RBUL=0.029947;RAY=0.19;
CB=0 0 ; FBR=RBUL 0; FLR=RAY 0;
FBZ=0 RBUL; FLZ=0 RAY;TOP=RAY RAY;
LBR=CB D 5 FBR;LLR=FBR D 12 FLR;
LBZ=CB D 5 FBZ;LLZ=FBZ D 12 FLZ;
AUX=RBUL*(SIN 45);P45=AUX AUX;
FB1=C 5 FBR CB P45;FB2=C 5 P45 CB FBZ;
FBUL=FB1 ET FB2;
BULLE=LBR FB1 FB2 LBZ DALLER PLAN;
TOIT=FLZ D 5 TOP;
SUR =FLR D 5 TOP;
SEP =P45 D 12 TOP;
LIQ1 =LLR SUR SEP FB1 DALLER PLAN;
LIQ2 =SEP TOIT LLZ FB2 DALLER PLAN;
EAU = LIQ1 ET LIQ2;
VA=0 0;CQ1=FLR PLUS VA;CQ2=TOP PLUS VA;
COQ=CQ1 D 5 CQ2;
RAC=RACCOR 0.001 SUR COQ;
PBUL=BULLE CHANGE POI1;
PBUL=PBUL DIFF ((FBUL CHANG POI1) ET CB);
PLIQ1=LIQ1 CHANGE POI1;
PLIQ1=PLIQ1 DIFF ((FB1 ET SEP ET SUR) CHANG POI1);
PLIQ2=LIQ2 CHANGE POI1;
PLIQ2=PLIQ2 DIFF ((FB2 ET SEP) CHANG POI1) ET FLZ);
```

```
PSEP=(SEP CHANG POI1) DIFF (TOP ET P45);  
ZALE=PBUL ET PLIQ1 ET PLIQ2 ET PSEP;  
TOUT=BULLE ET EAU ET RAC ET COQ ET ZALE;  
SORTIE TOUT;  
FIN;
```

**Physical properties:****Water:**

$\rho = 1000 \text{ Kg/m}^3$  : density;  
 $c = 1500 \text{ m/s}$  : velocity of sound.

**Bubble:**

$\rho = 482 \text{ Kg/m}^3$  : density;  
 $p_{ini} = 288 \text{ MPa}$  : initial pressure;  
 $\gamma = 1.535$  :  $C_p/C_v$  ratio.

**Cylinder:**

$\rho = 7900 \text{ Kg/m}^3$  : density;  
 $E = 190 \text{ GPa}$  : Young's modulus;  
 $\nu = 0.3$  : Poisson's ratio;  
 $\sigma_{elas} = 265 \text{ MPa}$  : elastic limit.

**Boundary conditions:**

- The upper part of the tank is very rigid, therefore displacements have been embedded along z.

**Computation :**

The step is automatic and the computation is done during the first two milliseconds.

In order to visualize the results more easily, the computation is followed by the drawing of the time dependant displacements of the generating line of the cylinder.

**List of the input file:**

TEST MANON 11 (1/4) ( IN A.L.E. WITH CAR4 ) !titre

echo

GIBI 9 TOUT

\$TRAC

AXIS ALE

DIME

BLOQ 50 RELA 1 2 NALE 200

TERM

GEOM CAR4 BULLE EAU COQU COQ FS2D RAC TERM

COMPL

EPAIS 1.25E-3 LECT COQ TERM

GRILLE

LAGRANGE LECT COQ FBUL TERM

ALE LECT BULLE EAU TERM

\$ POUR LE CONTACT "FS"

FS LECT RAC TERM

\$ POUR L'EAU

LIGNE BASE LECT CQ2 P45 TERM

LIST LECT PSEP TERM

PLAN BASE LECT CQ1 CQ2 P45 FBR TERM

LIST LECT PLIQ1 TERM

PLAN BASE LECT CQ2 FLZ FBZ P45 TERM

LIST LECT PLIQ2 TERM

\$ POUR LA BULLE

PLAN BASE LECT CB FBR P45 FBZ TERM

LIST LECT PBUL TERM

MATERIAUX

VMIS ISOT RO 7900. YOUNG 190E9 NU .3 ELAS 265E6

TRAC 5 265E6 .00139

352E6 .0202

481E6 .105

559E6 .2214

600E6 .349

LECT COQ TERM

FLUI RO 1000. C 1500. PINI 1E5 PREF 1E5 PMIN 0

LECT EAU TERM

GAZP RO 482. GAMMA 1.535 PINI 2.88E8 PREF 1E5

LECT BULLE TERM

LINK RENUM

BLOQ 1 LECT LBZ LLZ CQ2 TERM

2 LECT TOIT LBR LLR TERM

23 LECT CQ1 CQ2 TERM

ECRITURE

TFREQ .25E-3 TRACE ALICE 10 TFREQ .5E-4

OPTION AUTO

NOTEST

CALCUL tini 0 nmax 1000 tfin 1.005E-3

SUITE

TEST MANON 11 (1/4) ( EN A.L.E. AVEC CAR4 )

RESULT 10

TEMPS 100 COURBE 5 TERM

SORTIE GRAPHIQUE

AXTEMPS 1E3 'T (MILLISEC.)'

COURBE 1 'DR-00 ' DEPLA COMP 1 NOEUD 5

COURBE 2 'DR-38 ' DEPLA COMP 1 NOEUD 6

COURBE 3 'DR-76 ' DEPLA COMP 1 NOEUD 4

COURBE 4 'DR-114 ' DEPLA COMP 1 NOEUD 3

COURBE 5 'DR-152 ' DEPLA COMP 1 NOEUD 2

DESSIN 5 1 2 3 4 5 AXES 1000 'DR-COQUE (MM)'

FIN



## 19.4 MODELLING OF PERFORATED PLATES

### Object:

This is a 2-D computation in A.L.E.

A plane wave is propagating in a cylindrical tube. It meets a perforated plate that generates partial reflections and head losses. The walls of the cylinder are supposed rigid and the plate flexible.

### Geometry and meshing:

Only the fluid in the tube is meshed. Elements with absorbant boundary conditions shall avoid reflected waves. At one end, a range of CL2D elements enables pressure source to be input in the form of a slope. These elements are superimposed to the absorbant elements.

The clamped plate is meshed with shell elements. The fluid is meshed in a continuous way to make it pass through the plate. The connecting elements of the fluid-structure junction ensure the coupling between the plate and the neighbouring nodes of the fluid. Another range of CL2D elements enables the characteristics of the grid to be input for the computation of the head losses.

$R = 500$  mm : radius of the cylinder;

$H = 1000$  mm : half height.

### GIBI data:

```
TITRE 'MAILLAGE D'UNE PLAQUE PERFOREES';
OPTI DIME 2 ELEM QUA4 NIVEAU 1;
PA=0 0 ; PB=500 0 ; PC=500 2000 ; PD=0 2000;
PE=0 1000 ; PF=500 1000;
ENT=PA D 5 PB ; SOR=PC D 5 PD ;
GRI=PF D 5 PE ; ENT2=ENT ;
LBF=PB D 10 PF ; LFC=PF D 10 PC ;
LDE=PD D 10 PE ; LEA=PE D 10 PA ;
LIQ1=ENT LBF GRI LEA DALLER PLAN ;
LIQ2=(INVE GRI) LFC SOR LDE DALLER PLAN ;
LIQ=LIQ1 ET LIQ2 ;
PCE=0 1000 ; PCF=500 1000 ;
COQ=PCF D 5 PCE;
FSE=RACC 0.00001 LIQ1 COQ ;
PLIQ=LIQ CHAN POI1;
PLIQ=PLIQ DIFF (GRI CHAN POI1);
AXE=LDE ET LEA ET PCE ;
BOR=LBF ET LFC ET PCF ;
LIQ=LIQ COUL BLEU;COQ=COQ COUL ROUG;
FSE=FSE COUL TURQ;ENT2=ENT2 COUL VERT;
```

```
TOUT=LIQ ET COQ ET FSE ET PLIQ ET AXE ET BOR ET ENT2;
SORT TOUT;
fin;
```

### Physical properties:

#### Fluid:

rho = 1000 Kg/m3 : density  
c = 1000 m/s : velocity of sound

#### Plate:

rho = 7800 Kg/m3 : density  
E = 190 GPa : Young's modulus  
nu = 0.3 : Poisson's ratio

### Computation:

The step is automatic and the computation is carried out for the first 6 milliseconds.

In order to visualize the results more easily, the computation is followed by two drawings. The first one describes the evolution of the pressures upstream and downstream, and the second describes the displacement of the center of the plate

### List of the input file:

```
PLAQUE PERFOREE SOUPLE (EULER) ALP=1 TAU=0 E=15
ECHO
GIBI 9 TOUT
AXIS EULER
DIMENSION
    BLOQ 60 NALE 100
    TERM
GEOM CAR1 LIQ CL2D ENT SOR GRI ENT2 COQU COQ FS2D FSE TERM
GRIL LAGR LECT COQ TERM
    ALE LECT LIQ TERM
    FS LECT FSE TERM
COMPLEMENT EPAIS 15 LECT COQ TERM
MATE LINE RO 7.8E-9 YOUNG 190E3 NU 0.3
    LECT COQ TERM
    FLUI RO 1E-9 C 1E6
    LECT LIQ TERM
    IMPE PIMP RO 1E-9 PRES -0.02
    TABP 3 0 0 0.002 1 1 1
```

```
      LECT ENT  TERM
IMPE ABSO RO 1E-9 C 1E6
      LECT SOR ENT2 TERM
IMPE GRIL RO 1E-9 C 1E6 ALP 1 TAU 0
      LECT GRI  TERM
LIAI RENUM
      BLOQ 1  LECT AXE  TERM
          1  LECT BOR  TERM
          2  LECT PCF  TERM
          3  LECT PCE PCF TERM
      FS LECT FSE TERM
IMPR FREQ 500
      TRAC ALIC 10  10
OPTI PAS AUTO
      NOTEST
CALCUL 0. 1E-5    1E-4    500    6E-3
SUITE
PLAQUE PERFOREE SOUPLE (EULER) ALP=1 TAU=0 E=15
RESULT 10
TEMPS 60 COURBE 7 TERM
SORTIE GRAPHIQUE
AXTEMPS 1E3 'T (MILLISEC.)'
COURBE  1 'P-3    '  ECROU COMP 1 ELEM  3
COURBE  2 'P-28   '  ECROU COMP 1 ELEM 28
COURBE  3 'P-48   '  ECROU COMP 1 ELEM 48
COURBE  4 'P-53   '  ECROU COMP 1 ELEM 53
COURBE  5 'P-73   '  ECROU COMP 1 ELEM 73
COURBE  6 'P-98   '  ECROU COMP 1 ELEM 98
COURBE  7 'DZ-64  '  DEPLA COMP 2 NOEU 64
DESSIN 6  1 2 3 4 5 6  AXES 10  'PRES. (BARS)'
DESSIN 1  7              AXES 1.  'DEPLA. (MM)'
FIN
```

## 20 DEVELOPMENT NOTES

### 20.1 PROGRAMMING GUIDELINES

The present document (User's Manual) deals primarily with a detailed description of all the input commands of the EUROPLEXUS code, in the order in which they appear in a typical input file. It is meant to be an exhaustive *reference* for Users of the code, but it is *not a tutorial*.

Some short indications are given in this Section, which might be useful to EUROPLEXUS code *developers* (i.e. to *programmers*), in the form of simple programming guidelines. Developers are assumed to have access to the EUROPLEXUS Consortium web site, which contains among other things all the available technical documentation.

Only an overview is presented here for brevity. Reference is made to several reports and other documents, listed in the bibliography and available on-line on the Consortium web site, which contain all the details.

#### 20.1.1 Programming Language

EUROPLEXUS is entirely written in the Fortran language. For historical reasons some parts of the code (the oldest ones) are still coded in Fortran 77 (F77). All new developments from 1999 onwards use the Fortran 90 (F90) language, mainly based upon F90 Modules.

A great effort of converting F77 parts to F90 is being performed, but the process is not yet complete. An example of successful conversion is the materials data structure, which has been completely ported to F90.

#### 20.1.2 F77 Programming

An introduction to F77 programming style in EUROPLEXUS is given in reference [150].

#### 20.1.3 F90 Programming

An introduction to F90 programming style and some tentative guidelines for the development of EUROPLEXUS are given in reference [175].

#### 20.1.4 Code Profiling

A code profiler is **not** readily available under Windows. A simple mechanism is set up to profile selected parts of the code (in a platform-independent way) by using the standard system routine `CPU_TIME`, which returns the job's CPU time in seconds.

This profiling cannot be totally automatic as e.g. under Linux, but it is set up in such a way:

- to have **virtually no impact** on the code standard version, i.e. when no profiling is activated.
- to require only a **very small** intervention on routines (or any code parts) to be profiled.

The mechanism is very simple. Assume we want to profile a routine `SUB1`. Then the following standard code is inserted at the beginning of the routine (before the first executable statement) and at the end of the routine (before the `RETURN` or the `END SUBROUTINE`):

```

SUBROUTINE SUB1
...
USE M_PROFILE
...

... routine body (executable statements)

END SUBROUTINE SUB1

```

Note that the two code portions to be inserted can be just copied (e.g. from LOOPELM.FF) and pasted as such. Only the name of the subroutine (SUB1) in the `CALL GET_PROFILE_INDEX` has to be changed.

To activate profiling of the routine, compile it with the `PROFILE` keyword activated for filtering. Under Windows the command is:

```
eplx_cmp -o -k PROFILE
```

Then link normally and run the code normally. At the end of the listing the profile table is printed, reporting the CPU time spent in each profiled portion of the code (both in absolute terms and in % of total CPU time), and the number of calls of each profiled routine.

in the routines, since they have no impact on normal code efficiency thanks to the standard EUROPLEXUS filtering mechanism.

Obviously, this type of profiling is not only limited to entire subroutines. The same mechanism can also be used to profile other code parts, e.g. a suspect DO loop. The name passed to `GET_PROFILE_INDEX` can be freely chosen by the user, it is not necessarily a subroutine name (maximum length is 32 characters, including any inter-word blanks).

Profiling instructions (filtered) will be gradually inserted in at least the most important routines of the code (CELEM, LOOPELM etc.) so that first-attempt profiling can become as simple as the following commands (under Windows):

```

eplx_modtree M_PROFILE      (search routines already containing M_PROFILE)
eplx_get_users              (copy all such routines in current directory)
<possibly add some other routines for profiling>
eplx_cmp -o -k PROFILE      (compile all local routines for profiling)
eplx_lk -o                  (link to produce a local executable eplx.exe)
eplx_bench -e eplx.exe ... (run tests and obtain profile)

```

Based on the results of this first-level profiling, other routines can then be added to the list for a refinement of profiling results.

The implementation uses a very small module `M_PROFILE` containing some very small **static** tables (no allocation / deallocation), with space for up to `NPROFMAX` (currently 100) routines to be profiled.

A permanent (unfiltered) profiling based upon the above mechanism has been added in the time loop routines: `CALCUL` for the normal case or `TLOOPP` for the case with spatial partitioning. Thus, by default the following main “blocks” of code are profiled:

- VFCC (cell-centred Finite Volumes)
- Finite Elements
- External Forces
- Coupled links (LINK COUP)

- Coupled links by the old “liaison” implementation (LINK LIAI)
- Decoupled links (LINK DECO)
- Outputs

This should give a first-level (and fully automatic) overview of where in the code the majority of CPU time is spent, for a given application. Then, refinement of the profiling can be performed if necessary by the technique explained above on the affected code portion.

## 20.2 PRECOMPILER KEYWORDS

Several precompiler keywords are used before the compilation, shown in the table below. Concerning the versions, 'X' indicates that it is used, '(X)' means that it is used in some cases and '-' that it is never used.

Keyword	Description	Versions	
		Win Ispra	L64 CEA
CADNA	Version CADNA	-	(X)
CHECK_DATA	SALOME: check only syntax validity of epX file	-	-
CYGWIN	Windows version with CYGWIN	-	-
ER_LIGHT_VERSION	Filtering for the research light version	-	(X)
ER_VERSION	Filtering for the research version	-	(X)
EPX2XML	Version EPX2ML DATA for Salome	-	(X)
FRANCAIS	Output: French version, otherwise English version	-	X
GFORTRAN	GFORTRAN Compiler	-	(X)
KAAPI	Need library KAAPI	-	X
MED	Need library MED	-	X
MFFT	Need library MFFT	X	X
MFRONT	Need library MFRONT	-	(X)
MKL	Need library MKL (MKL is part of the Intel compiler)	X	X
MPI	Version MPI	(X)	X
NCOCO	Need library NCOCO	X	(X)
OGL	OpenGL (Windows)	X	-
ONLYF90	Fortran 90 pur, no extension Fortran 2000, 2003, etc....	-	X
PETSC	Use of PETSc (EDF)	-	(X)
PROD_VERSION	Filtering for the production version	-	(X)
PROFILE	Performance profile calculation	-	-
QWIN	Quick Win	-	-
SOFA	Need library SOFA	-	-
T_LINUX	Version OGL for Linux	-	(X)
UNIX32	Not windows version	-	X
W32	Windows 32 bit	(X)	-
W64	Windows 64 bit	X	-
WIN	Windows (32 and 64 bit)	X	-
XLF	XLF (IBM/AIX) compiler	-	X

Keywords for the manual

Keyword	Description	Versions	
		HTML	PDF
FIGURES	Including the figures	X	X
HEVEA	HAVEA version	X	-
LATEX1	?	-	-
PDFLATEX	PDFLATEX version	-	X
SALOME	Special treatment for SALOME menus	-	-
TTH	TTH	Z	-

## 20.3 MANUAL CREATION

This documentation is produced starting from a  $\text{\LaTeX}$  source file that is continuously updated and maintained. A formatted version of the manual, such as the present one, can be obtained for on-line consulting either via PDF or via HTML.

A table of contents and an index can be found at the beginning and at the end of this documentation, respectively, and may help to rapidly retrieve information.

### 20.3.1 PAGE NUMBERING CONVENTIONS

Each page of the present manual contains information in both an header (at the page top) and a footer (at the page bottom).

The header contains in its middle part a ‘page identification’ used to identify each page or sequence of pages dealing with a specific topic. This is composed by a letter or group of letters, indicating a ‘section’ of the manual, and a number. The same information is also repeated in the left part of the footer. The number is not the actual page number (this is given in the right part of the footer, instead), since this is subject to frequent changes as text is added or reformatting is performed. Cross references in the text are generally done by referring to the page identification.

The numbers appearing in the page identification are not incremented by 1, but have larger increments (usually 10) in order to leave space for future pages.

The page header contains in its right part a date indicating when a certain page has been inserted or modified in the manual.

The date in the centre of the footer corresponds to the manual edition date.



## 20.4 ACKNOWLEDGEMENTS, LICENSES

Several libraries are used in the EUROPLEXUS code.

### 20.4.1 lib\_vtk\_io

This library is used to write the vtk-files that are used in order to allow an output for ParaView. The library is written by Stefano Zaghi and can be find here:

<https://github.com/szaghi/Lib-VTK-IO>.

Copyright (c) 2014, Stefano Zaghi All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided by the copyright holders and contributors "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the copyright holder or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

### 20.4.2 Blas, Lapack

The packages can be obtained from <http://www.netlib.org/blas> and <http://www.netlib.org/lapack>.

Copyright (c) 1992-2007 The University of Tennessee. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer listed in this license in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holders nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided by the copyright holders and contributors "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the copyright owner or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

### 20.4.3 OpenMPI

Copyright (c) 2004-2010 The Trustees of Indiana University and Indiana University Research and Technology Corporation. All rights reserved.

Copyright (c) 2004-2010 The University of Tennessee and The University of Tennessee Research Foundation. All rights reserved.

Copyright (c) 2004-2010 High Performance Computing Center Stuttgart, University of Stuttgart. All rights reserved.

Copyright (c) 2004-2008 The Regents of the University of California. All rights reserved.

Copyright (c) 2006-2010 Los Alamos National Security, LLC. All rights reserved.

Copyright (c) 2006-2010 Cisco Systems, Inc. All rights reserved.

Copyright (c) 2006-2010 Voltaire, Inc. All rights reserved.

Copyright (c) 2006-2011 Sandia National Laboratories. All rights reserved.

Copyright (c) 2006-2010 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.

Copyright (c) 2006-2010 The University of Houston. All rights reserved.

Copyright (c) 2006-2009 Myricom, Inc. All rights reserved.

Copyright (c) 2007-2008 UT-Battelle, LLC. All rights reserved.

Copyright (c) 2007-2010 IBM Corporation. All rights reserved.

Copyright (c) 1998-2005 Forschungszentrum Juelich, Juelich Supercomputing Centre, Federal Republic of Germany

Copyright (c) 2005-2008 ZIH, TU Dresden, Federal Republic of Germany

Copyright (c) 2007 Evergrid, Inc. All rights reserved.

Copyright (c) 2008 Chelsio, Inc. All rights reserved.

Copyright (c) 2008-2009 Institut National de Recherche en Informatique. All rights reserved.

Copyright (c) 2007 Lawrence Livermore National Security, LLC. All rights reserved.

Copyright (c) 2007-2009 Mellanox Technologies. All rights reserved.

Copyright (c) 2006-2010 QLogic Corporation. All rights reserved.

Copyright (c) 2008-2010 Oak Ridge National Labs. All rights reserved.

Copyright (c) 2006-2010 Oracle and/or its affiliates. All rights reserved.

Copyright (c) 2009 Bull SAS. All rights reserved.

Copyright (c) 2010 ARM Ltd. All rights reserved.

Copyright (c) 2010-2011 Alex Brick . All rights reserved.

Copyright (c) 2012 The University of Wisconsin-La Crosse. All rights reserved.

Copyright (c) 2013-2014 Intel, Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer listed in this license in the documentation and/or other materials provided with the distribution.

- Neither the name of the copyright holders nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

The copyright holders provide no reassurances that the source code provided does not infringe any patent, copyright, or any other intellectual property rights of third parties. The copyright holders disclaim any liability to any recipient for claims brought against recipient by any third party for infringement of that parties intellectual property rights.

This software is provided by the copyright holders and contributors "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the copyright owner or

contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

#### 20.4.4 Glut

Copyright (c) Mark J. Kilgard, 1994, 1997. The programs are not in the public domain, but they are freely distributable without licensing fees. These programs are provided without guarantee or guarantee expressed or implied.

#### 20.4.5 f90gl, f90glu, f90glut

The package can be taken from here <http://math.nist.gov/f90gl/>.

The research software provided on this web site ('software') is provided by NIST as a public service. You may use, copy and distribute copies of the software in any medium, provided that you keep intact this entire notice. You may improve, modify and create derivative works of the software or any portion of the software, and you may copy and distribute such modifications or works. Modified works should carry a notice stating that you changed the software and should note the date and nature of any such change. Please explicitly acknowledge the National Institute of Standards and Technology as the source of the software.

The software is expressly provided 'as is.' NIST makes no warranty of any kind, express, implied, in fact or arising by operation of law, including, without limitation, the implied warranty of merchantability, fitness for a particular purpose, non-infringement and data accuracy. NIST neither represents nor warrants that the operation of the software will be uninterrupted or error-free, or that any defects will be corrected. NIST does not warrant or make any representations regarding the use of the software or the results thereof, including but not limited to the correctness, accuracy, reliability, or usefulness of the software.

You are solely responsible for determining the appropriateness of using and distributing the software and you assume all risks associated with its use, including but not limited to the risks and costs of program errors, compliance with applicable laws, damage to or loss of data, programs or equipment, and the unavailability or interruption of operation. This software is not intended to be used in any situation where a failure could cause risk of injury or damage to property. The software was developed by NIST employees. NIST employee contributions are not subject to copyright protection within the United States.

## 21 BIBLIOGRAPHY

This section provides a list of the bibliography related to EUROPLEXUS (from December 1999 on) and its ancestors: PLEXIS-3C (including its own ancestors, the codes of the EURDYN series), and CASTEM-PLEXUS.

The section is subdivided into six sub-sections:

- Some selected references of the EURDYN series of codes,
- References pertaining to PLEXIS-3C and EUROPLEXUS development performed at JRC,
- References pertaining to CASTEM-PLEXUS and EUROPLEXUS developments performed at CEA,
- References pertaining to PLEXUS and EUROPLEXUS developments performed at Samtech.
- References pertaining to CASTEM-PLEXUS and EUROPLEXUS developments performed at EDF,
- References pertaining to EUROPLEXUS developments performed at ONERA.

The documents from CEA contains proprietary or other confidential or privileged informations. The contents may only be used, reproduced, published or communicated to others as provided within the terms of the agreements between CEA and the receiving party.

It is also recalled that an updated on-line version of the EUROPLEXUS User manual is available on the Internet and may be consulted interactively from a workstation or terminal.

## References

- [1] J. Donéa, S. Giuliani. *EURDYN: Finite element codes for dynamic analysis of large displacement, small strain problems with material nonlinearities*, 3rd Int. Conference on SMiRT, paper M2/5, London, 1–5 September, 1975.

### ESSENTIAL EURDYN BIBLIOGRAPHY (before 1986)

- [2] J. Donéa, S. Giuliani, J.P. Halleux. *Theoretical aspects of the EURDYN computer programs for non-linear transient dynamic analysis of structural components*, EUR 5473e, C.C.R. Ispra, May 1976.
- [3] J. Donéa, S. Giuliani. *EURDYN computer programs for transient dynamic analysis of large-displacement, small-strain problems with material non-linearities (User's manual)*, EUR 5474e, C.C.R. Ispra, May 1976.
- [4] J. Donéa, P. Fasoli-Stella, S. Giuliani. *Finite element solution of transient fluid-structure problems in Lagrangian coordinates*, International Meeting on fast reactor safety and related physics, Chicago, USA, CONF-761001, Vol. 3, pp. 1427–1435, 5–8 Oct. 1976.
- [5] J. Donéa, P. Fasoli-Stella, S. Giuliani. *Lagrangian and Eulerian finite element techniques for transient fluid-structure interaction problems*, 4th Int. Conference on SMiRT, paper B1/2, S. Francisco, USA, 15–19 August, 1977.
- [6] J. Donéa, S. Giuliani, J.P. Halleux. *Prediction of the nonlinear dynamic response of structural components using finite elements*, International Seminar on Extreme Load Condition and Limit Analysis Procedures for Structural Reactor Safeguards and Containment Structures, Berlin, 8–11 Sept. 1975. Nuclear Engineering and Design, Vol. 37, pp. 95–114, 1976.
- [7] J. Donéa, S. Giuliani. *The computer code EURDYN-1M (Release 1). Part 2: User's manual*, EUR 6425en, C.C.R. Ispra, 1979.
- [8] J. Donéa, P. Fasoli-Stella, S. Giuliani, J.P. Halleux, A.V. Jones. *An Arbitrary Lagrangian Eulerian finite element procedure for transient dynamic fluid-structure interaction problems*, 5th Int. Conference on SMiRT, paper B1/3, Berlin, 13–17 August, 1979.
- [9] J. Donéa, H. Laval. *An improved formulation of the parabolic isoparametric element for explicit transient analysis*, Earthquake Eng. and Struc. Dynamics, Vol. 7, pp 23–29, 1979.
- [10] F. Casadei, J.P. Halleux. *1-D numerical simulation of the Large Dynamic Test Facility*. Technical Note N. I.06.01.79.77, July 1979.
- [11] F. Casadei. *Risoluzione di problemi di impatto monodimensionale con il codice EURDYN-1D*. Ispra, May 1979.
- [12] F. Casadei. *Approccio numerico a un problema di perforazione percussiva con il codice EURDYN-2*. Ispra, May 1979.
- [13] F. Casadei. *Sviluppo di un modello di Mohr-Coulomb per le rocce*. Ispra, May 1979.
- [14] J. Donéa. *Finite element analysis of transient dynamic fluid-structure interaction*, Chapter in Advanced Structural Dynamics, Editor J. Donéa, Applied Science Editor, 1980.
- [15] J. Donéa, S. Giuliani, J.P. Halleux. *Recent improvements of the nonlinear transient dynamic structural computer programs EURDYN*, EUR 6694, C.C.R. Ispra, 1980.

- [16] J. Donéa, P. Fasoli-Stella, S. Giuliani, J.P. Halleux, A.V. Jones. *The computer code EURDYN-1M (Release 1) for transient dynamic fluid-structure interaction. Part 1: governing equations and finite element modelling*, EUR 6751, C.C.R. Ispra, 1980.
- [17] J. Donéa, S. Giuliani. *Fluid-structure interaction in problems with interfaces involving sharp corners*, 6th Int. Conference on SMiRT, paper B1/3, Paris, 17–21 August, 1981.
- [18] J. Donéa, S. Giuliani, J.P. Halleux, P. Granet, P. Hamon, P. Permezel. *Loading and response of fast-reactore core subassemblies*, CONFAVRE-3 Internal Loading and Containment of Fast Breeder Reactors, Ispra, 24–25 August, 1981.
- [19] J. Donéa, S. Giuliani, J.P. Halleux, P. Granet, P. Hamon, P. Permezel. *Response of fast reactor core subassemblies to pressure transient*, Nuclear Engineering and Design, Vol. 68, N. 1, pp 153–174, 1981.
- [20] S. Giuliani. *An algorithm for continuous rezoning of the hydrodynamic grid in Arbitrary Lagrangian Eulerian computer codes*, Nuclear Engineering and Design, Vol. 72, pp. 205–212, September 1982.
- [21] J. Donéa, S. Giuliani, J.P. Halleux. *An Arbitrary Lagrangian-Eulerian finite element method for transient dynamic fluid-structure interaction*, 2nd International Conference on FENOMECH, invited paper, University of Stuttgart, 25–28 Aug. 1981. Computer Methods in Applied Mechanics and Engineering, Vol. 33, pp. 689–723, 1982.
- [22] J. Donéa. *Arbitrary Lagrangian-Eulerian finite element methods*, Chapter 10 in Computational Methods for Transient Analysis, Editors T. Belytschko and T.J.R. Hughes, North Holland Publishing Company, Amsterdam, 1983.
- [23] J.P. Halleux, F. Casadei. *TPLOT : an interactive data management system for transient problems. 2nd edition*, EUR 9553 EN, 1984.
- [24] F. Casadei, J.P. Halleux. *Some calculations on the effect of misalignment on the behaviour of a modified Hopkinson-bar device*, Technical Note N. I.07.C1.84.04, Jan. 1984.
- [25] J.P. Halleux, F. Casadei. *Transient large strain analysis of plane and axisymmetric structures by means of biquadratic finite elements*, 2nd Int. Conf. on Numerical Methods for Nonlinear Problems, Barcelona, Apr. 9–13, 1984.
- [26] F. Casadei, J.P. Halleux. *Identification of the Large Dynamic Test Facility by the code EURDYN-1D*, Technical Note N. I.07.C1.84.60, May 1984.
- [27] F. Casadei. *1-D Hopkinson-bar calculations*. Technical Note N. I.06.C1.84.109, Aug. 1984.
- [28] F. Casadei, J.P. Halleux. *Updating of the 1-D numerical model of L.D.T.F. with UNI-38NCD4 material*. Technical Note N. I.06.C1.84.141, Oct. 1984.
- [29] J.P. Halleux, F. Casadei. *Transient large strain analysis of plane and axisymmetric structures by means of biquadratic finite elements*. Engineering Computations, Vol. 1, pp. 351–358, Dec. 1984.
- [30] F. Casadei, J.P. Halleux. *1-D numerical model of L.D.T.F. with maraging steel material*. Technical Note N. I.06.C1.84.186, Dec. 1984.
- [31] P. Pegon, J.P. Halleux, J. Donéa. *A discussion of stress rates and their application in finite strain plasticity*, Proceedings of NUMETA'85 Conference, Editors J. Middleton and G.N. Pande, pp. 315–325, Balkema, 1985.

- [32] J.P. Halleux, J. Donéa. *A discussion of Cauchy stress formulation for large strain analysis*, Proceedings of Europe-US Symposium on FE in Nonlinear Problems, Editor P.G. Bergan, paper 1.7 Vol. 1, University of Trondheim, Norway, 1985.
- [33] F. Casadei, J.P. Halleux. *EURDYN-1D : a computer code for the one-dimensional non-linear dynamic analysis of structural systems. Description and users' manual*. EUR 10115 EN, 1985.
- [34] F. Casadei. *Scoping 1-D calculations for the Large Dynamic Test Facility*. Technical Note N. I.06.C1.85.114, July 1985.
- [35] F. Casadei, C. Delzano, G. Verzeletti. *L.D.T.F.: a large dynamic test facility for the investigation of the dynamic behaviour of structures*. 8th Int. Conf. on S.M.i.R.T., Brussels, Aug. 19–23, 1985.
- [36] F. Casadei, J.P. Halleux. *Response of an elastoplastic circular plate submitted to dynamic pressure loading: a code validation exercise*. Technical Note N. I.06.C1.85.122, September 1985.
- [37] F. Casadei, J. Donéa, S. Giuliani, J.P. Halleux, G. Verzeletti. *Recenti sviluppi nei metodi computazionali e sperimentali per l'analisi dinamica di strutture nel campo nonlineare*. XIII Convegno Nazionale dell'Associazione Italiana per l'Analisi delle Sollecitazioni, presso l'I.S.M.E.S, Bergamo, 23–27 Settembre 1985.

## PLEXIS-3C/EUROPLEXUS BIBLIOGRAPHY

## 1986:

- [38] F. Casadei, G. Tamborini. *1-D numerical model of L.D.T.F. with dovetail specimen's attachment*. Technical Note N. I.06.C1.86.52, June 1986.
- [39] H. Molinaro. *Rapport de Stage. 14 Avril 1986 - 6 Juin 1986*. Technical Note N. I.06.C1.86.53, June 1986.
- [40] F. Casadei. *A bibliographic study of finite elements for the elasto-plastic analysis of 3-D shell-like structures subjected to static and dynamic loading*. Technical Note N. I.06.C1.86.79, Aug. 1986.
- [41] F. Casadei, J. Donéa, J.P. Halleux, L.G. Lamain. *Some experience with shell analysis*. 3rd Int. Conf. on Numerical Methods for Nonlinear Problems, Dubrovnik, Sep. 15–18, 1986.
- [42] J. Donéa. *On an alternative rate of Cauchy stress*, First World Congress on Computational Mechanics, Austin, Texas, September 22–26, 1986.
- [43] F. Casadei, C. Delzano, G. Magonette, J.P. Halleux, G. Verzeletti. *Dynamic testing of large AISI-316 L steel specimens behaviour using L.D.T.F.* 12th MPA-Seminar: Safety and Availability of Plant Technology, Staatliche Materialpruefungsanstalt, Stuttgart, Oct. 9–10, 1986.
- [44] F. Casadei, J.P. Halleux, H. Molinaro. *Preliminary results of the numerical simulation of plastic test LDTF21 by 1-D and 2-D models*, Technical Note N. I.06.C1.86.98, Oct. 1986.

## 1987:

- [45] J.P. Halleux, F. Casadei. *Transient large strain finite element analysis of solids*. In 'Computational Methods for Non-Linear Problems', 2nd Volume in the Series: Recent Advances in Non-Linear Computational Mechanics, Ed. C. Taylor, D.R.J. Owen and E. Hinton, Pineridge Press, Swansea, 1987.
- [46] F. Casadei, C. Delzano, G. Magonette, J.P. Halleux, G. Verzeletti. *Dynamic testing of large AISI-316 L steel specimens behaviour using LDTF*, Nuclear Engineering and Design, Vol. 102, pp. 463–474, 1987.
- [47] F. Casadei, J.P. Halleux, G. Verzeletti, A.G. Youtsos. *Application of numerical models to dynamic flow of steel specimens in the large dynamic test facility*, International Conference on Computational Plasticity Models, Software and Applications, Barcelona, Spain, April 6–10, 1987.
- [48] J. Donéa, A.G. Youtsos, F. Casadei. *A stress update algorithm for the theory of viscoplasticity based on total strain and overstress*, International Conference on Computational Plasticity Models, Software and Applications, Barcelona, Spain, April 6–10, 1987.
- [49] P.M. Jones, F. Casadei, J.P. Halleux. *Progress in numerical modelling of dynamic specimen behaviour in the L.D.T.F.*, 13th MPA-Seminar: The Contribution of Component and Large Specimen Testing to Structural Integrity Assessment, Staatliche Materialpruefungsanstalt, Stuttgart, Oct. 8–9, 1987.
- [50] F. Casadei. *Numerical simulation of axisymmetric steel specimens tests in LDTF*, Technical Note N. I.87.122, Oct. 1987.



**1988:**

- [51] F. Casadei, G. Verzeletti. *Dynamic analysis of structural components: some recent advances in experiments and computations*, 4th Cairo University Conference on Mechanical Design and Production (MPD-4), Cairo, Dec. 27–29, 1988.
- [52] J. Donéa, S. Giuliani. *An explicit ALE finite element formulation for 3D transient dynamic fluid-structure interaction problems*, EUR 11936en, C.C.R. Ispra, 1988.
- [53] S. Giuliani, A. Rossi. *Test problems for the validation of computer codes TRAFU-2D and TRAFU-3D*, Technical Note I.88.125, C.C.R. Ispra, October 1988.
- [54] S. Giuliani, J. Donéa. *The computer code TRAFU-3D (Release 1) for transient convective heat transfer: User's manual*, Technical Note I.88.144, C.C.R. Ispra, December 1988.

**1989:**

- [55] H. Bung, F. Casadei, J.P. Halleux, M. Lepareux. *PLEXIS-3C: a computer code for fast dynamic problems in structures and fluids*, 10th International Conference on Structural Mechanics in Reactor Technology, Anaheim, U.S.A., August 14–18, 1989.
- [56] F. Casadei, J.P. Halleux. *Validation of the PLEXIS-3C computer code on academic and real test cases in structural dynamics*, 10th International Conference on Structural Mechanics in Reactor Technology, Anaheim, U.S.A., August 14–18, 1989.
- [57] F. Casadei, A. Daneri, G. Toselli. *Use of PLEXUS as a LMFBR primary containment code for the CONT benchmark problem*, 10th International Conference on Structural Mechanics in Reactor Technology, Anaheim, U.S.A., August 14–18, 1989.
- [58] P.M. Jones, F. Casadei, J.P. Halleux. *Progress in numerical modelling of dynamic specimen behaviour in the L.D.T.F.*, Nuclear Engineering and Design, Vol. 112, pp. 51–64, 1989.
- [59] J. Donéa. *Finite strain plasticity in transient dynamic calculations*, Proc. 5th Int. Symp. on Numerical Methods in Engineering, Vol. 1, pp. 147–154, Springer Verlag, 1989.
- [60] A.G. Youtsos, J. Donéa, G. Verzeletti. *Viscoplastic behaviour of stainless steels AISI 316L and 316H*, Acta Mechanica, Vol. 76, pp. 161–187, 1989.
- [61] J.P. Halleux. *Computational Treatment of Transient Problems in Nonlinear Structural Mechanics*, International Centre for Mechanical Sciences, Udine, Courses and Lectures, No. 300, Chapter 2 in “Advances in Computational Nonlinear Mechanics”, edited by I. St. Doltsinis, Springer Verlag, pp. 81–108, 1989.

**1990–1991:**

- [62] F. Casadei. *The network printing installation of AMD at building 42*, Technical Note N. I.91.12, February 1991.
- [63] F. Casadei. *Filters for text and graphics processing and printing - Part I : Descriptions*, Technical Note N. I.91.19, February 1991.
- [64] F. Casadei. *Filters for text and graphics processing and printing - Part II : Sources*, Technical Note N. I.91.20, February 1991.

- [65] F. Casadei. *EXTRACT: a tool for the management of FORTRAN programs*, Technical Note N. I.91.21, February 1991.
- [66] F. Casadei. *T-Rend: a Starbase graphics interface for interactive visualization of tridimensional finite element data - Release 1.0*, Technical Note N. I.91.22, February 1991.
- [67] F. Casadei, J.P. Halleux and A. Huerta. *Dynamic response of fluid-structure systems by PLEXIS-3C*, European Conference on New Advances in Computational Structural Mechanics, Giens, France, April 2–5, 1991.
- [68] J. Donéa. *Hydrodynamic models for transient fluid-structure interaction*, Proc. European Conference on New Advances in Computational Structural Mechanics, pp. 355–359, Giens (F), 2–5 April 1991.
- [69] F. Casadei. *The current PLEXIS-3C and EURDYN-1D interface to EDF*, Technical Note N. I.91.85, July 1991.
- [70] A. Huerta, F. Casadei. *Arbitrary Lagrangian-Eulerian formulation for large boundary motion in nonlinear continuum mechanics*, Technical Note N. I.91.95, August 1991.
- [71] F. Casadei. *Automatic Processing and Printing of the PLEXIS-3C User's Manual*, Technical Note N. I.91.125, October 1991.
- [72] F. Casadei, P. Di Giamberardino, J.P. Halleux, A. Poggianti. *Acquisizione di conescenze avanzate nel campo della dinamica rapida non lineare* ENEA, Dipartimento Reattori Innovativi, Rapporto N. CT-WHE-00001, December 1991.

**1992:**

- [73] F. Casadei. *TPlot: an interactive data management system for transient problems*, Technical Note N. I.92.96, March 1992.
- [74] J. Donéa, A. Huerta, F. Casadei. *Finite element models for transient dynamic fluid-structure interaction*, in New Advances in Computational Structural Mechanics, Studies in Applied Mechanics 32, (Ed. P. Ladevèze and O.C. Zienkiewicz), Elsevier, 1992.
- [75] A. Huerta, F. Casadei, J. Donéa. *An arbitrary Lagrangian-Eulerian stress update procedure for coining simulations*, 4-th Int. Conf. on Numerical Methods in Industrial Forming Processes, NUMIFORM '92, Sophia Antipolis, France, 14–18 Sept. 1992.
- [76] A. Huerta, F. Casadei, J. Donéa. *An arbitrary Lagrangian-Eulerian stress update procedure for coining simulations*, in 'Numerical Methods in Industrial Forming Processes', Eds. J.L. Chenot, R.D. Wood and O.C. Zienkiewicz, ISBN 9054100877, Balkema, Rotterdam, 1992.
- [77] F. Casadei *et al.* *Reference Methods for the Evaluation of Structural Stability*. Extract from the 1991 Annual Report of the Safety Technology Institute. EUR 14803 EN, pp. 86–106, 1992.

**1993:**

- [78] F. Casadei. *PLEXIS-3C solutions to a shock tube problem*, Technical Note N. I.93.33, March 1993.
- [79] F. Casadei. *A nonlinear 3-D shell finite element implementation for transient problems*, Technical Note N. I.93.41, April 1993.

- [80] F. Casadei. *Considerations about a data management and document production system for AMU*, Technical Note N. I.93.55, April 1993.
- [81] F. Casadei, J.P. Halleux (Eds.). *PLEXIS-3C User's Manual (Preliminary Version)*, Technical Note N. I.93.83, June 1993.
- [82] F. Casadei. *Implementation of Compressible Fluid Models in PLEXIS-3C. Part I - Models*, Technical Note N. I.93.86, June 1993.
- [83] F. Casadei. *Implementation of 3D Degenerated Shell Elements in PLEXIS-3C*, Technical Note N. I.93.88, July 1993.
- [84] F. Casadei. *Generalisation of a Two-Dimensional Rectilinear Beam/Conical Shell Element to Large Membrane Strains. Part I - Theory*, Technical Note N. I.93.89, July 1993.
- [85] F. Casadei. *A Miscellanea of New Models in PLEXIS-3C*, Technical Note N. I.93.90, July 1993.
- [86] F. Casadei. *Treatment of Essential Boundary Conditions in PLEXIS-3C*, Technical Note N. I.93.91, July 1993.
- [87] F. Casadei. *A Triangular Plate Element for the Nonlinear Dynamic Analysis of Thin 3D Structural Components*, Technical Note N. I.93.92, July 1993.
- [88] F. Casadei. *Implementation of A.L.E. Fluid-Structure Interaction Models in PLEXIS-3C*, Technical Note N. I.93.139, November 1993.
- [89] F. Casadei. *Simulation of a Gas Explosion in a Reactor Containment by PLEXIS-3C*, Technical Note N. I.93.148, November 1993.
- [90] F. Casadei. *Simulation of an Explosion in an Underground Cell by PLEXIS-3C*, Technical Note N. I.93.149, November 1993.
- [91] F. Casadei. *Post-Processing of PLEXIS-3C Calculated Results by CASTEM2000*, Technical Note N. I.93.150, November 1993.
- [92] F. Casadei *et al.* *Reference Methods for the Evaluation of Structural Stability*. Extract from the 1992 Annual Report of the Safety Technology Institute. EUR 15055 EN, pp. 126–146, 1993.

**1994:**

- [93] F. Casadei. *A Filter for the Treatment of MIF Files Produced by CASTEM2000*, Technical Note N. I.94.14, January 1994.
- [94] F. Casadei. *New Boundary Condition Models for Compressible Fluid Flows in PLEXIS-3C*, Technical Note N. I.94.75, May 1994.
- [95] F. Casadei, J.P. Halleux. *Automatic coupling of transient fluid/gas flows with deformable structures of arbitrary shape*, Third World Congress on Computational Mechanics, WCCM III, Chiba, Japan, August 1–5, 1994.
- [96] A. Huerta, F. Casadei. *New ALE Applications in Non-Linear Fast-Transient Solid Dynamics*, Engineering Computations, Vol. 11, pp. 317–345, Aug. 1994.

- [97] F. Casadei, J.P. Halleux. *On the treatment of fluid-structure interactions of the permanent type in PLEXIS-3C*, Second International Conference on Computational Structures Technology, Athens, Greece, 30 August–1 September 1994.
- [98] J. Donéa. *The ALE description in transient fluid-structure problems*, Proc. 2nd Int. Conf. on Computational Structures Technology, B.H.V. Topping and M. Papadrakakis, Editors, Athens, August 30– September 1, 1994.
- [99] J.J. López Cela. *Simulation of Compression Tests on the LDTF by PLEXIS-3C*. Technical Note N. I.94.137, September 1994.
- [100] F. Casadei *et al.* *Reference Methods for the Evaluation of Structural Stability*. Extract from the 1993 Annual Report of the Safety Technology Institute. EUR 15669 EN, pp. 131–156, 1994.

**1995:**

- [101] A. Soria, F. Casadei. *Modelling of Arbitrary Lagrangian-Eulerian Multicomponent Flow with Fluid-Structure Interaction in PLEXIS-3C*, Special Publication N. I.95.01, Jan. 1995.
- [102] J.J. López Cela, P. Pegon, F. Casadei. *Brittle Material Law with Von Mises Yield Surface and Softening Behaviour*, Special Publication N. I.95.09, Feb. 1995.
- [103] F. Casadei. *Simulation of a Steam Explosion in a Cavity by PLEXIS-3C*, Special Publication N. I.95.12, Feb. 1995.
- [104] A. Huerta, F. Casadei, J. Donéa. *ALE Stress Update in Transient Plasticity Problems*, presented at the 4th Int. Conf. on Computational Plasticity — Fundamentals and Applications, Barcelona, 3–6 April, 1995.
- [105] F. Casadei. *Simulation of a Steam Explosion in an Axisymmetric Cavity with Lateral Opening*, Special Publication N. I.95.33, May 1995.
- [106] F. Casadei. *Simulazione di Fenomeni di Pirolisi dovuti ad archi elettrici nei Trasformatori*, Special Publication N. I.95.36, May 1995.
- [107] F. Casadei, J.P. Halleux. *On the treatment of fluid-structure interactions of the permanent type in PLEXIS-3C*, (Reprint of the Paper presented at the Second International Conference on Computational Structures Technology, Athens, Greece, 30 August–1 September 1994), Technical Note N. I.95.54, May 1995.
- [108] F. Casadei, J.P. Halleux. *Fluid-Structure Interactions in Fast Transient Dynamics*, presented at the 3rd U.S. National Congress on Computational Mechanics (USNCCM III), Dallas, Texas, June 12–14, 1995.
- [109] F. Casadei, J.P. Halleux. *Modification of a Triangular Plate Element for the Nonlinear Dynamic Analysis of Thin 3D Structural Components*, Technical Note N. I.95.68, June 1995.
- [110] F. Casadei, J.P. Halleux (Eds.) *The PLEXIS-3C Developer's Handbook. Volumes I and II*, Technical Note N. I.95.74, June 1995.
- [111] F. Casadei. *1-D/Multi-D Coupled Calculations and Simplified Crash Simulations with PLEXIS-3C*, Technical Note N. I.95.81, June 1995.

- [112] F. Casadei, J.P. Halleux (Eds.) *The PLEXIS-3C Sample Input Files Handbook (Preliminary Version June 1995)*, Technical Note N. I.95.83, June 1995.
- [113] F. Casadei, J.P. Halleux. *Numerical Simulation of Fast Transient Dynamic Events including Fluid-Structure Interactions with PLEXIS-3C*, oral communication at the 13-th SMiRT Conference, Porto Alegre, Brasil, 13–18 Aug. 1995.
- [114] F. Casadei, J. Donéa, A. Huerta. *Arbitrary Lagrangian-Eulerian Finite Elements in Non-Linear Fast Transient Continuum Mechanics*, EUR 16327 EN, 1995.
- [115] F. Casadei, J.P. Halleux. *An Algorithm for Permanent Fluid-Structure Interaction in Explicit Transient Dynamics*, Computer Methods in Applied Mechanics and Engineering, Vol. 128, Nos. 3–4, pp. 231–289, December 1995.
- [116] F. Casadei, J. P. Halleux (Eds.). *PLEXIS-3C User's Manual (Preliminary Version March 1995)*. Special Publication N. I.95.22, March 1995.
- [117] F. Casadei. *Transient Dynamic Analysis of Fluid-Structure Systems with PLEXIS-3C*. Special Publication N. 95.53, December 1995.
- [118] F. Casadei *et al.* *Reference Methods for the Evaluation of Structural Reliability. Human Capital and Mobility Programme*. Extract from the 1994 Annual Report of the Safety Technology Institute. EUR 16251 EN, pp. 118–139 and 144–146, 1995. Also appeared as Special Publication N. 95.32, December 1995.

**1996:**

- [119] F. Casadei, J.J. López Cela. *A Multilayer Formulation for Shell Elements in PLEXIS-3C*, Technical Note N. I.96.14, February 1996.
- [120] J.J. López Cela, P. Pegon, F. Casadei. *Brittle Material Law with Drucker Prager Yield Surface and Softening Behaviour*, Technical Note N. I.96.34, February 1996.
- [121] F. Chillè, A. Sala, F. Casadei. *Containment of Blast Phenomena in Underground Electrical Power Plants*, 5th International Conference on 'Structures Under Shock and Impact', SUSI 96, CISM Udine, Italy, July 3–5, 1996.
- [122] F. Casadei, L. Papa, P. Pegon. *Use of AVS as a Visualization Tool for the Postprocessing of PLEXIS-3C and CASTEM 2000 Finite Element Results*, Technical Note N. I.96.206, November 1996.
- [123] V. Días Da Silva, F. Casadei. *An Implementation of the Cam-Clay Elasto-Plastic Model Using a Backward Interpolation and Viscoplastic Regularization*, Technical Note N. I.96.239, December 1996.
- [124] F. Casadei *et al.* *Reference Methods for the Evaluation of Structural Stability*. Extract from the 1995 Annual Report of the Safety Technology Institute. EUR 16363 EN, pp. 110–137, 1996. Also appeared as Special Publication N. S.P.I. 96.41, July 1996.

**1997:**

- [125] J.J. López Cela, P. Pegon, F. Casadei. *Fast Transient Analysis of Reinforced Concrete Structures with Drucker-Prager Model and Viscoplastic Regularization*, 5-th International Conference on Computational Plasticity, Complas-5, Barcelona, Spain, March 17–20, 1997.

- [126] F. Casadei. *Generalization of the Finite Element Model for Compressible Fluids in PLEXIS-3C to Multi-Phase Flows*, Technical Note N. I.97.33, March 1997.
- [127] F. Casadei. *LOOM: a Command Language for Object-Oriented Software Development in Computational Mechanics*, Technical Note N. I.97.75, May 1997.
- [128] V. Días Da Silva, F. Casadei. *Implementation in PLEXIS-3C of Orthotropic Material Behaviour for Use with Layered Shell Elements*, Technical Note N. I.97.82, May 1997.
- [129] F. Casadei, A. Sala. *Simulation of Electrical Arcs in SF6-Insulated Current Transformers by PLEXIS-3C*, Technical Note N. I.97.83, May 1997.
- [130] F. Casadei, A. Sala, J.P. Halleux. *Enhancements in the numerical Modeling of Fluids and Fluid-Structure Interactions in PLEXIS-3C*, Technical Note N. I.97.85, May 1997.
- [131] F. Casadei, A. Sala. *Development of Rupture Disk and Energy Injection Models for the Simulation of Electric Transformers in PLEXIS-3C*, Technical Note N. I.97.86, May 1997.
- [132] F. Casadei, J.P. Halleux. *Transmission of Finite Element Fluid Pressure Loads Across Warped Fluid-Structure Interfaces*, 4th U.S. National Congress on Computational Mechanics (USNCCM IV), "Minisymposium on Computer Simulation of Fluid-Structure Interaction Problems" organized by C. Farhat, C. Felippa and R. Ohayon, S. Francisco, California, August 6–8, 1997.
- [133] J.J. López Cela, F. Casadei, P. Pegon. *Fast Transient Analysis of Thin Shell Reinforced Concrete Structures with Drucker-Prager Model*, 14-th International Conference on Structural Mechanics in Reactor Technology, SMiRT-14, Lyon, France, August 17–22, 1997.
- [134] F. Casadei. *Recent Advances in the Modelling of Fast Transient Fluid-Structures Interactions of the Permanent Type*, Paper B-187 at the 14-th International Conference on Structural Mechanics in Reactor Technology, SMiRT-14, Lyon, France, August 17–22, 1997.
- [135] A. Sala, F. Casadei. *Modellazione Numerica 3D di una Miscela di Gas in PLEXIS-3C*, Technical Note N. I.97.97, June 1997.
- [136] A. Soria, F. Casadei. *Arbitrary Lagrangian-Eulerian Multicomponent Compressible Flow with Fluid-Structure Interaction*. International Journal for Numerical Methods in Fluids, Vol. 25, pp. 1263–1284, December 1997.
- [137] F. Casadei, J. Avotins. *A Language for Implementing Computational Mechanics Applications*, TOOLS Pacific '97 Conference (Technology of Object-Oriented Languages and Systems), Melbourne, Australia, November 24–27, 1997.
- [138] F. Casadei, E. Gabellini. *Implementation of a 3D Coupled Spectral-Element/Finite-Element Solver for Wave Propagation and Soil-Structure Interaction Simulations. Part I - Models*, EUR 17730 EN, 1997.
- [139] F. Casadei et al. *Earthquake Engineering and Dynamics of Structures. Industrial Processes and Clean Technologies*. Extract from the 1996 Annual Report of the Institute for Systems, Informatics and Safety. EUR 17321 EN, pp. 32–39 and 40–49, 1997.

**1998:**

- [140] F. Chillè, A. Sala, F. Casadei. *Containment of Blast Phenomena in Underground Electrical Power Plants*, Advances in Engineering Software, Vol. 29, No. 1, pp. 7–12, February 1998.
- [141] F. Casadei, E. Gabellini. *Implementation of a 3D Coupled Spectral-Element/Finite-Element Solver for Wave Propagation and Soil-Structure Interaction Simulations. Part II - Numerical Examples*, EUR 18051 EN, 1998.
- [142] F. Casadei, E. Gabellini, F. Maggio, A. Quarteroni. *Wave propagation in complex media by the mortar approximation*, Fourth International Conference on Mathematical and Numerical Aspects of Wave Propagation, in ‘Mathematical and Numerical Aspects of Wave Propagation’, J. DeSanto Ed., SIAM, pp. 314–318, Colorado School of Mines, Golden, Colorado, USA, June 1–5, 1998.
- [143] F. Casadei, E. Gabellini, F. Maggio, A. Quarteroni. *3D Seismic Modelling of Complex Media by the Mortar Method*, oral communication (presented by F. Maggio) at the 23rd General Assembly of the European Geophysical Society, Nice, France, 20–24 April 1998.
- [144] F. Casadei. *Implementation of the TPLLOT Data Management System Under MS-Windows NT*, Technical Note N. I.98.44, March 1998.
- [145] F. Casadei. *Implementation of the PLEXIS-3C System Under MS-Windows NT*, Technical Note N. I.98.78, May 1998.
- [146] F. Casadei, G. Fotia, E. Gabellini, F. Maggio, A. Quarteroni. *Un Solutore Ibrido per la Elastodinamica Tridimensionale*. Oral communication presented by G. Fotia (with an abstract in the proceedings) at the IV Congresso Nazionale della Società Italiana di Matematica Applicata e Industriale, Giardini Naxos, Messina, Italy, 1–5 June 1998.
- [147] S. Pedretti, F. Casadei. *Implementation of a Critical State Soil Model (CAMC)*, Technical Note N. I.98.81, May 1998.
- [148] F. Casadei, J.P. Halleux. *Compte-Rendu of an Introductory Course on Transient Dynamics and on the Use of the PLEXIS-3C Code for Industrial Applications*, Technical Note N. I.98.91, June 1998.
- [149] F. Casadei, G. Fotia, E. Gabellini, F. Maggio, A. Quarteroni. *An Advanced Hybrid Numerical Solver for 3D Elastodynamics*. Oral communication ICO98-87 presented by G. Fotia (with an abstract in the proceedings) at the International Conference on Spectral and High-Order Methods, ICOSAHOM 98, Herzliya (Tel Aviv), Israel, 22–26 June 1998.
- [150] F. Casadei. *An Introduction to the Data Structure of the PLEXIS-3C Software System*, Technical Note N. I.98.116, July 1998.
- [151] A. Rodríguez-Ferran, F. Casadei, A. Huerta. *ALE Stress Update for Transient and Quasi-static Processes*, International Journal for Numerical Methods in Engineering, 43, pp. 241–262, 1998.
- [152] F. Casadei. *Animation of PLEXIS-3C Results*, Technical Note N. I.98.227, November 1998.
- [153] F. Casadei. *Optimisation of ALE mesh rezoning algorithms in PLEXIS-3C*, Technical Note N. I.98.228, November 1998.

- [154] F. Casadei. *Review of Rigid-Surface Boundary Condition Models in PLEXIS-3C Results*, Technical Note N. I.98.237, November 1998.
- [155] E. Faccioli, R. Paolucci, S. Pedretti, M. Vanini, F. Chávez, F. Maggio, E. Gabellini, G. Verzeletti, J. Molina, F. Casadei, G. Giacinto, F. Roli, G. Gazetas: *3D Site Effects and Soil-Foundation Interaction in Earthquake and Vibration Risk Evaluation (TRISEE)*, Joint European Union / Japan Workshop on Seismic Risk, Chania, Crete (Greece), March 24–26, 1998. Proceedings appeared in “Earthquake Strong Ground Motion Evaluation — Application for Earthquake Disaster Mitigation”, Edited by C.P. Providakis and M. Yeroyanni, ISBN 960-9968-0-8, pp. 67–80, 1998.
- [156] J.P. Halleux, F. Casadei. *An Introductory Course on Transient Dynamics and on the Use of the PLEXIS-3C Code for Industrial Applications*. Course given at JRC Ispra, 13 January to 28 May 1998 (15 Sessions).
- [157] A. Sala. *A Comparison Between Two Fluid Solvers in a Fluid-Structure Interaction Problem Simulated By PLEXIS-3C*, ENEL internal report, 1998.
- [158] (\*) G. Giuseppetti, G. Mazzà, F. Chillè, A. Sala. *The Mitigation of the Effects of Blast Phenomena in Underground Hydroelectric Powerplants. Methodology and Case Histories*, III Conference on Modeling, Testing and Monitoring of Hydro Powerplants, Aix-en-Provence, 5–7 October, 1998.
- [159] F. Chillè. *Sviluppo nel codice PLEXIS-3C di moduli di interfaccia pre/post-processing per I-DEAS*, ENEL internal report, 1998.

**1999:**

- [160] F. Casadei. *Estensione del Modello di Pirolisi in PLEXIS-3C ai Fluidi Multifase e Multi-componente*, Technical Note N. I.99.27, February 1999.
- [161] F. Casadei. *Further Development and Validation of Multi-Phase Multi-Component Compressible Fluid Models in PLEXIS-3C*, Technical Note N. I.99.28, February 1999.
- [162] A. Sala, F. Casadei, A. Soria. *A 3D Finite Volume Numerical Model of Compressible Multicomponent Flow for Fluid-Structure Interaction Applications*, ‘IV Congreso de Métodos Numéricos en Ingeniería’, Sevilla, Spain, 7–10 June 1999.
- [163] F. Casadei, A. Sala. *Finite Element and Finite Volume Simulation of Industrial Fast Transient Fluid-Structure Interactions*, ‘European Conference on Computational Mechanics — Solids, Structures and Coupled Problems in Engineering — ECCM ’99’, Munich, Germany, August 31 – September 3, 1999.
- [164] F. Casadei, G. Fotià, F. Maggio, A. Quarteroni, J. Sabadell. *Seismic Wave Simulation for Complex Applications Using a Hybrid Higher Order Method*, (oral communication) presented by F. Maggio at the ‘Fourth International Conference on Theoretical and Computational Acoustics, ICTCA ’99’, Stazione Marittima, Trieste, Italy, May 10–14, 1999.
- [165] F. Casadei. *TPLLOT99: A New Version of the TPLLOT Data Management System under MS-Windows NT*, Technical Note N. I.99.86, May 1999.
- [166] F. Casadei, G. Fotià, E. Gabellini, F. Maggio, A. Quarteroni. *Numerical Methods for Seismic Wave Propagation and Seismic Response Analysis*, Chapter 2.1 in ‘3D Site Effects and Soil-Foundation Interaction in Earthquake and Vibration Risk Evaluation’, Final Report of the TRISEE Project, Edited by E. Faccioli, R. Paolucci and M. Vanini, EC DG



XII, 1994–98 Environment and Climate Programme — Climate and Natural Hazards Unit, July 1999.

- [167] B. Moras. *Constitutive Equations of Strain Rate Sensitive Materials for the Automotive Industry. PLEXIS-3C Implementation Report*, Technical Note N. I.99.54, March 1999.
- [168] F. Casadei, G. Fotia, E. Gabellini, F. Maggio, A. Quarteroni. *Hybrid Methods for Computational Tools for Large Scale Wave Propagation in Earthquake Engineering*, Oral communication presented by F. Maggio (with an abstract in the proceedings) at the SIMAI Workshop on Mathematical Modelling for Environment, Rome, EETI 99, 21 and 24 September 1999.

#### 2000:

- [169] F. Casadei, A. Sala, F. Chillè. *Computational Methods for Fluid-Structure Interaction in Fast Transient Industrial Applications*, V Congresso Nazionale della Società Italiana di Matematica Applicata e Industriale (SIMAI), Ischia, June 5-9, 2000.
- [170] F. Casadei, F. Chillè, M. Gèradin, A. Sala. *Interactions Fluide-Structure en Dynamique Rapide - Applications Industrielles*, International Colloquium in Mechanics of Solids, Fluids, Structures and Interactions, Nha Trang, Vietnam, 14–18 August 2000.
- [171] A. Combescure, H. Bung, F. Casadei, K. Kayvantash, G. Winkenmuller. *Modélisation en Dynamique Rapide - De la Théorie à la Pratique*, Course held at Institut pour la Promotion des Sciences de l'Ingénieur (IPSI), 26, Rue de la Pépinière, Paris, 17–19 October 2000 (proceedings available).
- [172] H. Bung, F. Casadei, J.P. Halleux, M. Lepareux. *Atelier Logiciel de EUROPLEXUS*, CEA's Report N. SEMT/DYN/RT/00-036/A, November 2000.
- [173] H. Bung, F. Casadei, J.P. Halleux, M. Lepareux. *Specifications for the Multi-Site Development of the EUROPLEXUS Computer Code*, Technical Note N. I.00.147, December 2000.
- [174] F. Casadei, J.P. Halleux, H. Bung, M. Lepareux. *Organisation of the EUROPLEXUS Mirror Site (Windows NT) at JRC Ispra*, Technical Note N. I.00.145, December 2000.
- [175] F. Casadei, J.P. Halleux, H. Bung, M. Lepareux. *Some Tentative Guidelines for the Development of the EUROPLEXUS Software System*, Technical Note N. I.00.146, December 2000.
- [176] F. Casadei, E. Gabellini, G. Fotia, F. Maggio, A. Quarteroni. *A Mortar Spectral/Finite Element Method for Complex 2D and 3D Elastodynamic Problems*, Submitted to Computer Methods in Applied Mechanics and Engineering, December 2000.
- [177] F. Casadei et al. *Safety of Building Structures and Means of Transport and Protection of Cultural Heritage*. Extract from the 1999 Annual Report of the Institute for Systems, Informatics and Safety. EUR 19512 EN, pp. 29–46, 2000.

#### 2001:

- [178] F. Casadei. *Algorithms for Fast Transient Fluid-Structure Interactions*, User Conference SAMTECH 2001, Paris, France, 30–31 January 2001.

- [179] F. Chillè, F. Polidoro. *Blast Load Effects on Civil Structures and Electrical Equipment: An Overview of ENEL Industrial Applications of Plexis-3C Code*, User Conference SAMTECH 2001, Paris, France, 30–31 January 2001.
- [180] F. Casadei, J.P. Halleux, A. Sala, F. Chillè. *Transient Fluid-Structure Interaction Algorithms for Large Industrial Applications*, Computer Methods in Applied Mechanics and Engineering, Vol. 190/24–25, pp. 3081–3110, March 2001.
- [181] F. Casadei. *A Proposal for the Reorganization of the Informatics Data Structure Related to Material Models in EUROPLEXUS* Technical Note N. I.01.68, July 2001.
- [182] F. Casadei. *A Module for Inverse Isoparametric Mappings in EUROPLEXUS*. Technical Note N. I.01.113, November 2001.
- [183] P. Koechlin, S. Moulin. *Modèle de comportement global des plaques en béton armé sous chargement dynamique de flexion : loi GLRC*. EDF Report HT-62/01/028/A, December 2001.
- [184] F. Casadei et al. *Safety of Building Structures and Protection of Cultural Heritage. Safety in Transient Structural Dynamics*. Extract from the 2000 Annual Report of the Institute for Systems, Informatics and Safety. EUR 19743 EN, pp. 78–87 and 88–93, 2001.

**2002:**

- [185] F. Casadei, J.P. Halleux, H. Bung, M. Lepareux. *Organisation of the EUROPLEXUS Mirror Site (MS Windows) at JRC Ispra - Second Edition*. Technical Note N. I.02.03, January 2002.
- [186] F. Casadei. *Permanent Fluid-Structure Interactions with Incompatible Interfaces*. Technical Note N. I.02.27, March 2002.
- [187] F. Casadei. *A Hierarchic Pinball Method for Contact-Impact in Fast Transient Dynamics*. VI Congresso Nazionale della Società Italiana di Matematica Applicata e Industriale (SIMAI 2002), Chia (Cagliari), Italy, 27–31 May 2002.
- [188] F. Casadei. *Time Step Management in the EUROPLEXUS Domain Decomposition Prototype*. Technical Note N. I.02.108, November 2002.
- [189] F. Casadei. *Domain Decomposition Prototype in EUROPLEXUS: An Evaluation of the Data Structure and Implementation*. Technical Note N. I.02.106, November 2002.
- [190] F. Casadei. *A Revision of the Liaisons Model in EUROPLEXUS for Domain Decomposition Calculations*. Technical Note N. I.02.107, November 2002.
- [191] F. Casadei. *Final Report Relative to the Collaboration Agreement Between the European Community and EDF/R&D - JRC Ref. N. 18921-2001-12 T1FS ISP FR*. Technical Note I.02.105, November 2002.

**2003:**

- [192] F. Casadei, S. Potapov. *Simulation of a Fast Decompression of the HDR Test Facility with a New Incompatible Fluid-Structure Interface Algorithm*. To be presented at the Second M.I.T. Conference on Computational Fluid and Solid Mechanics, M.I.T., Cambridge, MA 02319, U.S.A., June 17-20, 2003.

- [193] F. Casadei, J.P. Halleux. *EUROPLEXUS: a Numerical Tool for Fast Transient Dynamics with Fluid-Structure Interaction*. SAMTECH Users Conference 2003, Toulouse, France, Febraury 3-4, 2003.
- [194] F. Casadei, P. Pegon. *Improvements in Animations Made by EUROPLEXUS and CAST3M*. Technical Note N. I.03.09, February 2003.
- [195] F. Casadei. *Adaptation of OpenGL Libraries for Use Within the EUROPLEXUS System*. Technical Note N. I.03.26, February 2003.
- [196] F. Casadei. *Generation of Bitmapped OpenGL Graphical Images Within EUROPLEXUS*. Technical Note N. I.03.68, April 2003.
- [197] F. Casadei, J.P. Halleux, H. Bung, M. Lepareux. *Organisation of the EUROPLEXUS Mirror Site (MS Windows) at JRC Ispra - Third Edition*. Technical Note N. I.03.70, April 2003.
- [198] F. Casadei. *A Guide to Programming Link Conditions in EUROPLEXUS*. Technical Note N. I.03.86, June 2003.
- [199] F. Casadei, J.P. Halleux. *EUROPLEXUS: A Study of the Performance of Various Elements in Transient Dynamic Analysis of Shell Structures*. Technical Note N. I.03.92, June 2003.
- [200] F. Casadei. *Improvements in the Module for Inverse Isoparametric Mappings of EUROPLEXUS*. Technical Note N. I.03.132, November 2003.
- [201] F. Casadei *et al.* *Assessment and Management of Natural and Technological Risks*. Extract from the 2002 Annual Report of the Institute for the Protection and Security of the Citizen. EUR 20750 EN, pp. 41–51, 2003.
- [202] F. Casadei, H. Bung. *Generation of Bitmapped OpenGL Graphical Images Within EUROPLEXUS - 2<sup>nd</sup> Edition*. Technical Note N. I.03.155, November 2003.
- [203] F. Casadei. *A Revision of the Liaisons Model in EUROPLEXUS for Domain Decomposition Calculations - Second Edition*. Technical Note N. I.03.164, November 2003.
- [204] F. Casadei, K. Mehr. *The EUROPLEXUS Web Site at JRC Ispra*. Technical Note N. I.03.163, November 2003.
- [205] F. Casadei. *The LOOM System Graphical User Interface*. Technical Note I.03.166, November 2003.
- [206] F. Casadei. *The LOOM System Vector Graphics Model*. Technical Note I.03.167, November 2003.
- [207] F. Casadei. *The LOOM System Raster Graphics Model*. Technical Note I.03.165, November 2003.
- [208] Ph. Buchet, F. Casadei, P. Pegon. *A Module for Advanced 3D OpenGL Graphics Rendering in Computational Mechanics*. Technical Note N. I.03.169, November 2003.
- [209] Ph. Buchet, F. Casadei, P. Pegon. *New Graphic Capabilities in Cast3m and Visual Cast3m*. JRC Special Publication N. I.03.219, December 2003.
- [210] F. Casadei. *A General Impact-Contact Algorithm Based on Hierarchic Pinballs for the EUROPLEXUS Software System*. Technical Note N. I.03.176, December 2003.

**2004:**

- [211] F. Casadei. *Fluid-Structure Interaction in Fast Transient Dynamics, Part 1 - Presentation slides and trace of proposed exercises/examples*. Proceedings of the Course “Fluid-Structure Interaction in Fast Transient Dynamics” given by the author at the University of Barcelona (Universitat Politècnica de Catalunya) on May 4-7, 2004, JRC Special Publication N. I.04.102, May 2004.
- [212] F. Casadei. *Fluid-Structure Interaction in Fast Transient Dynamics, Part 2 - Auxiliary Course Notes*. Proceedings of the Course “Fluid-Structure Interaction in Fast Transient Dynamics” given by the author at the University of Barcelona (Universitat Politècnica de Catalunya) on May 4-7, 2004, JRC Special Publication N. I.04.103, May 2004.
- [213] M.F. Robbe, F. Casadei. *Comparison of various models for the simulation of a Core Disruptive Accident in the MARA 10 mock-up*. Nuclear Engineering and Design, Vol. 232, issue 3, pp. 301–326, August 2004.
- [214] F. Casadei, S. Potapov. *Permanent Fluid-Structure Interaction with Nonconforming Interfaces in Fast Transient Dynamics*. Computer Methods in Applied Mechanics and Engineering, Vol. 193, issues 39-41, pp. 4157-4194, October 2004.
- [215] F. Casadei, H. Bung. *A Library to Make Bitmapped OpenGL Graphical Images and AVI Animations within EUROPLEXUS*. Technical Note N. I.04.162, December 2004.
- [216] Ph. Buchet, F. Casadei, P. Pegon. *A Module for Advanced 3D OpenGL Graphics Rendering in Computational Mechanics – 2<sup>nd</sup> Edition*. Technical Note N. I.04.161, December 2004.

**2005:**

- [217] F. Casadei. *Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems, Part 1 - Presentation slides and trace of proposed exercises/examples*. Proceedings of the Course “Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems” given by the author at JRC Ispra on April 11-15, 2005, JRC Special Publication in press.
- [218] F. Casadei. Proceedings of a Course on *Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples*. Course given at JRC Ispra on April 11-15, 2005, JRC Special Publication S.P.I.05.93, May 2005.
- [219] F. Casadei. Proceedings of a Course on *Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples*. Course given at Universitat Politècnica de Catalunya, Barcelona, on May 17-20, 2005, JRC Special Publication N. I.05.139, June 2005.
- [220] F. Casadei. *Use of EUROPLEXUS for Building Vulnerability Studies. Progress Report 1*. Technical Note N. I.05.50, July 2005.
- [221] F. Casadei. *Validation of the EUROPLEXUS Pinball Impact-Contact Model on an Indentation Problem*. Technical Note N. I.05.51, July 2005.
- [222] F. Casadei, J.P. Halleux, H. Bung, M. Lepareux. *Organisation of the EUROPLEXUS Mirror Site (MS Windows) at JRC Ispra - Fourth Edition*. JRC Publication N. S.P.I.05.166.EN, October 2005.

- [223] F. Casadei. *Adaptation of OpenGL Libraries for Use Within the EUROPLEXUS System. Second Edition.* JRC Publication N. S.P.I.05.170.EN, October 2005.
- [224] F. Casadei, H. Bung, M. Lepareux. *Implementation of Multi-Language Output in EUROPLEXUS.* Technical Note N. FC\_01.2005, November 2005 (also registered as PUBSY No. 006092, January 2007).
- [225] F. Casadei. *Recent Improvements in EUROPLEXUS Curve Plotting Capabilities.* Technical Note N. FC\_02.2005, December 2005 (also registered as PUBSY No. 006093, January 2007).

**2006:**

- [226] G. Juárez, G. Ayala, F. Casadei. *Finite Element Variational Formulation for Bending Elements with and without Discontinuities.* III European Conference on Computational Mechanics, Solids, Structures and Coupled Problems in Engineering, C.A. Mota Soares *et al.* (eds.), Lisbon, Portugal, 5-8 June 2006.
- [227] F. Casadei. *Proceedings of a Course on Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples.* Course given at Universitat Politècnica de Catalunya, Barcelona, on May 15-18, 2006 and at Royal Military Academy, Brussels, on June 26-30, 2006, JRC Special Publication N. PB/2006/IPSC/1956, June 2006.
- [228] M.F. Robbe, M. Lepareux, F. Casadei. *Simulation of the MARA 10 test representing a core disruptive accident.* Computer Assisted Mechanics and Engineering Sciences, Vol. 13, pp. 269-304, 2006.
- [229] J.P. Halleux, F. Casadei. *Spatial Time Step Partitioning in EUROPLEXUS.* EUR Report 22464 EN, 2006.
- [230] P. Iorio. *Esperienza Gioconda del Centro Comune di Ricerca di Ispra: analisi dei dispositivi sperimentali finalizzata al loro recupero e smantellamento sicuro.* Tesi di Laurea, Università degli Studi di Roma "La Sapienza", Facoltà di Ingegneria, Dipartimento di Energia Nucleare e Conversioni di Energia, 21 Dicembre 2006.

**2007:**

- [231] J.P. Halleux, F. Casadei. *An Algorithm for Spatial Partitioning in Explicit Time Integration — Part I - Basics.* Submitted for publication in Computer Methods in Applied Mechanics and Engineering, January 2007.
- [232] F. Casadei, J.P. Halleux. *An Algorithm for Spatial Partitioning in Explicit Time Integration — Part II - Treatment of Boundary Conditions and Extension to ALE.* Submitted for publication in Computer Methods in Applied Mechanics and Engineering, January 2007.
- [233] F. Casadei. *Domain Decomposition Technique in EUROPLEXUS — Part I : Summary of Basic Formulation.* Technical Note, PUBSY No. 006090, January 2007.
- [234] F. Casadei, J.P. Halleux. *Explicit Time Step Spatial Partitioning in Nonlinear Transient Dynamics.* ECCOMAS Thematic Conference on Computational Methods in Structural Dynamics and Earthquake Engineering, COMPDYN 2007, Rethymno, Crete, June 13-16, 2007.

- [235] F. Casadei, A. Anthoine. *Use of EUROPLEXUS for Building Vulnerability Studies. Progress Report 2.* Technical Note, PUBSY No. 006972, March 2007.
- [236] G. Ayala, G. Juárez, F. Casadei. *Model for soil amplifications studies involving soil failure.* ANIDIS 2007 - XII Convegno - L'Ingegneria Sismica in Italia, Pisa, Italy, 10-14 June 2007.
- [237] F. Casadei. *Proceedings of a Course on Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples.* Course given at Universitat Politècnica de Catalunya, Barcelona, on May 14-17, 2007, PUBSY No. 007357, June 2007.
- [238] F. Casadei. *Implementation of Fast Search Algorithms in EUROPLEXUS.* Technical Note, PUBSY No. JRC38086, October 2007.
- [239] F. Casadei. *Use of EUROPLEXUS for Building Vulnerability Studies. Progress Report 3.* Technical Note, PUBSY No. JRC40393, October 2007.
- [240] F. Casadei, H. Bung. *Organisation of the EUROPLEXUS Mirror Site (MS Windows) at JRC Ispra - Fifth Edition.* Technical Note, PUBSY No. JRC40467, October 2007.
- [241] M. Larcher. *Simulation of the Effects of an Air Blast Wave.* Technical Note, PUBSY No. JRC41337, November 2007.
- [242] G. Giannopoulos. *Implementation of Failure Models for Shell Elements in EUROPLEXUS.* Technical Note, PUBSY No. JRC42336, 2007.
- [243] G. Giannopoulos. *Aspects of Geometrical Representation of Large Structures and Spaces for Blast Analysis.* Technical Note, PUBSY No. JRC42363, 2007.

**2008:**

- [244] F. Casadei. *Fast Transient Fluid-Structure Interaction with Failure and Fragmentation.* Extended abstract submitted for presentation at the 8th World Congress on Computational Mechanics (WCCM8), Venice, Italy, June 30 – July 5, 2008.
- [245] M. Larcher. *Simulation of Air Blast Waves and the Loading of Glass Sheets.* Extended abstract submitted for presentation at the International Conference on Impact Loading of Lightweight Structures, Trondheim, Norway, June 17-19, 2008.
- [246] F. Casadei, H. Bung. *A Library to Make Bitmapped OpenGL Graphical Images and AVI Animations within EUROPLEXUS — Fourth Edition.* Technical Note, PUBSY No. JRC42523, January 2008.
- [247] F. Casadei, J.P. Halleux. *Spatial Time Step Partitioning in Explicit Fast Transient Dynamics.* EUR Technical Report 23062 EN, PUBSY No. JRC42581, 2008.
- [248] M. Larcher. *Implementation of New Materials in EUROPLEXUS.* Technical Note, PUBSY No. JRC43399, March 2008.
- [249] M. Larcher. *Postprocessing of EUROPLEXUS results by ParaView.* Technical Note, PUBSY No. JRC44523, April 2008.
- [250] F. Casadei. *Use of EUROPLEXUS for Building Vulnerability Studies. Progress Report 4.* Technical Note, PUBSY No. JRC44636, April 2008.

- [251] C. Giry, G. Solomos, F. Casadei. *Numerical analysis of the JRC Large Hopkinson Bar*. Technical Note, PUBSY No. JRC46866, September 2008.
- [252] F. Casadei. *Proceedings of a Course on Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples*. Course given at Universitat Politècnica de Catalunya, Barcelona, on May 19-23, 2008, PUBSY No. JRC45967, June 2008.
- [253] M. Larcher. *Simulation of Air Blast Waves in Urban Environment*. Technical Note, PUBSY No. JRC43400, October 2008.
- [254] M. Larcher. *Pressure Time Functions for the Description of Air Blast Waves*. Technical Note, PUBSY No. JRC46829, October 2008.
- [255] V. Faucher, F. Casadei. *Time-dependent kinematic constraints using Lagrange Multipliers for explicit structural dynamics on distributed memory clusters*. Submitted for publication in the International Journal for Numerical Methods in Engineering, October 2008.
- [256] F. Casadei. *Use of EUROPLEXUS for Building Vulnerability Studies. Progress Report 5*. Technical Note, PUBSY No. JRC48440, November 2008.
- [257] F. Casadei, H. Bung. *Organisation of the EUROPLEXUS Mirror Site (MS Windows) at JRC Ispra - Sixth Edition*. Technical Note, PUBSY No. JRC48621, November 2008.
- [258] E. Paffumi, N. Taylor. *Structural Response of a Large Pressure Vessel to Dynamic Loading. Fast Transient 3-D Dynamic Analysis of the Effect of a Blast Wave*. JRC EUR Report EUR 23451 EN, 2008.
- [259] M. Larcher. *Simulation of Several Glass Types Loaded by Air Blast Waves*. Technical Note, PUBSY No. JRC48240, December 2008.

**2009:**

- [260] F. Casadei. *Specifications for the implementation in Samcef Field of new features related to the study of explosion effects*. JRC internal document, January 2009.
- [261] G. Giannopoulos, F. Casadei. *Risk Assessment of Explosion Events Using EUROPLEXUS*. Technical Note, PUBSY No. JRC50385, March 2009.
- [262] G. Giannopoulos, G. Solomos. *Blast loading analysis of a metro train carriage*. Technical Note, PUBSY No. JRC49904, March 2009.
- [263] M. Larcher. *Risk analysis about explosions in trains*. Abstract of a paper to be presented at the LWAG 2009 Conference, Light Weight Armour for Defence and Security, University of Aveiro, Portugal, 18-19 May 2009.
- [264] M. Larcher. *Simulation of laminated glass loaded by air blast waves*. Paper to be presented at the DYMAT 2009 Conference, 9-th International Conference on the Mechanical and Physical Behaviour of Materials under Dynamic Loading, Brussels, Belgium, 7-11 September 2009.
- [265] M. Larcher. *Simulations of a Metro Carriage Exposed to an Internal Detonation*. Technical Note, PUBSY No. JRC50327, March 2009.

- [266] F. Casadei, G. Giannopoulos, M. Larcher, G. Solomos. *Transport Infrastructure Vulnerability Studies with EUROPLEXUS. Concluding Report*. Technical Note, PUBSY No. JRC51065, in publication.
- [267] M. Larcher. *Simulations of a Metro Carriage exposed to an Internal Detonation*. Paper to be presented at the 11th Samtech User Conference, Musée Dapper, Paris, March 31 – April 1, 2009.
- [268] P. Sotiropoulos, D. Grosset, G. Giannopoulos, F. Casadei. *AUV docking system for existing underwater control panel*. Paper to be presented at the OCEANS '09 IEEE Conference, Bremen, Germany, 11–14 May 2009.
- [269] M. Larcher. *Inputs Command Reading in EUROPLEXUS*. Technical Note, PUBSY No. JRC51609, May 2009.
- [270] G. Giannopoulos, M. Larcher, F. Casadei, G. Solomos. *Risk assessment of the fatality due to explosion in land mass transport infrastructure by fast transient dynamic analysis*. Submitted for publication in the Journal of Hazardous Materials, May 2009.
- [271] F. Casadei, J.P. Halleux. *Binary spatial partitioning of the central-difference time integration scheme for explicit fast transient dynamics*. International Journal for Numerical Methods in Engineering, 78, 1436–1473, 2009.
- [272] M. Larcher, F. Casadei. *Explosions in complex geometries — a comparison of several approaches*. Submitted for publication in Engineering Structures, June 2009. Published in the International Journal of Protective Structures, Vol. 1, N. 2, pp. 169–195, PUBSY No. JRC46723, 2010.
- [273] F. Casadei. *Proceedings of a Course on Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples*. Course given at Universitat Politècnica de Catalunya, Barcelona, on May 11–15, 2009, PUBSY No. JRC52299, June 2009.
- [274] F. Casadei, N. Leconte. *Fluid-Structure Interaction with Cell-Centred Finite Volumes in EUROPLEXUS*. Technical Note, PUBSY No. JRC53799, September 2009.
- [275] G. Solomos, F. Casadei, M. Larcher, G. Giannopoulos, M. G radin. *RAILPROTECT Project Final Executive Summary Report*. JRC Scientific and Technical Report, PUBSY No. JRC56334, 2009.
- [276] N. Leconte. *Numerical modelling of the compression Hopkinson bar of ELSA*. Technical Note, PUBSY No. JRC53820, September 2009.
- [277] M. Larcher. *New developments of the risk function in EUROPLEXUS and their use for long train carriages*. Technical Note, PUBSY No. JRC53181, October 2009.
- [278] M. Larcher. *Determination of the Risk Inside of the Land Mass Transport Systems*. Oral presentation in: Euro-Atlantic Stakeholder Conference (EASC09), 30 September 2009, Stockholm (Sweden). Organised by the Swedish Civil Contingencies Agency (MSB) in cooperation with the US Department of Homeland Security Science and Technology Directorate. PUBSY No. JRC52858, 2009.



**2010:**

- [279] F. Casadei, M. Larcher. *On Some Computational Methods for the Simulation of Structures Subjected to Blast Loading and Fragmentation*. Paper submitted to ECCM 2010, IV European Conference on Computational Mechanics, Paris, France, May 16-21, 2010.
- [280] M. Larcher, F. Casadei. *Experimental and numerical investigations about laminated glass loaded by air blast waves*. Paper submitted to ECCM 2010, IV European Conference on Computational Mechanics, Paris, France, May 16-21, 2010.
- [281] N. Leconte, F. Casadei. *Fluid/Structure Interaction with Node-Centered Finite Volumes in EUROPLEXUS*. Technical Note, PUBSY No. JRC56577, January 2010.
- [282] M. Larcher. *EUROPLEXUS - MPI Version at the JRC*. Technical Note, PUBSY No. JRC56677, February 2010.
- [283] M. Larcher, F. Casadei, G. Solomos. *Risk analysis of explosions in trains by fluid-structure calculations*. Journal of Transportation Security, Vol. 3, pp. 57-71, Springer, 2010.
- [284] F. Casadei, N. Leconte. *Current Status of Fluid-Structure Interaction with Finite Elements and Finite Volumes in EUROPLEXUS*. Technical Note, PUBSY No. JRC57800, 2010.
- [285] M. Larcher, G. Manara. *Influence of air damping on structures especially glass*. Technical Note, PUBSY No. JRC57330, 2010.
- [286] M. Larcher, F. Casadei. *Explosions in complex geometries — a comparison of several approaches*. JRC EUR Report 24288 EN, 2010.
- [287] F. Casadei, N. Leconte. *Coupling finite elements and finite volumes by Lagrange multipliers for explicit dynamic fluid-structure interaction*. International Journal for Numerical Methods in Engineering, Vol. 86, pp. 1–17, 2011.
- [288] F. Casadei, N. Leconte. *Some Notes on the Recent Revision of Node-Centered Finite Volume Fluid Models in EUROPLEXUS*. Technical Note, PUBSY No. JRC58352, May 2010.
- [289] N. Leconte, F. Casadei. *Modifications of the NCFV model of EUROPLEXUS for Fluid/Structure Interaction*. Technical Note, PUBSY No. JRC58667, July 2010.
- [290] G. Solomos, F. Casadei, G. Giannopoulos, M. Larcher. *Assessment of explosion effects in railway stations*. International conference “Urban Habitat Constructions Under Catastrophic Events”, Naples, Italy, 16-18 September 2010.
- [291] T. Dyngeland. *Simulation of a Standard ISO Steel Container Subjected to Blast Loading*. JRC Scientific and Technical Report, PUBSY No. JRC56386, 2010.
- [292] F. Casadei. *Proceedings of a Course on Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples*. Course given at Universitat Politècnica de Catalunya, Barcelona, on May 31 - June 04, 2010, PUBSY No. JRC59017, June 2010.
- [293] N. Leconte, F. Casadei. *Fluid/Structure Interaction with Node-Centered Finite Volumes in EUROPLEXUS. Second Release*. Technical Note, PUBSY No. JRC59306, July 2010.
- [294] F. Casadei, N. Leconte. *Use of FLSR Fluid-Structure Interaction with Node-Centered Finite Volumes in EUROPLEXUS*. Technical Note, PUBSY No. JRC59686, August 2010.

- [295] N. Leconte, A. Saltelli, F. Casadei. *Some comments on the coupling of sensitivity analysis tools to EUROPLEXUS*. Technical Note, PUBSY No. JRC59841, August 2010.
- [296] M. Larcher, F. Casadei, G. Solomos. *Influence of venting areas on the air blast pressure inside tubular structures like railway carriages*. Journal of Hazardous Materials, 183, pp. 839–846, 2010.
- [297] F. Casadei, P. Díez, F. Verdugo. *A Data Structure for Adaptivity in EUROPLEXUS*. Technical Note, PUBSY No. JRC60795, in publication (2010).
- [298] M. Larcher, G. Solomos, F. Casadei, N. Gebbeken. *Simulation terroristischer Anschläge in Massenverkehrsmitteln* (in German). Bautechnik, Vol. 88(4), pp. 225–232, DOI: 10.1002/bate.201110023, 2011.
- [299] M. Larcher, F. Casadei, N. Gebbeken, G. Solomos. *Influence of venting areas in tubular structures like train carriages*. The First International Conference of Protective Structures, Manchester, September 29–October 1, 2010.
- [300] M. Larcher, N. Gebbeken, M. Teich, G. Solomos. *Simulation of laminated glass loaded by air blast waves*. International Symposium on the Application of Architectural Glass, Munich, Germany, October 4–5, 2010.
- [301] F. Casadei, P. Díez, F. Verdugo. *Adaptivity in FE Models for Fluids in EUROPLEXUS*. Technical Note, PUBSY No. JRC61622, November 2010.

**2011:**

- [302] M. Larcher, M. Teich, N. Gebbeken, G. Solomos, F. Casadei, G.A. Falcon, S.L. Sarmiento. *Simulation of laminated glass loaded by air blast waves*. PUBSY No. JRC63182. PROTECT 2011 Conference, Lugano, Switzerland, August 30 – September 1, 2011. Also published in Applied Mechanics and Materials, Vol. 82, pp. 69–74, 2011.
- [303] F. Casadei, M. Larcher, N. Leconte. *Strong and weak forms of a fully non-conforming FSI algorithm in fast transient dynamics for blast loading of structures*. PUBSY No. JRC60824. COMPDYN 2011, III ECCOMAS Thematic Conference on Computational Methods in Structural Dynamics and Earthquake Engineering, Corfu, Greece, May 25–28, 2011.
- [304] F. Casadei, P. Díez, F. Verdugo. *Adaptive 3D Refinement and Un-refinement of 8-node Solid and Fluid Hexahedra in EUROPLEXUS*. Technical Note, PUBSY No. JRC63833, March 2011.
- [305] F. Casadei. *Some perspectives for the simulation of impact against offshore pipelines by EUROPLEXUS*. Technical Note, PUBSY No. JRC63847, March 2011.
- [306] F. Casadei, T. Dyngeland, P. Pegon. *Simulation of Blast Loading on a 20-ft Steel ISO Container with FSI by EUROPLEXUS*. Technical Note, PUBSY No. JRC63868, March 2011.
- [307] M. Larcher. *Risk analysis about explosions in train carriages*. International Journal of Materials Engineering Innovation, Vol. 2, No. 2, pp. 110–123, 2011.
- [308] F. Casadei, P. Díez, F. Verdugo. *Implementation of a 2D Adaptivity Indicator for Fast Transient Dynamics in EUROPLEXUS*. Technical Note, PUBSY No. JRC64506, April 2011.

- [309] F. Casadei, P. Díez, F. Verdugo. *An algorithm for mesh refinement and un-refinement in fast transient dynamics*. Paper submitted for publication in the International Journal for Numerical Methods in Engineering, April 2011. Published in the International Journal of Computational Methods as DOI 10.1142/S0219876213500187, Vol. 10, No. 4, pp. 1350018-1 – 1350018-31, 2013.
- [310] F. Verdugo, P. Díez, F. Casadei. *Natural quantities of interest in linear elastodynamics for goal oriented error estimation and adaptivity*. Proceedings of the V International Conference on Adaptive Modeling and Simulation (ADMOS 2011), D. Aubry and P. Díez (Eds), Paris, France, 6–8 June 2011.
- [311] F. Casadei. Proceedings of a Course on *Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples*. Course given at Universitat Politècnica de Catalunya, Barcelona, on May 9-13, 2011, PUBSY No. JRC65139, 2011.
- [312] V. Faucher, F. Casadei, S. Potapov, F. Crouzet, R. Ortiz. *Planification du développement du programme EUROPLEXUS. Expression des besoins par les représentants techniques du Consortium*. Rapport Technique DEN/DANS/DM2S/SEMT/DYN/RT/11-009/A, PUBSY No. JRC65151, May 2011.
- [313] N. Leconte, F. Casadei, B. Langrand. *Modelling of riveted structures subjected to blast loading*. PUBSY No. JRC65425. Third International Conference on Impact Loading of Lightweight Structures (ICILS2011), Valenciennes, France, June 28 – July 1, 2011.
- [314] M. Larcher, F. Casadei, G. Solomos, N. Gebbeken. *Can Venting Areas Mitigate the Risk Due to Air Blast Inside Railway Carriages?* International Journal of Protective Structures, Vol. 2, N. 2, pp. 221–230, PUBSY No. JRC62703, June 2011.
- [315] M. Larcher, F. Casadei, G. Giannopoulos, G. Solomos, J.-L. Planchet, A. Rochefrette. *Determination of the risk due to explosions in railway systems*. Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit, Special Issue “Engineering a Secure Railway”, Vol. 225, N. 4, pp. 373-382, DOI: 10.1243/09544097JRRT385, PUBSY No. JRC56669, July 2011.
- [316] F. Verdugo, P. Díez, F. Casadei. *General form of the natural quantities of interest for goal oriented error assessment and adaptivity in linear elastodynamics*. Submitted for publication in the International Journal for Numerical Methods in Engineering, DOI: 10.1002/nme, PUBSY No. JRC65788, July 2011.
- [317] F. Casadei, N. Leconte. *FLSW : A Weak, Embedded-Type Fluid-Structure Interaction Model with CCFV in EUROPLEXUS*. Technical Note, PUBSY No. JRC65826, July 2011.
- [318] F. Casadei. *Implementation of Parabolic Lagrange 2D and 3D Continuum Elements in EUROPLEXUS*. Technical Note, PUBSY No. JRC66266, August 2011.
- [319] F. Casadei, P. Díez, F. Verdugo. *Further Development of 2D Adaptivity Error Indicators in EUROPLEXUS*. Technical Note, PUBSY No. JRC66337, September 2011.
- [320] G. Solomos, F. Casadei, G. Giannopoulos, M. Larcher. *Experimental and Numerical Simulation Activities for the Assessment of Explosion Effects in a Train Station*. Internal Security (semi-annual Journal of the Police Academy in Szczytno), January–June 2011 Volume, pp. 49–62, August 2011.

- [321] M. Larcher, G. Solomos, F. Casadei, N. Gebbeken. *Experimental and numerical investigations of laminated glass subjected to blast loading*. International Journal of Impact Engineering, Vol. 39, pp. 42–50, 2012.
- [322] F. Casadei. *Numerical simulation of blast effects on structures at JRC-ELSA*. Proceedings of a Workshop on Modeling and Behaviour of Light-Weight Protective Structures, SIMLab, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2–3 December 2010. Edited by T. Krauthammer, M. Langseth and T. Børvik. Published in 2011.
- [323] N. Leconte, F. Casadei, B. Langrand. *Numerical methods for the modelling of blast-loaded full-scale riveted structures*. Submitted for publication in the International Journal of Impact Engineering, October 2011.
- [324] F. Casadei, G. Valsamos, P. Díez, F. Verdugo. *Implementation of Adaptivity in 2D Cell Centred Finite Volumes in EUROPLEXUS*. Technical Note, PUBSY No. JRC67859, December 2011.
- [325] F. Casadei, P. Díez, F. Verdugo. *Implementation of Adaptivity in 3D Cell Centred Finite Volumes in EUROPLEXUS*. Technical Note, PUBSY No. JRC68168, December 2011.
- [326] F. Casadei, P. Díez, F. Verdugo. *Testing Adaptivity in 2D Cell Centred Finite Volumes with the CDEM Combustion Model in EUROPLEXUS*. Technical Note, PUBSY No. JRC68333, December 2011.

**2012:**

- [327] G. Valsamos, F. Casadei, G. Solomos. *Numerical Simulation of Pochhammer-Chree Wave Dispersion in Cylindrical Rods by EUROPLEXUS*. Technical Note, PUBSY No. JRC68457, January 2012.
- [328] F. Casadei. *A Set of Functions to Manipulate Bitmaps and Animations for Use with EUROPLEXUS*. Technical Note, PUBSY No. JRC70484, April 2012.
- [329] F. Casadei. *Proceedings of a Course on Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples*. Course given at Universitat Politècnica de Catalunya, Barcelona, on May 7-11, 2012, PUBSY No. JRC71459, 2012.
- [330] F. Casadei, K. Rakvåg, M. Kristoffersen. *Simulation of Gas Discharge Through a Deformable Plate with Orifices by EUROPLEXUS*. Technical Note, PUBSY No. JRC71900, June 2012.
- [331] G. Valsamos, F. Casadei, G. Solomos. *Numerical Simulation of Wave Dispersion Curves in Cylindrical Rods by EUROPLEXUS*. EUROMECH Colloquium 540 — Advanced Modelling of Wave Propagation in Solids, Prague, Czech Republic, October 1–3, 2012.
- [332] G. Valsamos, F. Casadei, G. Solomos. *Explosion Scenarios Simulations for a Real Metro Station Structure by Europlexus*. Technical Note, PUBSY No. JRC78055, 2012.

**2013:**

- [333] F. Casadei, G. Solomos, G. Valsamos, H. Bung, A. Beccantini. *Organisation of the EUROPLEXUS Mirror Site (MS Windows) at JRC Ispra - Seventh Edition*. Technical Note, EUR 25821, PUBSY No. JRC79295, 2013.
- [334] F. Casadei. *Proceedings of a Course on Numerical Simulation of Fast Transient Dynamic Phenomena in Fluid-Structure Systems. Presentation slides and trace of proposed exercises/examples*. Edited by G. Solomos and G. Valsamos. Course given at Universitat Politècnica de Catalunya, Barcelona, on April 15-19, 2013, PUBSY No. JRC81589, 2013.
- [335] G. Valsamos, F. Casadei, G. Solomos. *A numerical study of wave dispersion curves in cylindrical rods with circular cross-section*. Applied and Computational Mechanics, Volume 7, pp. 99–114, 2013.
- [336] M. Kristoffersen, F. Casadei, T. Børvik, M. Langseth, G. Solomos, O.S. Hopperstad. *Numerical simulation of submerged and pressurised X65 steel pipes – COMPLAS XII*. XII International Conference on Computational Plasticity. Fundamentals and Applications. COMPLAS XII. E. Oñate, D.R.J. Owen, D. Peric and B. Suárez (Eds). Barcelona, Spain, September 3–5, 2013.
- [337] M. Larcher, M. Peroni, G. Solomos, N. Gebbeken, P. Bieber, J. Wandelt, N. Tran *Dynamic Increase Factor of Masonry Materials: Experimental Investigations* 15th International Symposium on Interaction of the Effects of Munitions with Structures (ISIEMS) Potsdam, Germany, September 17–20, 2013.
- [338] M. Kristoffersen. *Europlexus applied to pipe impact including fluid-structure interaction*. Technical Note, PUBSY No. JRC86875, 2013.

**2014:**

- [339] M. Kristoffersen, F. Casadei, T. Børvik, M. Langseth, O.S. Hopperstad. *Impact against empty and water-filled X65 steel pipes – Experiments and simulations*. International Journal of Impact Engineering, 71, pp. 73-78, 2014.
- [340] G. Valsamos, F. Casadei, G. Solomos, M. Larcher. *Numerical Simulation of Explosion Scenarios in a Real Metro Station*. Paper 157, Proceedings of the Second International Conference on Railway Technology: Research, Development and Maintenance (Railways 2014), Ajaccio, Corsica, France, 8-11 April 2014. J. Pombo (Editor), Civil-Comp Press, Stirlingshire, Scotland.
- [341] F. Casadei, G. Valsamos, M. Larcher, A. Beccantini. *Combination of Mesh Adaptivity with Fluid-Structure Interaction in Europlexus*. Technical Note, PUBSY No. JRC89728, EUR Report 26617 EN, 2014.
- [342] F. Casadei, M. Larcher, G. Valsamos. *Adaptivity with Simplex Elements in Europlexus*. Technical Note, PUBSY No. JRC89894, EUR Report 26630 EN, 2014.
- [343] F. Casadei, M. Larcher, G. Valsamos. *Adaptivity in CEA's Fluid Elements in Europlexus*. Technical Note, PUBSY No. JRC89953, EUR Report 26632 EN, 2014.
- [344] F. Casadei, B. Langrand, M. Larcher, G. Valsamos. *Pinball-based Contact-Impact Model with Parabolic Elements in EUROPLEXUS*. Technical Note, PUBSY No. JRC89913, EUR Report 26629 EN, 2014.

- [345] M. Larcher, F. Casadei, G. Valsamos. *Adaptivity in Shell/Beam/Bar Elements in Europlexus*. Technical Note, PUBSY No. JRC90456, EUR Report 26697 EN, 2014.
- [346] M. Larcher, G. Valsamos, G. Solomos. *Numerical material modelling for the blast actuator*. Technical Note, PUBSY No. JRC86348, EUR Report 26407 EN, 2014.
- [347] G. Valsamos, M. Larcher, G. Solomos, A. Anthoine. *Numerical simulations in support of the blast actuator development*. Technical Note, PUBSY No. JRC86464, EUR Report 26430 EN, 2014.
- [348] F. Casadei, M. Larcher, G. Valsamos, V. Faucher. *Implementation of Assembled Surface Normals and of a Penalty Contact Formulation in the Pinball Model of Europlexus*. Technical Note, PUBSY No. JRC90939, EUR Report 26714 EN, 2014.
- [349] N. Leconte. *Modelling of Eulerian Fluid Strong Shocks using EUROPLEXUS*. Technical Note, University of Valenciennes, LAMIH UMR CNRS 8201, 2014.
- [350] M. Larcher, F. Casadei, G. Solomos. *Simulation of blast waves by using mapping technology in EUROPLEXUS*. Technical Note, PUBSY No. JRC91102, EUR Report 26735 EN, 2014.
- [351] M. Kristoffersen. *Impact against X65 offshore pipelines*. PhD Thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2014.
- [352] Van Doormaal A, Haberacker C, Huesken G, Larcher M, Saarenheimo A, Solomos G, Stolz A, Thamie L, Bedon C. *Numerical simulations for classification of blast loaded laminated glass: possibilities, limitations and recommendations - ERNCIP Thematic Group: Resistance of structures to explosion effects*. Technical Note, PUBSY No. JRC94928, 2014.

**2015:**

- [353] V. Faucher, F. Casadei. *Adaptive mesh refinement with combined criteria for fast transient fluid-structure dynamics*. Abstract submitted to the 5th International Conference on Computational Methods in Structural Dynamics and Earthquake Engineering, COMPDYN 2015, Crete, Greece, 25–27 May 2015.
- [354] F. Casadei, M. Larcher, G. Valsamos. *Implementation of Linear Lagrange 2D and 3D Continuum Elements for Solids in EUROPLEXUS*. Technical Note, PUBSY No. JRC94373, EUR Report 27070 EN, 2015.
- [355] V. Karlos, M. Larcher, G. Solomos. *Analysis of the blast wave decay coefficient in the Friedlander equation using the Kingery-Bulmash data*. Technical Note, PUBSY No. JRC94784, 2015.
- [356] G. Valsamos, F. Casadei, M. Larcher, G. Solomos. *Implementation of Flying Debris Fatal Risk Calculation in EUROPLEXUS*. Technical Note, PUBSY No. JRC94805, EUR Report 27135 EN, 2015.
- [357] F. Casadei, M. Larcher, G. Valsamos. *Visualization of Fluid-Structure Interaction Pressure acting on Structures with EUROPLEXUS*. Technical Note, PUBSY No. JRC95654, EUR Report 27207 EN, 2015.
- [358] F. Casadei, V. Faucher, M. Larcher, G. Valsamos. *Combined Fluid and Structure Mesh Adaptivity with Fluid-Structure Interaction in EUROPLEXUS*. Technical Note, PUBSY No. JRC96043, EUR Report 27276 EN, 2015.

- [359] F. Casadei, G. Valsamos, M. Larcher. *On the interpretation and post-processing of mesh-adaptive numerical results in EUROPLEXUS*. Technical Note, PUBSY No. JRC96386, EUR Report 27316 EN, 2015.
- [360] F. Casadei, V. Faucher, G. Valsamos, M. Larcher. *Decoupled formulation of constraints on adaptive hanging nodes in EUROPLEXUS*. Technical Note, PUBSY No. JRC96807, EUR Report 27347 EN, 2015.
- [361] F. Casadei, V. Aune, G. Valsamos, M. Larcher. *Description of the elasto-plastic material routine SGDI*. Technical Note, PUBSY No. JRC97557, EUR Report 27434 EN, 2015.
- [362] F. Casadei, G. Valsamos, M. Larcher. *Some notes on elasto-plasticity models in Europlexus ancestor codes*. Technical Note, PUBSY No. JRC97564, EUR Report 27593 EN, 2015.
- [363] F. Casadei, V. Aune, G. Valsamos, M. Larcher. *Some notes on the organization of routines for solid materials in EUROPLEXUS*. Technical Note, PUBSY No. JRC97565, EUR Report 27683 EN, 2015.
- [364] F. Casadei, V. Aune, G. Valsamos, M. Larcher. *Testing of the Johnson-Cook material model VPJC in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC98848, EUR Report 27594 EN, 2015.
- [365] V. Karlos, G. Solomos, M. Larcher. *Effect of axial load on steel members under blast*. The First International Conference on Structural Safety under Fire and Blast; Glasgow (UK): ASRANet Ltd; 2015. p. 55-63, 2015.
- [366] G. Valsamos, M. Larcher, G. Solomos. *Numerical Simulations in Support of the Blast Actuator Development Part II*. Technical Note, PUBSY No. JRC94812, 2015.
- [367] Jung A, Larcher M, Jirousek O, Koudelka P, Solomos G. *Strain-rate Dependence for Ni/Al Hybrid Foams*. T11th International DYMAT Conference; Les Ulis Cedex A (France): EDP Sciences; p. 04030, 2015.
- [368] Karlos V, Solomos G. *Dynamic response of steel beams and columns under blast loading*. Technical Note, PUBSY No. JRC96401, 2015.
- [369] Peroni M, Solomos G, Caverzan A, Larcher M, Valsamos G. *Assessment of dynamic mechanical behaviour of reinforced concrete beams using a Blast Simulator*. EPJ Web of Conferences; EPJ Web of Conferences; p. 01010, 2015.

**2016:**

- [370] V. Aune, G. Valsamos, F. Casadei, M. Larcher, M. Langseth, T. Børvik. *Inelastic response of thin aluminium plates exposed to blast loading*. Extended abstract of oral presentation at the 1st International Conference on Impact Loading of Structures and Materials, ICILSM 2016, Turin, Italy, 22–26 May 2016.
- [371] F. Casadei, V. Aune, G. Valsamos, M. Larcher. *Generalization of the pinball contact/impact model for use with mesh adaptivity and element erosion in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC101013, EUR Report 27888 EN, 2016.
- [372] Martin Larcher, Michel Arrigoni, Chiara Bedon, J. C. A. M. van Doormaal, Christof Haberacker, Götz Hüsken, Oliver Millon, Arja Saarenheimo, George Solomos, Laurent Thamie, Georgios Valsamos, Andy Williams, Alexander Stolz. *Design of Blast-Loaded*

- Glazing Windows and Facades: A Review of Essential Requirements towards Standardization.* Advances in Civil Engineering, vol. 2016, Article ID 2604232, 14 pages, 2016. doi:10.1155/2016/2604232, 2016.
- [373] V. Aune, F. Casadei, G. Valsamos, T. Børvik. *Formulation and Implementation of the VPJC Material Model in EUROPLEXUS.* JRC Technical Report, PUBSY No. JRC102336, EUR Report 27982 EN, 2016.
- [374] V. Aune, G. Valsamos, F. Casadei, M. Larcher, M. Langseth, T. Børvik. *Numerical study on the structural response of blast-loaded thin aluminium and steel plates.* International Journal of Impact Engineering 99 (2016), pp. 131–144.
- [375] Larcher M; Valsamos G. *Fast transient numerical simulations with EUROPLEXUS.* Training in: Course; 25-27 October 2016; Cadarche (France); F4E, ITER-IO (Organiser). 2016. JRC103470
- [376] Peroni M; Jung A; Larcher M; Solomos G. *High strain-rate properties of hybrid aluminium and polyurethane foams.* ICILSM2016 - Proceedings 1st International Conference on Impact Loading of Structures and Materials; Politecnico di Torino (Organiser). Politecnico di Torino; 2016. JRC100933
- [377] Casadei F, Aune V, Valsamos G, Larcher M. *Accounting for large membrane strains in Q4GS and T3GS elements in EUROPLEXUS.* EUR 27836. Luxembourg (Luxembourg): Publications Office of the European Union; 2016. JRC101010
- [378] Karlos V; Solomos G; Larcher M. *Analysis of the blast wave decay coefficient using the Kingery-Bulmash data.* INTERNATIONAL JOURNAL OF PROTECTIVE STRUCTURES 7 (3); 2016. p. 409-429. JRC102274
- [379] Larcher M. *Planning for resilience: a thematic overview on security issues.* Oral presentation in: 3rd European Conference on Sustainable Urban Mobility Plans; 12-13 April 2016; Bremen (Germany); City of Bremen together with DG MOVE (Organiser). 2016. JRC101022
- [380] Larcher M, Valsamos G, Solomos G. *Misuse of drones by terrorists as a new threat.* Ispra (Italy): European Commission; 2016. JRC98222
- [381] Casadei F, Aune V, Daude F, Galon P, Valsamos G, Larcher M. *Shock tube tests with coupled 1D-3D models in EUROPLEXUS.* EUR 27890. Luxembourg (Luxembourg): Publications Office of the European Union; 2016. JRC101011
- [382] Larcher M; Forsberg R; Bjoernstig U; Holgersson A; Solomos G. *Effectiveness of finite-element modelling of damage and injuries for explosions inside trains.* JOURNAL OF TRANSPORTATION SAFETY AND SECURITY 8 (S1); 2016. p. 83-100. JRC89005
- [383] Casadei F, Valsamos G, Larcher M. *Testing of the GLIS contact model in EUROPLEXUS.* EUR 27889. Luxembourg (Luxembourg): Publications Office of the European Union; 2016. JRC101012
- [384] Larcher M; Solomos G. *Vulnerability of railway systems to explosion terrorist attacks and protection techniques.* Oral presentation in: UIC Global Rail Security Congress 2016; 23-25 November 2016; Helsinki. 2016. JRC104000
- [385] Karlos V; Solomos G; Larcher M. *Analysis of blast parameters in the near-field for spherical free-air explosions.* EUR 27823. Luxembourg (Luxembourg): Publications Office of the European Union; 2016. JRC101039



- [386] Larcher M and Arrigoni M and Bedon C and Van Doormaal A and Haberacker C and Huesken G and Millon O and Saarenheimo A and Solomos G and Thamie L and Valsamos G and Williams A and Stolz A *Design of blast-loaded glazing windows and facades: a review of essential requirements towards standardization*. ADVANCES IN CIVIL ENGINEERING, pages 2604232, 2016.
- [387] Larcher M; Valsamos G. *Blast assessment tool - BLAssTo - V 1.0*. European Commission; 2016. JRC104200
- [388] F. Casadei, P. Saez, P. Díez, M. Larcher, G. Valsamos. *Scoping calculations for the simulation of human head impact with EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112659, 2018.
- [389] F. Casadei, P. Saez, P. Díez, M. Larcher, G. Valsamos. *Simulation of rat and monkey head slice deceleration with EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112660, 2018.
- [390] F. Casadei, G. Valsamos, M. Larcher, V. Aune. *Scoping calculations for the simulation of shock-loaded perforated plates with EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112046, 2018.

**2017:**

- [391] Karlos, V., Solomos, G. and Larcher, M. *Probabilistic analysis of steel columns under blast induced loads* EUROSTEEL 2017, 13-15 September 2017, Scandic Copenhagen Hotel, EUROSTEEL 2017 proceedings, 2017, JRC106080.
- [392] Lori G; Morison C; Larcher M; Belis J. *Sustainable facade design for glazed buildings in a blast resilient urban environment*. Springer; 2017. JRC106471
- [393] Jung A, Luksch J, Felten M, Reis M, Sory D, Pullen A, Proud W, Larcher M, Valsamos G and Solomos G. *Investigation of strain-rate effects in Al foams and Ni-Al hybrid foams on different scales* In: 88th GAMM Annual Meeting; 06-10 March 2017; Ilmenau (Weimar (Germany)). Proceedings in Applied Mathematics and Mechanics; GAMM (Organiser). Wiley; 2017. JRC106983
- [394] Larcher, M., Valsamos, G. and Karlos, V. *Access control points: Reducing a possible blast impact by meandering*. ADVANCES IN CIVIL ENGINEERING, ISSN 1687-8086, 2017, JRC107762
- [395] Karlos, V., Solomos, G. and Larcher, M. *Validity of blast parameters in the near-field*. 9th National Conference of Steel Structures, 05-07 October 2017, Medical School of University of Thessaly - Larisa, 9th National Conference of Steel Structures, 2017, JRC108107.
- [396] Cambriani, A., Dagata, E., Freis, D., Larcher, M. and Valsamos, G. *Accident Analyses for the Radioisotope Heater Unit Prototype*. European Commission - Joint Research Centre, Karlsruhe, 2017, JRC110377.
- [397] V. Aune, G. Valsamos, F. Casadei, M. Langseth, T. Børvik. *On the dynamic response of blast-loaded steel plates with and without pre-formed holes*. International Journal of Impact Engineering 108 (2017), pp. 27–46.
- [398] V. Aune, G. Valsamos, F. Casadei, M. Larcher, M. Langseth, T. Børvik. *Use of damage-based mesh adaptivity to predict ductile failure in blast-loaded aluminium plates*. International Conference on Dynamic Fracture of Ductile Materials, Dynfrac 2017, DYMAT

- 23<sup>rd</sup> Technical Meeting, 12-14 September 2017, Trondheim, Norway. Also in *Procedia Engineering* 197 (2017) 3-12.
- [399] F. Casadei, P. Saez, P. Diez, M. Larcher, G. Valsamos. *Simulation of rat head deceleration with EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC106754, 2017.
- [400] V. Aune, G. Valsamos, F. Casadei, M. Larcher, M. Langseth, T. Børvik. *Numerical simulations of blast-loaded plates using combined fluid and structure mesh adaptivity*. Extended abstract for the 30th Nordic Seminar on Computational Mechanics, NSCM-30, J Høsberg, N.L. Pedersen (Eds.), 2017.
- [401] G. Valsamos, V. Aune, F. Casadei, M. Larcher. *Numerical simulation of shock tube tests with EPX*. EPX Users Meeting, 2017.
- [402] F. Casadei, G. Valsamos, M. Larcher, V. Aune. *Generalization of the flying debris model for use with mesh adaptivity and element erosion in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117282, 2019.
- [403] F. Casadei, G. Valsamos, M. Larcher. *Preliminary considerations for the formulation of a new model of rigid bodies in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117292, 2019.
- [404] F. Casadei, G. Valsamos, M. Larcher. *Formulation and implementations of a new model of rigid bodies in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117293x, 2019.
- [405] F. Casadei, V. Faucher, G. Valsamos, M. Larcher. *A new strategy for computing structural influence domains in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117296, 2019.
- [406] F. Casadei, G. Valsamos, M. Larcher, V. Aune. *Implementation of friction in the pinball-based contact-impact model of EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112047, 2018.
- [407] F. Casadei, G. Valsamos, M. Larcher. *Post-processing of EUROPLEXUS results by the POV-Ray ray tracing visualizer*. JRC Technical Report, PUBSY No. JRC117295, 2019.
- [408] F. Casadei, M. Larcher, G. Valsamos, V. Faucher. *Implementation of Assembled Surface Normals and of a Penalty Contact Formulation in the Pinball Model of Europlexus – Revision 1*. JRC Technical Report, PUBSY No. JRC111429, EUR Report 29217 EN, 2018.
- [409] F. Casadei, M. Larcher, G. Valsamos, B. Langrand. *New simulations of cellular materials crushing with EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112050, 2018.
- [410] F. Casadei, G. Valsamos, M. Larcher, V. Faucher. *Development of graphical visualization for mesh-adaptive calculations in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117403, 2019.
- [411] F. Casadei, G. Valsamos, M. Larcher. *Visualization of links in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112661, 2018.
- [412] F. Casadei, V. Aune, G. Valsamos, M. Larcher. *Contact detection by pseudo-nodal pinballs with mesh adaptivity in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112048, 2018.

- [413] F. Casadei, G. Valsamos, M. Larcher. *Adaptivity in fully integrated 8-node hexahedra and in 6-node prism elements in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112662, 2018.
- [414] F. Casadei, G. Valsamos, M. Larcher. *A technique based on mesh adaptivity for the simulation of explosive detonations in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112663, 2018.
- [415] F. Casadei, M. Larcher, G. Valsamos, V. Faucher. *Implementation of the LINK DECO RIGI directive under MPI in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC111615, EUR Report 29206 EN, 2018.
- [416] F. Casadei, M. Larcher, G. Valsamos, V. Faucher. *Implementation of the LINK DECO BLOQ directive under MPI in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC111619, EUR Report 29208 EN, 2018.

**2018:**

- [417] F. Casadei, G. Valsamos, M. Larcher, V. Aune. *Characterization of a shock tube facility by EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112051, 2018.
- [418] F. Casadei, M. Larcher, G. Valsamos, V. Aune. *A solution mapping algorithm in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112052, EUR Report 29259 EN, 2018.
- [419] V. Aune, F. Casadei, G. Valsamos, M. Langseth, T. Børvik. *A shock tube used to study the dynamic response of blast-loaded plates*. Presented at 18th International Conference on Experimental Mechanics (ICEM18), Brussels 1–5 July 2018. Also in Proceedings, 2 (2018) 503, doi:10.3390/ICEM18-05395.
- [420] F. Casadei, G. Valsamos, M. Larcher, R. Ortiz. *Mesh quality assessment in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC112389, 2018.
- [421] F. Casadei, G. Valsamos, M. Larcher, V. Faucher. *Decoupled formulation of constraints on adaptive hanging nodes in EUROPLEXUS – Revision 1*. JRC Technical Report, PUBSY No. JRC112665, 2018.

**2019:**

- [422] G. Valsamos, F. Casadei, G. Solomos, M. Larcher. *Risk assessment of blast events in a transport infrastructure by fluid-structure interaction analysis*. Safety Science 118 (2019), 887–897.
- [423] V. Aune, G. Valsamos, F. Casadei, T. Børvik. *Aspects of Fluid-Structure Interaction on the dynamic response of blast-loaded metallic plates*. Paper to be presented at the Light Weight Armour for Defense & Security (LWAG 2019) Conference, ENSAIT, Roubaix (F), October 8–9, 2019.
- [424] F. Casadei, M. Larcher, G. Valsamos. *Air blast (AIRB) model implementation with adaptivity and erosion in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117289, 2019.

- [425] F. Casadei, G. Valsamos, M. Larcher, C. Duñó Nosas, P. Saez, P. Díez. *Simulation of rat, monkey and human head under blast loading with EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117290, 2019.
- [426] F. Casadei, G. Valsamos, M. Larcher. *Further development of the new model of rigid bodies in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117294, 2019.
- [427] F. Casadei, G. Valsamos, M. Larcher, V. Aune. *Some ideas for the formulation and implementation of a node splitting algorithm in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117300, 2019.
- [428] F. Casadei, M. Larcher, G. Valsamos, V. Faucher. *Implementation of the LINK DECO RIGI directive under MPI in EUROPLEXUS – Revision 1*. JRC Technical Report, PUBSY No. JRC117472, 2019.
- [429] F. Casadei, M. Larcher, G. Valsamos, V. Faucher. *Implementation of the LINK DECO BLOQ directive under MPI in EUROPLEXUS – Revision 1*. JRC Technical Report, PUBSY No. JRC117473, 2019.
- [430] F. Casadei, G. Valsamos, M. Larcher, V. Faucher. *Coupled, decoupled and MPI formulations of the ARMA constraint in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117474, 2019.
- [431] F. Casadei, G. Valsamos, M. Larcher, V. Faucher. *Coupled, decoupled and MPI-coupled formulations of the GLUE constraint in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117475, 2019.
- [432] F. Casadei, G. Valsamos, M. Larcher, V. Faucher. *Coupled, decoupled and MPI formulations of the DEPL, VITE and ACCE constraints for imposed motions in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117476, 2019.
- [433] F. Casadei, M. Larcher, G. Valsamos. *An attempt to use the FreeGLUT and f03gl OpenGL graphical libraries in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117477, 2019.
- [434] F. Casadei, G. Valsamos, M. Larcher, V. Aune. *Fluid-Structure Interaction with 3D beam and bar elements in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117478, 2019.
- [435] F. Casadei, V. Faucher, G. Valsamos, M. Larcher. *MPI parallel calculations under MS-Windows in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC117479, 2019.
- [436] V. Faucher, F. Casadei, G. Valsamos, M. Larcher. *An alternative definition of 3D shell bilateral contact conditions in the GLIS contact model of EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC118564, 2019.
- [437] F. Casadei, G. Valsamos, M. Larcher, V. Faucher. *Generalization of the FLSW and FLSR directives to deal with several Fluid-Structure Interaction zones in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC118565, 2019.
- [438] F. Casadei, G. Valsamos, M. Larcher, V. Faucher. *Implementation of 3D Adaptivity Indicators for Fast Transient Dynamics in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC118658, 2019.

**2020:**

- [439] G. Solomos, M. Larcher, G. Valsamos, V. Karlos, F. Casadei. *A survey of computational models for blast induced human injuries for security and defence applications*. JRC Technical Report, PUBSY No. JRC119310, EUR 30039, 2020.
- [440] F. Casadei, G. Valsamos, M. Larcher, V. Faucher, A. Cambriani. *On some recent applications and developments concerning contact models in EUROPLEXUS*. JRC Technical Report, PUBSY No. JRC119981, in publication.
- [441] G. Valsamos, M. Larcher, F. Casadei. *A blast assessment tool for urban environments*. JRC Technical Report, PUBSY No. JRC120958, 2020.
- [442] G. Valsamos, M. Larcher, F. Casadei, V. Karlos. *A numerical framework to support the certification of barrier testing*. JRC Technical Report, PUBSY No. JRC120307, EUR 30165 EN, 2020.
- [443] V. Aune, G. Valsamos, F. Casadei, M. Langseth, T. Børvik. *Fluid-structure interaction effects during the dynamic response of clamped thin steel plates exposed to blast loading*. Paper submitted to the International Journal of Mechanical Sciences, August 2020.

## CASTEM-PLEXUS/EUROPLEXUS BIBLIOGRAPHY

## 1978:

- [444] **P. Verpeaux, C. Chavant**  
Programme PLEXUS. Formulation théorique et organisation générale. *Rapport DEDR/DEMT 78.060*

## 1979:

- [445] **C. Chavant, P. Verpeaux**  
Programme PLEXUS. Comparaisons à des cas connus. *Rapport DEDR/DEMT 79.009*
- [446] **C. Chavant, A. Hoffmann, P. Verpeaux, J. Dubois**  
PLEXUS : A general computer code for explicit Lagrangian computation. *SMIRT 5, B2-8*

## 1980:

- [447] **H. Bung, C. Berriaud, A. Hoffmann**  
Calcul de perforation d'une dalle en béton par un projectile rigide. *Rapport DEDR/DEMT 80.103*

## 1981:

- [448] **M. Lepareux, B. Schwab**  
Code PLEXUS. Réaction sodium-eau en 3D. *Rapport DEDR/DEMT 81.004*
- [449] **M. Ruznievski**  
L'élément poutre dans PLEXUS. *Rapport DEDR/DEMT 81.014*
- [450] **M. Lepareux, A. Hoffmann, H. Van Crasbeck**  
Chute d'un chateau de transport sur l'aire de déchargement de PEGASE. *Rapport DEDR/DEMT 81.015*
- [451] **H. Bung, M. Lepareux**  
Programme PLEXUS. Choc d'un tuyau contre un mur. Calcul 2D et 3D. *Note DEDR/DEMT 81.053*
- [452] **M. Lepareux**  
Chute d'un chateau de transport sur l'aire de déchargement de PEGASE (additif au rapport 81.015). *Rapport DEDR/DEMT 81.053*
- [453] **M. Lepareux**  
Sureté des usines. Proposition d'un modèle de comportement pour l'étude des voiles ou coques en béton, en cas de choc. Programme PLEXUS. *Rapport DEDR/DEMT 81.063*
- [454] **M. Lepareux**  
Système CEASEMT (CASTEM). Programme PLEXUS. Présentation résumée des possibilités. *Rapport DEDR/DEMT 81.066*
- [455] **S. Iche, C. Duretz**  
Etude de la chute accidentelle d'un étui sur le collet d'un chateau de transport. *Note DEDR/DEMT 81.705*

- [456] **J. Vinsonneau**  
Etude de la chute d'un emballage SPX1 sur un tourillon de manutention. *Note DEDR/DEMT 81.716*
- 1982:**
- [457] **M. Lepareux, H. Bung**  
Le programme PLEXUS et la méthode ALE. *Rapport DEDR/DEMT 82.012*
- [458] **M. Lepareux, C. Chavant**  
Système CEASEMT. Le modèle de Drucker-Prager pour la rupture du béton dans le programme PLEXUS. *Rapport DEDR/DEMT 82.014*
- [459] **R. Ladouceur, M. Lepareux**  
Calcul de dépressurisation de la ligne VVP depuis le générateur de vapeur jusqu'à la brèche de rupture circonférentielle située en aval du supportage principal. *Rapport DEDR/DEMT 82.020*
- [460] **M. Lepareux, B. Schwab**  
Filière à neutrons rapides. Circuit "Regain". Efficacité d'un niveau libre à la base du générateur de vapeur pour atténuer l'amplitude des ondes de pression. *Rapport DEDR/DEMT 82.033*
- [461] **J.L. Lieutenant, M. Lepareux**  
Calcul PLEXUS 3D. Impact sur une tuyauterie de l'échangeur intermédiaire de SPX1. *Rapport DEDR/DEMT 82.046*
- [462] **B. Schwab**  
Validation de PLEXUS. Propagation d'ondes planes dans un tuyau avec changement de section. *Note DEDR/DEMT 82.068*
- [463] **J.L. Lieutenant, M. Lepareux**  
Poinçonnement de tubes. Chute d'un tube sur un poinçon fixe. *Rapport DEDR/DEMT 82.071*
- [464] **M. Lepareux et al.**  
Fiches de validation du programme PLEXUS. *Rapport DEDR/DEMT 82.075*
- [465] **R. J. Gibert, B. Schwab**  
Validation de PLEXUS. Piquage d'une tuyauterie sur un réservoir. Aspects mono- et bi-dimensionnels. *Note DEDR/DEMT 82.083*
- [466] **M. Mallet, H. Bung**  
Le traitement numérique des ondes de chocs par la méthode ALE programmée dans PLEXUS. *Rapport DEDR/DEMT 82.084*
- [467] **T. Yuritzinn, Th. Charras**  
Mécanique de la rupture. Simulation numérique de la propagation d'une fissure. *Rapport DEDR/DEMT 82.088*
- [468] **B. Schwab**  
Générateur de vapeur de Creys-Malville. Comparaison entre un modèle tridimensionnel (PLEXUS) et un modèle monodimensionnel (METCAR), pour traiter la propagation d'ondes dans la partie haute du générateur de vapeur. *Note DEDR/DEMT 82.089*

- [469] **B. Schwab, J.L. Lieutenant**  
Validation de PLEXUS. Piquage d'une tuyauterie sur un réservoir. Aspects tridimensionnels. *Note DEDR/DEMT 82.100*
- [470] **H. Bung, M. Lepareux**  
Programme PLEXUS. Algorithme pour traiter les lignes de glissement. *Note DEDR/DEMT 82.124*
- [471] **H. Bung, M. Lepareux**  
Programme PLEXUS. Algorithme de remaillage automatique. *Note DEDR/DEMT 82.125*
- 1983:**
- [472] **H. Bung, M. Lepareux**  
Impact d'un projectile sur une cible en béton. *Note DEDR/DEMT 83.001*
- [473] **M. Lepareux, A. Hoffmann**  
Système CEASEMT. Programme PLEXUS. Exposé fait à l'ENEA. Rome le 20 décembre 1982. *Rapport DEDR/DEMT 83.003*
- [474] **R.J. Gibert, B. Schwab**  
Réaction sodium-eau. Etude de l'élément absorbant (CL2D ou CL3D) de PLEXUS en vue de représenter des plaques perforées traversées par une onde plane. *Rapport DEDR/DEMT 83.011*
- [475] **P. Aillaud, M. Lepareux**  
Code PLEXUS. Explosion hydrogène. *Rapport DEDR/DEMT 83.019*
- [476] **A. Hoffmann, M. Lepareux**  
Robotique - Le calcul des robots - Une approche à l'aide de CASTEM. *Rapport DEDR/DEMT 83.023*
- [477] **M. Lepareux, J.L. Lieutenant**  
Programme PLEXUS. Collaboration CEA/UKAEA. Réponse élastoplastique d'une plaque soumise à un chargement dynamique. *Rapport DEDR/DEMT 83.025*
- [478] **M. Lepareux, J.L. Lieutenant**  
Programme PLEXUS. Calcul des dalles en béton armé soumises à l'impact d'un avion (Lear Jet). *Rapport DEDR/DEMT 83.026*
- [479] **T. Yuritzinn, M. Lepareux**  
Programme PLEXUS. Amortissement linéaire. *Rapport DEDR/DEMT 83.027*
- [480] **M. Lepareux, H. Bung**  
Système CEASEMT. Programme PLEXUS. Présentation résumée (printemps 1983). *Rapport DEDR/DEMT 83.034*
- [481] **J.L. Lieutenant, Ph. Jamet**  
Impact d'un Lear Jet sur des dalles en béton armé de type CPI et P'4. Première évaluation des coefficients de sécurité. *Rapport DEDR/DEMT 83.052*
- [482] **P. Guillemot**  
Etudes de chocs sur des cylindres en aluminium. *Rapport DEDR/DEMT 83.056*
- [483] **M. Lepareux et al.**  
Fiches de validation du programme PLEXUS. Janvier 1984. *Rapport DEDR/DEMT 83.068*



- [484] **J. Brochard, M. Lepareux**  
Calcul préliminaire de la tenue mécanique de l'enceinte d'un réacteur PWR lors d'une détonation d'hydrogène. *Note DEDR/DEMT 83.091*
- [485] **J.L. Lieutenant, P. Verpeaux**  
PLEXUS 3D. Comportement dynamique élastoplastique de dalles en béton armé. Dalles 8, 9 et 10. *Note DEDR/DEMT 83.110*
- [486] **J.L. Lieutenant, P. Verpeaux**  
Calcul de 2 dalles (dimensionnement élastique dynamique P'4) en fonction de la vitesse d'impact d'un Lear Jet. *Note DEDR/DEMT 83.112*
- [487] **H. Bung, M. Lepareux, B. Schwab**  
Programme PLEXUS. Introduction d'un matériau "eau-vapeur" traité en fluide compressible diphasique homogène. *Note DEDR/DEMT 83.129*
- [488] **S. Iche, Ch. Duretz**  
Etude de la chute accidentelle d'un étui de transport sur le collet d'un emballage mono-alvéole. *Note DEDR/DEMT/SYST/TCR 83.001*
- [489] **Y. Blanchet**  
Programme PLEXUS. Réaction sodium-eau dans un GV - Premier calcul d'interprétation de l'expérience BA-11. *Note DEDR/DRNR/SYTC/LMCC 83.1046*
- 1984:**
- [490] **M. Lepareux, J.L. Lieutenant**  
Programme PLEXUS. Calcul d'une dalle en béton armé soumise à l'impact d'un avion (Mirage IV). *Note DEDR/DEMT 84.006*
- [491] **B. Schwab, M. Lepareux**  
Décompression du générateur de vapeur 68-19 du palier P4 en cas de rupture de la tuyauterie de vapeur. *Rapport DEDR/DEMT 84.019*
- [492] **B. Schwab**  
Calcul et comparaison aux essais du régime transitoire consécutif à la décompression d'un tube RIC en vue du calcul mécanique du fouettement de ce tube. Application aux réacteurs à eau sous pression. *Note DEDR/DEMT 84.028*
- [493] **H. Bung, A. Forestier, M. Lepareux**  
Programme PLEXUS. Traitement des fluides incompressibles ou peu compressibles. *Rapport DEDR/DEMT 84.028*
- [494] **D. Robin, Th. Charras**  
Programme PLEXUS. Calcul de la fissuration dynamique d'une éprouvette pré-contrainte en statique. *Rapport DEDR/DEMT 84.053*
- [495] **A. Forestier**  
Etude des ondes de choc et quelques applications à PLEXUS. *Rapport DEDR/DEMT 84.084 (TTMF/84.038)*
- [496] **A. Forestier, M. Lepareux**  
PLEXUS - Cas de l'explosion hydrogène. *Rapport DEDR/DEMT 84.085 (TTMF/84.039)*

- [497] **H. Bung, M. Lepareux**  
Programme PLEXUS. Réaction sodium-eau. Contrôle anti-voilement (anti-hourglass).  
*Rapport DEDR/DEMT 84.107*
- [498] **B. Schwab, M. Lepareux**  
PLEXUS. Traitement de la réaction sodium-eau dans les tuyauteries (Calcul eulérien 1D).  
*Rapport DEDR/DEMT 84.119*
- [499] **M. Lepareux, J.L. Lieutenant**  
PLEXUS - Calcul statique d'une dalle en béton armé. *Rapport DEDR/DEMT 84.162*
- [500] **M. Lepareux, J.L. Lieutenant**  
PLEXUS. Poinçonnement d'une plaque en acier inox. *Rapport DEDR/DEMT 84.184*
- [501] **A. Kerboudj**  
Amélioration des méthodes de calcul dans les interactions fluide-structure lors d'un transitoire intense. *Rapport DEDR/DEMT 84.185*
- [502] **M. Lepareux, B. Schwab**  
Validation des modèles de cavitation. *Rapport DEDR/DEMT 84.212*
- [503] **J.P. Rossi, J.A. Clementi, M. Lepareux, A. Hoffmann**  
Robotique. Mécanismes spatiaux formés de corps déformables et indéformables. Programme PLEXUS. *Rapport DEDR/DEMT 84.222*
- [504] **M. Lepareux, B. Schwab, H. Bung**  
PLEXUS. Réaction sodium-eau. Validation de la méthode A.L.E. Interprétation de l'essai MANON 11. *Rapport DEDR/DEMT 84.246*
- [505] **M. Lepareux, H. Bung**  
PLEXUS - Coque épaisse avec cisaillement en 2D et axisymétrique. *Rapport DEDR/DEMT 84.265*
- [506] **M. Lepareux, H. Bung**  
PLEXUS. Coque épaisse avec cisaillement en 2D et axisymétrique. *Rapport DEDR/DEMT 84.266*
- [507] **A. Hoffmann, M. Lepareux, B. Schwab, A. Forestier, H. Bung**  
PLEXUS. A general computer program for fast dynamic analysis. (Porto-Alegre, Brésil).  
*Rapport DEDR/DEMT 84.295*
- [508] **A. Forestier, E. Houssini-Skalli**  
PLEXUS - Premiers calculs de l'explosion hydrogène dans le cas d'une enceinte bidimensionnelle axisymétrique. *Rapport DEDR/DEMT 84.304 (TTMF/84.104)*
- [509] **Y. Blanchet, Y. Thaon**  
Validation de PLEXUS-2D : Effets de la modélisation de la membrane dans l'essai de réaction sodium-eau BA-11. *Note DEDR/DRNR/SYTC/LMCC 84.1035*

**1985:**

- [510] **B. Schwab, M. Lepareux**  
PLEXUS. Réaction sodium-eau. Traitement des faisceaux de tubes en hélice (option 2D lagrangienne). *Rapport DEDR/DEMT 85.014*

- [511] **R. Rivière, M. Lepareux, J.L. Lieutenant**  
Analyse du dimensionnement du génie civil d'un voile des structures internes du bâtiment réacteur P'4. *Rapport DEDR/DEMT 85.019*
- [512] **H. Bung, Ph. Jamet, M. Lepareux**  
PLEXUS. Modèle de comportement simplifié pour le béton. *Rapport DEDR/DEMT 85.020*
- [513] **M. Lepareux, H. Bung**  
PLEXUS. Non linéarités géométriques des éléments massifs. *Rapport DEDR/DEMT 85.021*
- [514] **J.L. Lieutenant, M. Lepareux**  
Programme PLEXUS. Calcul de la poutre sous butée J. *Rapport DEDR/DEMT 85.056*
- [515] **B. Schwab, M. Lepareux**  
PLEXUS. Réaction sodium-eau. Traitement des membranes de sécurité. *Rapport DEDR/DEMT 85.072*
- [516] **H. Bung, M. Lepareux**  
Etude de l'impact d'un conteneur contre un mur indéfiniment rigide. *Rapport DEDR/DEMT 85.073*
- [517] **B. Schwab, M. Finck, M. Blanchet**  
Projet 1500. Réaction sodium-eau. Calculs des effets locaux dans le générateur de vapeur à l'aide de PLEXUS. *Rapport DEDR/DEMT 85.081*
- [518] **B. Schwab, M. Finck, M. Blanchet, A. Lapicoré**  
Three dimensionnal sodium-water reaction calculations with fluid-structures interactions for LMFBFR steam generators (3rd international conference on boilers dynamics and control in nuclear power stations). *Rapport DEDR/DEMT 85.273*
- [519] **M. Lepareux, P. Gendre, Ph. Ravier**  
Calcul de l'impact d'une pompe sur le fond de cuve de SPX avec le programme PLEXUS. *Rapport DEDR/DEMT 85.323*
- [520] **M. Lepareux, V. Motreff, J. Sevagen**  
Robotique - Mécanismes spatiaux - Corps déformables et indéformables. *Rapport DEDR/DEMT 85.324*
- [521] **B. Schwab, M. Finck**  
Réaction sodium-eau - Calcul d'une réaction au droit du corps central du G.V. de SPX2. *Rapport DEDR/DEMT 85.332*
- [522] **A. Hoffmann, M. Lepareux, H. Bung, A. Barraco, A. Cuny**  
Dynamique des systèmes articulés déformables et indéformables. Application à la robotique (colloque "Tendances actuelles en calcul de structures" - Bastia 6-8 nov.85). *Rapport DEDR/DEMT 85.367*
- [523] **J.L. Lieutenant, M. Lepareux, J. Rivière**  
Calcul de l'impact d'un Boeing sur l'enceinte de SPX1. *Rapport DEDR/DEMT 85.387*
- [524] **J.L. Lieutenant, M. Lepareux, J. Rivière**  
Calcul de l'impact d'un Lear-jet sur l'enceinte de SPX1. *Rapport DEDR/DEMT 85.388*
- [525] **A. Forestier, S. Goldstein**  
Internal air-hydrogen detonation: calculation on a containment — (SMIRT 8, Bruxelles). *Rapport DEDR/DEMT 85.426*

- [526] **J.L. Lieutenant, M. Lepareux**  
Programme PLEXUS - Poinçonnement dynamique de plaques en acier. *Rapport DEDR/DEMT 85.442*
- [527] **J.L. Lieutenant, M. Lepareux, Ph. Jamet**  
Programme PLEXUS - Sûreté des usines - Calcul de la flexion sous impact de dalles en béton armé. *Rapport DEDR/DEMT 85.446*
- [528] **M. Lepareux, B. Schwab, A. Hoffmann, Ph. Jamet, H. Bung**  
Un programme de calcul général pour l'analyse dynamique rapide - Cas des tuyauteries. (Colloque "Tendances actuelles en calcul de structures" - Bastia, 6-8 nov.85). *Rapport DEDR/DEMT 85.458*
- [529] **M. Lepareux, H. Bung, Ph. Pasquet**  
Programme PLEXUS - Ecrasement d'un caisson sous l'impact d'un projectile. *Rapport DEDR/DEMT 85.504*
- [530] **Y. Blanchet**  
Validation de la méthode A.L.E. dans PLEXUS-2D sur l'essai de réaction sodium-eau SI-1-5 (sans interaction fluide-structure). *Note DEDR/DRNR/SYTC/LMCC 85.1001*
- [531] **Y. Blanchet**  
Validation de la méthode A.L.E. dans PLEXUS-2D sur l'essai de réaction sodium-eau BA-11 (sans interaction fluide-structure). *Note DEDR/DRNR/SYTC/LMCC 85.1003*
- [532] **Y. Blanchet, Y. Thaon**  
Validation de la méthode A.L.E. dans PLEXUS-2D sur l'essai de réaction sodium-eau BA-11 (avec interaction fluide-structure). *Note DEDR/DRNR/SYTC/LMCC 85.1008*
- [533] **Y. Blanchet, Y. Thaon**  
Validation de PLEXUS-3D Lagrange : Comparaison avec les codes SADCAT et WHAM sur un problème d'interaction fluide-structure. *Note DEDR/DRNR/SYTC/LMCC 85.1020*
- 1986:**
- [534] **M. Khardi**  
Précalcul au fouetttement. *Rapport DEDR/DEMT 86.008*
- [535] **A. Hoffmann, Ph. Jamet, M. Lepareux, A. Millard, B. Barbé, Ph. Maurel**  
Ultimate flexural behaviour of reinforced concrete shells under static and dynamic loading. (Communication présentée au SMIRT 8). *Rapport DEDR/DEMT 86.030*
- [536] **M. Lepareux, B. Schwab, J.L. Lieutenant, A. Millard**  
Fouetttement des tubes-guides d'instrumentation sous cuve. Calculs effectués avec les codes TEDEL et PLEXUS. *Rapport DEDR/DEMT 86.110*
- [537] **E. Lurin, Th. Charras**  
Formulation d'un nouveau modèle de fissuration dynamique dans PLEXUS. Premières applications à une éprouvette de laboratoire. *Rapport DEDR/DEMT 86.130*
- [538] **H. Bung, M. Lepareux**  
Dynamique des robots - Mécanismes spatiaux : Articulations. *Rapport DEDR/DEMT 86.201*

- [539] **M. Lepareux, F. Finck, B. Schwab**  
PLEXUS - Réaction sodium-eau - Traitement des faisceaux de tubes en hélice - Option 3D. *Rapport DEDR/DEMT 86.291*
- [540] **A. Hoffmann, M. Lepareux, Ph. Jamet, B. Schwab, A. Forestier**  
PLEXUS - A general computer program for fast dynamic analysis. *Rapport DEDR/DEMT 86.295*
- [541] **E. Lurin, Th. Charras, P. Verpeaux**  
Formulation d'un nouveau modèle de fissuration dans PLEXUS (Complément au rapport DEMT 86.130). *Rapport DEDR/DEMT 86.318*
- [542] **M. Lepareux, J.L. Lieutenant**  
Calcul de l'impact d'une pompe sur le fond de cuve SPX avec le programme PLEXUS (2ième partie). *Rapport DEDR/DEMT 86.328*
- [543] **F. Sangianni, E. Naddeo, M. Lepareux**  
Modélisation des asservissements dans PLEXUS. *Rapport DEDR/DEMT 86.337*
- [544] **E. Lurin, Th. Charras, Ph. Jamet**  
Simulation numérique d'essais de fissuration dynamique sur éprouvette. *Rapport DEDR/DEMT 86.358*
- [545] **B. Schwab, Y. Berthion**  
Boucle OPERA - Calculs à l'aide de PLEXUS des débits critiques aux divers emplacements de rupture envisagés sur la boucle. *Rapport DEDR/DEMT 86.398*
- [546] **B. Schwab**  
PLEXUS 1D - Décompression accidentelle de la ligne VVP-P4 en cas de rupture guillotine en aval du support SORENAM. *Rapport DEDR/DEMT 86.437*
- [547] **D. Brochard**  
Etude du comportement des faisceaux de tubes baignés par du fluide à l'aide d'une méthode d'homogénéisation. *Rapport DEDR/DEMT 86.444*
- [548] **M. Lepareux, J.L. Lieutenant**  
Programme PLEXUS - Calcul d'écrasement d'une tuyauterie R.U.R. en air. *Rapport DEDR/DEMT 86.458*

**1987:**

- [549] **A. Combescure, Th. Charras, P. Verpeaux**  
Eléments multicouches dans CASTEM 2000 et PLEXUS. *Note DEDR/DEMT 87.003*
- [550] **B. Schwab**  
Boucle OPERA - Dépressurisation des écrans neutrophages. *Rapport DEDR/DEMT 87.115*
- [551] **B. Schwab, F. Finck**  
Conséquences d'une réaction sodium-eau violente sur la partie inférieure du générateur de vapeur de SPX2 - Premiers calculs PLEXUS. *Rapport DEDR/DEMT 87.129*
- [552] **B. Schwab, A. Baudouin**  
R.C.V.S. cuve courte. Efforts dynamiques sur tubes-guides associés à une décompression suite à un A.P.R.P. *Rapport DEDR/DEMT 87.178*

- [553] **B. Schwab, A. Baudouin**  
Calcul des coefficients d'anisotropie acoustique pour le plénum supérieur du R.C.V.S. version cuve courte. *Rapport DEDR/DEMT 87.179*
- [554] **A. Barraco, B. Cuny, A. Hoffmann, Ph. Jamet, M. Lepareux, H. Bung**  
Computer simulation of manipulator dynamics using different control laws (Communication présentée à la "Second international conference on robotics and factories of the future", San Diego, 28-31 juil. 1987). *Rapport DEDR/DEMT 87.202*
- [555] **J.L. Lieutenant**  
Calcul de l'impact d'un pont roulant sur l'enceinte SPX1. *Rapport DEDR/DEMT 87.204*
- [556] **A. Barraco, B. Cuny, A. Hoffmann, Ph. Jamet, M. Lepareux, H. Bung**  
Conception mécanique, cinématique et dynamique des robots. (Communication présentée à la conférence "Les robots industriels" à la Société Française des Mécaniciens, 25-26 mars 1987. Revue Française de Mécanique, numéro spécial 1987). *Rapport DEDR/DEMT 87.297*
- [557] **M. Lepareux, Ph. Matheron, Ph. Jamet, J.L. Lieutenant, J. Couilleaux, G. Lazare-Chopard, J. Rivière**  
Ultimate behaviour of reinforced concrete shells under static and dynamic loading. (Communication présentée au 9ième SMIRT, Lausanne, 17-21 aout 1987). *Rapport DEDR/DEMT 87.360*
- [558] **Ph. Jamet, M. Lepareux, Ph. Matheron, J.L. Lieutenant, J. Couilleaux, D. Duboelle, J. Aguilar**  
Experimental and numerical studies of impacts on stainless steel plates subjected to rigid missiles at low velocity. (Communication présentée au 9ième SMIRT, Lausanne, 17-21 aout 1987). *Rapport DEDR/DEMT 87.361*
- [559] **H. Bung, Ph. Jamet**  
Conséquences mécaniques d'une explosion BORAX dans SILOE. *Rapport DEDR/DEMT 87.418*
- [560] **B. Schwab**  
PLEXUS 1D - Décompression accidentelle de la ligne VVP-P4 en cas de rupture guillotine en aval du support SORENAM. *Rapport DEDR/DEMT 87.553*
- [561] **J.L. Lieutenant, M. Lepareux**  
Programme PLEXUS - Poinçonnement dynamique de plaques en acier d'épaisseur 6 mm. *Rapport DEDR/DEMT 87.555*
- [562] **J.L. Lieutenant, M. Lepareux**  
Programme PLEXUS - Poinçonnement dynamique de plaques en acier d'épaisseur 10 mm. *Rapport DEDR/DEMT 87.556*
- [563] **P. Caumette, M. Lott, J-F. Ivars**  
Quelques cas simples d'interaction fluide-structure - Tests du code PLEXUS. *Note DEDR/DRE/STRE/LCP 87.825*
- 1988:**
- [564] **J.L. Lieutenant, M. Lepareux**  
Programme PLEXUS. Sureté des usines. Calcul à la ruine d'une dalle en béton armé avec un modèle massif tridimensionnel. *Rapport DEDR/DEMT 88.142*

- [565] **A. Forestier**  
Schémas du second ordre en bidimensionnel pour un fluide compressible avec un maillage arbitraire. *Rapport DEDR/DEMT 88.263*
- [566] **A. Forestier**  
Application d'un solveur de Riemann dans PLEXUS pour un gaz réel. Application au gaz de Van der Waals. *Rapport DEDR/DEMT 88.321*
- [567] **Ch. Lepretre**  
Calcul à la ruine des structures en béton armé. Mise au point d'un modèle béton en contraintes planes. *Rapport DEDR/DEMT 88.330*
- [568] **D. Dubucq, C. Euverte**  
PLEXUS. Application à la robotique. Validation et convivialité. *Rapport DEDR/DEMT 88.331*
- [569] **B. Schwab, M. Lepareux**  
PLEXUS-1D. Traitement des appareils à pertes de charge variables placés sur une tuyauterie. Application au clapet anti-retour. *Rapport DEDR/DEMT 88.332*
- 1989:**
- [570] **M. Lepareux, B. Schwab**  
Calcul de la tenue du confinement d'un réacteur à neutrons rapides en cas de fusion du coeur. Première application au cas européen (Benchmark CONT). *Rapport DEDR/DEMT 89.062*
- [571] **H. Bung, F. Gantenbein**  
Etude de l'impact d'un avion sur un bâtiment réacteur. *Rapport DEDR/DEMT 89.086*
- [572] **A. Barraco, B. Cuny, Ph. Jamet, A. Combescure, M. Lepareux, H. Bung**  
PLEXUS. Software for the numerical analysis of the dynamical behavior of rigid and flexible mechanisms. *Rapport DEDR/DEMT 89.307*
- [573] **H. Bung, R. Galon**  
Programme PLEXUS. Eléments coques multicouches. *Rapport DEDR/DEMT 89.370*
- [574] **A. Forestier, M. Billard, C. Lecomte**  
Etude de la détonation hydrogène. Comparaison pour des calculs plans et axisymétriques des effets de concentration en présence d'obstacles internes. *Rapport DEDR/DEMT 89.384*
- [575] **A. Forestier, M. Billard, C. Lecomte**  
Etude d'une détonation localisée dans une casemate. *Rapport DEDR/DEMT 89.407*
- [576] **J.P. Joli, D. Rolnik, M. Lepareux**  
Simulation dynamique de systèmes articulés asservis avec le programme PLEXUS. Application au bras manipulateur de la navette spatiale HERMES. *Rapport DEDR/DEMT 89.427*
- [577] **R. Galon**  
Nouvelles directives d'impression dans PLEXUS. *Rapport DEDR/DEMT 89.434*
- [578] **A. Forestier, H. Bung**  
Boundary conditions in a TVD scheme for Euler equation in a finite element formulation. *Rapport DEDR/DEMT 89.436*

**1990:**

- [579] **H. Makil, B. Schwab, M. Lepareux, A. Combescure**  
Hydro-mechanical analysis of a primary pipe, coupled to a reactor vessel, during a depressurization. *Rapport DRN/DMT 90.044*
- [580] **H. Bung, R. Galon**  
Modification du modèle Steinberg-Guinan. *Rapport DRN/DMT 90.061*
- [581] **R. Galon, H. Bung, M. Lepareux**  
Programme PLEXUS. Algorithme de traitement des surfaces de glissement (3D). *Rapport DRN/DMT 90.092*
- [582] **H. Bung, A. Forestier, M. Lepareux, B. Schwab**  
Programme PLEXUS. Fiches théoriques numéro 1. *Rapport DRN/DMT 90.094*
- [583] **M. Lepareux**  
Comportement des tuyauteries lors d'une dépressurisation consécutive à une rupture accidentelle. *Rapport DRN/DMT 90.154*
- [584] **H. Bung, A. Forestier, M. Lepareux, B. Schwab**  
Programme PLEXUS. Fiches théoriques numéro 3. *Rapport DRN/DMT 90.155*
- [585] **H. Bung, A. Forestier, M. Lepareux, B. Schwab**  
Programme PLEXUS. Fiches théoriques numéro 2. *Rapport DRN/DMT 90.159*
- [586] **H. Bung, M. Lepareux, R. Galon**  
Programme PLEXUS. Prise en compte du frottement dans les lignes et les surfaces de glissement. *Rapport DRN/DMT 90.203*
- [587] **D. Buttin, P. Damilot**  
Modélisation d'un véhicule utilitaire. *Rapport DRN/DMT 90.207*
- [588] **R. Le Tourmelin, L. Chelart**  
Programme PLEXUS. Etude d'une maquette fonctionnelle de bras manipulateur. *Rapport DRN/DMT 90.256*
- [589] **P. Grasset, B. Jurczynski**  
Programme PLEXUS. Modélisation des mécanismes asservis. *Rapport DRN/DMT 90.257*
- [590] **P. Chellapandi, M. Lepareux**  
Programme PLEXUS. Effects of the radial expansion for the fluid-structure couplings of pipes. *Rapport DRN/DMT 90.292*
- [591] **H. Bung, R. Galon**  
Programme PLEXUS. Matériaux orthotropes linéaires pour les éléments massifs. *Rapport DRN/DMT 90.301*
- [592] **A. Forestier**  
Traitement de la combustion dans un écoulement non visqueux. Cas de la réaction oxygène-hydrogène. *Rapport DRN/DMT 90.378*
- [593] **H. Bung, Ph. Boyce, M. Lepareux**  
Tenue des structures aux chocs (Entretiens Science et Défense 1990) *Rapport DRN/DMT 90.400*



- [594] **M. Lepareux, H. Bung, PH. Jamet**  
Programme PLEXUS. Dynamic behaviour of deformable and rigid articulated systems (SMIRT 10). *Rapport DRN/DMT 90.403*
- 1991:**
- [595] **A. Forestier**  
Application du code PLEXUS à l'étude d'un jet sur une géométrie 2D de puits de cuve. *Rapport DRN/DMT 91.002*
- [596] **P. Chellapandi, M. Lepareux**  
Programme PLEXUS . Some studies on momentum transport equations. *Rapport DRN/DMT 91.005*
- [597] **R. Galon, M. Lepareux, H.Bung**  
Programme PLEXUS. Remplissage d'un volume par des billes équidistantes. *Rapport DRN/DMT 91.007*
- [598] **H.Bung, A. Forestier, R. Galon, M. Lepareux**  
Programme PLEXUS. Examples manual. *Rapport DRN/DMT 91.011*
- [599] **M. Lepareux, J.F. Flobert**  
Programme PLEXUS. Matériau ADCJ : loi de comportement à trois constituants. *Rapport DRN/DMT 91.056*
- [600] **R. Galon**  
Programme PLEXUS. Sorties sur fichier de type "ALICE". *Rapport DRN/DMT 91.057*
- [601] **H.Bung, M. Lepareux, R. Galon**  
Programme PLEXUS. Controle anti-voilement pour les éléments CUBE (3D). *Rapport DRN/DMT 91.070*
- [602] **R. Galon, H.Bung**  
Programme PLEXUS. Sorties sur fichiers pour post-traitement par CASTEM 2000. *Rapport DRN/DMT 91.127*
- [603] **M. Lepareux**  
Programme PLEXUS. Comportement des structures sous impact (exposé GRECO du 16/05/1989). *Rapport DRN/DMT 91.134*
- [604] **M. Lepareux, A. Forestier, H.Bung**  
Programme PLEXUS. Note succinte de présentation. *Rapport DRN/DMT 91.144*
- [605] **H. Makil**  
Analyse des effets mécaniques d'une dépressurisation brutale dans un circuit de refroidissement d'un réacteur à neutrons rapides. *Rapport DRN/DMT 91.171*
- [606] **F. Casadei, A. Daneri, G. Toselli**  
Use of PLEXUS as a LMFBR primary containment code for the CONT benchmark problem. *Rapport DRN/DMT 91.176*
- [607] **M. Lepareux, R. Galon, J.F. Flobert**  
Programme PLEXUS. Fiches théoriques numéro 5. *Rapport DRN/DMT 91.216*

- [608] **M.S. Mougery, J.F. Flobert**  
Programme PLEXUS. Nouvelles tables de l'eau (0°C - 2000°C et 0 - 30 Kbars). *Rapport DRN/DMT 91.218*
- [609] **M. Billard, L. Luan, A. Forestier**  
Etude de la détonation hydrogène dans des géométries allemandes. *Rapport DRN/DMT 91.220*
- [610] **M. Lepareux, H. Bung**  
Traitement du contact billes-massifs *Rapport DRN/DMT 91.232*
- [611] **P. Damilot, D. Buttin**  
Programme PLEXUS. Modélisation et expérimentation de mécanismes flexibles. *Rapport DRN/DMT 91.233*
- [612] **R. Galon, H. Bung**  
Programme PLEXUS. Loi de comportement pour une modélisation par des billes. *Rapport DRN/DMT 91.255*
- [613] **A. Combescure, H. Bung, M. Lepareux, R. Galon, A. Forestier**  
Lois de comportement et modélisation des impacts. *Rapport DRN/DMT 91.292*
- [614] **A. Forestier**  
Van Leer and second order schemes. Application to gas dynamic. *Rapport DRN/DMT 91.456*
- 1992:**
- [615] **J.P. Magnaud, M. Lepareux**  
Premier modèle de comportement "corium" pour l'analyse de l'explosion vapeur. *Rapport DRN/DMT 92.082*
- [616] **A. Forestier**  
TVD scheme for an Euler formulation extension to a viscous or reactive fluid. *Rapport DRN/DMT 92.117*
- [617] **M. Lepareux et al.**  
Programme PLEXUS. Fiches théoriques N° 4. *Rapport DRN/DMT 92.299*
- [618] **L. Razafindrazaka**  
Analyse de l'essai Excobulle. *Rapport DRN/DMT 92.305*
- [619] **A. Forestier**  
Stable or unstable aspects in a combustion treatment : case of oxygen-hydrogen reaction. *Rapport DRN/DMT 92.342*
- [620] **N. Azouz**  
Analyse dynamique des systèmes souples poly-articulés. *Rapport DRN/DMT 92.344*
- [621] **P. Gonzales, A. Forestier**  
Implicit schemes for P-system equations via Roe's linearisation. *Rapport DRN/DMT 92.355*
- [622] **R. Galon, H. Bung, M. Lepareux**  
Programme PLEXUS. Matériau Puff pour les éléments "billes". *Rapport DRN/DMT 92.374*

- [623] **H. Bung, M. Matheron, A. Combescure, A. Aguilar, L. Luan**  
Experimental and numerical studies of impact on stainless steel plates submitted to rectangular flat nose missiles. *Rapport DRN/DMT 92.526*
- [624] **M. Pascal, N. Azouz, A. Combescure**  
Dynamic analysis of general flexible multibody systems. *Rapport DRN/DMT 92.559*
- [625] **P. Galon, M. Lepareux**  
Programme PLEXUS. Calcul du débit initial constant dans les tuyauteries avec prise en compte des pertes de charge localisées ou réparties. *Rapport DRN/DMT 92.657*
- 1993:**
- [626] **H. Bung, A. Forestier, M. Lepareux, P. Galon**  
Programme PLEXUS. Fiches de validation. *Rapport DRN/DMT 93.035*
- [627] **L. Razafindrazaka**  
Modèle bille de corium dans PLEXUS. *Rapport DRN/DMT 93.103*
- [628] **P. Galon, M. Lepareux, H. Bung**  
Programme PLEXUS. Gaz de Van der Waals et échanges thermiques (1D). *Rapport DRN/DMT 93.128*
- [629] **P. Galon, M. Lepareux, H. Bung**  
PLEXUS. Modèle d'endommagement pour les matériaux élastoplastiques. d'explosion vapeur. *Rapport DRN/DMT 93.278*
- [630] **C. Berriaud**  
Perforation des enceintes en béton par un projectile dur. Rapport de synthèse. *Rapport DRN/DMT 93.299*
- [631] **A. Letellier, A. Forestier**  
Le problème de Riemann en fluide quelconque. *Rapport DRN/DMT 93.451*
- [632] **A. Forestier, A. Caroli**  
Analysis of detonation in air-H<sub>2</sub>, by means of the Plexus code. *Rapport DRN/DMT 93.495*
- [633] **W. Breitung, R. Redlinger, A. Forestier**  
Report on Numerical Modeling on Hydrogen-Air Detonation. *Rapport DRN/DMT 93.496*
- [634] **A. Forestier**  
TVD scheme for an Euler formulation. Extension to a viscous or reactive fluid (version 2). *Rapport DRN/DMT 93.509*
- [635] **A. Forestier**  
Implicit methods for hyperbolic system. Application to P-system and gas dynamic. *Rapport DRN/DMT 93.510*
- [636] **P. di Giamberardino, H. Bung, M. Lepareux**  
Rigid missile impact on reinforced concrete slabs. Numerical analysis by means of a 3D local model. *Rapport DRN/DMT 93.589*
- [637] **P. Galon, M. Lepareux, H. Bung**  
A new method for the treatment of impact and penetration problems. *Rapport DRN/DMT 93.597*

- [638] **P. di Giamberardino, H. Bung, M. Lepareux, Ph. Matheron**  
Rigid missile impact on reinforced concrete slabs. Experimental results and numerical analyses by means of a global model. *Rapport DRN/DMT 93.611*
- [639] **F. Corsi, P. di Giamberardino, H. Bung, M. Lepareux, Ph. Matheron**  
Experimental tests and numerical calculations of reinforced concrete slabs subjected to rigid missile impact. *Rapport DRN/DMT 93.629*
- [640] **A. Forestier**  
Un schéma aux volumes finis pour les écoulements diphasiques compressibles. *Rapport DRN/DMT 93.631*

**1994:**

- [641] **P.Galon, M.Lepareux**  
Programme PLEXUS. Calcul de l'état initial en équilibre de solides et de fluides soumis à la pesanteur. *Rapport DRN/DMT 94.003*
- [642] **P.Galon, A.Forestier**  
PLEXUS. Schéma de Godunov et de Van Leer pour la résolution des équations d'Euler 3D. *Rapport DRN/DMT 94.054*
- [643] **P.Joli**  
Modélisation et simulation de systèmes à topologie variable due à des liaisons de contact. *Rapport DRN/DMT 94.058*
- [644] **P.Galon, M.Lepareux, H.Bung**  
Impact d'un projectile formé de billes sur une plaque déformable. *Rapport DRN/DMT 94.081*
- [645] **M.Lepareux, D.Thiault, J.M. Michelin**  
PLEXUS - R : Une extension de PLEXUS à la robotique. *Rapport DRN/DMT 94.138*
- [646] **V. Hairabetian, M. Lepareux**  
Calcul d'impact d'une tuyauterie. *Rapport DRN/DMT 94.141*
- [647] **P.di Giamberardino, M.Lepareux**  
Vapour loop - Fluid-structure dynamic analysis of partial breaks in pipelines. *Rapport DRN/DMT 94.177*
- [648] **N.Azouz**  
Modélisation des structures souples poly-articulées - Application à la simulation des robots. *Rapport DRN/DMT 94.178*
- [649] **A. Combescure**  
Grands logiciels de mécanique. *Rapport DRN/DMT 94.223*
- [650] **M. Lepareux**  
Programme PLEXUS. Matériau "EAU". Modèle homogène équilibré. *Rapport DRN/DMT 94.398*
- [651] **P. Galon-Bruère, M. Lepareux**  
Etude de propagations d'ondes en milieu non confiné sans obstacle. *Rapport DRN/DMT 94.484*

- [652] **M. F. Robbe, M. Lepareux, H. Bung**  
Programme PLEXUS. Notice théorique. *Rapport DRN/DMT 94.490*
- [653] **A. Forestier, J.M. Hérard, X. Louis**  
Un système non strictement hyperbolique décrivant une modélisation de turbulence compressible. *Rapport DRN/DMT 94.554*
- 1995:**
- [654] **P.Galon, H. Bung, M.Lepareux**  
Programme PLEXUS. Matériau linéaire dans le modèle "billes". *Rapport DRN/DMT 95.002*
- [655] **P. Galon-Bruère, M. Lepareux, L. Razafindrazaka**  
Etude d'une détonation hydrogène dans une cuve de stockage. *Rapport DRN/DMT 95.020*
- [656] **P.Galon, H. Bung**  
Programme PLEXUS. Interface entre modèle "billes" et éléments finis massifs. *Rapport DRN/DMT 95.043*
- [657] **J. Kichenin**  
Utilisation des codes TRIO-EF et PLEXUS en vibrations couplées aux écoulements. *Rapport DRN/DMT 95.091*
- [658] **D. Moulin, T. Yuritzinn, G. Sert**  
Crack initiation in transportation cask subjected to impacts loads - Experimentation and calculations. *Rapport DRN/DMT 95.169*
- [659] **M.F. Robbe**  
Analyse de la tenue mécanique d'une cuve de REP en cas d'explosion vapeur. Modèle EAU métastable. *Rapport DRN/DMT 95.184*
- [660] **L. Razafindrazaka**  
Programme PLEXUS. Liste des documents. *Rapport DRN/DMT 95.231*
- [661] **F. Bliard, M. Lepareux**  
Programme PLEXUS. Modélisation d'un composant de type "POMPE". *Rapport DRN/DMT 95.269*
- [662] **C. Caroli, P. Galon, A. Forestier**  
Improvement of the H2 detonation model in Plexus and verification against experimental results. *Rapport DRN/DMT 95.364*
- [663] **P. Galon, H. Bung**  
Programme PLEXUS. Couplage des modèles d'endommagement P/Y et Chaboche-Lemaitre avec des lois de comportement Von Mises dynamique et Steinberg-Guinan. *Rapport DRN/DMT 95.399*
- [664] **P. Galon**  
PLEXUS. Comparaison des schémas de Godunov et de Van Leer 2D et 3D non réactif. Etude du tube à choc. *Rapport DRN/DMT 95.412*
- [665] **P. Galon, M. Lepareux, W. Breitung, H. Massier, M. Moschke, R. Redlinger, A. Vesper**  
Hydrogen-air detonation experiments in complex geometry and their analysis with different codes. *Rapport DRN/DMT 95.515*

- [666] **P. Galon, L. Razafindrazaka**  
Propagations tridimensionnelles d'ondes en milieu non confiné sans obstacle. *Rapport DRN/DMT 95.528*
- [667] **P. Galon, L. Razafindrazaka**  
Détonation hydrogène-air. Comparaisons calculs/essais cylindre tronqué. *Rapport DRN/DMT 95.608*
- [668] **H. Bung.** *Plexus : un logiciel dynamique pour les chocs et impacts*. Papier présenté au seminaire phi2as IPSI le 21 mars 1995

**1996:**

- [669] **P. Galon, M. Lepareux, W. Breitung, H. Redlinger et al.**  
Hydrogen-air detonation. Final report. Detonation subgroup. *Rapport DRN/DMT 96.057*
- [670] **P. Galon, L. Razafindrazaka**  
Detonation Hydrogène-air dans un tube. Comparaisons calculs/expériences pour des configurations géométriques avec obstacles. *Rapport DRN/DMT 96.061*
- [671] **X. Louis**  
Modélisation numérique de la turbulence compressible. *Rapport DRN/DMT 96.067*
- [672] **M.F. Robbe, M. Lepareux**  
Programme PLEXUS. Matériau "EAU". Modèle homogène non-équilibré. *Rapport DRN/DMT 96.142*
- [673] **L. Razafindrazaka, P. Galon**  
Détonation hydrogène-air - Comparaisons calculs/essais pour une cible conique. *Rapport DRN/DMT 96.152*
- [674] **L. Razafindrazaka, P. Galon**  
Détonation hydrogène-air - Comparaisons calculs/essais pour une cible cylindrique. *Rapport DRN/DMT 96.187*
- [675] **L. Benchikh**  
PLEXUS Robotique. Conception généralisée des systèmes mécatroniques complexes. *Rapport DRN/DMT 96.196*
- [676] **M. Lepareux**  
Analyse simplifiée de la tenue d'un fond de cuve a une explosion vapeur. *Rapport DRN/DMT 96.210*
- [677] **H. Bung, M. Lepareux**  
Procédure GIBIANE de calcul dynamique explicite. I - Cas linéaire. *Rapport DRN/DMT 96.294*
- [678] **J. Bonini, H. Bung, M. Lepareux**  
Programme PLEXUS - Développement d'un algorithme de traitement des impacts par la méthode des multiplicateurs de Lagrange. *Rapport DRN/DMT 96.296*
- [679] **M. Lepareux**  
Explosion vapeur en fond de cuve. Influence de la propagation et de la souplesse des parois. *Rapport DRN/DMT 96.304*

- [680] **M.F. Robbe**  
Programme PLEXUS. Etude théorique sur la modélisation des composants internes dans un ADC (modèle de porosité). *Rapport DRN/DMT 96.309*
- [681] **M.F. Robbe**  
Programme PLEXUS. Réflexion pour améliorer la documentation relative à PLEXUS. *Rapport DRN/DMT 96.310*
- [682] **F. Bliard, H. Bung**  
Programme PLEXUS. Introduction d'un modèle Symonds et d'un modèle libre dans le matériau Von Mises Dynamique. *Rapport DRN/DMT 96.341*
- [683] **H. Bung**  
Programme PLEXUS. Algorithme de traitement des auto-contacts. *Rapport DRN/DMT 96.351*
- [684] **A. de Gayffier**  
Les lois de comportement des matériaux solides en grandes déformations dans Castem 2000 et Plexus. *Rapport DRN/DMT 96.359*
- [685] **A. Letellier**  
Contribution à la modélisation des impacts d'oiseaux sur les aubes des réacteurs d'avions. *Rapport DRN/DMT 96.447*
- [686] **A. de Gayffier**  
Etude des modèles rhéologiques du béton disponibles dans PLEXUS et CASTEM 2000. *Rapport DRN/DMT 96.449*
- [687] **M. Lepareux, A. de Gayffier, H. Bung, A. Letellier**  
Réflexions sur les méthodes de la dynamique explicite dans Plexus et Castem 2000. *Rapport DRN/DMT 96.473*
- [688] **M. Lepareux**  
Explosion vapeur en cuve. Réponse dynamique de la structure. Rappel des ordres de grandeur. *Rapport DRN/DMT 96.567*
- [689] **E. Seinturier, M.F. Robbe**  
Programme PLEXUS - Fiche de validation CIR-03 sur les tuyaux. *Rapport DRN/DMT 96.616*
- [690] **F. Bliard, M.F. Robbe**  
Programme PLEXUS. Introduction d'un modèle de porosité dans le matériau ADCR. *Rapport DRN/DMT 96.620*
- 1997:**
- [691] **A. de Gayffier, H. Bung, J. Bonini, M. Lepareux**  
Dynamique explicite dans CASTEM 2000 - Spécification des développements. *Rapport DRN/DMT 97.027*
- [692] **M. Lepareux**  
Programme PLEXUS - Les modèles de débit critique. *Rapport DRN/DMT 97.029*
- [693] **H. Bung, M. Lepareux, A. de Gayffier**  
Procédure GIBIANE de calcul dynamique explicite :  
II - Cas non-linéaires. *Rapport DRN/DMT 97.048*

- [694] **M. Lepareux**  
Programme PLEXUS - Traitement des plaques perforées. *Rapport DRN/DMT 97.085*
- [695] **C. Strub, T. Grunenwald, D. Badellon**  
Simulation numérique d'essais d'écrasement de cylindres impactés. *Rapport DRN/DMT 97.294*
- [696] **T. Yuritzinn**  
Analyse exploratoire du comportement d'une maquette d'emballage de transport sous impacts. *Rapport DRN/DMT 97.236*
- [697] **A. de Gayffier, Y. Nuttinck**  
Méthode de décomposition de domaines par multiplicateurs de Lagrange pour la dynamique des structures. Application à la simulation de l'ébranlement d'un bâtiment réacteur soumis à l'impact d'un avion. *Rapport DRN/DMT 97.344*
- [698] **E. Vallejo, M. Lepareux**  
APRP structures fixes - Influence de la finesse du maillage 3D de la cuve. *Rapport DRN/DMT 97.354*
- [699] **L. Doveil, O. Bonnard**  
Comparaison de résultats PLEXUS et DREXUS sur le calcul des poutres en statique. *Rapport DRN/DMT 97.363*
- [700] **C. Strub**  
Analysis of SLUG IMPACT against the reactor pressure vessel head - Interpretation of BERDA test 07 with PLEXUS using the method of particles and forces for the slug discretization. *Rapport DRN/DMT 97.364*
- [701] **J. Bonini, H. Bung**  
Programme PLEXUS - Fiche Théorique - Modélisation des impacts avec frottements par la méthode des multiplicateurs de Lagrange. *Rapport DRN/DMT 97.367*
- [702] **J. Bonini**  
Programme PLEXUS - Fiches de validation - Modélisation des impacts avec frottements par la méthode des multiplicateurs de Lagrange. *Rapport DRN/DMT 97.368*
- [703] **M.F. Robbe**  
Programme PLEXUS - Fiche de validation "CIR-CADRE". *Rapport DRN/DMT 97.378*
- [704] **M.F. Robbe, O. Bonnard**  
Programme PLEXUS - Fiches de validation des poutres en statique. *Rapport DRN/DMT 97.594*
- [705] **M.F. Robbe, O. Bonnard**  
Programme PLEXUS - Fiche de validation des poutres (poids propre). *Rapport DRN/DMT 97.595*
- [706] **M.F. Robbe, M. Lepareux, N. Vivien, G. Cenerino**  
Screening calculations on the vessel lower head behaviour due to an in-vessel steam explosion. *Actes 14<sup>e</sup> Int. Conf. on Structural Mechanics In Reactor Technology, (SMIRT 14) article PW/9, Lyon, France (17-22 août 1997).*
- [707] **M.F. Robbe, N. Vivien, M. Valette**  
A steam explosion assessment by thermalhydraulic and mechanical linked computations. *Actes 14<sup>e</sup> Int. Conf. on Structural Mechanics In Reactor Technology, (SMIRT 14) article P02/4, Lyon, France (17-22 août 1997).*



**1998:**

- [708] **P. Galon, E. Studer**  
Hydrogen Combustion Loads - PLEXUS calculations. *Publication DRN/DMT/SEMT/TTMF/ PU/98.01/A*
- [709] **M.F. Robbe, O. Bonnard**  
Programme PLEXUS - Fiches de validation : poutres en statique. *Rapport DRN/DMT/SEMT/DYN/ RT/98.001/A*
- [710] **M. Lepareux, F. Bliard**  
Programme PLEXUS - Appuis non linéaires. *Rapport DRN/DMT/SEMT/DYN/ RT/98.006/A*
- [711] **C. Strub**  
Introduction dans PLEXUS d'un modèle de comportement de matériaux élasto-plastiques endommageables. *Rapport DRN/DMT/SEMT/DYN/ RT/98.026/A*
- [712] **M. Lepareux, E. Treille**  
Qualification d'une loi de comportement de type ADC avec porosité. *Rapport DRN/DMT/SEMT/DYN/ RT/98.032/A*
- [713] **F. Luzeau, C. Strub**  
Etude du comportement du couvercle d'une cuve REP impactée par du corium - Analyse de l'essai BERDA 7 avec la méthode particulière SPH de PLEXUS. *Rapport DRN/DMT/SEMT/DYN/ RT/98.034/A*
- [714] **M. Merelo, C. Strub**  
Développement dans PLEXUS d'un modèle de comportement plastique couplé à l'endommagement sensible à la vitesse de déformation. *Rapport DRN/DMT/SEMT/DYN/ RT/98.036/A*
- [715] **A. Legay, D. Leray**  
PLEXUS : Introduction de la viscosité dans la méthode SPH. *Rapport DRN/DMT/SEMT/DYN/ RT/98.040/A*
- [716] **R. Zecchiero**  
Qualification du code PLEXUS envers les effets mécaniques d'une réaction sodium-eau. *Note Technique DRN/DEC/SECA/LECC/ NT/98.030*
- [717] **E. Treille, M. Lepareux**  
Programme PLEXUS - Sortie sur fichier au format universel. *Rapport DRN/DMT/SEMT/DYN/ RT/98.051/A*
- [718] **E. Treille, M. Lepareux**  
Programme PLEXUS - Post-traitement par MATLAB. *Rapport DRN/DMT/SEMT/DYN/ RT/98.052/A*
- [719] **M.F. Robbe**  
Aide au dimensionnement de VULCANO en cas d'explosion de vapeur - Calculs 1D avec PLEXUS. *Rapport DRN/DMT/SEMT/DYN/ RT/98.054/A*
- [720] **M.F. Robbe, M. Lepareux.**  
PLEXUS - Fiches de validation "CIR-RETRECIS-RAMP" et "CIR-RETRECIS-TRIA". *Rapport DRN/DMT/SEMT/DYN/ RT/98.060/A*

- [721] **M.F. Robbe, M. Lepareux, C. Trollat**  
Hydrodynamic loads on a PWR primary circuit due to a LOCA - Pipe computations with the CASTEM-PLEXUS code. *Rapport DRN/DMT/SEMT/DYN/ RT/98.061/A*
- [722] **M.F. Robbe, N. Vivien, M. Valette.**  
A steam explosion assessment by thermohydraulic and mechanical linked computations. *Rapport DRN/DMT/SEMT/DYN/ RT/98.062/A*
- [723] **M.F. Robbe, M. Lepareux, N. Vivien, G. Cenerino.**  
Screening calculations on the vessel lower head behaviour due to an in-vessel steam explosion. *Rapport DRN/DMT/SEMT/DYN/ RT/98.063/A*
- [724] **F. Bliard, M. Lepareux**  
Support de cours pour la formation à l'usage de PLEXUS - Thème : calcul de tuyauteries. *Rapport DRN/DMT/SEMT/DYN/ RT/98.066/A*
- [725] **T. Yuritzinn**  
Procédure d'identification des paramètres du modèle d'endommagement en statique et en dynamique. *Rapport DRN/DMT/SEMT/LISN/ RT/98.067/A*
- [726] **T. Laporte**  
Risques d'éjection ou de rupture du couvercle en cas d'explosion vapeur en cuve : Analyse des essais BERDA, estimation d'ordre de grandeur. *Rapport DRN/DMT/SEMT/LESY/ RT/98.035/A*
- 1999:**
- [727] **M.F. Robbe**  
PLEXUS - Fiche de qualification : "CIR-PINCE-SECTION". *Rapport DRN/DMT/SEMT/DYN/ RT/99.001/A*
- [728] **M.F. Robbe**  
A porosity method to model the internal structures of a reactor vessel. *Rapport DRN/DMT/SEMT/DYN/ RT/99.003/A*
- [729] **M.F. Robbe, F. Bliard**  
A porosity model to represent the influence of structures on a fluid flow. Application to a Hypothetical Core Disruptive Accident. method to model the internal structures of a reactor vessel. *Rapport DRN/DMT/SEMT/DYN/ RT/99.005/A*
- [730] **M.F. Robbe**  
Plexus - Fiche de qualification : "CIR-DIAPHRAGME". *Rapport DRN/DMT/SEMT/DYN/ RT/99.010/A*
- [731] **G. Marchaud, M. Lepareux**  
Plexus - Fiche de qualification des modèles de débit critique : "CIR-DCRI". *Rapport DRN/DMT/SEMT/DYN/ RT/99.013/A*
- [732] **P. Galon, C. Strub**  
Modélisation de l'essai mini-Charpy : Comparaison des calculs dynamiques (PLEXUS) et "quasi-statiques" (CASTEM 2000). *Rapport DRN/DMT/SEMT/DYN/ RT/99.019/A*
- [733] **P. Galon, M. Lepareux**  
Comportement dynamique des tuyauteries - Analyse d'un essai de fouettement au moyen de Plexus. *Rapport DRN/DMT/SEMT/DYN/ RT/99.028/A*

- [734] **M.F. Robbe, F. Bliard**  
A porosity model to represent the influence of structures on a fluid flow. Application to a hypothetical core disruptive accident. *Actes 7<sup>e</sup> Int. Conf. On Nuclear Engineering, (ICONE 7) article 7819, Tokyo, Japan (19-23 avril 1999).*
- [735] **M.F. Robbe**  
A porosity method to model the internal structures of a reactor vessel. *Actes 15<sup>e</sup> Int. Conf. on Structural Mechanics In Reactor Technology, (SMIRT 15) article B03/2, Seoul, Corea (15-20 août 1999).*
- [736] **M.F. Robbe, M. Lepareux, C. Trollat**  
Hydrodynamic loads on a PWR primary circuit due to a LOCA. Pipe computations with the CASTEM-PLEXUS code. *Actes 15<sup>e</sup> Int. Conf. on Structural Mechanics In Reactor Technology, (SMIRT 15) article J05/4, Seoul, Corea (15-20 août 1999).*
- [737] **M.F. Robbe, P. Galon, T. Yuritzinn**  
CASTEM-PLEXUS: Un logiciel de dynamique rapide pour évaluer l'intégrité des structures en cas d'accident. *Actes 4<sup>e</sup> Conf. INSTRUC, Prévision et estimation de la durée de vie des structures mécaniques, pp 224-233, Courbevoie, France (23-24 novembre 1999).*

**2000:**

- [738] **P. Galon**  
Sur les limites de validité de la méthode particulière SPH. *Rapport DRN/DMT/SEMT/DYN/ RT/00.008/A*
- [739] **H. Bung**  
PLEXUS : atelier logiciel KSTM. *Rapport DRN/DMT/SEMT/DYN/ RT/00.010/A*
- [740] **T. Yuritzinn, P. Galon, T. Grunenwald, P. Izquierdo, A. Le Maoult, M.P. Valeta**  
Impact resistance of 304L type stainless steel plates : identification of Lemaitre and Chaboche damage model's parameters at different strain rates. Comparison with experimental results. *Rapport DRN/DMT/SEMT/LISN/ PU/00.012/A - 6ième DYMAT, Cracovie, 25-29 Septembre 2000.*
- [741] **P. Galon, M. Lepareux**  
Couplage 1D/3D : analyse d'un essai de fouetttement au moyen d'EUROPLEXUS. *Rapport DRN/DMT/SEMT/DYN/ RT/00.015/A*
- [742] **P. Galon, P. Izquierdo**  
Modélisation de la perforation de plaques en acier inoxydable 304L avec le modèle d'endommagement de Lemaitre et Chaboche : comparaisons avec les résultats expérimentaux. *Rapport DRN/DMT/SEMT/DYN/ RT/00.018/A*
- [743] **P. Galon**  
Calculs dynamiques d'un essai Charpy miniature avec prise en compte ou non du frottement. *Rapport DRN/DMT/SEMT/DYN/ RT/00.024/A*
- [744] **M. Lepareux**  
EUROPLEXUS - Fiche de qualification : CIR-GRIL-PCHA. *Rapport DRN/DMT/SEMT/DYN/ RT/00.026/A*

- [745] **M. Lepareux**  
EUROPLEXUS - Fiche de qualification : "CIR-TCHOC-GP". *Rapport DRN/DMT/SEMT/DYN/ RT/00.029/A*
- [746] **M. Lepareux**  
Explosions des mélanges de fluides - Notions de base et application à EUROPLEXUS. *Rapport DRN/DMT/SEMT/DYN/ RT/00.031/A - Cours IPSI, 17-19 octobre 2000, Paris*
- [747] **M. Lepareux**  
EUROPLEXUS - Traitement des piquages en 3D. *Rapport DRN/DMT/SEMT/DYN/ RT/00.033/A*
- [748] **H. Bung, F. Casadei, M. Lepareux, J.P. Halleux**  
Atelier logiciel de EUROPLEXUS. *Rapport DRN/DMT/SEMT/DYN/ RT/00.036/A*
- [749] **P. Galon, M. Lepareux, H. Bung, T. Grunenwald**  
Simulations of slug impact with smoothed particle hydrodynamics, some remarks on frequently asked questions. *Rapport DRN/DMT/SEMT/DYN/ PU/00.036 - 6ième DYMAT, Cracovie, 25-29 Septembre 2000.*
- [750] **H. Bung**  
EUROPLEXUS : etude bibliographique sur les modèles de béton en dynamique rapide. *Rapport DRN/DMT/SEMT/DYN/ RT/00.044/A*
- [751] **M. F. Robbe, S. V. Potapov**  
A pipe-model to assess the hydrodynamic effects of a blowdown in a 4-loop PWR. *Actes 8<sup>e</sup> Int. Conf. On Nuclear Engineering (ICONE 8), article 8156, Baltimore, USA (2-6 avril 2000).*
- [752] **M. F. Robbe, M. Lepareux, C. Trollat**  
Assessment of the hydrodynamic loads due to a LOCA in a 3-loop PWR. CASTEM-PLEXUS computations. *Actes 8<sup>e</sup> Int. Conf. On Nuclear Engineering (ICONE 8), article 8199, Baltimore, USA (2-6 avril 2000).*
- [753] **M. F. Robbe, M. Lepareux**  
Scoping calculations of an in-vessel steam explosion. CASTEM-PLEXUS computations. *Actes 8<sup>e</sup> Int. Conf. On Nuclear Engineering (ICONE 8), article 8203, Baltimore, USA (2-6 avril 2000).*
- [754] **M. F. Robbe**  
Bibliographic synthesis of the experimental and numerical German and French research programs concerning slug impact. *Actes 8<sup>e</sup> Int. Conf. On Nuclear Engineering (ICONE 8), article 8751, Baltimore, USA (2-6 avril 2000).*
- [755] **M. F. Robbe, Y. Cariou, M. Lepareux, E. Treille**  
Validation of the Core Disruptive Accident constitutive law of the Castem-Plexus code on the Mara8 test. *International colloquium in mechanics of solids, fluids, structures and interactions, pp 735-744, Nha Trang, Vietnam (14-18 août 2000).*
- [756] **M. F. Robbe, Y. Cariou, E. Treille, M. Lepareux**  
Qualification of the numerical simulation of a Hypothetical Core Disruptive Accident on the MARA10 test-facility. *International colloquium in mechanics of solids, fluids, structures and interactions, pp 745-755, Nha Trang, Vietnam (14-18 août 2000).*

- [757] **M. F. Robbe, Y. Cariou, M. Lepareux, E. Treille**  
Description of the MARS small scale replica of a Fast Breeder Reactor and of the numerical models used to simulate a Hypothetical Core Disruptive Accident in the test-facility. *International colloquium in mechanics of solids, fluids, structures and interactions*, pp 725-734, Nha Trang, Vietnam (14-18 août 2000).
- [758] **M. F. Robbe**  
Bibliographic review of the computer codes and experiments dealing with steam explosion in Europe. *Actes European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS)*, Barcelone, Espagne (11-14 septembre 2000).
- [759] **S. V. Potapov, M. F. Robbe, F. Tephany**  
Hydrodynamic consequences of a LOCA in a 4-loop PWR. *Actes European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS)*, Barcelone, Espagne (11-14 septembre 2000).
- [760] **M. F. Robbe, Y. Cariou, M. Lepareux, E. Treille**  
Numerical simulation of a Hypothetical Core Disruptive Accident in the MARA8 mock-up with the CASTEM-PLEXUS computer code. *Actes European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS)*, Barcelone, Espagne (11-14 septembre 2000).
- [761] **M. F. Robbe, Y. Cariou, E. Treille, M. Lepareux**  
Numerical interpretation of the MARA10 test simulating a Hypothetical Core Disruptive Accident in a mock-up schematizing a reactor. *Actes European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS)*, Barcelone, Espagne (11-14 septembre 2000).
- [762] **M. F. Robbe, M. Lepareux, C. Trollat**  
Assessment of the hydrodynamic loads due to a LOCA in a 3-loop PWR - Pipe computations with the code CASTEM-PLEXUS. *Actes International Conference on Numerical Methods in Continuum Mechanics 2000*, Liptovsky Jan, Slovaquie (19-24 septembre 2000).
- [763] **M. F. Robbe, M. Lepareux**  
Estimation of the mechanical consequences of an in-vessel steam explosion by means of parametric computations. *Actes International Conference on Numerical Methods in Continuum Mechanics 2000*, Liptovsky Jan, Slovaquie (19-24 septembre 2000).
- [764] **M. F. Robbe, N. Vivien, M. Valette, E. Berglas**  
Estimation of the mechanical consequences of a steam explosion with a local approach - Thermohydraulic and mechanical linked computations. *Actes International Conference on Numerical Methods in Continuum Mechanics 2000*, Liptovsky Jan, Slovaquie (19-24 septembre 2000).
- [765] **M. F. Robbe, Y. Cariou, M. Lepareux, E. Treille**  
Numerical simulation of a Hypothetical Core Disruptive Accident in the MARA8 mock-up with the CASTEM-PLEXUS computer code. *Actes International Conference on Numerical Methods in Continuum Mechanics 2000*, Liptovsky Jan, Slovaquie (19-24 septembre 2000).
- [766] **M. F. Robbe, Y. Cariou, E. Treille, M. Lepareux**  
Numerical interpretation of the MARA10 test simulating a Hypothetical Core Disruptive Accident in a mock-up schematizing a reactor. *Actes International Conference on Numerical Methods in Continuum Mechanics 2000*, Liptovsky Jan, Slovaquie (19-24 septembre 2000).

- [767] **M. F. Robbe, Y. Cariou, M. Lepareux, E. Treille**  
Computation of the MARS test simulating a Hypothetical Core Disruptive Accident in a small scale replica of a Fast Breeder Reactor. *Actes International Conference on Numerical Methods in Continuum Mechanics 2000, Liptovsky Jan, Slovaquie (19-24 septembre 2000)*.
- [768] **M. F. Robbe, Y. Cariou, M. Lepareux, E. Treille**  
Comparison of a Core Disruptive Accident simulation with the MARS experimental test. *Actes International Conference on Numerical Methods in Continuum Mechanics 2000, Liptovsky Jan, Slovaquie (19-24 septembre 2000)*.
- 2001:**
- [769] **P. Galon**  
Exemple détaillé de la modélisation et de la mise en données d'un problème complexe : Analyse d'un essai de fouettement avec Couplage 1D/3D. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.001/A*
- [770] **P. Galon, T. Tanguy**  
Modélisation de l'écrasement d'un cylindre troué avec EUROPLEXUS. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.002/A*
- [771] **P. Galon, P. Izquierdo**  
Simulation de l'écrasement de cylindres troués : comparaisons calculs/expériences. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.003/A*
- [772] **M. Lepareux**  
EUROPLEXUS : fiche de qualification FLU\_123D\_ABSO. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.007/A*
- [773] **M. Lepareux**  
EUROPLEXUS : fiche de qualification CIR\_COUDE. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.008/A*
- [774] **M. Lepareux**  
EUROPLEXUS : fiche de qualification CIR\_TE\_DISSYM. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.010/A*
- [775] **P. Galon**  
Mesure des champs de déformations 3D sur une coque trouée en acier inoxydable 304L. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.012/A*
- [776] **P. Galon, P. Izquierdo, P. Ruffet**  
Essais d'écrasement de cylindres troués en acier inoxydable 304L. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.013/A*
- [777] **M. Lepareux**  
EUROPLEXUS : fiche de qualification tube à choc en eau CIR\_TCHOC\_EAU. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.018/A*
- [778] **P. Galon, M. Lepareux**  
Documentation théorique : Modélisation des tuyauteries dans EUROPLEXUS. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.021/A*

- [779] **H. Bung**  
EUROPLEXUS : loi de comportement du bois sous impact, modèle 2D et axisymétrique. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.022/A*
- [780] **M. Lepareux**  
EUROPLEXUS : fiche de qualification - Modèle de pompe : CIR\_ANNEAU\_POMPE. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.023/A*
- [781] **M. Lepareux**  
EUROPLEXUS : Compléments de validation du modèle de réaction sodium-eau. *Rapport DEN/DM2S/SEMT/DYN/ RT/01.029/A*
- [782] **M.F. Robbe, M. Lepareux, E. Treille, Y. Cariou**  
Numerical simulation of a Core Disruptive Accident and comparison with the simple test Mara8. *Actes ANS/HPS Student Conference "2001: a nuclear odyssey", Texas A&M University, USA (29 mars - 1er avril 2001).*
- [783] **M.F. Robbe, E. Treille, Y. Cariou, M. Lepareux**  
Simulation of a Core Disruptive Accident in the scale model Mara10 schematizing a reactor. *Actes ANS/HPS Student Conference "2001: a nuclear odyssey", Texas A&M University, USA (29 mars - 1er avril 2001).*
- [784] **M.F. Robbe, Y. Cariou, M. Lepareux, E. Treille**  
Numerical simulation of a Core Disruptive Accident in a scale model of a LMFBFR *Actes ANS/HPS Student Conference "2001: a nuclear odyssey", Texas A&M University, USA (29 mars - 1er avril 2001).*
- [785] **M.F. Robbe, S. Potapov, F. Tephany**  
Simulation of the blowdown induced by a LOCA in a 4-loop PWR *Actes ANS/HPS Student Conference "2001: a nuclear odyssey", Texas A&M University, USA (29 mars - 1er avril 2001).*
- [786] **M.F. Robbe, S. Potapov, F. Tephany**  
Modeling of the hydraulic flows in a 4-loop PWR with a pipe-geometry. *Actes 9<sup>e</sup> Int. Conf. On Nuclear Engineering, (ICONE 9) article 9864, Nice, France (9-12 avril 2001).*
- [787] **M.F. Robbe, S. Potapov, F. Tephany**  
Estimation of the hydrodynamic effects of a LOCA in a 4-loop PWR. *Actes 9<sup>e</sup> Int. Conf. On Nuclear Engineering, (ICONE 9) article 9682, Nice, France (9-12 avril 2001).*
- [788] **M.F. Robbe, F. Bliard**  
Homogenization of the internal structures of a reactor with the cooling fluid. *Actes 9<sup>e</sup> Int. Conf. On Nuclear Engineering, (ICONE 9) article 9681, Nice, France (9-12 avril 2001).*
- [789] **M.F. Robbe, Y. Cariou**  
Modeling of the MARS mock up simulating a Core Disruptive Accident in a Fast Breeder Reactor. *Actes 9<sup>e</sup> Int. Conf. On Nuclear Engineering, (ICONE 9) article 9685, Nice, France (9-12 avril 2001).*
- [790] **M.F. Robbe, M. Lepareux**  
Simulation of a Hypothetical Core Disruptive Accident in the MARS test-facility. *Actes 9<sup>e</sup> Int. Conf. On Nuclear Engineering, (ICONE 9) article 9678, Nice, France (9-12 avril 2001).*

- [791] **M.F. Robbe, E. Treille, M. Lepareux**  
Estimation of the mechanical effects of a Core Disruptive Accident in a LMFBR. *Actes 9<sup>e</sup> Int. Conf. On Nuclear Engineering, (ICONE 9) article 9699, Nice, France (9-12 avril 2001).*
- [792] **M.F. Robbe, Y. Cariou, E. Treille, M. Lepareux**  
Qualification of the numerical simulation of a Core Disruptive Accident in the MARS mock-up. *Actes 9<sup>e</sup> Int. Conf. On Nuclear Engineering, (ICONE 9) article 9686, Nice, France (9-12 avril 2001).*
- 2002:**
- [793] **P. Galon**  
EUROPLEXUS : Fiche de qualification TUBE\_PISTON. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.002/A*
- [794] **M. Lepareux**  
EUROPLEXUS : Fiche de qualification CIR\_RACCORD\_1D3D. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.005/A*
- [795] **N. Bousquet, H. Bung, A. Combescure, M. Lepareux, S. Potapov**  
EUROPLEXUS : Une méthode de décomposition de domaines en dynamique explicite. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.007/A*
- [796] **P. Galon**  
EUROPLEXUS : fiche de qualification ESSAI\_MARA\_02. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.008/A*
- [797] **P. Galon**  
EUROPLEXUS : fiche de qualification - Réponse d'un tuyau à paroi mince à un pic de pression. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.009/A*
- [798] **X. Delaune**  
Validation du matériau BOIS axisymétrique dans EUROPLEXUS. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.012/A*
- [799] **P. Galon**  
EUROPLEXUS - Fiche de qualification : Simulation de l'écrasement de cylindres troués en acier inoxydable 304L - Comparaisons calculs - expériences. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.013/A*
- [800] **P. Galon**  
EUROPLEXUS - Fiche de qualification : Comparaisons des coques 3D. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.020/A*
- [801] **P. Galon**  
EUROPLEXUS - Fiche de qualification : Couplages coques-massifs en 3D. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.023/A*
- [802] **M. Lepareux**  
EUROPLEXUS - Fiche de qualification : Plaque perforée avec IFS. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.027/A*
- [803] **X. Delaune**  
Modélisation simplifiée du contact coque/bois dans les capots de protection des emballages de transport. *Rapport DEN/DM2S/SEMT/DYN/ RT/02.030/A*



**2003:**

- [804] **M. Lepareux**  
EUROPLEXUS - Modélisation des appuis au moyen du matériau "SUPPORT". *Rapport DEN/DM2S/ SEMT/DYN/ RT/03.002/A*
- [805] **M. Lepareux**  
EUROPLEXUS - Modélisation des explosions en milieu liquide : matériaux "ADCR" et "ADCJ". *Rapport DEN/DM2S/ SEMT/DYN/ RT/03.004/A*
- [806] **P. Galon, F.Bliard**  
EUROPLEXUS - Fiche de qualification : Comportement des supports. *Rapport DEN/DM2S/ SEMT/DYN/ RT/03.006/A*
- [807] **M. Lepareux**  
EUROPLEXUS - Modèle de porosité monophasique. *Rapport DEN/DM2S/ SEMT/DYN/ RT/03.009/A*
- [808] **P. Galon, S. Potapov**  
Méthode des volumes finis pour les écoulements compressibles - Analyse bibliographique. *Rapport DEN/DM2S/ SEMT/DYN/ RT/03.013/A*
- [809] **P. Galon, F.Bliard**  
EUROPLEXUS - Fiche de qualification : Amortissements. *Rapport DEN/DM2S/ SEMT/DYN/ RT/03.014/A*
- [810] **M.F. Robbe, S. Potapov**  
Modeling of the depressurisation induced by a pipe rupture in the primary circuit of a nuclear plant. *Revue Européenne des éléments finis, Volume 12 - n°4/2003, pages 459-485.*

**2004:**

- [811] **M. Lepareux**  
Comportement des internes de cuve en cas d'APRP - Etat de l'art. *Rapport DEN/DM2S/SEMT/DYN/ RT/04.002/A*
- [812] **P. Galon**  
Analyse des ruptures de tuyauteries sous pression : calcul d'une tuyauterie générique avec Europlexus. *Rapport DEN/DM2S/SEMT/DYN/ RT/04.004/A*
- [813] **M. Lepareux**  
EUROPLEXUS : Modèle de porosité diphasique. *Rapport DEN/DM2S/SEMT/DYN/ RT/04.009/A*
- [814] **P.E Dumouchel, P. Galon, D. Combescure**  
Etude du fouettement d'une tuyauterie 3D avec Europlexus et construction de modèles simplifiés. *Rapport DEN/DM2S/SEMT/DYN/RT/04.011/A*
- [815] **P. Galon**  
Plan de développement des volumes finis dans Europlexus. *Rapport DEN/DM2S/SEMT/DYN/ PT/04.017/A*

**2005:**

- [816] **M. Lepareux, N. Visseaux**  
Analyse du cloisonnement sous APRP - Cas d'une brèche en sortie de pompe. *Rapport DEN/DM2S/SEMT/DYN/ RT/05.014/A*
- [817] **D. Combescure, P. Galon, J. Aubert**  
Etude de l'impact d'un tuyau coudé sur une plaque. *Rapport DEN/DM2S/SEMT/DYN/ RT/05.008/A*
- [818] **M. Lepareux, N. Visseaux**  
Analyse du cloisonnement sous APRP - Cas d'une brèche en entrée de cuve. *Rapport DEN/DM2S/SEMT/DYN/ RT/05.015/A*
- [819] **A. Suffis.**  
*Développement d'un modèle d'endommagement à taux de croissance contrôlé pour la simulation robuste de ruptures sous impacts.* Thèse INSA-Lyon soutenue le 21 juin 2004.

**2006:**

- [820] **P. Galon**  
Rapport de synthèse : Méthode des volumes finis en ALE dans EUROPLEXUS - Nouvelles structures informatiques associées - Cas tests. *Rapport DEN/DM2S/SEMT/DYN/ RT/06.001/A*
- [821] **M. Lepareux, N. Visseaux**  
Analyse du cloisonnement sous APRP - Cas d'une brèche en entrée de cuve - Maillage 3D actualisé. *Rapport DEN/DM2S/SEMT/DYN/ RT/06.009/A*
- [822] **D. Combescure, J. Rousseau, P. Galon**  
élaboration de modèles simplifiés d'une tuyauterie impactant une plaque ou une autre plaque. *Rapport DEN/DM2S/SEMT/DYN/ RT/06.014/A*
- [823] **P. Galon**  
Méthode des volumes finis en ALE dans EUROPLEXUS - Interaction Fluide/Structure *Rapport DEN/DM2S/SEMT/DYN/ RT/06.016/A*
- [824] **M.F. Robbe, M. Lepareux, F. Casadei**  
Simulation of the MARA 10 test representing a core disruptive accident. *Computer Assisted Mechanics and Engineering Sciences, Vol. 13, pp. 269-304, 2006.*
- [825] **F. Gatuingt**  
Modèle de comportement BETON DYNAR LMT. *Internal report.*
- [826] **T. Menouillard, J. Réthoré, A. Combescure, H. Bung**  
Efficient explicit time stepping for the eXtended Finite Element Method (X-FEM). *Int. J. Numer. Meth. Engng 2006; 68;911-939*

**2007:**

- [827] **P. Galon**  
Méthode des volumes finis en ALE dans EUROPLEXUS - Couplage 1D (EF) - 3D (VF). *Rapport DEN/DM2S/SEMT/DYN/ RT/07.005/A*

- [828] **V. Faucher**  
Plan de développement parallèle du code EUROPLEXUS. *Rapport DEN/DM2S/SEMT/DYN/RT/07-015/A*.
- [829] **P. Galon, J. Nunziati**  
Méthode des volumes finis dans EUROPLEXUS - Extension du schéma à l'ordre deux en temps et en espace. *Rapport DEN/DM2S/SEMT/DYN/ RT/07.016/A*
- [830] **T. Menouillard**  
*Dynamique explicite pour la simulation numérique de propagation de fissure par la méthode des éléments finis étendus*. Thèse INSA-Lyon soutenue le 20 septembre 2007.
- 2008:**
- [831] **P. Galon, J. Nunziati**  
Méthode des volumes finis dans EUROPLEXUS - Extension du schéma à l'ordre deux en temps et en espace. *Rapport DEN/DM2S/SEMT/DYN/ RT/08.001/A*
- [832] **P. Galon, A. Coussin**  
Volumes finis dans EUROPLEXUS - Introduction et validation de 2 nouveaux matériaux pour modéliser les explosions. *Rapport DEN/DM2S/SEMT/DYN/ RT/08.002/A*
- [833] **P. Galon, V. Faucher, F. Casadei, S. Potapov**  
*Modelling Complex Fluid-Structure Interaction Problems with EUROPLEXUS Fast Dynamic Software*. Extended abstract submitted for presentation at the 8th World Congress on Computational Mechanics (WCCM8), Venice, Italy, June 30 – July 5, 2008.
- [834] **V. Faucher, H. Bung, F. Casadei**  
*Fully Coupled Time-Varying Links Using Lagrange Multipliers for Fast Transient Dynamics Using EUROPLEXUS*. Extended abstract submitted for presentation at the 8th World Congress on Computational Mechanics (WCCM8), Venice, Italy, June 30 – July 5, 2008.
- [835] **H. Bung**  
*Serveur web de l'atelier logiciel Europlexus*. Rapport DEN/DM2S/SEMT/DYN/RT/08.019/A.
- [836] **V. Faucher**  
*Parallélisme à mémoire distribuée dans EUROPLEXUS - Avancement des développements*. Rapport DEN/DM2S/SEMT/DYN/RT/08.010/A.
- [837] **F. Debaud, P. Galon**  
Validation de l'implémentation de la méthode des volumes finis dans Europlexus. Simulation des essais MARA 2 et HDR V32. *Rapport DEN/DM2S/SEMT/DYN/ RT/08.015/A*
- [838] **P. Galon**  
Méthode des volumes finis dans EUROPLEXUS - modélisation d'une impédance. *Rapport DEN/DM2S/SEMT/DYN/ RT/08.020/A*
- [839] **P. Galon, A. Beccantini**  
Méthode des volumes finis dans EUROPLEXUS - Introduction d'un solveur bas mach. *Rapport DEN/DM2S/SEMT/DYN/ RT/08.021/A*
- [840] **T. Menouillard, J. Réthoré, A. Combescure, H. Bung**  
Mass lumping strategies for X-FEM explicit dynamics: Application to crack propagation. *Int. J. Numer. Meth. Engng* 2008; 74;447-474

**2009:**

- [841] **P. Galon**  
Europlexus - Bilan des modélisations filaires Eléments Finis. Hiérarchisation des modèles en vue du transfert vers le formalisme Volumes Finis. *Rapport DEN/DM2S/SEMT/DYN/RT/09.015/A*
- [842] **V. Faucher**  
*Performances de l'algorithme du programme EUROPLEXUS. Mise à disposition d'une version parallèle à mémoire distribuée. Rapport DEN/DM2S/SEMT/DYN/RT/09-016/A.*
- [843] **M. Chambart**  
*Endommagement anisotrope et comportement dynamique des structures en béton armé jusqu'à la ruine* Thèse LMT-Cachan soutenue le 24 septembre 2009.

**2010:**

- [844] **A. Beccantini, E. Studer**  
*The reactive Riemann problem for thermally perfect gases at all combustion regimes.* Int. J. Numer. Meth. Fluids; **64**:269-313, 2010.
- [845] **A. Beccantini, K. Tang**  
*The Reactive Discrete Equation Method for the (reactive) Euler Equations : the minmod reconstruction versus the antidiffusive approach.* CEA technical report SFME/LTMF/10-007/A, 2010.
- [846] **P. Galon, R. Gadenne, G. Tatoyan**  
Implémentation et validation dans le code de dynamique rapide Europlexus de 3 nouveaux matériaux pour représenter le comportement des liquides et des mélanges gaz/liquides en présence de chocs forts. *Rapport DEN/DM2S/SEMT/DYN/ RT/10.004/A*
- [847] **D. Solyga, P. Galon**  
Introduction d'une nouvelle méthode de reconstruction du second ordre en temps et en espace basée sur les variables primitives pour traiter les transitoires intenses dans Europlexus. *Rapport DEN/DM2S/SEMT/DYN/ RT/10.007/A*
- [848] **V. Faucher**  
Implementation dans EUROPLEXUS d'une approche anti-diffusive pour la simulation d'écoulements multi-composants  
*CEA technical report SEMT/DYN/10-014/A, 2010.*
- [849] **V. Faucher**  
Parallelisme a memoire distribuee pour les Volumes Finis Cell-Centered dans EUROPLEXUS - Cas particulier de l'ALE  
*CEA technical report SEMT/DYN/10-015/A, 2010.*
- [850] **Z. Gao, P. Galon, F. Daude**  
Introduction and validation of three different Multiphase flows models in the fast transient dynamic Code Europlexus. *Rapport DEN/DM2S/SEMT/DYN/ RT/10.019/A*
- [851] **V. Faucher, P. Galon, P. Izquierdo**  
Groupe de travail 'Analyse de la traversée d'une plaque perforée par une onde de dépressurisation' - Rapport de synthèse  
*CEA technical report SEMT/DYN/10-027/A, 2010.*

- [852] **V. Faucher**  
Assistance a la mise en donnees EUROPLEXUS dans SALOME avec XDATA - Synthese des actions 2010  
*CEA technical report SEMT/DYN/10-030/A, 2010.*
- [853] **V. Faucher**  
Description et evaluation des modeles de calcul EUROPLEXUS pour l'analyse des consequences mecaniques d'un Accident par Perte de Refrigerant Primaire  
*CEA technical report SEMT/DYN/10-031/A, 2010.*
- 2011:**
- [854] **F. Caleyron**  
*Simulation numérique par la méthode SPH de fuites de fluide consécutives à la déchirure d'un réservoir sous impact.* Thèse soutenue le XX septembre 2011, Université de Lyon, LaMCoS - UMR CNRS 5514 - INSA de Lyon.
- [855] **K. Tang**  
*Discrete Equations Method for 1D stiffened gas flows I: Interface problems.* CEA technical report DEN/DANS/DM2S/STMF/LATF/RT/11-002/A 2011.
- [856] **V. Faucher**  
Parallelisme a memoire distribuee dans EUROPLEXUS - Application aux elements discrets et à leurs interactions avec les elements finis  
*CEA technical report SEMT/DYN/11-013/A, 2011.*
- [857] **V. Faucher**  
Parallelisme a memoire distribuee dans EUROPLEXUS - Integration des modelisations filaires pour les tuyauteries  
*CEA technical report SEMT/DYN/11-017/A, 2011.*
- [858] **V. Faucher**  
Mise en donnees pour EUROPLEXUS dans SALOME avec XDATA - Description de l'outil et de son utilisation  
*CEA technical report SEMT/DYN/11-022/A, 2011.*
- [859] **P. Galon, F. Daude**  
Rapport de synthèse : Méthode des Volumes Finis Filaires dans Europlexus - Nouvelles structures informatiques associées - Cas tests. *Rapport DEN/DANS/DM2S/SEMT/DYN/RT/11.026/A*
- [860] **V. Faucher**  
Performances du code EUROPLEXUS pour le calcul de structure - Analyse de l'utilisation du partitionnement spatial des elements  
*CEA technical report SEMT/DYN/11-027/A, 2011.*
- [861] **F. Crouzet, V. Faucher, P. Piteau, P. Galon, P. Izquierdo**  
LOCA Simulation: Analysis of Rarefaction Waves Propagating through Geometric Singularities  
*ASME PVP 2011, Baltimore (USA), July 17 - 21, 2011.*

**2012:**

- [862] **K. Tang, A. Beccantini, C. Corre**  
*Combining of discrete equations method and upwind downwind-controlled splitting for non-reacting and reacting two-fluid computations.* Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii, July 9-13, 2012.
- [863] **A. Leroux.**  
Modèle multiaxial d'endommagement anisotrope: Gestion numerique de la rupture et application à la ruine des structures en béton armé sous impacts. *Thèse soutenue le 21 Novembre 2012 LaMSID-UMR EDF/CNRS/CEA.*
- [864] **K. Tang**  
*Combining Discrete Equations Method and Upwind Downwind-Controlled Splitting for Non-Reacting and Reacting Two-Fluid Computations.* Thèse de Doctorat, Université de Grenoble, December 2012.
- [865] **V. Faucher**  
Elements discrets pour la simulation de la ruine des structures en beton arme avec EUROPLEXUS - Etat des lieux et actions de support pour la these d' A. Masurel  
*CEA technical report SEMT/DYN/12-022/A, 2012.*
- [866] **Y. Drymer, P. Galon, P. Guimbal**  
Tentative application of the EUROPLEXUS code in the modeling of sudden depressurization of generic irradiation device. *Rapport DEN/DANS/DM2S/SEMT/DYN/RT/11.024/A*
- [867] **V. Faucher**  
Module EPXDATA - Corrections/ameliorations et interpretation de la syntaxe de commande du programme EUROPLEXUS pour la relecture des jeux de donnees existants  
*CEA technical report SEMT/DYN/12-029/A, 2012.*
- [868] **P. Galon, F. Daude**  
Méthode des Volumes Finis Filaires dans Europlexus - Compléments I. *Rapport DEN/DANS/DM2S/SEMT/DYN/ RT/12.031/A*
- [869] **F. Caleyron, A. Combescure, V. Faucher, S. Potapov**  
Dynamic Simulation of Damage-Fracture Transition in SPH Shells  
*Int. J. for Num. Meth. In Engrg, 90(6), 2012.*
- [870] **L. Daudeville, V. Faucher, S. Potapov**  
Advanced simulation of damage of reinforced concrete structures under impact  
*European Journal of Environmental and Civil Engineering, 16(9), 2012.*
- [871] **V. Faucher, F. Crouzet, P. Piteau, P. Galon, P. Izquierdo**  
Numerical and experimental analysis of transient wave propagation through perforated plates for application to the simulation of LOCA in PWR  
*Nuclear Engineering and Design, 253(2), 2012.*
- [872] **V. Faucher**  
Contribution of ANR RePDyn project to the parallel simulation of fast transient accidental phenomena at nuclear reactor scale  
*Complex Systems Design and Management Conference, Paris, France, December 12-14, 2012.*

- [873] **F. Crouzet, F. Daude, P. Galon, P. Helluy, J.M Hérard, O. Hurisse, Y. Liu**  
On the Computation of the Baer-Nunziato Model - AIAA paper 2012-3355. *42nd AIAA Fluid Dynamics Conference and Exhibit, New orleans, Louisiana, 25-28 june 2012.*
- 2013:**
- [874] **A. Beccantini, F. Casadei, P. Galon**  
*Improvement of the FLSW model for Cell-Centered Finite Volumes in EUROPLEXUS. Report DEN/DANS/DM2S/STMF/LATF/NT/13-019/A, April 2013.*
- [875] **A. Beccantini, F. Casadei**  
*Spatial time-step partitioning for Cell-Centered Finite Volumes in EUROPLEXUS. Report DEN/DANS/DM2S/STMF/LATF/NT/13-020/A, April 2013.*
- [876] **V. Faucher**  
Calcul Haute-Performance avec le programme EUROPLEXUS - Synthèse des développements réalisés dans le cadre du projet ANR REPDYN  
*CEA technical report SEMT/DYN/13-012/A, 2013.*
- [877] **V. Faucher**  
Vérification de la version parallèle à mémoire distribuée d'EUROPLEXUS sur les tests industriels EDF  
*CEA technical report SEMT/DYN/13-016/A, 2013.*
- [878] **P. Galon, F. Daude**  
Développement d'un modèle diphasique à deux pressions dans Europlexus et vérifications numériques - Partie convective *Rapport DEN/DANS/DM2S/SEMT/DYN/ RT/13.024/A*
- [879] **V. Faucher**  
Mise en données du code EUROPLEXUS à l'aide de la plateforme SALOME - Version 1.1.0 du module EPXDATA  
*CEA technical report SEMT/DYN/13-025/A, 2013.*
- [880] **P. Galon, F. Daude**  
Méthode des Volumes Finis Filaires dans Europlexus - Compléments II. *Rapport DEN/DANS/DM2S/SEMT/DYN/ RT/13.026/A*
- [881] **V. Faucher, S. Kokh**  
Extended Vofire Algorithm for Fast Transient Fluid-Structure Dynamics with Liquid-Gas Flows and Interfaces  
*Journal of Fluids and Structures, volume 30, pages 126-153, 2013.*
- [882] **F. Caleyron, A. Combescure, V. Faucher, S. Potapov**  
SPH modeling of fluid-solid interaction for dynamic failure analysis of fluid-filled thin shells  
*Journal of Fluids and Structures, volume 30, pages 102-125, 2013.*
- [883] **V. Faucher**  
Advanced parallel strategy for strongly coupled fast transient fluid-structure dynamics with dual management of kinematic constraints  
*Advances in Engineering Software, volume 67, pages 70-89, 2013.*
- [884] **R. Pelee de Saint Maurice, V. Faucher, T. Elguedj, A. Combescure, B. Prabel**  
Propagation de fissures 3 dimensions et en dynamique rapide  
*11eme Colloque National sur le Calcul des Structures, Giens (France), 13-17 mai, 2013.*

- [885] **V. Faucher**  
Simulation de l'Accident de Dimensionnement du Confinement dans un reacteur de IVe generation avec une approche parallele hybride dans EUROPLEXUS  
*11eme Colloque National sur le Calcul des Structures, Giens (France), 13-17 mai, 2013.*
- [886] **T. Gautier, F. Lementec, V. Faucher, B. Raffin**  
X-KAAPI: a Multi Paradigm Runtime for Multicore Architectures  
*Sixth International Workshop on Parallel Programming Models and Systems Software for High-End Computing (P2S2), Lyon, 1er octobre, 2013.*
- [887] **R. Pelee de Saint Maurice, T. Elguedj, A. Combescure, B. Prabel, V. Faucher**  
Dynamic simulation of crack propagation in 3D by X-FEM method  
*XFEM 2013, Lyon (France), 11-13 septembre, 2013.*
- [888] **V. Faucher, P. Galon, A. Beccantini, F. Crouzet, F. Debaud, T. Gautier**  
Hybrid parallel strategy for the simulation of fast transient accidental situations at reactor scale  
*Supercomputing for Nuclear Applications + Monte Carlo 2013, Paris, 27-31 octobre, 2013.*
- [889] **F. Crouzet, F. Daude, P. Galon, P. Helluy, J.M Hérard, O. Hurisse, Y. Liu**  
Approximate Solutions of the Baer-Nunziato Model. *ESSAIM: proceedings, EDP Sciences, Vol 40, p 63-82, juillet 2013.*
- [890] **A. Velikorodny, E. Studer, S. Kudriakov, A. Beccantini.** Combustion modeling in large scale volumes. *ICHS 2013, Brussels (Belgium), September 9-11, 2013.*
- [891] **E. Studer, A. Beccantini, S. Kudriakov, A. Velikorodny.** Hydrogen combustion modelling in large-scale geometries. *Proceedings of ICON21, Chengdu (China), July 29-August 2, 2013.*
- 2014:**
- [892] **K. Tang, A. Beccantini, C. Corre**  
Combining Discrete Equations Method and upwind downwind-controlled splitting for non-reacting and reacting two-fluid computations: One dimensional case.  
*Computer & Fluids, volume 93, pages 74-90, 2014.*
- [893] **K. Tang, A. Beccantini, C. Corre**  
Combining Discrete Equations Method and upwind downwind-controlled splitting for non-reacting and reacting two-fluid computations: Two dimensional case.  
*Computer & Fluids, volume 103, pages 132-155, 2014.*
- [894] **F. Daude, P. Galon, Z. Gao, E. Blaud**  
Numerical experiments using a HLLC-type scheme with ALE formulation for compressible two-phase flows five-equation models with phase transition.  
*Computer & Fluids, volume 94, pages 112-138, 2014.*
- [895] **A. Menasria, F. Daude, P. Galon**  
Implementation of a HLLC-type Riemann solver for the twophase flow model of Baer-Nunziato in the fast transient Europlexus software.  
*Rapport DEN/DANS/DM2S/SEMT/DYN/ NT/14.005/A, 2014.*
- [896] **P. Galon, F. Daude**  
Volumes Finis Filaires dans Europlexus Couplage 1D-3D.  
*Rapport DEN/DANS/DM2S/SEMT/DYN/ NT/14.008/A, 2014.*



- [897] **E. Blaud, P. Galon, F. Daude**  
Implémentation et validation d'un modèle à 5 équations avec changement de phase dans le code de dynamique rapide Europlexus.  
*Rapport DEN/DANS/DM2S/SEMT/DYN/ NT/14.015/A, 2014.*
- [898] **P. Galon, F. Daude**  
Extension du modèle diphasique à deux pressions aux Volumes-Finis filaires.  
*Rapport DEN/DANS/DM2S/SEMT/DYN/ NT/14.025/A, 2014.*
- [899] **F. Bliard**  
Modélisation de la perforation du bois avec EUROPLEXUS : Mise en oeuvre d'un processus de décohésion des éléments reposant sur un critère de rupture.  
*Rapport DEN/DANS/DM2S/SEMT/DYN/ NT/14-019/A, 2014.*
- 2015:**
- [900] **P. Galon, A. Beccantini, E. Marechal, A. Boutayeb**  
Introduction d'un nouveau modèle de détonation pour les mélanges gazeux dans Europlexus.  
*Rapport DEN/DANS/DM2S/SEMT/DYN/NT/15-014/A, 2015.*
- [901] **V. Faucher, P. Galon, A. Beccantini, F. Crouzet, F. Debaud, T. Gauthier**  
Hybrid parallel strategy for the simulation of fast transient accidental situations at reactor scale.  
*Annals of Nuclear Energy , volume 82, pages 188-194, 2015.*
- [902] **F. Crouzet, F. Daude, P. Galon, J.M Hérard, O. Hurisse, Y. Liu**  
Validation of a two-fluid model on unsteady liquid-vapor water flows.  
*Computer & Fluids, volume 119, pages 131-142, 2015.*
- [903] **H. Lochon, F. Daude, P. Galon, J.M Hérard**  
Computation of steam-water transients using a two-fluid seven-equation model.  
*12th International Conference - Pressure Surges, pages 69-81, 2015.*
- 2016:**
- [904] **F. Daude, P. Galon**  
On the computation of the Baer-Nunziato model using ALE formulation with HLL- and HLLC-type solvers towards fluid-structure interactions.  
*Journal of Computational Physics, volume 324, pages 189-230, 2016.*
- [905] **H. Lochon, F. Daude, P. Galon, J-M Hérard**  
Comparison of two-fluid models on steam-water transients.  
ESAIM Mathematical Modelling and Numerical Analysis 50(6):1631-1657, 2016.
- [906] **H. Lochon, F. Daude, P. Galon, J-M Hérard**  
HLLC-type Riemann solver with approximated two-phase contact for the computation of the Baer-Nunziato two-fluid model.  
*Journal of Computational Physics, volume 326, pages 733-762, 2016.*
- [907] **P. Galon**  
Europlexus : éléments de note théorique et synthèse des documents de référence.  
*Rapport DEN/DANS/DM2S/SEMT/DYN/ RT/16.012/A, 2016.*

**2018:**

- [908] **M. Deka, S. Brahmachary, R. Thirumalaisamy, A. Dalal, G. Natarajan.**  
A new Green–Gauss reconstruction on unstructured meshes. Part I: Gradient reconstruction.  
*J. Comput. Phys.* (2019)

**2019:**

- [909] **V. Faucher, F. Casadei, G. Valsamos, M. Larcher.**  
High resolution adaptive framework for fast transient fluid-structure interaction with interfaces and structural failure – Application to failing tanks under impact.  
*International Journal of Impact Engineering*, 127 (2019), 62–85.
- [910] **H. Mathis.**  
A thermodynamically consistent model of a liquid-vapor fluid with a gas.  
*ESAIM: M2AN*, 53 (2019), 63–84.

---

**SAMTECH PLEXUS/EUROPLEXUS BIBLIOGRAPHY****1999:**

- [911] E. Gabellini. *PLEXUS - User Material Feasibility Study*. Rapport Samtech 180, July 1999.
- [912] E. Gabellini. *PLEXUS - Bushing Element Development*. Rapport Samtech 181, August 1999.

**2000:**

- [913] E. Gabellini. *PLEXUS - Mesh-Merging Development ("Gluing" shell-shell)*. Rapport Samtech, October 2000.

**2001:**

- [914] E. Maillard, F. Dubois, N. Kill, E. Gabellini. *Simulation numerique d'impact d'oiseau sur une structure représentative de bord d'attaque de voilure*. Sonaca/Samtech paper, 2001.

**2003:**

- [915] F. Lachaud, P. Jetteur. *Modélisation du comportement des composites à fibres courtes : application à l'impact basse vitesse*. Colloque "COMPOSITES 2003", CNAM, Paris, 22 Octobre 2003.

**2009:**

- [916] N. Kill. *Modèle matériau pour l'élément cohésif d'Europlexus*. Rapport Samtech, April 2009.

**EDF CASTEM-PLEXUS/EUROPLEXUS BIBLIOGRAPHY****1982:**

- [917] **A. Thuon, J-P. Kirchgessner**

Interprétation d'un essai d'écrasement dynamique de tuyauterie (essai CEMETE) par le code de calcul PLEXUS. *Note Technique EDF/SEPTEN/TA/NT 82.20/A.*

**1983:**

- [918] **T.H. Chau**

Validation du programme PLEXUS pour l'étude du flambage dynamique. *Note Technique EDF/SEPTEN/TA/NT/83.21/A.*

**1984:**

- [919] **T.H. Chau, J-N. Laclau**

Etude dynamique des membranes de sûreté - Pression de flambage dynamique et tenue à un pic de pression. *Note Technique EDF/SEPTEN/MS/NT 84.21/A.*

**2004:**

- [920] **S. Potapov**

Simulation of hydrodynamic transients in piping systems containing swing check valves. *Proceedings of the Conference "Flow Induced Vibration", Ecole Polytechnique, Paris, 6-9th July 2004.*

- [921] **S. Potapov and E. Altstadt**

Coupled fluid-structure analysis with EUROPLEXUS fast dynamics software of the CWHTF experiment. *Proceedings of the 6th International Conference on Nuclear Thermal Hydraulics, Operations and Safety (NUTHOS-6), Nara, Japan, October 4-8, 2004.*

**2005:**

- [922] **S. Potapov and P. Galon**

Modelling of Aquitaine II pipe whipping test with the EUROPLEXUS fast dynamics code. *Nuclear Engineering and Design, Volume 235, Issues 17-19, August 2005, Pages 2045-2054.*

- [923] **S. Moulin, S. Potapov**

Modélisation du contact dans EUROPLEXUS. Théorie et éléments de validation, *Rapport EDF HT-62/05/012/A.*

**2006:**

- [924] **S. Potapov**

Introduction de l'amortissement de Rayleigh dans le code de dynamique rapide EUROPLEXUS, *Rapport EDF HT-62/06/008/A.*

- [925] **P. Koechlin**  
Validation d'un critère de perforation sur les essais Meppen, *Rapport EDF H-T62-2006-02120-FR*.
- [926] **P. Koechlin**  
Failure criterion for reinforced concrete beams and plates subjected to membrane force, bending and shear, *Rapport EDF H-T62-2006-02537-EN*.
- 2007:**
- [927] **S. Potapov**  
Correction de la vitesse des ondes de pression dans un élément de tuyauterie à paroi élastique dans le code EUROPLEXUS, *Rapport EDF H-T62-2007-00449-FR*.
- [928] **P. Koechlin**  
Modèle de comportement membrane-flexion et critère de perforation pour l'analyse de structures minces en béton armé sous choc mou, *Thèse de doctorat de l'Université Pierre et Marie Curie - Paris VI*, 2007.
- [929] **P. Koechlin, S. Potapov**  
Global Constitutive Model for Reinforced Concrete Plates, *J. Engrg. Mech., Vol. 133, Issue 3, pp. 257-266* (March 2007).
- [930] **S. Potapov, P. Koechlin, L. Flandi**  
Validation de la loi élasto-plastique endommogéante GLRC DAMA PERF d'EUROPLEXUS sur l'essai II-2 de la campagne MEPPEN, *Rapport EDF H-T62-2007-02949-FR*.
- 2008:**
- [931] **D. Markovic**  
Implantation d'un nouvel élément fini de coque épaisse (T3GS) dans EUROPLEXUS, *Rapport EDF H-T62-2008-00080-FR*.
- [932] **B. Maurel**  
Modélisation par la méthode SPH de l'impact d'un réservoir rempli de fluide, *Thèse de doctorat de l'Institut National des Sciences Appliquées de Lyon*, 2008.
- 2009:**
- [933] **F. Crouzet**  
Développement dans EUROPLEXUS d'un modèle de brèche dynamique, *Rapport EDF H-T63-2009-01828-FR*.
- [934] **S. Potapov**  
Modélisation du comportement de structures sous impact via la méthode des éléments discrets. Cas-tests., *Rapport EDF H-T62-2008-02020-FR*.
- [935] **J. Rousseau**  
Modélisation numérique du comportement dynamique de structures sous impact sévère avec un couplage éléments discrets / éléments finis, *Thèse de doctorat de l'Université Joseph Fourier*, 2009.

**2010:**

- [936] **L. Idoux**  
Etude du comportement non linéaire d'un puits de cuve sous explosion vapeur. Analyse de la modélisation 3D avec EUROPLEXUS. *Rapport EDF H-T62-2010-01356-FR.*
- [937] **J.-P. Devos**  
Conditions limites non réfléchissantes 3D pour les équations d'Euler compressibles et un gaz idéal parfait dans la nouvelle version volumes finis du module fluide du code EUROPLEXUS - Partie 1 : Mise en oeuvre. *Rapport EDF H-T63-2010-00160-FR.*
- [938] **J.-P. Devos**  
Conditions limites non réfléchissantes 3D pour les équations d'Euler compressibles et un gaz idéal parfait dans la nouvelle version volumes finis du module fluide du code EUROPLEXUS - Partie 2 : Validations. *Rapport EDF H-T63-2010-00173-FR.*

**2011:**

- [939] **S. Moulin**  
Mise en oeuvre des plaques excentrées dans EUROPLEXUS (T3GS et Q4GS). Développement et Validation. *Rapport EDF H-T62-2011-00779-FR.*
- [940] **F. Crouzet**  
Prise en compte dans le code EUROPLEXUS de la correction de vitesse des ondes pour un élément de tuyauterie avec fluide. *Rapport EDF H-T63-2011-02733-FR.*
- [941] **A. Masurel**  
Recherche d'un VER pour la méthode des éléments discrets en élasticité linéaire. *Rapport EDF de stage de fin d'études UPMC, 2011.*

**2012:**

- [942] **S. Fayolle**  
Introduction d'un cisaillement transverse non linéaire dans GLRC DAMA. *Rapport EDF H-T62-2012-00003-FR.*
- [943] **F. Daude**  
Développement d'un modèle diphasique à deux pressions dans EUROPLEXUS et vérifications numériques - Partie convective. *Rapport EDF H-T63-2012-01003-FR.*

**2013:**

- [944] **S. Potapov**  
Mise en oeuvre dans EUROPLEXUS d'un élément CUBB basé sur la méthode B-bar. *Compte-rendu EDF CR-AMA-13.179.*
- [945] **S. Potapov**  
Mise en oeuvre dans EUROPLEXUS d'un élément ressort linéaire à deux noeuds (RL3D). *Compte-rendu EDF CR-AMA-13.233.*

- [946] **L. Idoux, J. Haelewyn**  
Projet MATUR - Lot 6 Risque-Missile-Turbine: Mise en oeuvre d'une étude d'impact perforant d'un disque avec ailettes sur une couronne. *Rapport EDF H-T62-2013-00371-FR*.
- [947] **Y. Liu, F. Daude, P. Galon**  
Applications IFS d'un modèle diphasique à deux pressions dans EUROPLEXUS et vérifications numériques. *Rapport EDF H-T63-2013-01854-FR*.
- 2015:**
- [948] **A. Masurel**  
Modélisation mixte éléments discrets / éléments finis de la dégradation de structures en béton armé sous impact sévère. *Thèse de doctorat de l'Université Grenoble Alpes, 2015*.
- [949] **S. Potapov, E. Cheignon**  
Modélisation des câbles adhérents, glissants et frottants dans EUROPLEXUS. *Rapport EDF H-T61-2015-04268-FR*.
- [950] **S. Potapov, M. Siavelis**  
Mise en oeuvre dans EUROPLEXUS du collage Arlequin plaque-3D. *Rapport EDF H-T61-2015-04524-FR*.
- [951] **T. Li, J.-J. Marigo, D. Guilbaud, S. Potapov**  
Variational Approach to Dynamic Brittle Fracture via Gradient Damage Models. *Applied Mechanics and Materials, Volume 784, pp. 334–341*.
- [952] **T. Li, J.-J. Marigo, D. Guilbaud, S. Potapov**  
Gradient damage modeling of brittle fracture in an explicit dynamics context. *International Journal for Numerical Methods in Engineering, 2016*.
- 2016:**
- [953] **F. Banci**  
Projet Dynamique Rapide - LP DN01.12 (partie 1) - Analyses complémentaires autour du modèle de béton DPDC (version 8) pour EUROPLEXUS, *NT-6125-1714-2016-15058-FR*.
- [954] **H. Lochon**  
Développement de modèles diphasiques à deux pressions dans Europlexus - Partie associée aux termes sources, *H-T61-2015-05315-FR*.
- [955] **S. Potapov, A. Masurel, L. Daudeville, P. Marin**  
Using a mixed DEM/FEM approach to model advanced damage og reinforced concrete under impact, *Int. J. Comp. Meth. and Exp. Meas., Vol. 4, No. 3 (2016) 258-268*.
- [956] **S. Potapov, A. Masurel, P. Marin, L. Daudeville**  
Mixed DEM/FEM Modeling of Advanced Damage in Reinforced Concrete Structures, *J. Eng. Mech., 04016110, 1-13, 2016*.
- [957] **T. Li, J.-J. Marigo, D. Guilbaud, S. Potapov**  
Gradient damage modeling of brittle fracture in an explicit dynamics context, *Int. J. Numer. Meth. Engng (2016)*.

[958] **O. Boiteau, G. Ferte**

Analyse du parallelisme et des solveurs lineaires d'EUROPLEXUS afin d'accelerer les simulations EDF: application a une etude d'APRP, *NT-6125-1106-2016-16151-FR*.



---

ONERA EUROPLEXUS BIBLIOGRAPHY**1997:**

- [959] J.-F. Maire, J.-L. Chaboche. *A new formulation of Continuum Damage Mechanics for composite materials*. Aerospace Science and Technology, vol 4, 1997, pp 247-257.

**2014:**

- [960] M.-A. Kouane-Nana. *Étude des mécanismes influents dans les lois de comportement des matériaux CMO pour la modélisation des phénomènes d'endommagement*. Technical report, RT 2/22401 DADS, September 2014.

**2015:**

- [961] B. Langrand, F. Casadei, G. Portemont, V. Marcadon, M. Larcher. *Comportement mécanique de matériaux architecturés par empilement de tubes*. ONERA Technical Report RT 5/23762 DADS, November 2015.
- [962] V. Marcadon, G. Portemont, B. Langrand, A. Iltchev, S. Kruch, S. Forest, F. Casadei. *DynaCell: Mechanical characterisation of cellular structures made of tube stackings (Final report)*. ONERA Technical Report RF 6/18367 DMSM, December 2015.

**2016:**

- [963] B. Langrand, F. Casadei, V. Marcadon, G. Portemont, S. Kruch. *FE modelling of cellular materials under compressive load*. 11th International Symposium on Plasticity and Impact Mechanics (IMPLAST2016), New Delhi, India, December 12–14, 2016.
- [964] R. Ortiz, J.F. Sobry, F. Casadei. *126 - Propeller Blade Debris Kinematics : Blade debris trajectory computation with Aerodynamic effects using new FSI formulations*. Conference Greener Aviation 2016, Bruxelles, 11–13 October 2016.

**2017:**

- [965] B. Langrand, F. Casadei, V. Marcadon, G. Portemont, S. Kruch. *Experimental and Finite Element analysis of cellular materials under large compaction levels*. International Journal of Solids and Structures, 128, pp. 99–116, 2017.

**2018:**

- [966] R. Ortiz, F. Casadei, S. Mouton, J-F. Sobry. *Propeller Blade Debris Kinematics. Blade debris trajectory computation with aerodynamic effects using new FSI formulations*. CEAS Aeronautical Journal. <https://doi.org/10.1007/s13272-018-0313-4>. Published online: 14 June 2018.

## 22 Keywords Index

The next pages contain an extensive index of all the keywords that appear in the various EU-ROPLEXUS input directives that have been presented in this manual.

In the PDF and HTML versions of the manual, index entries contain links, and the relevant information may be reached by simply clicking on the link.

## Index

?, 1277, 1287, 1288, 1299  
\_RIGI<nnn>, 255  
1) Select color, 1380  
1) Select material, 1386  
2) Apply it to, 1381, 1386  
  
A, 567, 577, 585, 587, 596, 648  
A1, 353, 1310  
A12, 353  
A13, 353  
A14, 1310  
A2, 353  
A23, 353  
A3, 353  
A6, 1310  
ABAQ, 943  
ABS, 902, 1166, 1228  
ABSE, 863, 864, 866, 867, 869, 871  
ABSI, 717, 1118  
ABSO, 665, 702, 733  
ABST, 667  
ABSZ, 719  
ACBE, 815  
ACCE, 166, 819, 1069, 1080, 1094, 1168, 1197,  
1211, 1218, 1244, 1288, 1323, 1326, 1330  
ACIE, 480  
ACMN, 1234  
ACMX, 1234  
ACOS, 1228  
ADAD, 895  
ADAP, 91, 111, 166, 825, 895, 915, 946, 987,  
1153, 1168, 1277, 1288  
Adaptivity post-processing, 1217  
ADAV, 1288  
ADC8, 75  
ADCJ, 577  
ADCR, 558  
ADD, 1228  
ADDC, 1228  
ADDF, 91, 997, 1042, 1131  
ADFM, 606  
ADFT, 1197, 1218, 1244, 1330  
ADHE, 817, 854, 1323  
ADJA, 1153  
ADNP, 895  
ADOU, 480  
ADQ4, 61  
ADTI, 1131  
ADUN, 815  
  
AFIL, 480  
AFLX, 186  
AFLY, 212  
AHGF, 592  
AINI, 690  
AIRB, 722, 997, 1064  
AIRE, 186  
AJOU, 1401  
AL, 278  
AL1, 506  
AL2, 506  
AL3, 506  
AL4, 506  
ALE, 91, 119, 153, 907, 908  
ALEN, 1310  
ALES, 1133  
ALF0, 592, 1140  
ALF1, 418  
ALF2, 445  
ALFA, 166, 445, 529, 1052  
ALFN, 533  
ALIC, 946, 972, 1080, 1112, 1204, 1392  
ALIT, 1080  
ALL, 1310  
ALLF, 885  
ALP1, 961  
ALP2, 961  
ALPH, 181, 248, 480, 563, 614, 648, 960, 961  
ALPHA, 206, 509  
AMAX, 690, 709, 751, 1310  
AMBI, 1310  
AMMB, 186  
AMOR, 270, 271, 815, 1115  
AMOY, 1234  
AMP, 1052, 1055, 1057  
AMPD, 1288  
AMPF, 1288  
AMPL, 871  
AMPV, 1288  
ANCI, 1403  
ANG1, 1049, 1050  
ANG2, 1049, 1050  
ANGL, 247, 251, 313, 722, 902, 1064, 1166  
ANGM, 1153  
ANTI, 1310  
Antialias lines, 1361  
APEX, 802  
APPL, 1310

- APPU, 69, 193, 271  
APRS, 186  
ARLQ, 818  
ARMA, 243, 291, 811, 815  
ARRE, 1207  
ARTI, 862  
ARX, 274  
ARY, 274  
ARZ, 274  
As current geometry, 1365  
ASFA, 1310  
ASHB, 73  
ASIN, 1228  
ASIS, 1310  
ASN, 1153  
ASN length menu, 1369  
ASPE, 1103  
ASSE, 758, 761, 763, 770, 1115  
AT1, 278  
AT2, 278  
ATRA, 307  
ATRS, 186  
ATUB, 709, 751  
AUTO, 153, 216, 233, 798, 805, 843, 1080, 1110, 1176, 1197, 1287, 1400, 1401  
Auto 1, 1376, 1377  
Auto 14, 1373, 1376, 1377  
Auto 6, 1373, 1376, 1377  
AUTR, 153, 155–159  
AVE, 1310  
AVER, 1228  
AVI, 1288, 1300  
AVS, 1080, 1092, 1288  
AX, 274, 282  
AXE, 864, 867, 869  
AXE1, 248, 251, 839, 873, 874  
AXE2, 248, 839, 873, 874  
AXES, 1262, 1267, 1269, 1271  
AXIA, 1115  
AXIS, 91, 969, 1288  
AXOL, 1288  
AXTE, 1214, 1259  
AY, 186, 274, 282  
AZ, 186, 274, 282  
  
B, 298, 519, 567, 577, 585, 587, 596, 648, 722, 1064, 1343  
Back, 1384  
Background, 1381  
BACO, 341, 345  
BAKE, 91  
BAND, 1024, 1028  
BARR, 57  
BARY, 254, 880, 1097, 1234  
BASE, 143–150, 548  
BBOX, 1310  
BC, 453, 454  
BDFO, 610  
BENS, 1277  
BET, 1052, 1055, 1057  
BET0, 592  
BETA, 181, 270, 313, 418, 427, 445, 468, 480, 515, 529, 533, 538, 558, 567, 587, 1052, 1055, 1057  
BETC, 817  
BETO, 480, 811, 815  
BETON, 312  
BFIK, 139  
BFLU, 825, 915  
BGLA, 1310  
BGRN, 1310  
BHAN, 1310  
Biblio, 1428, 1462  
BIFU, 51, 233, 853, 1323  
BILA, 1197, 1238  
BILL, 70, 252, 406, 581, 848, 850  
BIMA, 243, 480  
BINA, 91, 1080  
Bitmap format, 1389  
BIVF, 52  
BL3S, 480  
BLAC, 1310  
Black, 1380  
Black plastic, 1386  
Black rubber, 1386  
Black rubber2, 1386  
BLAN, 204, 1262  
BLAP, 1310  
BLAR, 1310  
BLEU, 204, 1262  
BLKO, 509  
BLMT, 319  
BLOQ, 794, 1042, 1050, 1310, 1323  
BLR2, 1310  
BLUE, 1310  
Blue, 1380  
Blue glass, 1355  
BM, 453, 454  
BMF, 596  
BMP, 1288, 1389  
BMPI, 91, 105  
BN, 454

- BODY, 895, 902, 1153  
BOIS, 371  
BOTH, 1145  
Bottom, 1384  
Bounding box, 1381  
BOX, 193, 198, 225  
BPEL, 322  
BPOV, 1288  
BR3D, 68  
BRAS, 1310  
Brass, 1386  
BREC, 52, 233, 533, 542, 925  
BRON, 1310  
Bronze, 1386  
BSHR, 72  
BSHT, 59, 72  
BT, 325, 453, 454  
BUBB, 626  
BULK, 460, 503, 506  
BUTE, 841  
BV, 325  
BWDT, 122  
BXX, 552  
BXY, 552  
BYY, 552  
  
C, 164, 494, 515, 525, 526, 546, 548, 552, 581, 585, 665, 717, 764, 940, 942  
C0, 556, 592  
C1, 418, 468, 503, 592  
C2, 503, 592  
C272, 78  
C273, 78  
C81L, 78  
C82L, 78  
CABL, 817  
CALC, 1118, 1190, 1277, 1287  
CAM1, 1308  
CAM2, 1308  
CAMC, 423  
CAME, 1277, 1306, 1310  
CAPA, 606  
CAR, 558, 577  
CAR1, 57, 154  
CAR4, 58, 154  
CARB, 603  
CART, 105, 1024, 1027, 1028, 1176  
CAS, 231  
CASE, 987  
CAST, 91  
Cast3m mesh color problem, 136  
  
CAVF, 52  
CAVI, 51, 233  
CBIL, 252  
CBOX, 243  
CBU, 558, 577  
CC, 709, 751  
CD, 930, 1168  
CDEM, 628  
CDEP, 1288  
CDES, 1310  
CDMR, 480  
CDNO, 1288  
CDOM, 1310  
CEA, 217  
CEAU, 563  
CECH, 533  
CELD, 243  
CELP, 742, 744  
CEND, 1234  
CENE, 1148  
CENT, 125, 192, 800, 803, 807, 857, 880, 882, 949, 1308, 1310  
Center, 1381  
Centre, 1384  
CERO, 1234  
CERR, 166, 1223, 1248, 1330  
CESC, 843  
CEXP, 1228  
CFLA, 628  
CFON, 764  
CFVN, 91  
CGAZ, 558, 577  
CGLA, 1310  
CH2, 526  
CHA1, 889  
CHA2, 889  
CHAI, 889  
CHAM, 974, 1080  
CHAN, 345  
CHAR, 587, 997  
CHEC, 1168  
CHOC, 571, 1139  
CHOL, 786, 789  
CHRO, 1310  
Chrome, 1386  
CHSP, 298  
CI12, 371  
CI23, 371  
CI31, 371  
CIBD, 836  
CIBL, 836

- CINI, 105, 1176  
CIRC, 186  
CKAP, 572  
CL, 592  
CL1D, 51  
CL22, 61  
CL23, 60  
CL2D, 57, 217  
CL2S, 217  
CL32, 75  
CL33, 75  
CL3D, 67, 217  
CL3I, 75  
CL3Q, 76  
CL3S, 217  
CL3T, 68  
CL92, 78  
CL93, 78  
CLAM, 572  
CLAS, 1097, 1138  
CLAY, 427  
CLB1, 434, 437, 441, 490  
CLB2, 434, 437, 441, 490  
CLB3, 437, 441  
CLB4, 437  
CLD3, 76  
CLD6, 76  
CLEN, 1223, 1248, 1330  
CLIM, 166, 387, 402  
CLMT, 1118  
CLOS, 198  
CLOU, 1310  
CLTU, 52  
CLVF, 663  
CLX1, 217  
CLX2, 217  
CLXS, 192  
CLY1, 217  
CLY2, 217  
CLZ1, 217  
CLZ2, 217  
CMAI, 843  
CMC3, 69, 247  
CMDF, 1110  
CMID, 1153  
CMIN, 526, 529, 558, 577  
CMPX, 271  
CMPY, 271  
CMPZ, 271  
CMU, 572  
CN, 719, 815  
CNA, 558, 577  
CNOD, 239  
CNOR, 1153  
CO, 480  
COA1, 434, 437, 441, 490  
COA2, 434, 437, 439, 441, 490  
COA3, 441  
COA4, 441  
COCY, 248  
CODG, 1118  
COE\_1, 371  
COE\_2, 371  
COE\_3, 371  
COEA, 768, 776  
COEB, 768, 776  
COEF, 235, 237, 271, 529, 533, 540, 548, 552, 681, 722, 748, 1063, 1064  
COEN, 439  
COH, 563  
COH2, 563  
COHE, 243, 443, 843  
COLL, 889, 1323  
COLO, 1262, 1310, 1343  
Color scheme menu, 1374, 1378  
Colored glass, 1355, 1365  
Colored trajectories, 1366  
Colored vectors, 1372  
Colors, 1352  
Colors menu, 1380  
COLS, 1310  
COMM, 421  
COMP, 193, 198, 554, 607, 610, 628, 648, 822, 858, 860, 974, 1182, 1197, 1218, 1223, 1234, 1244, 1248, 1300, 1304  
COMP1, 614  
Component, 1375  
CON2, 563  
COND, 193, 198, 540, 606  
CONE, 190, 193, 198, 225, 802, 836, 1310  
CONF, 587, 722, 1064  
CONS, 563, 997, 1002, 1153  
CONT, 150, 166, 243, 371, 542, 797, 807, 889, 946, 957, 980, 1069, 1080, 1094, 1153, 1168, 1197, 1211, 1223, 1248, 1288, 1300, 1323, 1330, 1401  
Contact normal length menu, 1369  
CONV, 83, 592, 1042, 1046  
COO2, 563  
Coogri, 162  
COOH, 563  
COOR, 103, 1069, 1197, 1211, 1218, 1244

COPO, 1310  
COPP, 1310  
Copper, 1386  
COPT, 843  
COPY, 1277  
Copy as is, 1388  
Copy poster (1800\*1800), 1388  
Copy public. (1600\*1200), 1388  
COQ3, 67  
COQ4, 67  
COQC, 58  
COQI, 74  
COQM, 820, 1323  
COQU, 57, 818, 848, 850, 1011, 1028  
CORR, 478, 485, 1143  
CORT, 247  
COS, 1052, 1055, 1057, 1228  
COSC, 1310  
COSE, 1310  
COTE, 836  
COUC, 962  
COUP, 210, 212, 731, 758, 761, 781, 785, 852, 1310  
COUR, 186, 525, 1197, 1214  
COXA, 742, 744, 746  
COXB, 742, 744, 746  
COYA, 742, 744, 746  
COYB, 742, 744, 746  
COZA, 742, 744, 746  
COZB, 742, 744, 746  
CP, 468, 494, 603, 614, 667  
CPLA, 91  
CPOI, 1310  
CPUT, 1110  
CQ, 592  
CQD3, 76  
CQD4, 76  
CQD6, 76  
CQD9, 76  
CQDF, 946, 978  
CQDX, 176, 179  
CQST, 946, 976  
CQUA, 1300, 1304  
CR, 548, 800  
CR1, 494  
CR2, 494  
CR2D, 485  
CRAK, 946, 986  
CREA, 628, 648  
CRGR, 874  
CRIT, 166, 334, 473, 854  
CROI, 91  
CROS, 814  
CRTM, 470  
CS, 667  
CSHE, 1145  
CSN1, 583  
CSN2, 583  
CSNA, 529  
CSON, 558, 577  
CSPB, 1153  
CSPH, 241  
CSTA, 302, 304, 307, 490, 1110  
CSTE, 243  
CSTI, 811  
CSTR, 1145  
CT, 548, 719  
CTIM, 43, 1078, 1080, 1103, 1106, 1211  
CUB6, 67  
CUB8, 68, 1153  
CUBB, 66  
CUBE, 67, 154, 722, 1064  
CULL, 1288  
CURR, 1080, 1204  
CURV, 166, 198  
CUVF, 73  
CUVL, 70  
CV, 521, 523, 554, 585, 614, 960, 1168  
CV0, 628, 648  
CV1, 607, 628, 648  
CV2, 607, 648  
CV3, 607  
CVEL, 122  
CVL1, 58, 154  
CVME, 563  
CVordp, 628, 648  
CVT1, 583  
CVT2, 583  
CX, 212, 548, 800  
CXX, 552  
CXY, 552  
CY, 212, 548, 800  
CYAN, 1310  
Cyan, 1380  
Cyan plastic, 1386  
Cyan rubber, 1386  
CYAP, 1310  
CYAR, 1310  
CYLI, 164, 193, 198, 225, 801, 836  
CYY, 552  
CZ, 212, 548, 800

- D, 164, 210, 291, 554, 587, 596  
D1, 190  
D2, 190  
DACT, 105, 1176  
DADC, 325  
DAMA, 448, 457  
DASH, 1262, 1288  
DASN, 1310  
DBLE, 843  
DBTE, 228  
DC, 302, 304, 307, 325, 490, 494  
DC1, 353  
DC12, 353  
DC13, 353  
DC2, 353  
DC23, 353  
DC3, 353  
DCEN, 1118  
DCMA, 1118  
DCMS, 501  
DCOU, 526, 529, 1214, 1256  
DCRI, 542, 678  
DDL, 836, 1024  
DEBI, 526, 738, 946, 965  
DEBR, 64, 78, 192, 208, 1122, 1197, 1310  
Debris trajectories, 1381  
DEBU, 494, 1310  
Debug, 1134  
DEBX, 563, 572, 575, 705  
DEBY, 563, 572, 575, 705  
DEBZ, 563, 705  
DECA, 722, 1064  
DECE, 1118  
DECO, 113, 371, 731, 781, 785, 843  
DECX, 271  
DECY, 271  
DEE, 1242  
DEFA, 1310  
Default, 1369, 1370  
Default for paper, 1380  
Default for screen, 1380  
DEFI, 1310, 1343  
DEFO, 1288  
DEGP, 705  
DELE, 208, 811, 824, 825, 915, 930, 995  
DELF, 958  
DELT, 1052  
DEMS, 641  
DENS, 166, 1277  
DEPL, 166, 819, 946, 948, 1004, 1005, 1020, 1069, 1080, 1094, 1145, 1168, 1197, 1211, 1218, 1244, 1288, 1323, 1326, 1330  
DEPO, 960  
DER, 1393  
DERF, 445  
DESC, 1080  
DETA, 231  
DEXT, 186  
DGRI, 208, 811, 824, 825, 915, 930, 995, 1153  
DHAN, 1168  
DHAR, 1259  
DHAS, 1310  
DHR, 603  
DHRO, 603  
DHTE, 228  
DIAG, 1153  
DIAM, 190, 533, 548, 895, 902  
DIAP, 674  
DIFF, 37, 40, 1228, 1310  
DIME, 107, 1206  
DIMN, 1097, 1234  
DIMX, 1097, 1234  
DIRE, 156, 216, 628, 841, 843, 1145  
DIRF, 371  
DIRX, 796, 1097  
DIRY, 796, 1097  
DIRZ, 796, 1097  
DISP, 216, 776, 1153  
DIST, 198, 690, 879, 1101, 1228, 1277  
DIV, 1228  
DIVC, 1228  
DIVG, 1110  
DKT3, 70  
DL, 628  
DLEN, 1055  
DMAS, 946, 983  
DMAX, 841, 874, 885  
DMIN, 895  
DMME, 1242  
DMMN, 1242  
DMOY, 1234  
DOMA, 822, 1176, 1182, 1310  
DOMD, 1172  
DONE, 307, 413  
DOT, 1310  
Dots 1, 1362  
Dots 2, 1362  
Dots 4, 1362  
Dots 8, 1362  
Double, 1369, 1370  
DPAR, 1134  
DPAS, 1134



- DPAX, 1134  
DPCA, 1134  
DPDC, 328  
DPEL, 1134  
DPEM, 1134  
DPGR, 1134  
DPHY, 228  
DPIN, 1153  
DPLA, 91, 371  
DPLE, 1134  
DPLG, 1143  
DPLM, 1134  
DPLS, 434  
DPMA, 709, 751, 1123  
DPMI, 709, 751  
DPRE, 105, 1176  
DPROP, 533  
DPRU, 696, 700, 715  
DPRV, 90  
DPSD, 1134  
DPSF, 418  
DRAG, 210, 212, 930  
Draw hidden as, 1355  
DRIT, 871, 1323  
DROI, 190  
DRPR, 443  
DRST, 90  
DRUC, 284  
DRXC, 776  
DRXN, 776  
DRYC, 776  
DRYN, 776  
DRZC, 776  
DRZN, 776  
DSAT, 228  
DSOR, 233  
DST3, 70  
DT, 1287  
DT1, 1238  
DTBE, 1110  
DTCR, 1242  
DTPB, 1153  
DTST, 1080  
DTUB, 235, 237  
DTUN, 105, 1176  
DTVA, 1110  
DTXC, 776  
DTXN, 776  
DTYC, 776  
DTYN, 776  
DTZC, 776  
DTZN, 776  
Dull iso surfaces, 1375  
DUMP, 1134, 1143, 1148, 1153, 1166, 1168, 1182, 1303  
DUP, 1228  
DVOF, 825  
DX, 193, 198, 225, 233, 1262  
DY, 193, 198, 225, 1262  
DYMS, 1123  
DYNA, 291  
DZ, 193, 198, 225  
  
E, 423, 457  
EA, 648  
EACH, 811  
EACT, 603  
EAFT, 587  
EAU, 533  
ECHO, 82, 1123, 1204  
ECIN, 1097, 1234  
ECLI, 572  
ECOQ, 1094  
ECRC, 1080, 1288  
ECRG, 1097, 1234  
ECRI, 1069, 1207, 1392  
ECRM, 1097, 1234  
ECRN, 1080, 1197, 1218, 1244, 1330  
ECRO, 117, 166, 972, 980, 1069, 1080, 1168, 1197, 1211, 1223, 1248, 1288, 1330  
ED01, 60  
ED1D, 51, 53, 1194  
ED41, 61  
ED\_1, 371  
ED\_2, 371  
ED\_3, 371  
EDCV, 371  
EDEF, 243, 1323  
EDSS, 1118  
EEXT, 1234  
EF, 445, 473  
EFSI, 1080, 1277, 1330

- EGAL, 808  
EIBU, 577  
EINJ, 1234  
EINT, 392, 567, 587, 592, 1234, 1256  
ELAS, 190, 302, 304, 307, 410, 413, 416, 431, 439, 490, 494, 501  
ELAT, 190  
ELCE, 1223  
ELCR, 1242  
ELDI, 70, 243, 852, 1018  
ELEC, 758, 763, 771  
ELEM, 37, 103, 166, 858, 972, 1063, 1079, 1094, 1101, 1131, 1153, 1223, 1288, 1300, 1310, 1401  
Element numbers, 1381  
Element outlines, 1381  
ELGR, 122  
ELIM, 843  
ELMX, 1242  
ELOU, 1310  
ELSN, 122  
ELST, 1242  
EM, 473  
EMAS, 709, 751, 1097, 1234  
EMAX, 1101  
EMER, 1310  
Emerald, 1386  
EMIN, 1101  
END, 1390  
ENDA, 111  
ENDD, 113  
ENDO, 353, 1097  
ENDP, 1195, 1274  
ENEL, 1197, 1223, 1248  
ENER, 558, 946, 960, 1069, 1211  
ENGR, 338, 929, 946, 995, 1173  
ENMA, 533  
ENTH, 672  
ENUM, 1310  
ENVE, 198  
EOBT, 336  
EP, 186  
EP1, 353  
EP12, 353, 371  
EP13, 353  
EP2, 353  
EP23, 353, 371  
EP3, 353  
EP31, 371  
EPAI, 103, 176, 179, 247, 542, 889, 1080, 1197  
EPCD, 316  
EPCS, 316  
EPDV, 1234  
EPS, 1288  
EPS (b&w), 1389  
EPS (color), 1389  
EPS1, 1143  
EPS2, 1143  
EPS3, 1143  
EPS4, 1143  
EPSB, 1288  
EPSD, 302, 304, 307, 490  
EPSI, 552  
EPSL, 334  
EPSM, 1234  
EPST, 166, 1069, 1080, 1168, 1197, 1211, 1223, 1248, 1288, 1330  
EPSY, 501  
EPT1, 313  
EPT2, 313  
EPT3, 313  
EPTR, 313  
EQUI, 152, 533, 538, 946, 981, 984  
EQVD, 1153  
EQVF, 1153  
EQVL, 1153  
ER, 468  
ERCE, 1153  
ERIM, 210, 212  
EROD, 1097, 1277  
EROS, 91, 222, 1103  
ERRI, 1223, 1248, 1330  
ERRO, 1131  
ERUP, 431  
ES, 554  
ESCL, 843  
ESET, 138  
ESLA, 709, 751  
ESUB, 740  
ETA, 416, 418, 556  
ETLE, 1223  
EULE, 91, 139  
EV, 554  
EVAL, 1103  
EXCE, 183, 186, 247  
EXCL, 895, 902, 1140  
EXP, 1228  
EXPA, 606  
EXPC, 1228  
EXPF, 1228  
EXTE, 843, 848, 850  
External Files, 49

- EXTZ, 1288  
EXVL, 563  
EYE, 1306
- FAC4, 1166  
FACE, 235, 237, 843, 915, 1014, 1028, 1153, 1310  
Faces menu, 1361  
FACI, 1153  
FACT, 125, 525, 526, 529, 997, 1004, 1310  
FACX, 125, 1253  
FACY, 125, 1253  
FACZ, 125  
FADE, 1310  
Fading, 1355  
FAIL, 166, 216, 289, 410, 418, 434, 478, 485, 846, 1069, 1080, 1211, 1223, 1288, 1330  
FANF, 1288  
FANT, 386, 1137  
FARF, 1118  
FAST, 811, 1166  
FBAC, 208, 930  
FCON, 889, 1148  
FCTE, 997, 1059  
FDEC, 1069, 1197, 1218, 1326, 1330  
FDYN, 997, 1063  
FELE, 1310  
FEND, 942  
FENE, 1214  
FERR, 91, 811  
FESE, 923, 1323  
FEXT, 1069, 1080, 1094, 1197, 1211, 1218, 1244, 1288, 1326, 1330  
FFMT, 771  
FFRD, 772  
FICH, 142, 175, 269, 793, 946, 947, 994, 1001, 1080, 1094, 1176, 1207, 1209, 1211, 1242, 1253, 1288  
FIEL, 1326, 1330  
Field, 1372, 1375  
FILE, 949, 959  
FILI, 1310  
FILL, 208, 212, 1310  
Fill elements type menu, 1376  
FILT, 1228  
FIMP, 997, 1061  
FIN, 1201  
FINI, 942, 1310, 1343  
FINT, 231, 984, 1069, 1080, 1197, 1211, 1218, 1244, 1288, 1326, 1330  
FIRS, 1045, 1303
- FIXE, 119, 907, 913  
FL23, 61, 154  
FL24, 61, 154  
FL2S, 217  
FL34, 75, 154  
FL35, 75, 154  
FL36, 75, 154  
FL38, 75, 154  
FL3S, 217  
FLFA, 548  
FLIA, 1069, 1080, 1197, 1211, 1218, 1244, 1326, 1330  
FLIR, 1097, 1234  
FLMP, 612, 1143  
FLS, 1153  
FLSR, 825, 1310, 1323  
FLSR domains menu, 1366  
FLSS, 1323  
FLST, 824, 1323  
FLSW, 915, 1080, 1148, 1310, 1323  
FLSW domains menu, 1367  
FLSX, 831, 1323  
FLU1, 61, 154  
FLU3, 74, 154  
FLUI, 166, 192, 208, 387, 392, 515, 581, 824, 825, 831, 915, 930, 1277  
FLUT, 592, 612  
FLUX, 1042, 1044  
FLVA, 1094  
Flying debris menu, 1366  
FNOD, 923  
FNOR, 1153  
FNUM, 690  
FOAM, 445  
FOCO, 758, 762  
Focus on, 1358  
Focus on menu, 1359  
FOLD, 1080  
FOLL, 1218, 1223  
FONC, 164, 307, 533, 540, 672, 678, 748, 758, 760–763, 768, 819, 871, 876–878, 934, 1256  
FOND, 680  
FORC, 683, 758, 760, 1004, 1006, 1022, 1024, 1059, 1061, 1218, 1221, 1247  
FORM, 87, 91, 995, 1080, 1094, 1204, 1209  
FOUR, 744  
FOV, 1306  
FPLT, 729  
FPS, 1300, 1304  
FRAC, 184

- FRAM, 1310  
 FRCO, 322  
 FRED, 1310  
 FREE, 1310  
 Free camera, 1358  
 Free edges, 1381  
 FREQ, 43, 270, 789, 940, 942, 1209, 1259, 1277, 1392, 1393  
 FRIC, 443  
 FRLI, 322  
 FROM, 993, 1115, 1303  
 FRON, 243, 852  
 Front, 1384  
 FROT, 766, 811, 817, 836, 843, 895, 902  
 FRQR, 91  
 FS, 139, 834, 1323  
 FS2D, 58  
 FS3D, 67  
 FS3T, 69  
 FSA, 118, 892, 1140, 1323  
 FSCP, 825, 915  
 FSCR, 1140  
 FSI, 28, 166  
 FSIN, 1288  
 FSMT, 771  
 FSR, 894, 1140, 1323  
 FSRD, 772  
 FSS, 119, 153, 907  
 FSSA, 118  
 FSSF, 118  
 FSSL, 118  
 FSTG, 1323  
 FSUI, 683  
 FTAN, 815  
 FTOT, 984  
 FTRA, 307  
 FUN2, 64  
 FUN3, 77  
 FUNE, 431  
 FVAL, 1214, 1273  
 FVIT, 1171  
  
 G, 423, 427, 962, 1343  
 G1, 503, 506, 509  
 G12, 341, 348, 353, 371, 421, 476  
 G13, 341, 348, 353, 371, 421, 476  
 G2, 503, 506, 509  
 G23, 341, 348, 353, 371, 421, 476  
 G3, 503, 506, 509  
 G4, 503, 506, 509  
 G5, 503, 506, 509  
  
 G6, 503, 506, 509  
 GAH, 563  
 GAH2, 563  
 GAM, 734  
 GAM0, 556, 1145  
 GAM1, 554  
 GAMA, 572, 703–705, 736, 738, 740, 742, 744, 746  
 GAMG, 558, 577  
 GAMI, 825, 915  
 GAMM, 445, 457, 480, 514, 519, 521, 523, 554, 563, 575, 583, 592, 614, 709, 727, 751, 825, 836, 843, 895, 902, 915, 1256, 1310  
 GAMZ, 554  
 GAN2, 563  
 GAO2, 563  
 GAOH, 563  
 GAP, 1153  
 GAR, 529, 558, 577  
 GARD, 1204  
 GAS, 603  
 GAUS, 1080, 1197, 1223, 1248, 1310  
 GAUZ, 1080, 1223, 1310  
 GAZD, 648  
 GAZP, 521, 953, 955  
 GBIL, 225  
 GBU, 529, 558  
 GEAU, 563  
 GENE, 592  
 GENM, 592  
 GEOM, 127, 134, 137, 138, 1310  
 Geometry, 1352  
 Geometry menu, 1357  
 GEOP, 186  
 GF, 501  
 GGAS, 514  
 GGLA, 1310  
 GHOS, 359  
 GIBI, 40, 91, 103  
 GINF, 503, 506, 509  
 GIUL, 1145  
 GLAS, 478, 1310  
 Glass, 1355  
 GLIN, 280  
 GLIS, 118, 817, 843, 867, 1153, 1323  
 GLOB, 161, 1310  
 GLRC, 448  
 GLUE, 885  
 GMIC, 220  
 GO, 1277  
 GOL2, 1310

- GOLD, 1310  
Gold, 1386  
Gold2, 1386  
GOTR, 1277, 1305  
GPCG, 786  
GPDI, 572  
GPIN, 902, 1310, 1323  
Gpinballs menu, 1368  
GPLA, 281  
GPNS, 1153  
GR05, 1310  
GR10, 1310  
GR15, 1310  
GR20, 1310  
GR25, 1310  
GR30, 1310  
GR35, 1310  
GR40, 1310  
GR45, 1310  
GR50, 1310  
GR55, 1310  
GR60, 1310  
GR65, 1310  
GR70, 1310  
GR75, 1310  
GR80, 1310  
GR85, 1310  
GR90, 1310  
GR95, 1310  
GRAD, 166, 946, 961, 1148  
GRAP, 1207, 1214  
GRAV, 423, 427, 946, 962, 965, 1002, 1131  
GRAY, 1310  
GREE, 1310  
Green, 1380  
Green glass, 1355  
Green plastic, 1386  
Green rubber, 1386  
GREP, 1310  
GRER, 1310  
Grey05, 1380  
Grey10, 1380  
Grey15, 1380  
Grey20, 1380  
Grey25, 1380  
Grey30, 1380  
Grey35, 1380  
Grey40, 1380  
Grey45, 1380  
Grey50, 1380  
Grey55, 1380  
Grey60, 1380  
Grey65, 1380  
Grey70, 1380  
Grey75, 1380  
Grey80, 1380  
Grey85, 1380  
Grey90, 1380  
Grey95, 1380  
GRFS, 139, 692  
GRID, 1153  
GRIL, 103, 139, 670  
GROU, 192, 204, 1080, 1310  
GRPS, 122  
GT, 193, 198  
GUID, 796  
GVDW, 585  
GX, 254, 628  
GY, 254, 628  
GZ, 254, 628  
GZPV, 575  
  
H, 206, 298, 628, 962  
H0, 628, 648  
H1, 164  
H2, 164  
Half, 1369, 1370  
HANG, 1310, 1323  
HARD, 895  
HARM, 934, 940  
HBIS, 1310  
HELI, 538  
HEMI, 836  
HEOU, 1310  
HEXA, 147, 1310  
HF, 930  
HFAC, 1310  
HFRO, 1310  
HGQ4, 71, 1118  
HGRI, 208, 811, 824, 825, 915, 930, 995, 1153  
Hidden, 1355  
HIDE, 1310  
Hide, 1370  
Hide all, 1354  
Hide back iso surfaces, 1361  
Hide element outlines, 1361  
Hide front faces, 1361  
Hide info, 1379  
HIGH, 1310  
High, 1385, 1386  
HILL, 476  
HINF, 1310

HIST, 1190  
HMEM, 655  
HOLE, 91  
HOMO, 91  
HOUR, 1115  
HPIN, 1277  
HY, 186  
HYDR, 1008, 1011, 1014, 1016  
HYPE, 460  
HZ, 186  
  
ICLI, 572  
ICOL, 1310  
IDAM, 501  
IDEA, 40, 91, 137, 1094  
IDEN, 1176, 1403  
IDOF, 1259  
IEXT, 709, 751  
IFAC, 1310  
IFIS, 313  
IFSA, 118  
IGRA, 572, 1310  
IGT1, 773  
IGT2, 773  
ILEN, 1223, 1248, 1330  
ILNO, 1218, 1244, 1330  
IMAT, 946, 987, 992  
IMAX, 592  
IMES, 501  
IMIN, 592  
IMPA, 210, 212, 836, 1323  
IMPE, 663  
IMPO, 704, 1118  
IMPU, 1097, 1234  
IMPV, 663  
IMT1, 769  
IMT2, 769  
INCL, 1140  
INCLUDE, 49  
INCR, 271  
INDE, 863, 864, 866, 867, 869, 871  
INDI, 166  
INNER, 254, 758, 760–763, 769, 858, 860  
INFI, 703, 734  
INFO, 1277  
INIS, 503, 506, 509, 786, 789  
INIT, 946, 1244, 1248, 1310  
Initial geometry menu, 1365  
Initial outline, 1381  
Initial wireframe, 1381  
INJE, 944  
  
INJI, 945  
INJM, 945  
INJQ, 945  
INJV, 945  
INOI, 1310  
INSI, 225, 987, 992  
INST, 1024, 1027, 1028  
INT, 1228  
INT4, 59  
INT6, 73  
INT8, 73  
INTE, 181, 822, 843, 1182, 1228, 1308  
Internal faces, 1381  
INTR, 37  
INTS, 771  
INV, 1228  
INVE, 223  
INWI, 1310  
IO, 1110  
IOPT, 105  
IOT, 1110  
IPA1, 786, 789  
IPA2, 786, 789  
IRD1, 772  
IRD2, 772  
IRES, 1097, 1234  
IRX, 274  
IRY, 274  
IRZ, 274  
ISCA, 1310  
ISO, 1310  
Iso default color, 1381  
Iso fill, 1375  
Iso fill elements, 1375  
Iso fill lines, 1375  
Iso lines, 1375  
Iso smooth, 1375  
Iso smooth elements, 1375  
Iso smooth lines, 1375  
Iso surface edges, 1381  
Iso surface outlines, 1381  
Iso surfaces, 1375  
Iso surfaces lines, 1375  
ISOD, 1310  
ISOE, 1310  
ISOL, 1310  
ISOT, 289  
Isovalues, 1352  
Isovalues menu, 1375  
ISPR, 291  
ITEP, 1131

- ITER, 592, 1131  
ITOT, 690  
IXX, 860  
IXY, 860  
IXZ, 860  
IY, 186  
IYY, 860  
IYZ, 860  
IZ, 186  
IZZ, 860  
  
J, 186  
JADE, 1310  
Jade, 1386  
JAUM, 1118  
JAUN, 204, 1262  
JCLM, 490  
JEU1, 839  
JEU2, 839  
JEUX, 839  
JOCO, 434  
JOIN, 1228, 1310  
JONC, 118  
JPRP, 776  
JWL, 567  
JWLS, 587  
JXX, 254  
JXY, 254  
JXZ, 254  
JYY, 254  
JYZ, 254  
JZZ, 254  
  
K, 423, 427, 501, 676, 684, 686, 764  
K0, 423, 427, 628, 770  
K0F, 628  
K200, 1080, 1211, 1214, 1269  
K2CH, 1123  
K2FB, 1123  
K2GP, 1123  
K2MS, 1123  
KBAR, 1148  
KC, 771  
KENE, 1148  
KER, 334  
Keystrokes, 1285, 1350  
KFIL, 91, 138  
KFON, 764  
KFR1, 766  
KFR2, 766  
KFRE, 1300, 1304  
KI, 770  
  
KINT, 592, 1133  
KL, 278  
KMAS, 1148  
KOIL, 603  
KP, 770  
KPER, 515  
KPX, 515  
KPY, 515  
KPZ, 515  
KQDM, 1148  
KR1, 766  
KR2, 766  
KRAY, 276  
KRES, 480  
KRX, 274  
KRXC, 776  
KRXN, 776  
KRY, 274  
KRYC, 776  
KRYN, 776  
KRZ, 274  
KRZC, 776  
KRZN, 776  
KSI, 678  
KSI0, 628  
KSKN, 480  
KT1, 278  
KT2, 278  
KTUB, 552  
KTXC, 776  
KTXN, 776  
KTYC, 776  
KTYN, 776  
KTZC, 776  
KTZN, 776  
KV, 770  
KX, 274, 282  
KY, 274, 282  
KZ, 274, 282  
  
LAGC, 91  
LAGR, 91, 119, 139, 907, 911  
LALP, 1148  
LAM, 423, 427  
LAMB, 334, 1310  
LAPI, 606  
LARG, 193  
LAST, 1045, 1080, 1106, 1303  
LATE, 359  
LAYE, 251, 774  
LBIC, 316

- LBMD, 1190  
LBMS, 1190  
LBNS, 1190  
LBPW, 1190  
LBST, 1190  
LCAB, 817  
LCAM, 1277  
LCD, 316  
LCHP, 45, 176, 179  
LCOF, 118  
LCOU, 1242  
LCS, 316  
LCT, 316  
LDIF, 1310  
LE, 628  
LECD, 46  
LECM, 91  
LECT, 37  
LEES, 91  
Left, 1384  
LEM1, 302  
LENE, 1148  
LENG, 1310  
Length, 1370  
Length menu, 1370, 1373  
LENM, 1256  
LENP, 1256  
LEVE, 987  
Level-set, 1186  
LFEL, 1223, 1248, 1330  
LFEV, 1223, 1248, 1330  
LFNO, 1218, 1244, 1330  
LFNV, 1218, 1244, 1330  
LFUN, 1118  
LIAI, 1145  
LIAJ, 1153  
LIBR, 127, 129, 387, 406, 515, 853, 1110, 1228  
Light ambient, 1385  
Light diffuse, 1385  
Light model ambient, 1386  
Light shininess, 1385  
Light specular, 1385  
Light X, 1384  
Light Y, 1384  
Light Z, 1384  
Lights/Mats, 1352  
Lights/Mats menu, 1383  
LIGN, 144, 965  
LIGR, 74, 758, 766  
LIGX, 1310  
LIGY, 1310  
LIGZ, 1310  
LIMA, 1310  
LIMI, 91, 289, 410, 418, 434, 478, 485  
LINE, 198, 241, 252, 276, 1115, 1262, 1310  
Lines menu, 1361  
LINK, 781, 785, 1182  
Links (coupled) menu, 1370  
LIQU, 546  
LIST, 103, 143–150, 190, 857, 880, 882, 949, 959, 965, 1214, 1271  
LLAG, 1148  
LMAM, 1310  
LMAS, 117, 1148  
LMAX, 825, 895, 915, 1238  
LMC2, 307  
LMIN, 1238  
LMST, 1118  
LN, 1228  
LNKS, 1153, 1310  
LNOD, 118  
LOAD, 997  
LOCA, 198, 776  
LOD1, 742  
LOG, 1123  
LOG10, 1228  
LOI, 256  
LONG, 814  
LONP, 742, 744  
LOOP, 1305  
LOOS, 885  
LORD, 831  
LOW, 1310  
Low, 1385, 1386  
LPH, 317  
LPRE, 1148  
LQDM, 1148  
LSGL, 485  
LSHI, 1310  
LSLE, 1277  
LSPC, 831  
LSPE, 1310  
LSQU, 934  
LT, 193, 198  
LT1, 313  
LT2, 313  
LT3, 313  
LTM, 243  
LTR, 313  
LUNG, 91  
LVEL, 1148  
LX, 217



- LY, 217  
LZ, 217
- M, 423, 427, 494, 1259  
M0, 1148  
MAC1, 1122  
MACN, 1122  
MACR, 212  
MAGE, 1310  
Magenta, 1380  
MAIL, 1401  
Main menu, 1352  
MAIT, 843  
MANU, 1123, 1400, 1403  
MAP2, 59, 921, 1323  
MAP3, 72, 921, 1323  
MAP4, 72, 921, 1323  
MAP5, 59, 921, 1323  
MAP6, 73, 921, 1323  
MAP7, 73, 921, 1323  
MAPB, 946, 994, 1080  
MAPM, 178  
MAPP, 91, 946, 995, 1080  
MAS2, 1153  
MASE, 1197, 1223, 1248  
MASL, 902, 1153  
MASN, 1197, 1218, 1244  
MASS, 177, 243, 254, 271, 274, 384, 396, 626, 690, 722, 860, 944, 1064, 1234, 1326, 1330  
MAST, 166, 854, 885  
MATC, 995  
MATE, 117, 256, 987, 992  
MAVI, 1277, 1303  
MAX, 1228, 1310  
MAXC, 1223, 1248, 1330  
MAXL, 164, 166, 1168  
MAZA, 323  
MBAC, 91  
MBET, 91  
MC, 1142  
MC23, 63, 154  
MC24, 63  
MC34, 77  
MC35, 77  
MC36, 77  
MC38, 77  
MCCS, 1330  
MCEF, 1069  
MCFF, 610  
MCFL, 1069  
MCGP, 607  
MCMA, 944  
MCMF, 1094, 1197, 1218, 1244, 1330  
MCMU, 1069  
MCOM, 946, 974  
MCOU, 341  
MCP1, 1330  
MCP2, 1330  
MCPR, 1197, 1218, 1244, 1330  
MCRO, 1197, 1218, 1244, 1330  
MCTE, 1197, 1218, 1244, 1330  
MCUX, 1218, 1244  
MCUY, 1218, 1244  
MCUZ, 1218, 1244  
MCVA, 1069, 1094, 1211  
MCVC, 1069, 1211  
MCVE, 1094  
MCVI, 1197  
MCVM, 1069  
MCVS, 1069, 1211  
MCXX, 1080, 1094, 1288  
MCxx, 114  
ME1D, 116  
MEAN, 155, 1145, 1228  
MEAS, 1101, 1277  
MEAU, 563  
MEC1, 1323  
MEC2, 1323  
MEC3, 1323  
MEC4, 1323  
MEC5, 1323  
MECA, 69, 91, 118, 159, 160, 387, 399, 758  
MED, 1080  
MEDE, 91  
MEDI, 1310  
Medium, 1385, 1386  
MEDL, 91, 946, 980  
MEMB, 57, 676  
MEMO, 1242  
MEMP, 1242  
MENS, 877, 1323  
MESH, 225, 1094, 1310  
META, 533  
MF1, 457  
MF2, 457  
MFRA, 610, 974  
MFRO, 972  
MFSR, 1140  
MGT1, 773  
MGT2, 773  
MHOM, 117, 552

MIDP, 1153  
MIEG, 556  
MIF, 83, 1262, 1288  
MIN, 1228, 1310  
MINM, 1214  
MINT, 466  
MK, 554  
MLEV, 895  
MLVL, 91  
MMOL, 628, 648  
MMT1, 769  
MMT2, 769  
MNTI, 120, 1078  
MOCC, 758, 763  
MOD, 678, 684, 686  
MODA, 1176  
MODE, 533, 709, 751, 1168  
MODI, 1145  
MODU, 1145  
MOH, 563  
MOH2, 563  
MOME, 1059, 1061  
MOMT, 1118  
MON2, 563  
MOO2, 563  
MOOH, 563  
MOON, 503  
MOPR, 1145  
MORT, 248, 822, 1182  
Mouse events, 1286, 1351  
MOVE, 1228  
MOY4, 1323  
MOY5, 1323  
MOYG, 1228  
MPEF, 848  
MPI, 1080  
MQUA, 1101, 1103  
MRAY, 276  
MRD1, 772  
MRD2, 772  
MS24, 64  
MS38, 77  
MTOT, 254  
MTRC, 1310  
MTTI, 120, 1078  
MTUB, 552  
MU, 509  
MU1, 506  
MU2, 506  
MU3, 506  
MU4, 506  
MUDY, 836, 843, 895, 902  
MUL, 1228  
MULC, 1228  
MULT, 544, 1182  
MUR, 706  
MUST, 836, 843, 895, 902  
MVRE, 1145  
MXIT, 494  
MXLI, 118  
MXSU, 1238  
MXTF, 1024  
MXTP, 1028  
N, 298, 468, 501, 554, 772  
N1, 1259  
NACT, 1238  
NAH2, 526, 684  
NALE, 114  
NASN, 1310  
NAVI, 91, 924, 1166, 1310, 1323  
Navigation menu, 1358  
NAVS, 91  
NBCO, 341, 345  
NBJE, 118  
NBLE, 114  
NBTU, 552  
NBU, 529, 558  
NBUL, 533  
NC, 603  
NCFS, 892, 1148  
NCOL, 1153  
NCOM, 607  
NCOT, 908  
NCOU, 243, 852  
NCT1, 911  
NCT2, 911  
NDDL, 109  
NEAR, 193, 198  
Near plane tolerance menu, 1360  
NEIG, 485  
NELE, 1044, 1046–1048  
NEND, 1234  
NEPE, 120  
NEPX, 91  
NEQV, 1153  
NERO, 1234  
NESP, 614, 628, 648  
NEZ, 836  
NF, 473  
NF34, 1118  
NFAI, 1288

- NFAL, 278  
NFAR, 274  
NFAS, 278  
NFAT, 274, 278, 282  
NFD1, 473  
NFD2, 473  
NFI, 913  
NFKL, 278  
NFKR, 274  
NFKS, 278  
NFKT, 274, 278, 282  
NFKX, 274  
NFKY, 274  
NFKZ, 274  
NFR, 473  
NFRA, 1308  
NFRO, 120  
NFSC, 1140  
NFSL, 688  
NFTO, 1300  
NGAS, 603, 612  
NGAU, 1131  
NGPZ, 115, 181, 184  
NGRO, 197  
NIMA, 1131, 1143  
NIND, 111, 166  
NISC, 1310  
NITE, 980  
NLEN, 1310  
NLHS, 628, 648  
NLIN, 1080  
NLIQ, 612  
NLIT, 448  
NM, 1259  
NMAX, 208, 811, 824, 825, 915, 930, 995, 1153, 1190, 1242  
NNUM, 1310  
No initial geometry, 1365  
No iso, 1375  
No shrinkage, 1363  
No trajectories, 1366  
No vectors, 1372  
NOAD, 1153, 1288  
NOBE, 1277  
NOCG, 902  
NOCL, 1300  
NOCO, 302, 490  
NOCR, 874, 1110  
NOCU, 1153, 1288  
NOD1, 198  
NODE, 37, 193, 198, 216, 223, 722, 843, 1016, 1059, 1061, 1063, 1064, 1101, 1310  
Node numbers, 1381  
NODF, 239  
NODP, 1052  
NODS, 239  
NODU, 1134, 1166, 1168  
NOE1, 839  
NOE2, 839  
NOEC, 82, 1123  
NOEL, 1079, 1288  
NOER, 222  
NOEU, 153, 155–159, 863, 864, 866, 867, 869, 871, 949, 1043, 1049, 1050, 1055, 1057, 1153, 1218, 1300  
NOEX, 926  
NOFB, 208, 930  
NOFO, 1176  
NOGR, 873, 874, 1153, 1288  
NOHO, 1115  
NOHP, 1277  
NOIR, 204, 1262  
NOIS, 1288  
NOLI, 1262  
NOMU, 1182  
NONA, 1166  
NONE, 37, 103, 1145  
NONP, 786  
NONU, 1288  
NOOB, 1288  
NOPA, 934  
NOPO, 1079  
NOPP, 1168  
NOPR, 1123  
NORB, 1153  
NORE, 786, 789, 1182  
NORM, 103, 150, 152, 1153, 1197, 1218, 1223, 1234, 1244, 1288, 1310, 1343  
NOSY, 1288  
NOTE, 1110  
NOUN, 166, 895  
NOUT, 1310  
NOUV, 1403  
NOXL, 1262  
NOYL, 1262  
NPAS, 972, 987, 992, 1110, 1244, 1248, 1274  
NPEF, 118  
NPFR, 120  
NPIN, 111  
NPOI, 109, 111, 113, 908, 911, 913  
NPSF, 1153

- NPTM, 307  
NPTO, 1310  
NPTS, 118  
NR, 798, 805  
NRAR, 1256  
NSET, 37, 138  
NSFA, 1310  
NSPE, 220  
NSPL, 1238  
NSPT, 1238  
NSTE, 1145  
NSTO, 1244, 1248, 1274  
NTHR, 111  
NTIL, 1148  
NTLE, 1218  
NTRA, 1122  
NTSM, 1118  
NTUB, 235, 237  
NU, 276, 280, 281, 298, 302, 304, 307, 384, 389, 410, 413, 416, 418, 423, 427, 431, 434, 439, 443, 445, 457, 468, 478, 485, 490, 494, 501, 509, 1145, 1259  
NU12, 341, 348, 353, 371, 421, 476  
NU13, 348, 353, 371, 421, 476  
NU23, 348, 353, 371, 421, 476  
NUF, 473  
NUFA, 768  
NUFB, 768  
NUFL, 1142  
NUFO, 271, 529, 681, 683, 1061  
NULT, 776  
NUM, 427, 473, 592  
NUME, 1300  
NUPA, 43, 1078, 1209  
NUSE, 1238  
NUSN, 1238  
NUSP, 1238  
NUST, 1238  
NUT, 296  
NVFI, 111  
NVMX, 1242  
NVSC, 1310  
NWAL, 1123  
NWAT, 1123  
NWK2, 1123  
NWSA, 1123  
NWST, 1123  
NWTP, 1123  
NWXP, 1123  
NX, 150, 164, 217, 798, 805, 1052, 1055, 1057, 1103, 1262  
NY, 150, 164, 217, 798, 805, 1052, 1055, 1057, 1103, 1262  
NZ, 150, 164, 217, 798, 805, 1052, 1055, 1057, 1103  
OBJE, 212, 216, 223, 987, 992, 995, 1080, 1101, 1123, 1218, 1223, 1277, 1288, 1310  
Object names, 1381  
Objects, 1352  
Objects menu, 1354  
OBJN, 1288  
OBSI, 1310  
Obsidian, 1386  
OBSO, 1080, 1204  
OCR, 423  
ODMS, 359  
OEIL, 1288  
OF34, 1118  
OFFS, 1288  
OGDE, 506  
OLDS, 1134  
OM1, 300  
OM2, 300  
OMEG, 567, 577, 587, 940, 969  
OMEM, 1288  
ON, 1310  
ONAM, 1310  
ONEF, 91  
OPNF, 87, 90, 1204  
OPOS, 722, 1064  
OPTI, 822, 1108, 1182  
Options menu, 1374  
ORDB, 150  
ORDP, 628, 648  
ORDR, 1142, 1148  
ORIE, 223, 889  
ORIG, 190, 533, 876, 965, 969  
ORSR, 377  
ORTE, 353  
ORTH, 348, 371  
Orthogonal, 1360  
ORTS, 251, 350, 458, 1080  
OTPS, 1148  
OUTL, 1310  
Outline, 1365  
Outlines, 1355  
OUTS, 225, 987, 992  
OUV1, 313  
OUV2, 313  
OUV3, 313  
P, 291

- P0, 427  
 P10, 1288  
 P1X, 801, 803  
 P1Y, 801, 803  
 P1Z, 801, 803  
 P2X, 801, 803  
 P2X2, 1118  
 P2Y, 801, 803  
 P2Z, 801, 803  
 P3X, 803  
 P3Y, 803  
 P3Z, 803  
 PACK, 1153  
 PAIR, 895, 902  
 PAPE, 1310  
 PARA, 307, 387, 389, 392, 396, 399, 402, 406, 577, 934, 937, 943, 1049, 1050  
 PARD, 786, 789  
 PARE, 1310  
 PARF, 287  
 PARG, 529, 558, 577  
 PARO, 540  
 PART, 37, 138, 208, 210, 1110  
 PAS, 37, 42, 1110  
 PAS0, 1148  
 PAS1, 1190  
 PASF, 1190  
 PASM, 1110, 1209  
 PASN, 1310  
 PATH, 90  
 PATM, 727  
 PAXI, 548  
 PB, 592  
 PBAS, 345  
 PBRO, 1310  
 PBU, 529, 558  
 PCAL, 464, 503, 506, 509  
 PCAS, 1123  
 PCD, 316  
 PCHA, 669  
 PCJ, 587  
 PCLD, 166, 1080, 1168, 1310  
 PCOM, 1123  
 PCON, 786, 789, 1310  
 PCOP, 1310  
 PCOU, 1214, 1259  
 PCT, 316  
 PDIS, 709, 751  
 PDOT, 494, 1310  
 PDV, 1097  
 PEAR, 1310  
 Pearl, 1386  
 PELE, 166  
 PELM, 926, 1323  
 PEMA, 166  
 PENA, 843, 895, 902  
 PENE, 1310  
 PENT, 317, 965  
 PEPR, 410, 418, 478, 485  
 PEPS, 166, 410, 418, 478, 485  
 PERF, 453, 1214, 1405  
 PERK, 1214  
 PERP, 1049, 1050, 1310  
 Perp contours, 1381  
 PERR, 166  
 Perspective, 1360  
 PESA, 843  
 PESC, 843  
 PEWT, 1310  
 Pewter, 1386  
 PEXT, 709, 751  
 PFAC, 727  
 PFCT, 748  
 PFEM, 61, 74  
 PFIN, 676, 678, 686  
 PFMA, 1197, 1218, 1330  
 PFMI, 1197, 1218, 1330  
 PFSI, 843, 1197, 1218, 1330  
 PFUN, 727  
 PGAP, 843  
 PGAZ, 558, 577  
 PGOL, 1310  
 PGRI, 1123  
 PH, 473  
 PHAN, 1168  
 PHIC, 480  
 PHID, 940  
 PHII, 480  
 PHIR, 940, 1153  
 PHIS, 825, 915  
 PI, 614  
 PIGL, 869  
 PIGM, 1343  
 PIMP, 672, 704, 736  
 PINB, 895, 1080, 1288, 1310, 1323  
 Pinballs menu, 1364  
 PINC, 1288  
 PINI, 392, 406, 514, 515, 519, 521, 523, 525, 526, 533, 538, 546, 548, 554, 556, 563, 572, 575, 581, 585, 587, 596, 603, 614, 628, 648, 676, 678, 684, 686, 953  
 PINS, 1153, 1310

- PITE, 786, 789  
 PIVO, 864  
 PKAP, 572  
 PL2T, 1080  
 PLAM, 572  
 PLAN, 145, 164, 798, 1055  
 PLAT, 836  
 PLAW, 1123  
 PLAY, 1195, 1274  
 PLEV, 212  
 Plexus, 1462  
 PLIE, 858  
 PLIN, 1110, 1123  
 PLIQ, 558, 577  
 PLOA, 1123  
 PLOG, 1110  
 PM, 607  
 PMAT, 58, 68, 387, 396  
 PMAX, 228, 709, 722, 751, 1064, 1277  
 PMED, 1123  
 PMES, 1123  
 PMET, 105, 1176  
 PMIN, 228, 515, 526, 548, 571, 581, 592, 614, 1277  
 PMOL, 603  
 PMS1, 709, 751  
 PMS2, 709, 751  
 PMTV, 1080  
 PMU, 572  
 PNA, 529, 558, 577  
 PNOL, 1110  
 POCH, 1080, 1204  
 POI1, 801, 803  
 POI2, 801, 803  
 POIN, 127, 193, 198, 223, 798, 802, 805, 843, 858, 877, 878, 972, 1052, 1079, 1094, 1097, 1101, 1299, 1310  
 Points, 1381  
 Points menu, 1362  
 POLA, 127  
 Polished bronze, 1386  
 Polished copper, 1386  
 Polished gold, 1386  
 Polished silver, 1386  
 POMP, 681  
 PONC, 57  
 POPE, 690  
 PORE, 515  
 PORO, 515  
 POS, 994  
 POSI, 1145, 1218, 1223, 1391, 1396  
 POST, 91, 1176  
 POUT, 67  
 POV-Ray, 1288, 1389  
 POVR, 1288, 1310  
 PPLT, 694  
 PPM, 1288  
 PPM (Ascii), 1389  
 PPM (binary), 1389  
 PPMA, 1288  
 PR6, 68  
 PRAD, 548  
 PRAY, 1288  
 PREF, 392, 402, 406, 514, 515, 519, 521, 523, 525, 526, 529, 533, 538, 546, 548, 554, 556, 558, 563, 567, 572, 575, 577, 581, 585, 587, 592, 614, 628, 648, 672, 676, 678, 684, 686, 688, 690, 703–705, 727, 734, 736, 738, 740, 748  
 PRES, 166, 410, 423, 427, 478, 485, 610, 672, 703–705, 734, 736, 738, 740, 742, 744, 746, 974, 1004, 1007, 1022, 1028, 1042, 1048, 1052, 1123, 1277  
 PRGL, 583  
 PRGR, 1123  
 PRIN, 1103, 1123  
 PRIS, 68, 148, 154, 1310  
 PROB, 91  
 PROD, 1228  
 Profiling, 1420  
 PROG, 42, 997, 1022  
 Programming, 1420  
 PROJ, 836, 1310  
 Projection menu, 1360  
 PROT, 1080, 1106, 1391, 1393, 1396  
 PRUP, 676  
 PRVF, 74  
 PRVL, 70  
 PS, 83, 1262, 1288  
 PSAR, 478, 485  
 PSAT, 529, 558, 577  
 PSCR, 1204  
 PSIL, 540, 1310  
 PSL, 628, 962  
 PSYS, 1131  
 PT1D, 412  
 PTOT, 529, 558, 577  
 PTRI, 548  
 PTSL, 962  
 PU, 628  
 PUFF, 554  
 PV, 526

- PVTK, 1080, 1390  
PWD0, 90  
PX, 802, 1103  
PY, 802, 1103  
PYRA, 149  
PYRO, 117, 603  
PYVF, 74  
PZ, 802, 1103  
PZER, 1063
- Q, 614, 1306  
Q41, 62, 154  
Q41L, 62  
Q41N, 62, 154  
Q42, 62, 154  
Q42G, 63  
Q42L, 63  
Q42N, 62, 154  
Q4G4, 71  
Q4GR, 71  
Q4GS, 72  
Q4MC, 73  
Q4VF, 59  
Q92, 59  
Q92A, 60  
Q93, 60  
Q95, 63  
QGEN, 1042, 1045  
QMAX, 709, 751  
QMOM, 751  
QMS, 1277  
QMUR, 572  
QP1, 457  
QP2, 457  
QPPS, 71  
QPRI, 614  
QR1, 494  
QR2, 494  
QTAB, 603, 944  
QUAD, 157, 241, 252, 1115  
QUAL, 1197, 1277  
QUAS, 1115  
QUEL, 186  
Quit, 1352
- R, 186, 193, 198, 225, 473, 585, 592, 607, 628, 648, 771, 825, 915, 1259, 1277, 1287, 1288, 1299, 1343  
R1, 193, 198, 225, 567, 577, 587, 596  
R2, 193, 198, 225, 567, 577, 587, 596  
RACC, 233, 235, 237  
RADB, 776  
RADI, 166, 807, 949, 1042, 1047  
RANG, 592  
RAP, 678  
RATM, 727  
RAYC, 186  
RAYO, 241, 252  
RBIL, 225  
RCEL, 1153  
RCON, 1168  
RCOU, 1214, 1253  
RDK2, 715  
RDMC, 700  
READ, 1123  
REB0, 1153  
REB1, 1153  
REB2, 1153  
REBC, 1153  
RECO, 1148  
RECT, 186  
RECU, 1277  
RED, 1310  
Red, 1380  
Red plastic, 1386  
Red rubber, 1386  
REDP, 1310  
REDR, 1310  
REDU, 485, 758, 763, 772  
REFE, 181, 1118, 1197, 1310  
References menu, 1360  
REGI, 1097, 1234  
REGU, 105, 1176  
RELA, 808, 1323  
REMA, 1400  
RENA, 1253  
REND, 1166, 1288, 1303  
RENU, 786, 789  
REPR, 91, 1391, 1396  
RESE, 225, 241, 252, 1168  
Reset lights, 1383  
Reset materials, 1383  
RESG, 282  
RESI, 494  
RESL, 278  
RESS, 271, 758, 764  
REST, 889  
RESU, 1097, 1204, 1234  
RETI, 1299  
RETO, 142, 269  
RETURN, 49  
REUS, 1288  
REVE, 843

- REWR, 91  
REZO, 1145  
RGAS, 603  
RHO, 953, 962, 1145, 1259  
RHO , 1197  
RIC, 1323  
RIEM, 746  
RIGH, 1306  
Right, 1384  
RIGI, 254, 500, 882, 1059, 1061, 1115, 1153  
RIIL, 1323  
RIIS, 1323  
RISK, 91, 210, 212, 1080, 1097, 1197, 1223, 1330  
RL3D, 72  
RLIM, 241  
RMAC, 212  
RMAS, 1097  
RMAX, 592, 1145  
RMIN, 592  
RNFR, 817  
RNUM, 445  
RO, 210, 276, 280, 281, 298, 300, 302, 304, 307, 353, 371, 389, 392, 406, 410, 413, 416, 418, 421, 423, 427, 431, 434, 437, 439, 441, 443, 468, 476, 478, 485, 490, 494, 500, 501, 503, 506, 509, 514, 515, 519, 521, 523, 525, 526, 546, 548, 552, 554, 556, 567, 575, 581, 585, 587, 592, 606, 665, 667, 672, 676, 681, 683, 684, 686, 690, 702–705, 717, 719, 733, 734, 736, 738, 740, 742, 744, 746  
RO1, 583  
RO2, 583  
RO\_0, 445  
RO\_F, 445  
ROAR, 529, 558, 577  
ROB, 105, 1176  
ROBU, 529, 558, 577  
ROBZ, 577  
ROE, 1142  
ROEX, 709, 751  
ROF, 208, 473, 930  
ROGA, 558, 577  
ROI, 614  
ROIL, 603  
ROIN, 563, 572  
ROLI, 558, 577, 1118  
ROLR, 480  
ROM, 473, 1256  
RONA, 529, 558, 577  
ROP, 742, 744, 1256  
ROR, 548  
ROS, 567, 587, 596  
ROSA, 529, 558, 577  
ROSE, 204, 1262  
ROT, 548  
ROTA, 271, 818, 876, 946, 969, 1002, 1343  
Rotating camera, 1358  
ROTU, 866  
ROUG, 204, 1262  
ROUT, 307, 934, 937, 987, 1022, 1024, 1028  
ROW, 423  
ROX, 548  
ROY, 548  
ROZ, 548  
RPAR, 786, 789  
RPOV, 1288  
RREF, 592  
RRIS, 1234  
RSEA, 529, 686  
RST1, 476  
RST2, 476  
RST3, 476  
RTMANGL, 205  
RTMRCT, 205  
RTMVF, 205  
RUBY, 1310  
Ruby, 1386  
RUDI, 696  
RUGO, 540, 546  
RUPT, 371  
RVAL, 709, 751  
RVIT, 1148  
RX, 1343  
RXDR, 87  
RY, 1343  
RZ, 1343  
RZIP, 1303  
S, 554, 556  
S0, 302, 304, 307, 490  
S24, 64  
S38, 77  
SAFE, 1166  
SAND, 79, 184, 251, 774  
SAUV, 91, 1080, 1106, 1112, 1123, 1392, 1393, 1405  
SAVE, 1288, 1390  
SAXE, 1244, 1248  
SBAC, 1310  
SBEA, 37



- SBOU, 1310  
SCAL, 125, 825, 895, 915, 1310, 1343  
Scale menu, 1373, 1377  
Scaled colored vectors, 1372  
Scaled vectors, 1372  
SCAS, 895  
SCAV, 1310  
SCCO, 1310  
SCEN, 1277, 1310  
SCLM, 91, 105  
SCOU, 1214, 1244, 1248  
SCRN, 1310  
SDFA, 1190  
SECT, 176, 179  
SEGM, 1008, 1028, 1310  
SEGN, 1228  
SELE, 1310, 1343  
Select camera, 1358  
Select groups, 1354  
Select meshes, 1354  
Select points, 1354  
Selected objects, 1386  
SELF, 843, 885, 895, 902, 1153  
SELG, 1310  
SELM, 1310  
SELO, 1310  
SELP, 1310  
SELV, 1310  
SET, 307, 934, 943  
SF, 445  
SFAC, 895, 902, 1310  
SFRE, 1310  
SFSI, 727  
SFT, 186  
SFY, 186  
SFZ, 186  
SG2P, 614  
SGBC, 325  
SGEO, 889  
SGMP, 614, 620  
SH3D, 72, 914, 1323  
SH3V, 72  
SHAD, 1064  
SHAR, 1166, 1310  
Sharp corners, 1381  
SHB8, 70  
SHCG, 902  
SHEA, 448, 1052  
SHEL, 1080  
SHFT, 192  
SHIF, 125, 808  
SHIFT, 1228  
SHIN, 1310  
Shiny iso surfaces, 1375  
SHIX, 125  
SHIY, 125  
SHIZ, 125  
SHOW, 1310  
Show, 1370  
Show all, 1354  
Show all domains, 1366–1368  
Show all links, 1370  
Show back faces, 1361  
Show backface outlines, 1361  
Show blocked fluxes, 1366, 1367  
Show bounding box, 1360  
Show camera values, 1379  
Show center, 1360  
Show cones, 1366–1368  
Show contact joints, 1364, 1368  
Show contact normals, 1364, 1368  
Show contact points, 1364, 1368  
Show contacting descendents, 1364  
Show contacting domains, 1368  
Show couplings, 1366, 1367  
Show debug info, 1379  
Show descendent ASNs, 1364  
Show element numbers, 1379  
Show free edges, 1361  
Show hexahedra, 1366–1368  
Show internal faces, 1361  
Show internal outlines, 1361  
Show iso scale, 1379  
Show iso surface outlines, 1361  
Show link joints, 1370  
Show nodal ASNs, 1364  
Show node numbers, 1379  
Show normals, 1366, 1367  
Show object names, 1379  
Show parent ASNs, 1364, 1368  
Show parents, 1364  
Show penetration rates, 1368  
Show penetrations, 1368  
Show perp contours, 1361  
Show pinball contacts, 1379  
Show pinballs as solid, 1364  
Show prisms, 1366–1368  
Show reference frame, 1360  
Show sharp corners, 1361  
Show spheres, 1366–1368  
Show vectors scale, 1379  
SHRI, 192, 1310

- Shrink 40%, 1363  
Shrink 60%, 1363  
Shrink 80%, 1363  
Shrink 99%, 1363  
Shrink by groups, 1363  
Shrink hidden faces, 1363  
Shrink isolines, 1363  
Shrink node numbers, 1363  
Shrink outlines, 1363  
Shrink pinballs, 1363  
Shrinkage menu, 1363  
SHTU, 1256  
SIGD, 298  
SIGE, 298, 468  
SIGL, 334  
SIGN, 1080, 1197, 1218, 1244, 1330  
SIGP, 445  
SIGS, 334  
SILV, 1310  
Silver, 1386  
SIMP, 554  
SIN, 1228  
SINT, 1310  
SIOU, 1310  
SISM, 1057  
SISO, 1310  
SIVE, 1310  
SIZE, 1288  
SK, 181  
SKEW, 1103  
SKIP, 946, 993, 1143  
SL, 628  
SLAV, 166, 854, 885  
SLER, 1277, 1308  
SLEV, 423, 427  
SLIM, 1153  
SLIN, 489  
SLIP, 152  
SLOW, 1277  
SLPC, 153  
SLPN, 153  
SLZA, 468  
SMAL, 1118  
SMAX, 1228  
SMAZ, 487  
SMEL, 1310  
SMIN, 688, 1228  
SMLI, 1310  
SMOO, 1310  
SMOU, 515  
SNAP, 1080  
SNOD, 923  
SNOR, 1153  
SO1, 353  
SO12, 353  
SO13, 353  
SO2, 353  
SO23, 353  
SO3, 353  
SOL2, 786  
SOLI, 858, 889, 1310, 1323  
SOLU, 494  
SOLV, 786, 789  
SOMM, 1228  
SORD, 1131  
SORT, 91, 1128, 1153, 1207  
SOUR, 525, 1052, 1055, 1057  
SP1, 300  
SP2, 300  
SPCO, 91  
SPEC, 158, 220, 997, 1051  
SPEF, 1323  
SPER, 1310  
SPHC, 71  
SPHE, 164, 193, 198, 225, 800, 807, 953, 955, 994, 1310  
Spheres 1, 1362  
Spheres 2, 1362  
Spheres 4, 1362  
Spheres 8, 1362  
Spheres physical, 1362  
SPHP, 1310  
SPHY, 850  
SPLA, 805  
SPLI, 91, 987, 1080, 1145, 1166, 1204, 1277  
SPLIB, 786, 789  
SPLINE, 887  
SPLT, 786  
SPRE, 1052  
SPTA, 1080  
SQRT, 1228  
SRRF, 434, 490  
SSHA, 1310  
SSHE, 37  
SSOL, 37  
SSWP, 942  
STAB, 1110  
STAC, 698, 764  
STAD, 1153  
STAK, 764  
STAT, 946, 981, 1115, 1153, 1166, 1168  
STEC, 603

- STEL, 1110  
STEP, 1110, 1242, 1259, 1303  
STFL, 192, 217, 825, 915  
STGN, 298  
STIF, 523  
STOP, 1277, 1390, 1401  
STRA, 166  
STRI, 193  
STRP, 241  
STRU, 166, 387, 389, 824, 825, 831, 848, 850, 892, 915, 930, 962, 1176, 1277  
STTR, 1122  
STUB, 690  
STWA, 1142  
SUB, 1228  
SUBC, 1228  
SUIT, 40, 1201  
SUIV, 143  
SULI, 1310  
Superposed elements, 136  
SUPP, 274, 1248, 1310  
SURF, 540, 889, 949, 987, 992, 1288, 1310  
SVAL, 709, 751  
SVIT, 889  
Switch light on, 1383  
SWPD, 942  
SWVA, 690  
SX, 802, 1057  
SY, 802, 1057  
SY\_1, 371  
SY\_2, 371  
SY\_3, 371  
SYMB, 1262  
SYME, 843  
SYMM, 1103  
SYMO, 291  
SYMXX, 1288  
SYMY, 1288  
SYNC, 1142  
SYSC, 1262  
SYXY, 1288  
SYXZ, 1288  
SYYZ, 1288  
SZ, 802, 1057  
  
T, 480, 1244, 1248, 1274, 1343  
T0, 164, 468, 628, 1052, 1055, 1057  
T1, 164  
T12M, 371  
T23M, 371  
T31M, 371  
  
T3GS, 69  
T3MC, 73  
T3VF, 59  
TABL, 307, 811, 934, 936, 943, 1004, 1021, 1080, 1112  
TABO, 742, 744  
TABP, 672  
TABT, 742, 744, 746  
TACH, 758, 763, 773  
TACT, 709, 751  
TAIT, 519  
TAPE, 1103  
TARR, 994  
TAU, 676, 678, 684, 686, 688  
TAU1, 503, 506, 509  
TAU2, 503, 506, 509  
TAU3, 503, 506, 509  
TAU4, 503, 506, 509  
TAU5, 503, 506, 509  
TAU6, 503, 506  
TAUC, 302, 304, 307, 490  
TAUL, 334  
TAUT, 722, 1064  
TAUX, 473, 548  
TBLO, 795, 1323  
Tc, 648  
TCLO, 709, 751  
TCOR, 533  
TCPU, 1242  
TCS, 325  
TD, 722, 1064  
TDEA, 709, 751  
TDEL, 958  
TDET, 596  
TE, 473  
TEAU, 228  
TEKT, 83  
TEMP, 43, 91, 610, 972, 974, 1204  
TEND, 587, 1190  
TERM, 37, 42, 103, 123, 129, 134, 137, 138, 1101, 1277  
TEST, 231, 1110  
TETA, 296, 946, 958  
TETR, 68, 146, 154  
TEVF, 74  
TEXT, 1262, 1310  
Text, 1352, 1381  
Text menu, 1379  
TF, 298  
TFAI, 1190  
TFER, 688

- TFIN, 1190  
TFRE, 43, 1078, 1209, 1277, 1392  
TGA, 1288, 1389  
TGGR, 873  
TGRA, 1148  
TH2O, 228  
THEL, 231  
THETA, 248  
THIC, 1262  
THRS, 166  
TIME, 43, 987, 992, 1078, 1197, 1256, 1277  
TIMP, 1042, 1043, 1206  
TINI, 298, 533, 538, 563, 572, 603, 628, 648, 1190  
TINT, 722, 1064  
TION, 1110  
TIT1, 1346  
TIT2, 1346  
TIT3, 1346  
TITL, 1277, 1346  
TITR, 533, 538  
TM, 468, 494  
TMAX, 166, 228, 628, 648, 940  
TMEN, 878, 1323  
TMIN, 166, 228, 940  
TMUR, 572  
TN, 815  
TNOD, 946, 959  
TNSN, 771  
TO, 1303  
TO1, 353  
TO12, 353  
TO13, 353  
TO2, 353  
TO23, 353  
TO3, 353  
TOIL, 603  
TOL, 198, 494, 1103, 1259  
TOLC, 1118  
TOLE, 1131, 1166, 1182, 1197  
TOLS, 1288  
Top, 1384  
TOPE, 709, 751  
TORE, 803  
TOTA, 1118  
TOUS, 37  
TOUT, 1097  
TPAR, 540  
TPLO, 1080, 1112  
TPOI, 122, 1043–1049  
TQ, 494  
TR, 494  
TR1M, 371  
TR2M, 371  
TR3M, 371  
TR\_1, 371  
TR\_2, 371  
TR\_3, 371  
TRAA, 418  
TRAC, 302, 304, 307, 410, 413, 416, 418, 431, 464, 503, 506, 509, 1214, 1262, 1277, 1288, 1392, 1405  
TRAJ, 210, 212, 1310  
Trajectories, 1366  
TRAN, 271, 814, 1310, 1343  
TRCO, 1310  
TREF, 603, 606  
TRIA, 57, 154  
TRID, 91  
TRIG, 795, 1110, 1168  
TRUP, 676, 678, 684, 686, 696, 925  
TSEU, 563  
TSL, 628  
TSTA, 587, 811  
TTHI, 121  
TUBE, 51, 684, 686  
TUBM, 69, 235, 855  
TULE, 334  
TULT, 776  
TURB, 91  
TURQ, 204, 1262, 1310  
Turquoise, 1386  
TUVF, 52  
TUYA, 52, 186, 857, 1323  
TUYM, 69, 237, 856, 1323  
TVAL, 1168  
TVF, 473  
TVL1, 58, 154  
TVMC, 473  
TX, 1343  
TXTR, 1310  
TY, 1343  
TYPE, 166, 271, 460, 506, 548, 742, 744, 746, 786, 789, 940  
TYPL, 243  
TYVF, 52  
TZ, 1343  
UCDS, 628  
UK, 468  
UNIL, 228, 231  
UNIM, 228, 231

- UNIV, 1080, 1204  
Unselect groups, 1354  
Unselect meshes, 1354  
Unselect points, 1354  
UNSP, 166  
UP, 1306  
Update cameras list, 1358  
UPDR, 1153  
UPDT, 166, 786, 843  
UPTO, 795, 878, 895, 993, 1110, 1115  
UPWM, 1133  
UPWS, 1133  
USER, 1310  
User, 1373, 1376, 1377  
USLG, 1310  
USLM, 1310  
USLP, 1310  
USPL, 1277  
UTIL, 1110, 1287  
UZIP, 1303  
  
V1LC, 248  
V1X, 313  
V1Y, 313  
V1Z, 313  
V2LC, 248  
V2X, 313  
V2Y, 313  
V2Z, 313  
V3X, 313  
V3Y, 313  
V3Z, 313  
VALI, 1197  
VANL, 1142  
VANN, 688  
VARI, 1080, 1094, 1153, 1256, 1277  
VAXE, 969  
VCON, 118  
VCOR, 533  
VCVI, 980, 1080, 1168, 1197, 1223, 1248, 1326, 1330  
VECT, 247, 876–878, 880, 882, 969, 1059, 1061, 1063, 1310  
Vectors, 1352, 1381  
Vectors menu, 1372  
VEL1, 610, 974  
VEL2, 610, 974  
VEL3, 610, 974  
VELO, 1042, 1050  
VEMN, 1097, 1234  
VEMX, 1097, 1234  
  
VERI, 786, 789, 924  
VERR, 863  
VERT, 204, 1262  
VFAC, 1145  
VFCC, 217, 946, 953, 955, 993, 1069, 1148, 1211  
VFLU, 1080, 1145  
VFX, 208, 930  
VFY, 208, 930  
VFZ, 208, 930  
VHAR, 940  
VIAR, 529  
VIBU, 529, 558  
VIDA, 1134  
VIDE, 1153  
VIEW, 1306  
VILI, 558, 577  
VINA, 529  
VINI, 1148  
VIS1, 413  
VIS2, 413  
VIS3, 413  
VIS4, 413  
VIS5, 413  
VIS6, 413  
VISC, 276, 515, 521, 523, 525, 540, 546, 548, 581, 583, 606, 1115, 1148  
VISL, 533, 538, 540  
VISO, 558, 577  
VISU, 731, 1153, 1207, 1274  
VISV, 533, 538, 540  
VITC, 952  
VITE, 166, 291, 819, 946, 949, 1069, 1080, 1094, 1168, 1197, 1211, 1218, 1244, 1288, 1323, 1326, 1330  
VITG, 949, 1218, 1244, 1288, 1326, 1330  
VITP, 742, 744  
VITX, 952, 953  
VITY, 952, 953  
VITZ, 952, 953  
VLIN, 1042, 1050  
VM1D, 412  
VM23, 410  
VMAX, 1242  
VMGR, 300  
VMIS, 186, 286, 287, 289, 291, 296, 410, 434, 478, 485  
VMJC, 434  
VMLP, 437  
VMLU, 439  
VMOY, 1097, 1234

- VMPR, 485  
VMSF, 416  
VMZA, 441  
VNOR, 742, 744, 746  
VOFI, 166, 1143  
VOIS, 241, 252, 863, 864, 866, 867, 869, 871  
VOLU, 233, 818, 915, 944, 987, 992, 1097, 1234  
VPJC, 494  
VPLA, 1042, 1050  
VR, 212  
VRIG, 946, 951  
VRUP, 1057  
VSCA, 1310  
VSWP, 1143  
VTG1, 742, 744, 746  
VTG2, 742, 744, 746  
VTIM, 1049  
VTRA, 969  
VX, 186, 210, 212, 278, 864, 867, 869, 873, 874  
VXD, 587  
VXFF, 592  
VY, 186, 210, 212, 278, 864, 867, 869, 873, 874  
VYD, 587  
VYFF, 592  
VZ, 186, 210, 212, 278, 864, 867, 869, 873, 874  
VZD, 587  
VZFF, 592  
  
WALI, 1123  
WARD, 1223, 1238, 1248  
WARP, 1103  
WATP, 1123  
WAUX, 1223, 1248  
WAVE, 164  
WBC, 1142  
WC, 445, 494  
WCIN, 1197, 1238  
WECH, 1197  
WEXT, 1097, 1197, 1238  
WFIL, 105, 1176  
WGRI, 980  
WHAN, 1168  
WHIP, 1310  
WHIR, 1310  
WHIT, 1310  
White, 1380  
White plastic, 1386  
White rubber, 1386  
Whole mesh, 1386  
WIMP, 1238  
WIN, 83  
Win/Copy, 1352  
Win/Copy menu, 1388  
Window 1024\*768, 1388  
Window 1200\*1200, 1388  
Window 1280\*1024, 1388  
Window 320\*240, 1388  
Window 600\*600, 1388  
Window 640\*480, 1388  
Window 800\*600, 1388  
WINJ, 1097  
WINT, 1097, 1197, 1238  
WIRE, 1310  
Wireframe, 1365  
WK20, 1123  
Write camera, 1358  
WSAU, 1123  
WSTB, 1123  
WSUM, 1238  
WSYS, 1238  
WTOT, 1238  
WTPL, 1123  
WXDR, 87  
WXPL, 1123  
  
X, 164, 198, 210, 722, 1052, 1057, 1064  
X-FEM, 1186  
X0, 193, 198, 217, 225  
X1, 193, 198, 225  
X2, 193, 198, 225  
XAR, 529  
XAXE, 1262, 1267, 1269, 1271  
XB, 193  
XBU, 529  
XCAR, 1186  
XCOR, 533  
XCUB, 1186  
XDET, 587, 596  
XDR, 91, 1080  
XFEM, 1186  
XG, 860  
XGRD, 1262  
XIG, 628  
XINF, 325  
XLOG, 1262  
XLVL, 1080  
XMAX, 1262  
XMGR, 1214, 1267  
XMIN, 1262  
XPLO, 1080  
XT1, 476  
XT2, 476

XT3, 476  
XZER, 1262  
  
Y, 164, 198, 210, 722, 1052, 1057, 1064  
Y0, 193, 198, 217, 225  
Y1, 193, 198, 225  
Y1C, 473  
Y2, 193, 198, 225  
Y2C, 473  
YA, 468  
YB, 193  
YDET, 587, 596  
YELL, 1310  
Yellow, 1380  
Yellow plastic, 1386  
Yellow rubber, 1386  
YELP, 1310  
YELR, 1310  
YETP, 91  
YG, 860  
YG1, 300, 341, 348, 353, 371, 421, 476  
YG2, 300, 341, 348, 353, 371, 421, 476  
YG3, 348, 353, 371, 421, 476  
YGRD, 1262  
YIG, 628  
YLOG, 1262  
YMAS, 628, 648  
YMAX, 468, 1262  
YMIN, 1262  
YOUN, 276, 280, 281, 298, 302, 304, 307, 384,  
389, 410, 413, 416, 418, 431, 434, 437,  
439, 441, 443, 445, 468, 478, 485, 490,  
494, 501, 1145, 1259  
YP, 468  
YSTA, 552  
YZER, 1262  
  
Z, 164, 198, 210, 722, 1052, 1057, 1064  
Z0, 193, 198, 217, 225  
Z1, 193, 198, 225  
Z2, 193, 198, 225  
ZAC0, 304  
ZAC1, 304  
ZAC2, 304  
ZAC3, 304  
ZAC4, 304  
ZALM, 304  
ZAN, 304  
ZB, 193  
ZDET, 587, 596  
ZERO, 1130  
ZETA, 694  
  
ZF, 423  
ZG, 860  
ZIG, 628  
ZIP, 1288  
ZMAX, 341  
ZMIN, 341  
ZONE, 1218, 1223, 1401  
ZOOM, 1299