

AERO-S Manual

Version 1.1

Edition 0.1/25 August 2023

Farhat Research Group (FRG)
Stanford University

This manual was prepared with Texinfo (<http://www.gnu.org/software/texinfo>).

Short Contents

- [AERO-S](#)
- [1 INTRODUCTION](#)
- [2 INSTALLATION](#)
- [3 ACOUSTIC TIME-DOMAIN ARTIFICIAL BOUNDARY *S*](#)
- [4 ACOUSTIC TIME-DOMAIN DIRICHLET BOUNDARY CONDITIONS *S*](#)
- [5 ACOUSTIC TIME-DOMAIN DISTRIBUTED NEUMANN BOUNDARY CONDITION *S*](#)
- [6 ACOUSTIC TIME-DOMAIN NEUMANN BOUNDARY CONDITIONS *S*](#)
- [7 ACOUSTIC TIME-DOMAIN ROBIN DISTRIBUTED BOUNDARY CONDITION *S*](#)
- [8 ACTUATORS *S*](#)
- [9 AEROELASTICITY](#)
- [10 AEROHEAT](#)
- [11 ATTRIBUTES *S*](#)
- [12 BEAM OFFSET](#)
- [13 BEAM REFERENCE FRAMES](#)
- [14 BINARY INPUT / OUTPUT](#)
- [15 BOUNDARY CONVECTION *S*](#)
- [16 BOUNDARY DISPLACEMENTS *S*](#)
- [17 BOUNDARY FLUXES](#)
- [18 BOUNDARY FORCES *S*](#)
- [19 BOUNDARY TEMPERATURES *S*](#)
- [20 BUCKLING](#)
- [21 CLUSTERING SNAPSHOTS OR CONSTRUCTING A REDUCED BASIS](#)
- [22 COMMENTS](#)
- [23 COMPOSITE \(OR ORTHOTROPIC SHELL OR ANISOTROPIC SOLID\)](#)
- [24 COMPOSITE \(ORTHOTROPIC SHELL OR ANISOTROPIC SOLID\) ELEMENT FRAMES](#)
- [25 CONDITION NUMBER](#)
- [26 CONSTRAINTS](#)
- [27 CONSTRUCTING A REDUCED MESH](#)
- [28 CONTACT SURFACES](#)
- [29 CONTROL STATEMENT](#)
- [30 CONWEP](#)
- [31 DENSITY-TEMPERATURE TABLE](#)
- [32 DISCRETE NODAL MASS AND INERTIA](#)
- [33 DYNAMIC ANALYSIS](#)
- [34 EIGENVALUE PROBLEMS](#)
- [35 ELEMENT TOPOLOGY *S*](#)
- [36 ENABLING REDUCTION AND HYPER REDUCTION USING LOCAL REDUCED-ORDER BASES](#)
- [37 END](#)
- [38 EQUATION SOLVER CONFIGURATION](#)
- [39 FIELD WEIGHTS FOR MESH DECOMPOSITION](#)
- [40 FLUID/STRUCTURE INTERFACE](#)
- [41 FORCE TIME TABLE-MECHANICS AND ACOUSTICS](#)
- [42 GEOMETRIC RIGID BODY MODES](#)
- [43 GEOMETRIC STIFFENING DUE TO PRESTRESS](#)
- [44 GLOBAL SIMULATION PARAMETERS](#)
- [45 GRAVITATIONAL ACCELERATION](#)
- [46 GROUPS](#)
- [47 HEAT CONDUCTION COEFFICIENT-TEMPERATURE TABLE](#)
- [48 HEAT ZERO ENERGY MODE](#)
- [49 HEAT ZERO ENERGY MODES FILTER](#)
- [50 HELMHOLTZ](#)
- [51 HELMHOLTZ ANALYSIS \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\)](#)
- [52 HELMHOLTZ ARTIFICIAL ABSORBING BOUNDARY CONDITIONS *S*](#)
- [53 HELMHOLTZ ARTIFICIAL ABSORBING BOUNDARY CONDITIONS \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [54 HELMHOLTZ AUXILIARY SURFACE FAR-FIELD PATTERN \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [55 HELMHOLTZ CONSTRAINED FICTITIOUS NODES \(PARTIALLY AXISYMMETRIC\) *S*](#)
- [56 HELMHOLTZ DIRICHLET BOUNDARY CONDITIONS *S*](#)
- [57 HELMHOLTZ DIRICHLET BOUNDARY CONDITIONS \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [58 HELMHOLTZ DISTRIBUTED NEUMANN BOUNDARY CONDITION *S*](#)
- [59 HELMHOLTZ FAR-FIELD PATTERN \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [60 HELMHOLTZ LINEAR MULTIPONT CONSTRAINTS](#)
- [61 HELMHOLTZ LINEAR MULTIPONT CONSTRAINTS \(PARTIALLY AXISYMMETRIC\) *S*](#)
- [62 HELMHOLTZ LOCATIONS WHERE TO COMPUTE THE KIRCHHOFF INTEGRAL](#)
- [63 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS *S*](#)
- [64 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS - FICTITIOUS DOMAIN \(PARTIALLY AXISYMMETRIC\) *S*](#)
- [65 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS - REAL DOMAIN \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [66 HELMHOLTZ NODES BELOW AUXILIARY SURFACE \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [67 HELMHOLTZ SCATTERER BOUNDARY *S*](#)
- [68 HELMHOLTZ WAVE NUMBER \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [69 HELMHOLTZ WET INTERFACE BOUNDARY *S*](#)
- [70 HYDROELASTIC FLUID/STRUCTURE BOUNDARY](#)
- [71 HYDROELASTIC FREE SURFACE BOUNDARY](#)
- [72 HYPER REDUCTION OF THE INERTIA FORCES ASSOCIATED WITH A CONSTANT MASS MATRIX](#)
- [73 IMPEDANCE ANALYSIS](#)
- [74 INITIAL ACCELERATIONS \(Not Supported Yet\)](#)
- [75 INITIAL DISPLACEMENTS *S*](#)
- [76 INITIAL DISPLACEMENT 6 COLUMNS \(IDISP6 completely spelled out\) *S*](#)
- [77 INITIAL OR SEED DISPLACEMENT FOR PITA \(IDISP6PITA completely spelled out\)](#)

- [78 INITIAL OR SEED VELOCITY FOR PITA \(IVEL6PITA completely spelled out\)](#)
- [79 INITIAL TEMPERATURES *S*](#)
- [80 INITIAL VELOCITIES *S*](#)
- [81 INTRUSIVE POLYNOMIAL CHAOS](#)
- [82 LINEAR MATERIAL PROPERTIES *S*](#)
- [83 LINEAR MULTIPoint CONSTRAINTS FOR MECHANICAL ANALYSIS](#)
- [84 LOAD](#)
- [85 LOADCASE DEFINITION](#)
- [86 LUMPED](#)
- [87 MASS EVALUATION](#)
- [88 MATERIAL LAW](#)
- [89 MATERIAL LAW USAGE](#)
- [90 MESH DECOMPOSITION](#)
- [91 NODAL CONTACT](#)
- [92 NODAL FRAMES](#)
- [93 NODES *S*](#)
- [94 NON INTRUSIVE POLYNOMIAL CHAOS](#)
- [95 NONLINEAR ANALYSIS](#)
- [96 NONLINEAR SUPERPOSITION OF RIGID AND DEFORMATION MODES](#)
- [97 OUTPUT OF RESULTS \(OUTPUT completely spelled out\)](#)
- [98 OUTPUT OF RESULTS 6 COLUMNS \(OUTPUT6 completely spelled out\)](#)
- [99 PARALLEL-IN-TIME ALGORITHM \(PITA\)](#)
- [100 PRELOAD](#)
- [101 PRESSURE](#)
- [102 PRINTMAT](#)
- [103 QUASISTATICS ANALYSIS](#)
- [104 RANDOM](#)
- [105 READING REDUCED BASES](#)
- [106 RECONSTRUCTING THE SOLUTION FROM THE REDUCED COORDINATES](#)
- [107 RENUMBERING](#)
- [108 RESTART](#)
- [109 RIGID BODY \(AND OTHER ZERO ENERGY\) MODES](#)
- [110 RIGID BODY MODES FILTER](#)
- [111 RUBBER DAMPING PROPERTIES TABLE](#)
- [112 SAVING EIGENMODES OR PROJECTING ONTO A BASIS](#)
- [113 SENSITIVITY ANALYSIS \(SENSITIVITY completely spelled out\)](#)
- [114 SENSORS *S*](#)
- [115 SLOSHING PROBLEMS](#)
- [116 SLOSHING ZERO ENERGY MODE](#)
- [117 SOURCE TIME TABLE-HEAT CONDUCTION](#)
- [118 SPECIFIC HEAT COEFFICIENT-TEMPERATURE TABLE](#)
- [119 STATICS](#)
- [120 STRESS-STRAIN ONE-DIMENSIONAL TABLE](#)
- [121 STRESS-STRAIN TWO-DIMENSIONAL TABLE](#)
- [122 STRUCTURAL DAMPING LOSS FACTOR TABLE](#)
- [123 SURFACE TOPOLOGY](#)
- [124 THERMAL EXPANSION TEMPERATURE TABLE](#)
- [125 THERMOELASTICITY](#)
- [126 TIED SURFACES](#)
- [127 USER DEFINED FORCES *S*](#)
- [128 USER DEFINED PRESCRIBED DISPLACEMENTS *S*](#)
- [129 WEIGHTS](#)
- [130 YOUNG MODULUS-STRAIN TABLE](#)
- [131 YOUNGS MODULUS-TEMPERATURE TABLE](#)
- [132 YIELD STRESS-EFFECTIVE PLASTIC STRAIN TABLE](#)
- [133 YIELD STRESS SCALING FACTOR-EFFECTIVE PLASTIC STRAIN RATE TABLE](#)

Table of Contents

- [AERO-S](#)
- [1 INTRODUCTION](#)
- [2 INSTALLATION](#)
- [3 ACOUSTIC TIME-DOMAIN ARTIFICIAL BOUNDARY *S*](#)
- [4 ACOUSTIC TIME-DOMAIN DIRICHLET BOUNDARY CONDITIONS *S*](#)
- [5 ACOUSTIC TIME-DOMAIN DISTRIBUTED NEUMANN BOUNDARY CONDITION *S*](#)
- [6 ACOUSTIC TIME-DOMAIN NEUMANN BOUNDARY CONDITIONS *S*](#)
- [7 ACOUSTIC TIME-DOMAIN ROBIN DISTRIBUTED BOUNDARY CONDITION *S*](#)
- [8 ACTUATORS *S*](#)
- [9 AEROELASTICITY](#)
- [10 AEROHEAT](#)
- [11 ATTRIBUTES *S*](#)
- [12 BEAM OFFSET](#)
- [13 BEAM REFERENCE FRAMES](#)
- [14 BINARY INPUT / OUTPUT](#)
- [15 BOUNDARY CONVECTION *S*](#)
- [16 BOUNDARY DISPLACEMENTS *S*](#)
- [17 BOUNDARY FLUXES](#)
- [18 BOUNDARY FORCES *S*](#)
- [19 BOUNDARY TEMPERATURES *S*](#)
- [20 BUCKLING](#)

- [21 CLUSTERING SNAPSHOTS OR CONSTRUCTING A REDUCED BASIS](#)
- [22 COMMENTS](#)
- [23 COMPOSITE \(OR ORTHOTROPIC SHELL OR ANISOTROPIC SOLID\)](#)
- [24 COMPOSITE \(ORTHO TROPIC SHELL OR ANISOTROPIC SOLID\) ELEMENT FRAMES](#)
- [25 CONDITION NUMBER](#)
- [26 CONSTRAINTS](#)
- [27 CONSTRUCTING A REDUCED MESH](#)
- [28 CONTACT SURFACES](#)
- [29 CONTROL STATEMENT](#)
- [30 CONWEP](#)
- [31 DENSITY-TEMPERATURE TABLE](#)
- [32 DISCRETE NODAL MASS AND INERTIA](#)
- [33 DYNAMIC ANALYSIS](#)
- [34 EIGENVALUE PROBLEMS](#)
- [35 ELEMENT TOPOLOGY *S*](#)
- [36 ENABLING REDUCTION AND HYPER REDUCTION USING LOCAL REDUCED-ORDER BASES](#)
- [37 END](#)
- [38 EQUATION SOLVER CONFIGURATION](#)
- [39 FIELD WEIGHTS FOR MESH DECOMPOSITION](#)
- [40 FLUID/STRUCTURE INTERFACE](#)
- [41 FORCE TIME TABLE-MECHANICS AND ACOUSTICS](#)
- [42 GEOMETRIC RIGID BODY MODES](#)
- [43 GEOMETRIC STIFFENING DUE TO PRESTRESS](#)
- [44 GLOBAL SIMULATION PARAMETERS](#)
- [45 GRAVITATIONAL ACCELERATION](#)
- [46 GROUPS](#)
- [47 HEAT CONDUCTION COEFFICIENT-TEMPERATURE TABLE](#)
- [48 HEAT ZERO ENERGY MODE](#)
- [49 HEAT ZERO ENERGY MODES FILTER](#)
- [50 HELMHOLTZ](#)
- [51 HELMHOLTZ ANALYSIS \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\)](#)
- [52 HELMHOLTZ ARTIFICIAL ABSORBING BOUNDARY CONDITIONS *S*](#)
- [53 HELMHOLTZ ARTIFICIAL ABSORBING BOUNDARY CONDITIONS \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [54 HELMHOLTZ AUXILIARY SURFACE FAR-FIELD PATTERN \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [55 HELMHOLTZ CONSTRAINED FICTITIOUS NODES \(PARTIALLY AXISYMMETRIC\) *S*](#)
- [56 HELMHOLTZ DIRICHLET BOUNDARY CONDITIONS *S*](#)
- [57 HELMHOLTZ DIRICHLET BOUNDARY CONDITIONS \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [58 HELMHOLTZ DISTRIBUTED NEUMANN BOUNDARY CONDITION *S*](#)
- [59 HELMHOLTZ FAR-FIELD PATTERN \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [60 HELMHOLTZ LINEAR MULTIPONT CONSTRAINTS](#)
- [61 HELMHOLTZ LINEAR MULTIPONT CONSTRAINTS \(PARTIALLY AXISYMMETRIC\) *S*](#)
- [62 HELMHOLTZ LOCATIONS WHERE TO COMPUTE THE KIRCHHOFF INTEGRAL](#)
- [63 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS *S*](#)
- [64 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS - FICTITIOUS DOMAIN \(PARTIALLY AXISYMMETRIC\) *S*](#)
- [65 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS - REAL DOMAIN \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [66 HELMHOLTZ NODES BELOW AUXILIARY SURFACE \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [67 HELMHOLTZ SCATTERER BOUNDARY *S*](#)
- [68 HELMHOLTZ WAVE NUMBER \(AXISYMMETRIC/PARTIALLY AXISYMMETRIC\) *S*](#)
- [69 HELMHOLTZ WET INTERFACE BOUNDARY *S*](#)
- [70 HYDROELASTIC FLUID/STRUCTURE BOUNDARY](#)
- [71 HYDROELASTIC FREE SURFACE BOUNDARY](#)
- [72 HYPER REDUCTION OF THE INERTIA FORCES ASSOCIATED WITH A CONSTANT MASS MATRIX](#)
- [73 IMPEDANCE ANALYSIS](#)
- [74 INITIAL ACCELERATIONS \(Not Supported Yet\)](#)
- [75 INITIAL DISPLACEMENTS *S*](#)
- [76 INITIAL DISPLACEMENT 6 COLUMNS \(IDISP6 completely spelled out\) *S*](#)
- [77 INITIAL OR SEED DISPLACEMENT FOR PITA \(IDISP6PITA completely spelled out\)](#)
- [78 INITIAL OR SEED VELOCITY FOR PITA \(IVEL6PITA completely spelled out\)](#)
- [79 INITIAL TEMPERATURES *S*](#)
- [80 INITIAL VELOCITIES *S*](#)
- [81 INTRUSIVE POLYNOMIAL CHAOS](#)
- [82 LINEAR MATERIAL PROPERTIES *S*](#)
- [83 LINEAR MULTIPONT CONSTRAINTS FOR MECHANICAL ANALYSIS](#)
- [84 LOAD](#)
- [85 LOADCASE DEFINITION](#)
- [86 LUMPED](#)
- [87 MASS EVALUATION](#)
- [88 MATERIAL LAW](#)
- [89 MATERIAL LAW USAGE](#)
- [90 MESH DECOMPOSITION](#)
- [91 NODAL CONTACT](#)
- [92 NODAL FRAMES](#)
- [93 NODES *S*](#)
- [94 NON INTRUSIVE POLYNOMIAL CHAOS](#)
- [95 NONLINEAR ANALYSIS](#)
- [96 NONLINEAR SUPERPOSITION OF RIGID AND DEFORMATION MODES](#)
- [97 OUTPUT OF RESULTS \(OUTPUT completely spelled out\)](#)
- [98 OUTPUT OF RESULTS 6 COLUMNS \(OUTPUT6 completely spelled out\)](#)
- [99 PARALLEL-IN-TIME ALGORITHM \(PITA\)](#)
- [100 PRELOAD](#)
- [101 PRESSURE](#)

- [102 PRINTMAT](#)
- [103 QUASISTATICS ANALYSIS](#)
- [104 RANDOM](#)
- [105 READING REDUCED BASES](#)
- [106 RECONSTRUCTING THE SOLUTION FROM THE REDUCED COORDINATES](#)
- [107 RENUMBERING](#)
- [108 RESTART](#)
- [109 RIGID BODY \(AND OTHER ZERO ENERGY\) MODES](#)
- [110 RIGID BODY MODES FILTER](#)
- [111 RUBBER DAMPING PROPERTIES TABLE](#)
- [112 SAVING EIGENMODES OR PROJECTING ONTO A BASIS](#)
- [113 SENSITIVITY ANALYSIS \(SENSITIVITY completely spelled out\)](#)
- [114 SENSORS *S*](#)
- [115 SLOSHING PROBLEMS](#)
- [116 SLOSHING ZERO ENERGY MODE](#)
- [117 SOURCE TIME TABLE-HEAT CONDUCTION](#)
- [118 SPECIFIC HEAT COEFFICIENT-TEMPERATURE TABLE](#)
- [119 STATIC](#)
- [120 STRESS-STRAIN ONE-DIMENSIONAL TABLE](#)
- [121 STRESS-STRAIN TWO-DIMENSIONAL TABLE](#)
- [122 STRUCTURAL DAMPING LOSS FACTOR TABLE](#)
- [123 SURFACE TOPOLOGY](#)
- [124 THERMAL EXPANSION TEMPERATURE TABLE](#)
- [125 THERMOELASTICITY](#)
- [126 TIED SURFACES](#)
- [127 USER DEFINED FORCES *S*](#)
- [128 USER DEFINED PRESCRIBED DISPLACEMENTS *S*](#)
- [129 WEIGHTS](#)
- [130 YOUNG MODULUS-STRAIN TABLE](#)
- [131 YOUNGS MODULUS-TEMPERATURE TABLE](#)
- [132 YIELD STRESS-EFFECTIVE PLASTIC STRAIN TABLE](#)
- [133 YIELD STRESS SCALING FACTOR-EFFECTIVE PLASTIC STRAIN RATE TABLE](#)

Up: [\(dir\)](#)

AERO-S

- [ATDARB](#)
- [ADDIR](#)
- [ADDNB](#)
- [ADNEU](#)
- [ADROB](#)
- [ACTUATORS](#)
- [AERO](#)
- [AFROH](#)
- [ATTRIBUTES](#)
- [BOFFSET](#)
- [EFRAMES](#)
- [BINARY](#)
- [CONVECTION](#)
- [DISPLACEMENTS](#)
- [FLUX](#)
- [FORCES](#)
- [TEMPERATURES](#)
- [BUCKLE](#)
- [ROBC](#)
- [COMPOSITE](#)
- [CFRAMES](#)
- [CONDITION](#)
- [CONSTRAINTS](#)
- [RMSHC](#)
- [CONTACTSURFACES](#)
- [CONTROL](#)
- [CONWEP](#)
- [DCIT](#)
- [DIMASS](#)
- [DYNAMICS](#)
- [EIGEN](#)
- [TOPOLOGY](#)
- [LOCROB](#)
- [SOLVERCNTL](#)
- [FWEIGHTS](#)
- [FSINTERFACE](#)
- [MFIT](#)
- [GRBM](#)
- [GEPS](#)
- [PARAMETERS](#)
- [GRAVITY](#)
- [GROUPS](#)
- [CCTT](#)

- [HZEM](#)
- [HZEMFILTER](#)
- [HELMHOLTZ](#)
- [AXIHELMHOLTZ](#)
- [HARB](#)
- [AXIHARB](#)
- [AXIHASFFP](#)
- [AXIHCFCNO](#)
- [HDIR](#)
- [AXIHDIR](#)
- [HDNB](#)
- [AXIHFP](#)
- [HLMPC](#)
- [AXIHLMP](#)
- [KIRLOC](#)
- [HNEU](#)
- [AXIHNEUF](#)
- [AXIHNEURD](#)
- [AXIHNOBAS](#)
- [HSCB](#)
- [AXIHWAN](#)
- [HWIB](#)
- [HEFSB](#)
- [HEFRS](#)
- [WEIGHTMASS](#)
- [IMPEDANCE](#)
- [IACCELERATIONS](#)
- [IDISPLACEMENTS](#)
- [IDISP6](#)
- [IDISP6PITA](#)
- [IVEL6PITA](#)
- [ITEMPERATURES](#)
- [IVELOCITIES](#)
- [INPC](#)
- [MATERIAL](#)
- [LMP](#)
- [LOAD](#)
- [LOADCASE](#)
- [LUMPED](#)
- [MASS](#)
- [MATLAW](#)
- [MATUSAGE](#)
- [DECOMPOSE](#)
- [NODALCONTACT](#)
- [NFRAMES](#)
- [NODES](#)
- [NONINPC](#)
- [NONLINEAR](#)
- [FLOATINGFRAME](#)
- [OUTPUT](#)
- [OUTPUT6](#)
- [PITA](#)
- [PRELOAD](#)
- [PRESSURE](#)
- [PRINTMAT](#)
- [QSTATIC](#)
- [RANDOM](#)
- [READMODE](#)
- [RODC](#)
- [RENUMBERING](#)
- [RESTART](#)
- [TRBM](#)
- [RBMFILTER](#)
- [RUBDAFT](#)
- [MODE](#)
- [SENSITIVITY](#)
- [SENSORS](#)
- [SLOSH](#)
- [SSEM](#)
- [HFFT](#)
- [SCFT](#)
- [STATICS](#)
- [SS1DT](#)
- [SS2DT](#)
- [SDETAFT](#)
- [SURFACETOPO](#)
- [TETT](#)
- [THERMOE](#)
- [TIEDSURFACES](#)
- [USDF](#)
- [USDD](#)

- [WEIGHTS](#)
- [YMSL](#)
- [YMTT](#)
- [YSST](#)
- [YSSFSRT](#)

Next: [INSTALLATION](#)

1 INTRODUCTION

This section compiles basic information on how to prepare an input data for a finite element analysis using the **AERO-S** code, and how to run this code.

The input data consists of command macros, clustered numerical data, and comment lines. Each data cluster is preceded by a command statement. The following rules of thumb are suggested:

- 1 The users are allowed to have as many comment lines as they wish, as long as the first column of each line starts with * (see [COMMENTS](#)).
- 2 The subsequent data clusters may appear in any order. The command statement preceding each cluster may be in lower, upper, or mixed characters.
- 3 The data in each cluster are free format.
- 4 **Only the first four letters of a command need to be specified unless otherwise noted.**

Any segment of a **AERO-S** input file — for example, the data corresponding to any command macro, or a command macro and its data, or any number of input lines — can be replaced by a statement of the form

`[INCLUDE filename (or "filename")]`

where `filename` is the name of the file containing the information. "`filename`" can contain a path or can be replaced by `<filename>` in which case the path is that of the environment variable `\$FEM_INCLUDE`. Furthermore, the included file `filename` can be compressed by gzip, bzip2, or zip.

Throughout this User's Reference Manual, the `*S*` symbol is used to designate a command that is distributed according to a given mesh partition when **SOWER** is applied to **AERO-S**'s standard ASCII Input Command Data file in order to decompose it into binary files for (parallel) distributed I/O (see **SOWER**'s User's Reference Manual). A change in the data following any command designated by this symbol requires re-executing **SOWER**.

AERO-S can be executed in serial or parallel mode. However, if the requested finite element analysis requires an equation solver, **AERO-S** runs in parallel mode on a given parallel architecture only if the chosen equation solver (see [STATICS](#)) can be executed in parallel mode on that parallel architecture. In the latter case, performing the core finite element analysis — in addition to the solution of the system(s) of equations — in parallel using MPI and/or OpenMP requires partitioning the finite element mesh using the [DECOMPOSE](#) command. If on the other hand the requested finite element analysis does not require an equation solver, **AERO-S** can be executed in parallel mode on either a shared or distributed memory system as long as the finite element mesh is partitioned using the [DECOMPOSE](#) command.

The command line for executing **AERO-S** in serial mode is

aeros

<code>[-d <decomposition_pathandfilename>]</code>	(Specifies a decomposition file).
<code>[-v <verbose_frequency>]</code>	(Turns on verbose and specifies frequency of printing on screen the FETI iteration count and subspace iteration count).
<code>[-c]</code>	(Outputs contact status on screen (FETI solver)).
<code>[-t]</code>	(Converts input file to XPost format).
<code>[-T]</code>	(Converts input file to XPost format after removing all numbering gaps).
<code>[-m]</code>	(Converts input file to XPost format after gathering each material in a separate element set).
<code>[-M]</code>	(Converts input file to XPost format after removing all numbering gaps and gathering each material in a separate element set).
<code>[-P]</code>	(Generates automatically XPost patterns for the various XPost element sets. This option is useful only in conjunction with the <code>-m</code> and <code>-M</code> options which can generate multiple XPost element sets. Also, automatically generates a global element set).
<code><filename.aeros.aicdf></code>	(AERO-S ASCII Input Command Data file containing the finite element structural model and analysis commands).

The command line for executing **AERO-S** in parallel mode on a shared memory system using OpenMP and `number_of_threads` threads is

aeros -n number_of_threads

```

[-d <decomposition_pathandfilename>]
[-v <verbose_frequency>]

[-c]
[-t]
[-T]

[-m]
[-M]

[-P]

<filename.aeros.aicdf>

```

(Specifies a decomposition file).
(Turns on verbose and specifies frequency of printing on screen the FETI iteration count and subspace iteration count).
(Outputs contact status on screen (FETI solver)).
(Converts input file to **XPost** format).
(Converts input file to **XPost** after removing all numbering gaps).
(Converts input file to **XPost** format after gathering each material in a separate element set).
(Converts input file to **XPost** format after removing all numbering gaps and gathering each material in a separate element set).
Generates automatically **XPost** patterns for the various **XPost** element sets. This option is useful only in conjunction with the -m and -M options which can generate multiple **XPost** element sets. Also, automatically generates a global element set).
(**AERO-S** ASCII Input Command Data file containing the finite element structural model and analysis commands).

The command line for executing **AERO-S** in parallel mode on a distributed system using MPI with `number_of_MPI_processes` MPI processes [and `number_of_threads_within_an_MPI_process` threads within an MPI process] is

```
mpirun -np number_of_MPI_processes aeros -n number_of_threads_within_an_MPI_process
```

```

[-d <decomposition_pathandfilename>]
[-v <verbose_frequency>]

[-c]
[-t]
[-T]

[-m]
[-M]

[-P]

<filename.aeros.aicdf>

```

(Specifies a decomposition file).
(Turns on verbose and specifies frequency of printing on screen the FETI iteration count and subspace iteration count).
(Outputs contact status on screen (FETI solver)).
(Converts input file to **XPost** format).
(Converts input file to **XPost** after removing all numbering gaps).
(Converts input file to **XPost** format after gathering each material in a separate element set).
(Converts input file to **XPost** format after removing all numbering gaps and gathering each material in a separate element set).
Generates automatically **XPost** patterns for the various **XPost** element sets. This option is useful only in conjunction with the -m and -M options which can generate multiple **XPost** element sets. Also, automatically generates a global element set).
(**AERO-S** ASCII Input Command Data file containing the finite element structural model and analysis commands).

Next: [ATDARB](#), Previous: [INTRODUCTION](#)

2 INSTALLATION

The installation of **AERO-S** on a given computing system requires the availability on that system of the following tools:

<code>C++ compiler g++</code>	Version 4.1.2 or higher.
<code>Fortran compiler gfortran</code>	Version 4.1.2 or higher.
<code>Flex utility</code>	Version 2.5 or higher. <code>Flex</code> is a lexical analyser required for building the parser of AERO-S 's input command data file.
<code>Bison utility</code>	Version 2.3 or higher. <code>Bison</code> is a parser generator required for building the parser of AERO-S 's input command data file.
<code>CMake utility</code>	Version 2.6 or higher. <code>CMake</code> is a cross-platform open-source build system. It is comparable to the Unix <code>Make</code> program in that the build process is ultimately controlled by configuration files (<code>CMakeLists.txt</code>). However unlike <code>Make</code> , it does not directly build the final software but instead generates standard build files such as <code>makefiles</code> for Unix and projects/workspaces for Windows Visual C++. The <code>cmake</code> version 2.6 utility can be obtained from http://www.cmake.org . (Note: a "README.cmake" file discussing details on <code>cmake</code> options for code configuration and installation is available in the directory containing the source code of AERO-S).

and following libraries:

<code>BLAS library</code>	<code>BLAS</code> is a set of Basic Linear Algebra Subprograms required by various operations performed in AERO-S .
<code>LAPACK library</code>	<code>LAPACK</code> is a high-performance Linear Algebra PACKAGE with advanced solvers.
<code>MPI library openmpi</code>	Version 1.2.6 or higher. <code>open MPI</code> is a high-performance implementation of the Message Passing Interface (MPI) required for performing interprocessor communication, among others. More specifically, AERO-S requires an MPI-2 implementation such as the one provided by the Open MPI project.

OpenMP API Open Multi-Processing is an Application Programming Interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++ and Fortran on many architectures, including Unix. As an option, **AERO-F** can be compiled with OpenMP to enable multi-threaded execution.

In addition, the following optional libraries extend the capabilities of **AERO-S**:

SPOOLES library	SPOOLES is a library for solving sparse real and complex linear systems of equations with a sparse direct solver, written in the C language using object oriented design.
MUMPS library	MUMPS is a library for solving sparse real and complex linear systems of equations with a multifrontal massively parallel sparse direct solver.
SUPERLU library	SUPERLU is a general purpose library for the direct solution of large sparse nonsymmetric systems on high performance machines.
ARPACK library	ARPACK is the Arnoldi PACKage for the solution of large-scale symmetric, nonsymmetric, and generalized eigenproblems.
ScalAPACK library	ScalAPACK is also known as the Scalable LAPACK. This library includes a subset of LAPACK routines redesigned for distributed memory MIMD parallel computers.
BLACS library	BLACS (Basic Linear Algebra Communication Subprograms) is a linear algebra oriented message passing interface designed for linear algebra.
PARPACK library	PARPACK is the parallel version of ARPACK used by AERO-S 's parallel eigensolver.
METIS library	METIS is a library of graph manipulation routines that can be used by AERO-S for reordering of a sparse matrix to reduce the number of fill-in entries created during factorization.
Zoltan library	The Zoltan library includes among other things a suite of dynamic load-balancing and parallel partitioning tools that are used by AERO-S for parallel proximity searches.
Eigen3 library	Eigen3 is a versatile C++ template library for linear algebra (vectors, matrices, and related algorithms (see http://eigen.tuxfamily.org)).

To install **AERO-S**, follow the procedure specified below:

- From the directory containing the source code of **AERO-S**, type `cmake -DAERO=1 .` Note the space and the "." after the command `cmake`. The "." specifies the current directory.
- Watch the computer screen and verify that all invoked components were found and all build options were correct. A sample computer screen output of the `cmake` command is:

```
-- The C compiler identification is GNU
-- The CXX compiler identification is GNU
-- Check for working C compiler: /usr/bin/gcc
-- Check for working C compiler: /usr/bin/gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- The Fortran compiler identification is GNU
-- Check for working Fortran compiler: /usr/bin/gfortran
-- Check for working Fortran compiler: /usr/bin/gfortran -- works
-- Detecting Fortran compiler ABI info
-- Detecting Fortran compiler ABI info - done
-- Checking whether /usr/bin/gfortran supports Fortran 90
-- Checking whether /usr/bin/gfortran supports Fortran 90 -- yes
-- Found MPI: /usr/lib/openmpi/lib/libmpi_cxx.so
-- Try OpenMP C flag = [-fopenmp]
-- Performing Test OpenMP_FLAG_DETECTED
-- Performing Test OpenMP_FLAG_DETECTED - Success
-- Try OpenMP CXX flag = [-fopenmp]
-- Performing Test OpenMP_FLAG_DETECTED
-- Performing Test OpenMP_FLAG_DETECTED - Success
-- Found OpenMP: -fopenmp
-- Looking for Fortran sgemm
-- Looking for Fortran sgemm - found
-- A library with BLAS API found.
-- Looking for Fortran cheev
-- Looking for Fortran cheev - found
-- A library with LAPACK API found.
-- A library with ARPACK API found.
-- A library with SPOOLES API found.
-- A library with EIGEN2 API found.
-- A library with METIS API found.
-- A library with MUMPS API found.
-- A library with BLACS API found.
-- A library with SCALAPACK API found.
-- A library with METIS API found.

=====
Summary of build options
-----
Distributed FETI:      YES
Aeroelastic:          YES
Mumps:                YES
Arpack:               YES
Spuoles:              YES
Acme:                 YES
Metis:                YES
OpenMP:               YES
Build type:            Release
Extra libraries:
=====

-- Configuring done
-- Generating done
-- Build files have been written to: /home/pavery/Codes/FEM
```

- If necessary, edit the `CMakeCache.txt` file to include the file paths to all required and desired optional components that were not automatically found by `cmake`. Typically, the compilers, the utilities `Flex` and `Bison`, the libraries `MPI`, `BLAS` and `LAPACK`, and the API `OpenMP` will be automatically found. However, it may be necessary to specify the paths for the libraries `SPOOLES`, `MUMPS`, `ScalAPACK`, `BLACS`, `ARPACK`, `PARPACK`, `METIS`, `Zoltan` and `Eigen3`.
- Then, also from the directory containing the source code of **AERO-S**, type `make`.

The successful completion of the procedure described above leads to the creation in the bin/ directory of **AERO-S**'s executable **aeros**.

Next: [ATDDIR](#), Previous: [INSTALLATION](#)

3 ACOUSTIC TIME-DOMAIN ARTIFICIAL BOUNDARY *S*

Command Statement:	ATDARB
--------------------	---------------

The **ATDARB** command statement is used to specify the artificial boundary Σ on which an Antoine absorbing condition is to be applied in the time domain, and the order of this absorbing boundary condition. The input format is given below.

ATDARB	ORDER
---------------	--------------

FACE	FACE_TYPE	CONNECTIVITY_NODES
-------------	------------------	---------------------------

ORDER	Order of Antoine's absorbing boundary condition in the time domain (real). Currently supported values are 0 and 1.	
FACE	Face (or edge in two dimensions) identification number whose type and connectivity are to be specified (integer). In practice, this identification number is ignored by AERO-S .	
FACE_TYPE		
1	2-node line segment. To be used with two-dimensional linear elements.	
2	3-node line segment. To be used with two-dimensional quadratic elements.	
3	3-node triangular face. To be used with three-dimensional linear tetrahedral element.	
4	4-node quad face. To be used with three-dimensional linear hexahedral element.	
6	6-node triangular face. To be used with three-dimensional quadratic tetrahedral element.	
10	n^2 -node arbitrarily higher-order quad face where n is the number of nodes on an edge of this face. To be used with element type 95.	
11	arbitrarily higher-order triangular face.	
12	arbitrarily higher-order line segment.	
13	edge of a full isoparametric triangular element where the nodes are numbered linearly along it.	
14	n^2 -node arbitrarily higher-order (spectral) quad face where n is the number of nodes on an edge of this face. To be used with spectral element type 105.	
CONNECTIVITY_NODES	These should be listed in a stacked fashion on a single line, and numbered clockwise (when looking from infinity in three dimensions).	

Next: [ATDDNB](#), Previous: [ATDARB](#)

4 ACOUSTIC TIME-DOMAIN DIRICHLET BOUNDARY CONDITIONS *S*

Command Statement:	ATDDIR
--------------------	---------------

The **ATDDIR** command statement is used to specify *nodal* Dirichlet boundary conditions for a time-domain acoustic scattering problem. The input format is given below.

ATDDIR

NODE#	VALUE
--------------	--------------

NODE# Node number where the Dirichlet boundary condition is specified (integer).

VALUE	Value of the specified boundary condition (real).
-------	---

Next: [ATDNEU](#), Previous: [ATDDIR](#)

5 ACOUSTIC TIME-DOMAIN DISTRIBUTED NEUMANN BOUNDARY CONDITION *S*

Command Statement:	ATDDNB
--------------------	---------------

The ATDDNB command statement is used to specify the surface of a scatterer on which a *distributed* Neumann boundary condition of the form $\frac{\partial p^s}{\partial n} = c$ is applied in a time-domain acoustic computation, and the value of the constant c .

The input format of this command is given below.

ATDDNB	CONSTANT
--------	----------

FACE	FACE_TYPE	CONNECTIVITY_NODES
------	-----------	--------------------

CONSTANT	Value of the constant c (real).	
FACE	Face (or edge in two dimensions) identification number whose type and connectivity are to be specified (integer). In practice, this identification number is ignored by AERO-S .	
FACE_TYPE		
1	2-node line segment. To be used with two-dimensional linear elements.	
2	3-node line segment. To be used with two-dimensional quadratic elements.	
3	3-node triangular face. To be used with three-dimensional linear tetrahedral element.	
4	4-node quad face. To be used with three-dimensional linear hexahedral element.	
6	6-node triangular face. To be used with three-dimensional quadratic tetrahedral element.	
10	n^2 -node arbitrarily higher-order quad face where n is the number of nodes on an edge of this face. To be used with element type 95.^M	
11	arbitrarily higher-order triangular face.	
12	arbitrarily higher-order line segment.	
13	edge of a full isoparametric triangular element where the nodes are numbered linearly along it.	
14	n^2 -node arbitrarily higher-order (spectral) quad face where n is the number of nodes on an edge of this face. To be used with spectral element type 105.	
CONNECTIVITY_NODES	These should be listed in a stacked fashion on a single line, and numbered clockwise (when looking from infinity in three dimensions).	

Next: [ATDROB](#), Previous: [ATDDNB](#)

6 ACOUSTIC TIME-DOMAIN NEUMANN BOUNDARY CONDITIONS *S*

Command Statement:	ATDNEU
--------------------	---------------

The ATDNEU command statement is used to specify the *nodal* Neumann boundary conditions for a time-domain acoustic scattering problem. The input format is given below.

ATDNEU

NODE#	VALUE
-------	-------

NODE#	Node number where the Neumann boundary condition is specified (integer).
VALUE	Value of the specified boundary condition (real).

Next: [ACTUATORS](#), Previous: [ATDNEU](#)

7 ACOUSTIC TIME-DOMAIN ROBIN DISTRIBUTED BOUNDARY CONDITION *S*

Command Statement: **ATDROB**

The **ATDROB** command statement can be used to specify the surface of a scatterer on which a *distributed* Robin boundary condition of the form $\alpha \frac{\partial p^s}{\partial n} + \beta p^s = \gamma$ is applied in a time-domain acoustic computation, and the three constants α , β , and γ .

The input format of this command is given below.

ATDROB	α	β	γ
--------	----------	---------	----------

FACE	FACE_TYPE	CONNECTIVITY_NODES
------	-----------	--------------------

α
 β
 γ

Non-zero constant premultiplying the normal derivative of the Robin boundary condition (real).
Constant premultiplying the unknown of the Robin boundary condition (float).

Constant right hand-side of the Robin boundary condition (float).

FACE

Face (or edge in two dimensions) identification number whose type and connectivity are to be specified (integer). In practice, this identification number is ignored by **AERO-S**.

FACE_TYPE

1

2-node line segment. To be used with two-dimensional linear elements.

2

3-node line segment. To be used with two-dimensional quadratic elements.

3

3-node triangular face. To be used with three-dimensional linear tetrahedral element.

4

4-node quad face. To be used with three-dimensional linear hexahedral element.

6

6-node triangular face. To be used with three-dimensional quadratic tetrahedral element.

10

n^2 -node arbitrarily higher-order quad face where n is the number of nodes on an edge of this face. To be used with element type 95.^M

11

arbitrarily higher-order triangular face.

12

arbitrarily higher-order line segment.

13

edge of a full isoparametric triangular element where the nodes are numbered linearly along it.

14

n^2 -node arbitrarily higher-order (spectral) quad face where n is the number of nodes on an edge of this face. To be used with spectral element type 105.

CONNECTIVITY_NODES

These should be listed in a stacked fashion on a single line, and numbered clockwise (when looking from infinity in three dimensions).

Next: [AERO](#), Previous: [ATDROB](#)

8 ACTUATORS *S*

Command Statement: **ACTUATORS**

The **ACTUATORS** command is used to prescribe at specified nodes and local degrees of freedom a time-variant source term (for example, a force field for a structural model) using a user-defined subroutine, and request the solution state at that node (for example, the displacement, velocity, and acceleration fields for a structural model). In this case, the user has to write his/her own algorithm for specifying the prescribed source term within a subroutine named "control.C". The **SENSORS** command must be used to pass to "control.C" the solution state at the nodes of interest (see **SENSORS**). The user should grab the special makefile for this command which is located within the AERO-S.d/Control.d directory, and use the **LOAD** command to activate this command.

For structural models, all prescribed forces and moments are interpreted by default as being of the *axial* type — that is, as being defined in the *fixed* nodal degree of freedom reference frames (see **NODES** and **NFRAMES**). However, if a node has rotational degrees of freedom, the user can specify that the forces and/or moments prescribed at this node are of the *follower* type — that is, they act in a direction that remains constant in the local frame attached to the node where they are applied. This local frame coincides with the nodal degree of freedom reference frame (see **NODES** and **NFRAMES**) in the undeformed configuration. In the deformed configuration, the orientation of this local frame is defined by the rotation of the node to which it is attached. In other words, the specified nodal force or moment "follows"

in this case the rotation of the node to which it is applied.

An example input file using this command can be found in [APPENDIX 1](#).

Note 1: By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: For structural models, specifying a follower force or moment leads to an unsymmetric tangent "load" stiffness matrix during a [NONLINEAR](#) analysis.

The syntax for invoking this option is given below.

ACTUATORS

NODE#	DOF#	TYPE
-------	------	------

NODE# Node number where an actuating force/moment is specified (integer).
DOF# Degree of freedom local number where an actuating force/moment is specified (integer).
TYPE For structural models, all specified nodal source terms are by default of the axial type. However, if this parameter is set to FOLLOWER and the node **NODE#** has rotational degrees of freedom, the source term specified at this node and degree of freedom **DOF#** is considered to be of the follower type (characters).

Next: [AEROH](#), Previous: [ACTUATORS](#)

9 AEROELASTICITY

Command Statement:	AERO
--------------------	------

The **AERO** command statement can be used to perform any of the following instructions:

- Request that **AERO-S** sends to **AERO-F** one or several displacement fields of the structure.
- Indicate that **AERO-S** is to interact with **AERO-F** to compute a flow-induced load and perform the corresponding static structural analysis, or perform a coupled static or dynamic aeroelastic simulation.
- Choose a staggered time-integration algorithm (**ALGORITHM** subcommand) and, if needed, a displacement predictor (**ALGORITHM** subcommand) defined by two coefficients α_0 and α_1 and the formula

$$u^{n+k^p} = u^n + \alpha_0 \Delta t_S \dot{u}^n + \alpha_1 \Delta t_S (\dot{u}^n - \dot{u}^{n-1})$$
 when the structural time-integrator is implicit, with $k = 1/2$ if **A6** is the chosen time-integration algorithm and $k = 1$ otherwise, or

$$u^{n+\frac{1}{2}^p} = u^n + \alpha_0 \Delta t_S \dot{u}^{n-\frac{1}{2}} + \alpha_1 \Delta t_S (\dot{u}^{n-\frac{1}{2}} - \dot{u}^{n-\frac{3}{2}})$$
 when the structural time-integrator is explicit, to perform a static or dynamic aeroelastic simulation.
 Here, u^{n+1^p} ($u^{n+\frac{1}{2}^p}$) is the predicted displacement field at time-step t^{n+1} ($t^{n+\frac{1}{2}}$) and Δt_S is the structural time-step size (or simply, time-step) resulting from some settings of the [DYNAMICS](#) command. It is relevant however only for dynamic simulations.
- Select an algorithm for computing the corrected pressure field P_c^{n+1} to be used in computing the aerodynamic forces acting on the structure at time-step t^{n+1} (**PRESSURE** subcommand) of a dynamic aeroelastic simulation.
- Specify the structure matcher file, when either the Arbitrary Lagrangian/Eulerian (ALE) framework is chosen for performing a coupled dynamic or static fluid-structure simulation *without element deletion*, or the Embedded Boundary Method (EBM) framework is used for this purpose and the embedded surface is inputted to the flow solver **AERO-F**. Note that the ALE framework does not support element deletion, and the EBM framework supports it only if the embedded discrete surface is identical to the wet surface of the finite element structural model and inputted under this command.
- Specify the discrete surface to be embedded in the fluid grid, when the EBM framework is chosen for performing a coupled dynamic or static fluid-structure simulation *with or without element deletion*, if this surface is identical to the wet surface of the finite element structural model and defined using the same nodes of this model. Otherwise, provided that the simulation is to be performed without

element deletion, the discrete surface to be embedded in the fluid grid should be inputted to **AERO-F** and the structure matcher file should be specified under this command.

The purpose of the displacement predictor and that of the force corrector are to compensate for the effects of time-lagging. More specifically, they improve the time-accuracy and numerical stability of the chosen staggered solution algorithm by minimizing the lack of energy conservation at the fluid-structure interface. The theoretical, algorithmic, and practical aspects of both of these "compensators" are described in *S. Piperno and C. Farhat, "Partitioned Procedures for the Transient Solution of Coupled Aeroelastic Problems - Part II: Energy Transfer Analysis and Three-Dimensional Applications," Computer Methods in Applied Mechanics and Engineering, Vol. 190, pp. 3147-3170 (2001)*

Before setting the values of α_0 and α_1 that complete the definition of the displacement predictor and choosing the force corrector, the user should note that:

- Currently, the fluid code **AERO-F** can send to **AERO-S** at t^{n+1} either the pressure field at t^{n+1} , P^{n+1} , or the averaged pressure field $\bar{P}^{n+1} = \frac{1}{\Delta t_s} \int_{t^n}^{t^{n+1}} P(t) dt$. Hence, P_c^{n+1} can be set at least to either of these two values. Therefore, the purpose of the **PRESSURE** subcommand is to propose additional choices for the corrected pressure field P_c^{n+1} . These can affect the order of time-accuracy of the chosen staggered solution algorithm (see C. Farhat, G. van der Zee and P. Geuzaine, "Provably Second-Order Time-Accurate Loosely-Coupled Solution Algorithms for Transient Nonlinear Computational Aeroelasticity," Computer Methods in Applied Mechanics and Engineering, Vol. 195, pp. 1973-2001 (2006)).
- The values of α_0 and α_1 that minimize the variation of energy exchanged at the fluid-structure interface and therefore improve time-accuracy and numerical stability of the chosen staggered solution algorithm depend on the chosen fluid and structure time-integrators, the chosen force corrector, and the selected staggered solution algorithm itself.
- When the Generalized α method is used for time-integrating linear or nonlinear structural dynamics problems (see [DYNAMICS](#)), **AERO-S** solves the following equilibrium problem

$$M\ddot{u}^{n+1-\alpha_m} + C\dot{u}^{n+1-\alpha_f} + F_{internal}(u^{n+1-\alpha_f}) = F_{external}^{n+1-\alpha_f}$$

- When the central difference method is used for time-integrating structural dynamics problems (see [DYNAMICS](#)), **AERO-S** solves the following equilibrium problem

$$M\ddot{u}^{n+1} + C\dot{u}^{n+1} + F_{internal}(u^{n+1}) = F_{external}^{n+1}$$

Note 1: **AERO-F** offers two different computational frameworks for fluid-structure interaction: an ALE framework, and an EBM framework. When a functional capability described below is meaningful, applicable, or supported by only one of these two computational frameworks, its designating keyword is followed by **[ALE]** in the case of the ALE computational framework, and by **[EMB]** in the case of EBM computational framework. When this capability is applicable, meaningful, and supported by both computational frameworks, its designating keyword is not followed by any symbol.

Note 2: The algorithms **PP** and **MPP** require the additional presence of [DYNAMICS](#) in the input file.

Note 3: The algorithm **B0** requires the additional presence of [QSTATICS](#) in the input file. In this case, **Problem.Type** in **AERO-F** should be set to **SteadyAeroelastic**.

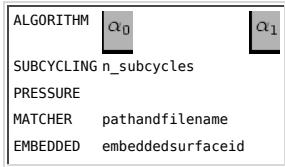
Note 4: For a static aeroelastic analysis, **A0**, **A4**, **A5**, **A6**, **B0** and **C0** are irrelevant as [QSTATICS](#), which is also needed in this case, is equipped with its own staggered solution algorithm. However, for parsing reasons, one of these staggered solution algorithms must be explicitly specified even if it will be ignored.

Note 5: The values of α_0 and α_1 derived in the paper by Piperno and Farhat mentioned above correspond to the case where the structure is linear and time-integrated by the Newmark algorithm with $\beta = 1/4$ and $\gamma = 1/2$. For other structural time-integrator choices, that analysis needs to be redone for choosing the parameters of the command **AERO**.

Note 6: If an aeroelastic analysis is requested with all of the **GEPs**, **IDISP6**, and **IDISPLACEMENTS** commands present in the input file, **AERO-S** interprets the **IDISPLACEMENTS** command and its content as the initialization of the incremental displacement field from the configuration **C*** (see [GEPs](#)) to the configuration **C** (see [GEPs](#)). In this case, it sends to the fluid code at each time-step the sum of the updated incremental displacement and the displacement inputted under the **IDISP6** command. Hence, this scenario is particularly suitable for the case where the fluid code is started from a deformed CFD mesh. If on the other hand an aeroelastic analysis is requested with only the **GEPs** and **IDISP6** commands present in the input file, **AERO-S** understands that the incremental displacement field is initialized to zero. However, it communicates in this case with the fluid code in a very special manner: at the first time-step, it sends to the fluid code the initial value of the incremental displacement field (which in this case is zero), and at each subsequent time-step, the sum of the updated incremental displacement and the displacement specified under the **IDISP6** command. Hence, the latter scenario is particularly suitable for the case where the fluid code is started from an undeformed CFD mesh.

The syntax for invoking this command is given below.

AERO



ALGORITHM

PP [ALE] This “Ping-Pong” algorithm sends the initial displacement of the structure, specified under either the [IDISP6](#) command or the [IDISPLACEMENTS](#) (possibly equipped with the sub-command `MODAL`), to the fluid code which receives it and deforms the CFD mesh accordingly (characters). For this purpose, the `DYNAMICS` command must also be present in the **AERO-S** input file. After the send occurs, the structure code exits gracefully, the fluid code computes the fluid mesh deformation associated with the specified structural displacement and outputs (if requested) the corresponding position and/or displacement of the fluid mesh (characters).

MPP This “Multi-Ping-Pong” algorithm sends several initial displacements of the structure at a time — for example, modal displacements — to the fluid code, which uses them to compute compatible fluid mesh deformations. The [READMODE](#) command must be used to input the initial displacements of interest — for example, a set of eigenmodes. If the second format of that command is used to specify the Reduced-Order Basis (ROB) to be used to generate the initial displacements, the coefficient α_1 of this command must be used to identify that ROB (see the field `rob_id` in [READMODE](#)).

If a coefficient α_0 is specified after `MPP`, the initial displacements are amplified by α_0 before they are sent to the fluid code. However, the fluid code scales back by $1/\alpha_0$ the corresponding fluid mesh displacements it computes before saving them in an output file, in order to preserve the effect of the mass normalization of the modal structural displacements. The deformed fluid mesh configurations can then be used to generate sources of excitations for the construction of fluid POD bases and ROMs, or for linearized (perturbation) flow simulations whose initial conditions involve a (modal) position or velocity of the fluid surface mesh.

As for the “Ping-Pong” case, the `DYNAMICS` command must also be present in the **AERO-S** input file, **AERO-S** exits gracefully after the send occurs, the fluid code computes the fluid mesh deformation associated with the specified structural displacements and outputs (if requested) the corresponding positions and/or displacements of the fluid mesh. In addition, for each inputted deformed fluid mesh position, the fluid code outputs an identification tag — for example, the frequency of the corresponding input structural mode in the case of modal displacements (characters).

A0 This is the basic sequential staggered solution algorithm (characters). When fluid subcycling is effected, the same algorithm has been referred to in the AIAA Paper 96-1388 by Farhat and co-workers as the `A1` algorithm (characters). It should not be used without a displacement predictor as it would reduce the overall order of time-accuracy compared to that intrinsic to the structural time-integrator. For the same reason, it is also not recommended — and as a matter of fact not available — when the structural time-integrator is explicit.

A4 This is the basic staggered solution algorithm with fluid-structure inter-parallelism (characters). It has been referred to as `A2` in the AIAA Paper 96-1388 by Farhat and co-workers. When this algorithm is specified, **AERO-F** sends to **AERO-S** $[P^n]$ or $[P^{n+1}]$ rather than the corresponding pressure fields at $[P^{n+1}]$. Currently, this algorithm is not available when an explicit structural time-integrator is chosen.

A5 This staggered solution algorithm also features inter-parallelism but offers a better accuracy than `A4` when both methods are used without a displacement predictor (characters). Currently, this algorithm is not available when an explicit structural time-integrator is chosen.

A6 This staggered solution algorithm has superior stability and accuracy properties. It is recommended when the fluid is time-advanced by an implicit time-integrator. It has been referred to in most papers by Farhat and co-workers as the ISS (Improved Serial Staggered algorithm) method (characters). It delivers good accuracy without any displacement predictor. However, it is provably second-order time-accurate when equipped with the displacement predictor defined by $\alpha_0 = 1/2$ and $\alpha_1 = 1/8$, the midpoint implementation of the Newmark time-integrator with $\beta = \frac{1}{4}$ and $\gamma = \frac{1}{2}$ (see [DYNAMICS](#)), and a second-order time-integrator for the discrete fluid problem. Also, this partitioned

solution procedure prefers a non time-averaged pressure field. Most importantly, it was designed to be used exclusively with the `NONCOLLOCATED` scheme of aerodynamic force evaluation (see below) (characters). Currently, it does not support an explicit structural time-integrator.

B0 This one-way coupled steady-state aeroelastic algorithm sends the initial displacement of the structure (if any) to the flow solver. After receiving it and updating its mesh, **AERO-F** performs any requested flow simulation and sends to **AERO-S** upon completion the flow-induced load. Then, **AERO-S** receives this load, performs a static analysis if requested, and finally outputs any requested data (such as the computed flow-induced load, and/or resulting displacement, stress, and strain fields).

C0 This staggered solution algorithm is designed for the case where the structural subsystem is time-integrated by the explicit central difference method (see [DYNAMICS](#)), and the fluid subsystem is time-integrated by either an explicit or an implicit scheme. As shown in C. Farhat, A. Rallu, K. Wang and T. Belytschko, “Robust and Provably Second-Order Explicit-Explicit and Implicit-Explicit Staggered Time-Integrators for Highly Nonlinear Fluid-Structure Interaction Problems,” International Journal for Numerical Methods in Engineering, (2010), it is genuinely second-order time-accurate when equipped with the second-order displacement predictor obtained by setting $\alpha_0 = 0.5$ and

$\alpha_1 = 0.375$ (see below), and a second (or higher)-order accurate explicit or implicit ALE fluid time-integrator.

Currently, this algorithm supports only explicit structural time-integrators, and is the only algorithm which supports element deletion in fluid-structure simulations.

When used with the `MPP` command, this coefficient can be specified to amplify the initial displacements to be sent to the fluid code (however, as noted above, the fluid mesh displacements computed by the fluid code are scaled back by $1/\alpha_0$ before they are saved in an output file). In this case, the default value is 1. Otherwise, this coefficient is part of

the construction of a predictor for the position of the structure. For $\alpha_0 = 0$ (and $\alpha_1 = 0$), no prediction is effected.

Setting $\alpha_0 = 1$ (and $\alpha_1 = 0$) generates a first-order prediction, and setting $\alpha_0 = 1$ (and $\alpha_1 = \frac{1}{2}$) generates a

second-order prediction. The optimal values of α_0 and α_1 depend on the specifics of the fluid and structure field time-integrators (see the aforementioned paper on interface energy conservation by Piperno and Farhat) (real). In this case, the default value is 0.5 for algorithm `c0`, and 0 for all other algorithms.

When used with the `MPP` command, this coefficient is interpreted as an integer that identifies among the reduced-order bases read using the `READMODE` command (see there the field `rob_idi`) that to be used with the algorithm `MPP` (integer). Otherwise, this coefficient is interpreted as a real coefficient that is part of the construction of a predictor for the position of the structure (real) and therefore should be used only as such (see above). In this case, its default value is 0.375 for algorithm `c0`, and 0 for all other algorithms.

SUBCYCLING This pair of sub-command keyword (characters) and corresponding numerical value (integer) specifies the structural subcycling factor, which defines the coupling fluid-structure time-step size (or simply, time-step) $\Delta t_{F/S}$ – that is, the time-interval during which structural and fluid computations are performed without any coupling, or equivalently, the time-interval between two couplings of the fluid and structural sub-systems – as follows

$$\Delta t_{F/S} = n_{subcycles} \Delta t_S$$

where Δt_S is the structural time-step resulting from some settings of the `DYNAMICS` command. Hence, `n_subcycles` can also be described as the number of time-steps that the structural analyzer performs per coupling fluid-structure time-step. The default value is 1.

PRESSURE This option, which can take either value `COLLOCATED` or `NONCOLLOCATED`, specifies how to compute the corrected pressure field. It is relevant only when `ALGORITHM` is set to `A6`, `A4`, `A5`, or `A6`.

`COLLOCATED` In this case, P_c^{n+1} is set to the received pressure field P^{n+1} (or \bar{P}^{n+1}), then converted to

$$P_c^{n+1-\alpha_f} = (1 - \alpha_f)P_c^{n+1} + \alpha_f P_c^n$$

before the aerodynamic forces are constructed and fed into the structural equations of dynamic equilibrium (see the introduction to this command). Therefore, this option covers the cases $P_c^{n+1} = P^{n+1}$ and $P_c^{n+1} = \bar{P}^{n+1}$, which explains the origin of the word “collocated” (characters).

NONCOLLOCATED In this case, which is also the default value of `PRESSURE`, the pressure field P_c^{n+1} is set to $P_c^{n+1} = \frac{1}{\gamma}P^{n+1} - \frac{(1-\gamma)}{\gamma}P_c^n$ (or $P_c^{n+1} = \frac{1}{\gamma}\bar{P}^{n+1} - \frac{(1-\gamma)}{\gamma}P_c^n$), then converted to $P_c^{n+1-\alpha_f} = (1 - \alpha_f)P_c^{n+1} + \alpha_f P_c^n$ (here, γ is the γ parameter of the Generalized `L` method). This specific choice for P_c^{n+1} is consistent with the quadrature rule of the Generalized `L` method for evaluating the work done by an external force in the time-interval $[t^n, t^{n+1}]$. In other words, the pressure forces fed into the structural equations of equilibrium are based on $P_c^{n+1-\alpha_f} = \frac{(1-\alpha_f)}{\gamma}P^{n+1} + (1 - \frac{(1-\alpha_f)}{\gamma})P_c^n$. For $\gamma = \alpha_f = \frac{1}{2}$, $P_c^{n+1-\alpha_f} = P^{n+1}$, which explains the origin of the word “non collocated” (characters).

MATCHER Name (including path, if needed) of the structural matcher file (characters). This file should be specified because the fluid-structure computation is to be performed without element deletion, using either the ALE or EBM framework. In the first case, the structural mesh should be matched with the fluid mesh. In the second case, it should be matched with the embedded discrete surface and this surface should be inputted to `AERO-F`.

EMBEDDED
`[EMB]` Integer identification number of a surface defined in `SURFACETOPO` using 3-noded triangles, 4-noded quadrilaterals, or a combination of such elements (integer). This parameter should be specified when the discrete surface to be embedded in the fluid mesh for the purpose of a fluid-structure computation (with or without element deletion) is identical to the wet surface of the finite element structural model and defined using the same nodes of this model. In this case, it is not necessary to specify above a structure matcher file.

Next: [ATTRIBUTES](#), Previous: [AERO](#)

10 AEROHEAT

Command Statement:	AEROH
--------------------	--------------

The **AEROH** command statement is used to indicate that **AERO-S** is to interact with **AERO-F** to perform an aerothermal (thermostructure-thermofluid) coupled simulation, to select a staggered time-integration algorithm (**ALGORITHM** subcommand), and specify the α_0 and α_1 parameters of the following prediction of the temperature of the structure at time-step t^{n+1} (**ALGORITHM** subcommand)

$$T_S^{n+1} = T_S^n + \alpha_0 \Delta t_S^H \dot{T}_S^n + \alpha_1 \Delta t_S^H (\dot{T}_S^n - \dot{T}_S^{n-1})$$

Note 1: Currently, the **AEROH** command supports only linear thermal analysis on the **AERO-S** side.

Note 2: The staggered solution algorithm α_0 discussed below for aerothermal analysis requires the additional presence in the input file containing this command statement of the **DYNAMICS** command in the *dynamic* case, or the **QSTATICS** command in the *static* case.

Note 3: For a static aerothermal analysis, the staggered solution algorithm α_0 must be specified only for parsing reasons, as **QSTATICS** has its own staggered (iterative) solution scheme.

The syntax for invoking this command is given below.

AEROH

ALGORITHM	α_0	α_1
-----------	------------	------------

ALGORITHM

α_0

This is the basic sequential aerothermal partitioned solution algorithm. It is similar to the α_0 algorithm of the **AERO** command.

α_0

For $\alpha_0 = 0$ (and $\alpha_1 = 0$), no prediction is effected. Setting $\alpha_0 = 1$ (and $\alpha_1 = 0$) generates a first-order prediction, and setting $\alpha_0 = 1$ (and $\alpha_1 = \frac{1}{2}$) generates a second-order prediction. However, the optimal values of α_0 and α_1 depend on the specifics of the fluid and structure field time-integrators (see *S. Piperno and C. Farhat, Partitioned Procedures for the Transient Solution of Coupled Aeroelastic Problems - Part II: Energy Transfer Analysis and Three-Dimensional Applications*, Computer Methods in Applied Mechanics and Engineering, Vol. 190, pp. 3147-3170 (2001)) (real).

α_1

See above (real).

Next: [BOFFSET](#), Previous: [AEROH](#)

11 ATTRIBUTES *\$*

Command Statement: **ATTRIBUTES**

The **ATTRIBUTES** command statement is used to label an element with an attribute identification number for linear elastic material and/or geometric properties, a composite or orthotropic shell, or anisotropic solid element attribute identification number, a frame or fiber-angle attribute identification number if this element has been defined as a composite or orthotropic shell or anisotropic solid element, and a hyper reduction coefficient when the mesh containing this element is sampled for the purpose of hyper reduction.

The attribute identification number for linear elastic material properties and/or geometric properties is required for all elements of the computational model. It is used in the **MATERIAL** command when specifying the linear elastic material properties and/or geometric properties of a group of elements (see [MATERIAL](#)).

An “empty” (or “phantom”) element — that is, an element with zero generalized stiffness and mass matrices — can be useful in aeroelastic and aerothermal computations to facilitate the exchange of elastodynamic, thermal, and aerodynamic data between the fluid, thermal, and structural analyzers. Such an element can be specified by assigning it a negative attribute number. In this case, it is not necessary to specify a material for such an element as its properties will be ignored.

It is also not necessary to assign an attribute number to a *massless* rigid element or simple joint element (types 118-125 and 127) for which the default constraint method (see [CONSTRAINTS](#)) is to be used for enforcing the associated constraints. On the other hand, an attribute number should always be assigned to a rigid element which has a mass (see [MATERIAL](#)), and a revolute-joint-with-joint-driver element (type 126) or a joint spring combination element (types 220-227), regardless of whether the default constraint method is to be used or not for this element.

The hyper reduction coefficients are computed by the **RMSHC** command and outputted in the result file **SAMPLMSH** (see [OUTPUT](#)) which includes the header **ATTRIBUTES**. Hence, this file can be simply included in the **AERO-S** input file using the **INCLUDE** command (see [INTRODUCTION](#)) for the purpose of hyper reduction. An alternative approach for this purpose that leads to faster CPU performance is to

use the reduced mesh generated by the command [RMSHC](#) and outputted in the file `SAMPLMSH.elementmesh.inc` (see [OUTPUT](#)).

AERO-S supports three input formats for this command that are described below. In the case of the first format, there should be as many lines as the number of finite elements. All formats can be mixed.

Note 1: This command is required in any **AERO-S** ASCII Input Command Data file.

Note 2: Only the following element types can be used as “empty” (or “phantom”) elements and therefore can be assigned negative attribute numbers: types 6 and 7 (beam elements); types 8, 88, 15, 1515, 16, 20, and 2020 (shell elements); types 19, 87, 128, and 129 (membrane elements); and types 46 and 49 (thermal elements).

Note 3: When phantom elements are used in the mesh, the FETI solvers are not guaranteed to work because of issues related to subdomain singularities. Hence, it is not recommended to use FETI solvers in such cases.

Note 4: Phantom elements should be used with care to avoid the generation of degrees of freedom without stiffness (and mass) and therefore avoid introducing artificial singularities in the solution of the problem of interest. In particular, it is strongly recommended to ensure that every node of every phantom element is also a node of a non-phantom element, and that every local degree of freedom of a phantom element is also a local degree of freedom of a non-phantom element. In other words, it is strongly recommended, for example, to avoid connecting a phantom shell element to a face of a solid (brick element) and use instead a phantom three-dimensional plane stress/plane strain element in this case.

Note 5: Only phantom shell elements and phantom plane stress/plane strain elements can transfer aeroelastic loads to a structure.

Note 6: If an element is attributed two different material laws using the [MATUSAGE/MATLAW](#) and [ATTRIBUTES/MATERIAL](#) commands, the material law defined in [MATLAW](#) and assigned in [MATUSAGE](#) takes precedence.

ATTRIBUTES

ELEMENT#	MAT_ATT#	CMP_ATT#	CMP_FRM#	HRC	HRCOEFF	[EXTFOL]
----------	----------	----------	----------	-----	---------	----------

or

ELEMENT#	MAT_ATT#	CMP_ATT#	THETA	θ_{Ref}	HRC	HRCOEFF	[EXTFOL]
----------	----------	----------	-------	----------------	-----	---------	----------

or

STARTING_ELEMENT#	ENDING_ELEMENT#	MAT_ATT#	CMP_ATT#	CMP_FRM#
-------------------	-----------------	----------	----------	----------

or

STARTING_ELEMENT#	ENDING_ELEMENT#	MAT_ATT#	CMP_ATT#	THETA	θ_{Ref}
-------------------	-----------------	----------	----------	-------	----------------

or

STARTING_ELEMENT#	ENDING_ELEMENT#	IDENTITY
-------------------	-----------------	----------

or

ELEMENT#	HRC	HRCCOEFF	[EXTFOL]
----------	-----	----------	----------

ELEMENT# Element number whose attribute numbers are to be specified (integer).
MAT_ATT# Group identification number (integer); **AERO-S** supports gaps in numbering. Identifies material properties.

CMP_ATT#	Composite (orthotropic or anisotropic) identification number (integer); AERO-S supports gaps in numbering. Identifies composite (orthotropic or anisotropic) properties. May be left blank if element is not a composite (orthotropic or anisotropic) element and must be set to -1 if the orthotropic and/or anisotropic material properties of element are specified using the command MATLAW instead of the command COMPOSITE .
CMP_FRM#	Composite frame identification number (integer); AERO-S supports gaps in numbering. May be left blank if element is not a composite (orthotropic or anisotropic) element.
THETA	Keyword that must be spelled as THETA (characters). This keyword announces that the next item on the same input line is the value θ_{Ref} of THETA . This unusual AERO-S input is necessary to avoid parsing conflicts.
θ_{Ref}	Reference angle in degrees between the axis defined by local nodes 1 and 2 of the element, and the material local x-axis defining the direction of the σ_{xx} and ϵ_{xx} of the constitutive law (real). When also using the LAYN and LAYC commands, the material x-axis corresponds to the direction of $E_1^{(k)}$ when $\theta_F^{(k)}$ is zero. If this angle rather than CMP_FRM# is specified, the composite frame is internally generated by AERO-S .
STARTING_ELEMENT#	First element of a sequence of elements that have the same MAT_ATT# , CMP_ATT# , and CMP_FRM# (integer).
ENDING_ELEMENT#	Last element of a sequence of elements that have the same MAT_ATT# , CMP_ATT# , and CMP_FRM# (integer).
MAT_ATT#	Group identification number. Identifies material properties (integer).
CMP_ATT#	Composite identification number (integer). Identifies composite (orthotropic or anisotropic) properties. May be left blank if element is not a composite (orthotropic or anisotropic) element.
CMP_FRM#	Composite frame identification number (integer). May be left blank if element is not a composite (orthotropic or anisotropic) element.
IDENTITY	This keyword signals to the AERO-S code that each element in the range delimited by STARTING_ELEMENT# and ENDING_ELEMENT# has a material attribute identification number equal to its element identification number. For example, the sequence 1 3 IDENTITY means that element#1 has the material attribute identification 1, element#2 has the material attribute identification 2, and element#3 has the material attribute identification 3.
HRC	Sub-command keyword to request the application of a specified hyper reduction coefficient to the element-level reduced internal forces and moments and acquiring this coefficient (characters).
HRCOEFF	Hyper reduction coefficient computed by the RMSHC command and stored in the result file SAMPLMSH under OUTPUT (float).
[EXTFOL]	Optional sub-command keyword to request the application of the specified hyper reduction coefficient to all of the element-level reduced internal forces and moments and the element-level reduced follower external forces and moments (see FORCES) (characters).

Next: [EFRAMES](#), Previous: [ATTRIBUTES](#)

12 BEAM OFFSET

Command Statement:	BOFFSET
--------------------	----------------

The **BOFFSET** command statement is used to specify a beam's neutral axis offset from the line passing through its two end nodes. This command may be used for both Euler-Bernoulli and Timoshenko beam elements. For beam elements not listed under **BOFFSET**, a zero offset is used. If the **BOFFSET** command is absent altogether, then an offset of zero is used for all beam elements. If a listed element is neither an Euler-Bernoulli nor a Timoshenko beam element, then the offset is ignored for that element. The input format is as follows.

BOFFSET

STARTING_ELEMENT#	ENDING_ELEMENT#	x	y	z
-------------------	-----------------	---	---	---

STARTING_ELEMENT#	First element of a sequence of elements that have the same offset specified by {x, y, z} (integer).
ENDING_ELEMENT#	Last element of a sequence of elements that have the same offset specified by {x, y, z} (integer). If ENDING_ELEMENT# is the same as STARTING_ELEMENT#, then the specified offset is applied to that element only (integer).
x	x component of the offset vector expressed in the global frame (float).
y	y component of the offset vector expressed in the global frame (float).
z	z component of the offset vector expressed in the global frame (float).

Next: [BINARY](#), Previous: [BOFFSET](#)

13 BEAM REFERENCE FRAMES

Command Statement: **EFRAMES**

The EFRAMES command statement is used to specify a beam's orientation which defines the position of the beam with respect to the global frame. The requirements for each beam frame are that one of the axis, **the local x-axis**, concurs with the longitudinal axis of the beam, and the remaining two axes complete an orthogonal triad. If this requirement is violated, AERO-S regenerates the local x-axis as well as the local z-axis.

For *flexible* beams, AERO-S also supports a run-time generation of frames that is activated either when the target flexible beam element is identified under this command and a third node is specified to generate the frame, or when a third node is found in the definition of a flexible beam element within the TOPOLOGY command. The only requirement for the third node is that it does not be colinear with the other two beam nodes that define the local x-axis (X). Under this command, the third node defines the X,Y plane. Under the TOPOLOGY command, it defines the X,Z plane. Using the third node option under the TOPOLOGY command also relieves the user from specifying the EFRAMES command. An example illustrating the third node option in the TOPOLOGY command can be found in FEM.d/fem_examples/Third_Node.d

Otherwise, the input format for this command is given below with the number of lines equal to the number of beams in the problem.

EFRAMES

ELEMENT#	$S_{1x} \ S_{1y} \ S_{1z}$	$S_{2x} \ S_{2y} \ S_{2z}$	$S_{3x} \ S_{3y} \ S_{3z}$
----------	----------------------------	----------------------------	----------------------------

or

ELEMENT#	THIRDNODE	third_node
----------	-----------	------------

ELEMENT# Element number where the beam frame is specified (integer).

$S_{1x} \ S_{1y} \ S_{1z}$ The first axis, **the local x-axis**, of the beam frame expressed in the global frame. This axis must concur with the longitudinal axis of the beam (floats).

$S_{2x} \ S_{2y} \ S_{2z}$ The second axis of the beam frame expressed in the global frame (floats).

$S_{3x} \ S_{3y} \ S_{3z}$ The third axis of the beam frame expressed in the global frame (floats).

THIRDNODE Keyword that must be spelled as THIRDNODE. This keyword announces that the next item on the same input line is the value third_node# of THIRDNODE. This unusual AERO-S input is necessary to avoid parsing conflicts (characters).

THIRD_NODE# Id number of an existing node to be considered as a third node of the *flexible* beam element identified by ELEMENT#, and which must be in the local x-y plane of the beam. The only requirement for the third node is that it does not be colinear with the other two beam nodes. The normalized vector node1 to node2 defines the local x axis. The normalized vector node1 to node3 defines the local y-axis. The local z-axis is automatically generated as cross product of the other two normalized axes.

Note 1: The above axes have to be normalized. By default, the beam frames concur with the global frame.

Next: [CONVECTION](#), Previous: [EFRAMES](#)

14 BINARY INPUT / OUTPUT

Command Statement: **BINARY**

When the finite element model of interest is very large, it may become necessary (for example, because of memory limitations), to organize the input data of an AERO-S computation in a set of binary distributed files, and more efficient to output its results in another set of binary distributed files. This requires:

- Using a version of aeros that is compiled in the distributed memory execution mode.
- Generating and using a mesh partition for the target simulation.
- Generating the binary distributed input files associated with the above mesh partition, as explained below.
- Inputting these binary distributed input files and if desired requesting the binary distributed format for the output files, using the command BINARY explained herein.

To decompose the input data contained in an initial instance of the ASCII Input Command Data file according to a generated mesh partition and reorganize it in a set of binary distributed input files, the user should use the specialized version of the software SOWER that is embedded in AERO-S. The generated binary distributed input files can then be inputted via this command in the final instance of the ASCII Input Command Data file. This command can also be used to request outputting the results marked for output in [OUTPUT](#) and/or [OUTPUT6](#) in the binary distributed format compatible with the generated mesh partition. In that case, the original SOWER (see SOWER's User's Reference Manual) should be used to convert these output files into the ASCII format suitable for postprocessing by XPost.

If memory is not an issue, **AERO-S** can also operate directly on the global set of input data — that is, on the standard content of the ASCII Input Command Data file.

Note 1: If **aeros** is not compiled in the distributed memory execution mode, it can only generate ASCII output files. Most of these are in a format that is suitable for postprocessing by the **XPost** software, but some are in a format that is suitable for postprocessing by **gnuplot**.

Note 2: Any command present in the initial instance of the ASCII Input Command Data file and whose description in this User's Reference Manual is marked by "***\$***" can be assumed to be contained in one of the aforementioned binary distributed input files. However, if that command contains both an algorithmic parameter as well as data, only the data will be contained in the appropriate distributed binary input file. Therefore, the command itself should be kept in the updated ASCII Input Command Data file together with the algorithmic parameter, but without the data. For example, the command **IDISPLACEMENTS** specifies both an amplification factor and an initial displacement data. The initial displacement data will be included in the aforementioned binary distributed input file, but the amplification factor will not. This is to allow the user to change simple things such as this amplification factor without having to regenerate the binary distributed input files.

Note 3: This data organization option does not support the sparse direct solver MUMPS (see [STATICS](#)).

The syntax for this command is as follows.

BINARY

BINARYINPUT	flagBIN	[<pathandfileprefix>]
BINARYOUTPUT	flagBOUT	
CPUMAP	<pathandfilename>	

BINARYINPUT	flagBIN	This pair of sub-command keyword (characters) and corresponding value (characters) can be used to choose between reading the binary distributed input files designated by <pathandfileprefix>, or the standard ASCII global input data contained in the initial instance of the ASCII Input Command Data file. flagBIN should be inputted on the same line as BINARYINPUT ; it can take one of the following values: On This setting chooses the specified binary distributed input files as input. In this case, the user should check first that all binary distributed input files have been generated and that the associated filenames and path are those referred to in <pathandfileprefix>. Off This setting, which is also the default setting, chooses the standard ASCII global data. In this case, the user should check first that all needed input data is included in the ASCII Input Command Data file. <pathandfileprefix> This entry is relevant only when flagBIN is set to on . By default, AERO-S expects the binary input data command to be contained in four binary distributed files named INPUT.dec , INPUT.con , INPUT.sub , and INPUT.msh . Alternatively, this data can be contained in four equivalent files that are named differently, as long as they share the same path and filename prefix — that is, these four binary distributed files must have the same path and can be named fileprefix.dec (substitute for INPUT.dec), fileprefix.con (substitute for INPUT.con), fileprefix.sub (substitute for INPUT.sub), and prefixfile.msh (substitute for INPUT.msh). In the latter case, <pathandfileprefix> specifies the common filename prefix (and file path if needed) (characters).
BINARYOUTPUT	flagBOUT	This pair of sub-command keyword (characters) and corresponding value (characters) can be used to choose between outputting the results specified in OUTPUT and/or OUTPUT6 in the binary distributed format, or the standard ASCII global format. flagBOUT should be inputted on the same line as BINARYOUTPUT ; it can take one of the following values: On This setting chooses the binary distributed format for all output files that support it. Off This setting, which is also the default setting, chooses the standard ASCII global format for all output files.
CPUMAP	<pathandfilename>	This pair of sub-command keyword (characters) and corresponding value (characters) can be used to input a CPU map file – that is, a file that specifies to which CPU each subdomain is to be assigned. The format of this file is the same as that of the decomposition file, with the number of subdomains replaced by the number of CPUs, the number of elements in a given subdomain replaced by the number of subdomains in a given CPU, and the element ID number in each subdomain replaced by the subdomain ID number in each CPU. The default for <pathandfilename> is CPUMAP , in which case this file must be located in the directory where the simulation is launched. If <pathandfilename> is not specified and a file named CPUMAP is not found in the aforementioned directory, a default subdomain-to-CPU map is internally generated by AERO-S .

Next: [DISPLACEMENTS](#), Previous: [BINARY](#)

15 BOUNDARY CONVECTION *S*

Command Statement:	CONVECTION [<LOADSET_ID>]
--------------------	----------------------------------

The **CONVECTION** command statement is used to specify nodal convection type boundary conditions. Each node can have only one degree of freedom. This command statement can be used to solve a prescribed boundary convection problem for both statics and dynamics. The input format is given below.

Note 1: This command contributes to the construction of the right-hand side vector only. For the left-hand side (stiffness matrix) contribution, **AERO-S** uses boundary convection elements and information specified in the **MATERIAL** command.

CONVECTION [LOADSET_ID]

NODE#	H-COEFF	AREA	TA
-------	---------	------	----

LOADSET_ID

Optional non-negative integer which identifies explicitly the "load" set to which the source term generated by this command belongs to (integer). The default value is 0. Hence, the CONVECTION command can be repeated as many times as desired within the same input file using each time a different value for LOADSET_ID and different data. The [LOADCASE](#) command can refer to LOADSET_ID to define one or multiple "load" cases for static analysis (see the [STATICS](#) command and the explanation of its sub-command keyword CASES), and/or the "load" case for dynamic analysis.

NODE#

Node number where the convection is specified (integer).

H_COEFF

Convection coefficient at the prescribed node (float).

AREA

Value of the node cross-sectional area (float).

TA

Ambient temperature around the node (float).

Next: [FLUX](#), Previous: [CONVECTION](#)

16 BOUNDARY DISPLACEMENTS *S*

Command Statement: **DISPLACEMENTS**

The DISPLACEMENTS command is used to prescribe nodal displacements and/or rotations, either directly, or via the definition of a surface using the command [SURFACETOPO](#). In the latter case, the specified displacement or rotation is applied to each specified local degree of freedom of each node of that surface. The user can specify up to three displacements per node if a node can have up to three degrees of freedom, and up to three displacements and three rotations if it can have up to six degrees of freedom.

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: This command can also be used to specify nodal Dirichlet boundary conditions for a time-domain or frequency-domain acoustic simulation by setting DOF# to 8 (see below). Hence, it can also serve as an alternative to the [ATDDIR](#) and [HDIR](#) commands.

Note 3: In the context of a linearized (perturbation) analysis, the displacements boundary conditions specified under this command are interpreted as displacement boundary perturbations.

Note 4: When the analysis is performed in the complex plane — for example, in a frequency response analysis of a damped system — this command can be used to prescribe only the real part of nodal displacements and/or rotations, as in this case it automatically sets their imaginary part to zero.

The following two formats are available for this command and can be mixed.

DISPLACEMENTS

NODE#	DOF#	VALUE
-------	------	-------

SURFACE	SURFACE#	DOF#	VALUE
---------	----------	------	-------

NODE#

Node number where the displacement or rotation is specified (integer).

DOF#

Degree of freedom local number where the displacement or rotation is specified (integer).

VALUE

Value of the specified displacement or rotation (real).

SURFACE

Keyword indicating that a surface defined in [SURFACETOPO](#) is to be identified next by its integer identification number (characters).

SURFACE#

Integer identification of the surface defined in [SURFACETOPO](#) where the force VALUE is specified (integer).

Next: [FORCES](#), Previous: [DISPLACEMENTS](#)

17 BOUNDARY FLUXES

Command Statement:	FLUX [LOADSET_ID]
--------------------	--------------------------

The **FLUX** command statement is used to specify nodal heat sources (for example, products of nodal areas and nodal values of finite element fluxes) either directly, or via the definition of a surface using the command [SURFACETOPO](#). In the latter case, a specified heat source is applied to each node of the identified surface. This command statement can be used to define both statics and dynamics heat problems with prescribed temperature flux boundary conditions. Its input format is given below.

FLUX [LOADSET_ID]

NODE#	VALUE
--------------	--------------

SURFACE	SURFACE#	VALUE
----------------	-----------------	--------------

LOADSET_ID

Optional non-negative integer which identifies explicitly the "load" set to which the source term generated by this command belongs to (integer). The default value is 0. Hence, the **FLUXES** command can be repeated as many times as desired within the same input file using each time a different value for **LOADSET_ID** and different data. The [LOADCASE](#) command can refer to **LOADSET_ID** to define one or multiple "load" cases for static analysis (see the [STATICS](#) command and the explanation of its sub-command keyword [CASES](#)), and/or the "load" case for dynamic analysis.

NODE#

Node number where the flux is specified (integer).

VALUE

Value of the prescribed boundary flux (float).

SURFACE

Keyword indicating that a surface defined in [SURFACETOPO](#) is to be identified next by its integer identification number (characters).

SURFACE#

Integer identification of the surface defined in [SURFACETOPO](#) where the nodal heat source **VALUE** is specified (integer).

Next: [TEMPERATURES](#), Previous: [FLUX](#)

18 BOUNDARY FORCES *S*

Command Statement:	FORCES [LOADSET_ID]
--------------------	----------------------------

The **FORCES** command is used to prescribe external nodal forces and/or moments, either directly, or via the definition of a surface using the [SURFACETOPO](#) command. In the latter case, the specified force or moment is applied to each specified local degree of freedom of each node of that surface. The user can specify up to three forces per node if a node has three degrees of freedom attached to it, and up to three forces and three moments if it has six degrees of freedom attached to it.

By default, all prescribed forces and moments are interpreted as being of the *axial* type — that is, as being defined in the *fixed* nodal degree of freedom reference frames (see [NODES](#) and [NFRAMES](#)).

However, if a node has rotational degrees of freedom, the user can specify that the forces and/or moments prescribed at this node are of the *follower* type — that is, they act in a direction that remains constant in the local frame attached to the node where they are applied. This local frame coincides with the nodal degree of freedom reference frame (see [NODES](#) and [NFRAMES](#)) in the undeformed configuration. In the deformed configuration, the orientation of this local frame is defined by the rotation of the node to which it is attached. In other words, the specified nodal force or moment "follows" in this case the rotation of the node to which it is applied.

Finally, the user can also time-vary the specified forces and moments using the [MFTT](#) command.

Note 1: By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: This command can also be used to specify nodal Neumann boundary conditions for a time-domain acoustic simulation by setting **DOF#** to 8 (see below). Hence, it can also serve as an alternative to the [ATDNEU](#) command (see [ATDNEU](#)).

Note 3: Specifying a follower force or moment leads to an unsymmetric tangent "load" stiffness matrix during a [NONLINEAR](#) analysis.

The following two formats are available for this command and can be mixed.

FORCES [LOADSET_ID]

NODE#	DOF#	VALUE TYPE
SURFACE	SURFACE#	VALUE TYPE

LOADSET_ID	Optional non-negative integer which identifies explicitly the "load" set to which the source term generated by this command belongs to (integer). The default value is 0. Hence, the FORCES command can be repeated as many times as desired within the same input file using each time a different value for LOADSET_ID and different data. The LOADCASE command can refer to LOADSET_ID to define one or multiple "load" cases for static analysis (see the STATICS command and the explanation of its sub-command keyword CASES), and/or the "load" case for dynamic analysis.	
NODE#	Node number where the force or moment is specified (integer).	
DOF#	Degree of freedom local number where the force or rotation is specified (integer).	
VALUE	Value of the specified force or moment (real).	
TYPE	By default, all specified nodal forces and moments are considered to be of the axial type. However, if this parameter is set to FOLLOWER and the node where a force or moment is specified has rotational degrees of freedom, this specified nodal force or moment is considered to be of the follower type (characters).	
SURFACE	Keyword indicating that a surface defined in SURFACETOPO is to be identified next by its integer identification number (characters).	
SURFACE#	Integer identification of the surface defined in SURFACETOPO where the force or moment VALUE is specified (integer).	

Next: [BUCKLE](#), Previous: [FORCES](#)

19 BOUNDARY TEMPERATURES *S*

Command Statement:	TEMPERATURES
--------------------	---------------------

The TEMPERATURES command statement is used to specify prescribed nodal temperature type boundary conditions either directly, or via the definition of a surface using the command [SURFACETOPO](#). In the latter case, a specified temperature is applied to each node of the identified surface. For structural analysis using AERO-S, this command statement causes AERO-S to construct a thermal load based on the prescribed nodal temperatures and the reference temperatures (see [MATLAW](#), see [TOPOLOGY](#), and use this load as usual in a structural analysis).

Note 1: The default value of a nodal temperature is the reference temperature T_a of the element containing this node (see [MATERIAL](#)).

The input format of this command is given below.

TEMPERATURES

NODE#	VALUE
-------	-------

SURFACE	SURFACE#	VALUE
---------	----------	-------

NODE#	Node number where the temperature is specified (integer).
VALUE	Value of the prescribed temperature (float).
SURFACE	Keyword indicating that a surface defined in SURFACETOPO is to be identified next by its integer identification number (characters).
SURFACE#	Integer identification of the surface defined in SURFACETOPO where the temperature VALUE is specified (integer).

Next: [ROBC](#), Previous: [TEMPERATURES](#)

20 BUCKLING

Command Statement:	BUCKLE
--------------------	---------------

The BUCKLE command statement is to be used together with the GEPS, IDISP6 (with $\alpha = -1$), and EIGEN command statements to request the buckling analysis of a given structure.

When the EIGEN, GEPS, and IDISP6 (with $\alpha = -1$) command statements are specified in the input file, the BUCKLE command statement signals to FEM that the eigenvalue problem to be solved is $KU = -\lambda K_G U$, where K_G is the geometric stiffness matrix associated with a

displacement field specified under the `IDISP6` command statement, and whose computation is triggered by the presence of the `GEPS` command statement in the input file. For each eigenvalue λ , the corresponding buckling load is λf where f is the load that created the displacement field specified under the `IDISP6` command.

An example input file that illustrates a buckling analysis can be found in `FEM.d/fem_examples/Buckle.d`. The syntax for invoking this option is given below.

BUCKLE

Next: [COMMENTS](#), Previous: [BUCKLE](#)

21 CLUSTERING SNAPSHOTS OR CONSTRUCTING A REDUCED BASIS

Command Statement: **ROBC**

The `ROBC` command can be used to perform *either* of the following tasks:

- Compress using SVD (Singular Value Decomposition) one or multiple specified sets of primal displacement, velocity, and/or acceleration solution snapshots, and/or one or multiple specified sets of previously computed primal or dual Reduced-Order Basis (ROB) vectors, to construct a global or a set of local primal ROBs.
- Compress using SVD one or multiple specified sets of dual (Lagrange multiplier) solution snapshots, **and/or** one or multiple specified sets of previously computed dual ROB vectors to construct a global or a set of local dual ROBs. In this case, SVD is appropriate only if the Lagrange multipliers are introduced to enforce equality constraints.
- Compress using NMF (Non-negative Matrix Factorization) one or multiple specified sets of dual (Lagrange multiplier) solution snapshots to construct a global or a set of local dual ROBs. In this case, NMF is appropriate if the Lagrange multipliers are introduced to enforce inequality constraints (e.g. contact).
- Cluster a set of displacement solution snapshots (and assign to constructed clusters other quantities such as associated velocity, acceleration, and/or Lagrange multiplier solution snapshots) using the k-means algorithm in order to:
 - Construct in a followup simulation a set of *local* ROBs, using this same `ROBC` command and the computational approach described in *D. Amsallem, M. Zahr and C. Farhat, "Nonlinear Model Order Reduction Based on Local Reduced-Order Bases," International Journal for Numerical Methods in Engineering*, Vol. 92, pp. 891-916 (2012).
 - Provide training data for the ECSW hyper reduction method described in *C. Farhat, P. Avery, T. Chapman and J. Cortial, "Dimensional Reduction of Nonlinear Finite Element Dynamic Models with Finite Rotations and Energy-Conserving Mesh Sampling and Weighting for Computational Efficiency," International Journal of Numerical Methods in Engineering*, Vol. 98, pp. 625-662 (2014) and analyzed in *C. Farhat, T. Chapman and P. Avery, "Structure-Preserving, Stability, and Accuracy Properties of the Energy-Conserving Sampling and Weighting (ECSW) Method for the Hyper Reduction of Nonlinear Finite Element Dynamic Models," International Journal for Numerical Methods in Engineering*, Vol. 102, pp. 1077-1110 (2015).
- Orthonormalize a previously computed global or local primal ROB with respect to the mass matrix arising from the current ASCII Input Command Data file.

In the first and second cases:

- If scaling is requested (see the `SSCALI` sub-command):
 - All inputted primal or dual snapshots are scaled using their associated weights – that is, the square roots of the quadrature weights of the time-continuous snapshot correlation matrix (see the explanation of the `SNAPFI` sub-command for more details) – before these snapshots are compressed using the SVD.
 - All inputted primal ROB vectors are scaled before data compression is performed using their associated nonzero singular values.
 - All inputted dual ROB vectors are scaled before data compression is performed using their associated nonzero singular values, if SVD is chosen as the data compression algorithm.
- If normalization is requested (see the `SNORMA` sub-command), all inputted primal or dual snapshots or ROBs are normalized using the infinity (max) norm, before the SVD is performed on these snapshots or ROBs.
- If both scaling and normalization are requested, only normalization is performed.

Each computed cluster of displacement (and other associated quantities) is outputted by this same command as described in the explanation of the `DO_CLUSTERING` sub-command keyword. On the other hand, any computed ROB can be outputted using the result keyword `ROBDAF` under [OUTPUT](#).

Note 1: Currently, model reduction of linear or nonlinear problems with equality or inequality constraints is supported only for implicit dynamics, and for those cases where such constraints are linear — that is, for those cases where such constraints arise from:

- [NODALCONTACT](#) with `mode_v = 1` or `mode_v = 3`.
- [LMPC](#) with `mode_v = 1`.
- Constraint function elements with `mode_v = 1` (see [TOPOLOGY](#)).

Note 2: In such cases, solution snapshot collection for the construction of a dual ROB can be performed only when the constraint method (see [CONSTRAINTS](#)) is chosen to be:

- multipliers equipped with the FETI DP solver (see [STATICS](#)).
- augmented equipped with the parallel `mumps` solver (see [STATICS](#)).

Note 3: This command can be executed in parallel by partitioning the mesh associated with the underlying finite element model of interest using the `DECOMPOSE` command — or the corresponding command line to request mesh partitioning or provide `aeros` a previously computed mesh partition (see [DECOMPOSE](#)).

The input format of this command is given below.

ROBC					
SNAPFI	<pathandfilenameS1>	<pathandfilenameS2>	<pathandfilenameSN>
VELSNAPFI	<pathandfilenameV1>	<pathandfilenameV2>	<pathandfilenameVN>
ACCSNAPFI	<pathandfilenameA1>	<pathandfilenameA2>	<pathandfilenameAN>
ROBFI	<pathandfilenameR1>	<pathandfilenameR2>	<pathandfilenameRN>
SSCALI	flagss	flagsr			
SNORMA	flagsn	flagrn			
COMPRESS	flagco				
DIMENS	dim				
ROMENR	L				
MNORMA	flagmn				
SKIP	freq	[offset]			
USE_NMF	maxnoi	cvtolr			
USE_NMF	numrob	incsiz	numini	maxnoi	cvtolr
USE_GREEDY					
USE_PON	maxnoi	cvtolr	maxini	stepln	
NSUBS	nblocks				
DO_CLUSTERING	nclusters				

SNAPFI

Sub-command keyword for specifying one or multiple ASCII or binary files containing solution snapshots and associated weights (characters). These snapshots can be:

- Primal (displacement, velocity, and/or acceleration) or dual (Lagrange multiplier) solution snapshots, if this command is used to construct a primal or dual, global or local ROB.
- Displacement solution snapshots, if this command is used for the purpose of clustering.

If the solution snapshots were generated in the time domain, the weight of a solution snapshot is the square root of the average of the time-steps separating this snapshot from the previous and next ones, except for the first and last outputted solution snapshots which are computed as follows: the weight of the first solution snapshot is the square root of the time-step separating this snapshot and the next one, and that of the last solution snapshot is computed as the square root of the time-step separating this snapshot and the previous one.

<pathandfilenameS>

Path and name of each ASCII or binary file containing solution snapshots and associated weights (characters). If more than one file is specified, all files are read. The union of all inputted solution snapshot vectors (and consistent ROB vectors inputted after the ROBFI sub-command keyword) is either:

- Compressed to construct the desired ROB: in this case, all inputted solution snapshots can be scaled by their associated weights or normalized using the infinity norm before they are compressed using the SVD, if requested using SSCLAI or SNORMA, respectively.
- Or clustered (see DO_CLUSTERING).

VELSNAPFI

Optional sub-command keyword for specifying one or multiple ASCII or binary files containing velocity solution snapshots (characters). It should be used only when this command is used for clustering solution snapshots. In this case, the velocity solution snapshots are regrouped in the same clusters as the associated displacement solution snapshots and saved in the binary files <pathandfilenameV1>.clusterj, where j is the cluster index.

<pathandfilenameV>

Path and name of each ASCII or binary file containing velocity solution snapshots (characters). If more than one file is specified, all files are read. The union of all inputted velocity solution snapshots is clustered when the DO_CLUSTERING sub-command keyword is used (see below).

ACCSNAPFI

Optional sub-command keyword for specifying one or multiple ASCII or binary files containing acceleration solution snapshots (characters). It should be used only when this command is used for clustering solution snapshots. In this case, the acceleration solution snapshots are regrouped in the same clusters as the associated displacement solution snapshots and saved in the binary files <pathandfilenameA1>.clusterj, where j is the cluster index.

<pathandfilenameA>

Path and name of each ASCII or binary file containing acceleration solution snapshots (characters). If more than one file is specified, all files are read. The union of all inputted acceleration solution snapshots is clustered when the DO_CLUSTERING sub-command keyword is used (see below).

ROBFI

Sub-command keyword (characters) for specifying one or multiple ASCII or binary files containing:

- ROB vectors.
- Associated weights (singular values, if these vectors were generated using SVD.)

ROBFI

Sub-command keyword (characters) for specifying one or multiple ASCII or binary files containing:

- ROB vectors.
- Associated weights (singular values, if these vectors were generated using SVD).

<pathandfilenameR>

Path and name of each ASCII or binary file containing ROB vectors (and when applicable, associated weights) to be considered for constructing the desired ROB (characters). If more than one file is specified, all files are read. The union of these ROB vectors (and potentially solution snapshots inputted after the SNAPFI sub-command keyword) is compressed by SVD or NMF, as specified, to construct the desired ROB. When requested using SCALI or SNORMA, all inputted ROB vectors are scaled by their associated weights or normalized using the infinity norm, respectively, if SVD is chosen as the data compression algorithm.

SCALI

Sub-command keyword for specifying whether to scale or not the snapshots inputted using SNAPFI, or the ROB vectors inputted using ROBFI by their associated weights, when this command is used for the purpose of constructing a desired ROB and SVD is chosen as the data compression algorithm (characters).

flagss

If this flag is set to YES, the snapshots inputted using SNAPFI are scaled by their associated weights before they are used to construct a ROB. If it is set to NO, which is the default setting, they are left as is.

flagsr

If this flag is set to YES, the ROB vectors inputted using ROBFI are scaled by their associated weights before they are used to construct a ROB. If it is set to NO, which is the default setting, they are left as is.

SNORMA

Sub-command keyword for specifying whether to normalize or not – using the infinity norm – the snapshots inputted via SNAPFI, or the ROB vectors inputted using ROBFI, when this command is used for the purpose of constructing a desired ROB and SVD is chosen as the data compression algorithm (characters).

flagsn

If this flag is set to YES, the snapshots inputted using SNAPFI are normalized using the infinity norm before they are used to construct a ROB. If it is set to NO, which is the default setting, they are left as is.

flagrn

If this flag is set to YES, the ROB vectors inputted using ROBFI are normalized using the infinity norm before they are used to construct a ROB. If it is set to NO, which is the default setting, they are left as is.

COMPRESS

Sub-command keyword relevant only when the ROBC command is used specifically for orthonormalizing a previously computed global or local primal ROB with respect to the mass matrix arising from the current ASCII Input Command Data file. Its purpose is to specify whether or not to compress the solution snapshots inputted under SNAPFI and/or ROBFI (characters).

flagco

This flag can have one of the following two settings:

- NO: In this case, the information inputted under SNAPFI or ROBFI should be the column vectors of a previously computed global or local primal ROB, and the objective of the ROBC command should be to orthonormalize this primal ROB with respect to the mass matrix arising from the current ASCII Input Command Data file.
- YES: This is the default setting. In this case, the information inputted under SNAPFI and/or ROBFI is compressed using SVD or NMF, as specified.

DIMENS

Sub-command keyword for specifying the desired dimension of the ROB to be constructed (characters). Note that if both ROMENR and this sub-command keyword are specified in the same ASCII Input Command Data file, this sub-command keyword takes priority. If none of them is specified and the ROB is constructed using SVD, the dimension of this ROB is set to the rank of the snapshot matrix.

dim

Dimension of the ROB to be constructed (integer). The default value is the minimum between the total number of degrees of freedom of the underlying finite element model, and the total number of vectors to be compressed.

ROMENR

Sub-command keyword that is relevant only when SVD is chosen as the data compression algorithm for constructing a global or local, primal or dual ROB. It requests using the criterion based on the relative energy of the solution snapshots captured by the first SVD vectors to truncate the SVD basis and therefore determine the appropriate dimension \underline{k} of the ROB to be constructed (characters). Specifically, \underline{k} is determined in this case so that the square of the relative error gives an indication of the magnitude of the "missing" information – that is

$$\frac{\sum_{i=1}^k \sigma_i^2(\mathbf{S})}{\sum_{i=1}^r \sigma_i^2(\mathbf{S})} = \mathcal{E}_{SVD} \Rightarrow 1 - \mathcal{E}_{SVD} = \frac{\sum_{i=k+1}^r \sigma_i^2(\mathbf{S})}{\sum_{i=1}^r \sigma_i^2(\mathbf{S})}$$

where $\mathbf{S} \in R^{N \times N_s}$ is the matrix of N_s solution snapshots, \underline{r} is its rank, \underline{N} is the size of the high-dimensional model, σ_i is the i -th singular value of \mathbf{S} , $k \leq r \leq N$, and \mathcal{E}_{SVD} is the desired ratio of the energy of the solution snapshots captured by the constructed ROB and total energy of the computed solution snapshots and is specified below. Note that if both DIMENS and this sub-command keyword are specified in the same ASCII Input Command Data file, DIMENS takes priority. If none of them is specified and the ROB is constructed using SVD, the dimension of this ROB is set to the rank of the snapshot matrix.

 \underline{k}

Desired value of the ratio \mathcal{E}_{SVD} of the energy of the solution snapshots to be captured by the constructed ROB, and total energy of the computed solution snapshots (real).

MNORMA

Sub-command keyword for specifying the SVD-based computation of the desired global or local, primal ROB (characters).

flagmn

This flag can have one of the following three settings:

- IM: This is the default setting. In this case, two equivalent but different ROBs are computed as follows and in this order:
 - First, one ROB that is orthonormalized with respect to the identity matrix.
 - Second, a counterpart ROB whose columns span the same subspace, but which is orthonormalized

with respect to the mass matrix. In this case, the construction of the second ROB does not incur the factorization of the high-dimensional mass matrix, but that of the reduced mass matrix. Note that if the high-dimensional mass matrix is singular, the reduced mass matrix may or may not be singular, depending on the content of the ROB.

- **MI:** In this case, two equivalent but different ROBs are computed as follows and in this order:
 - First, one ROB that is orthonormalized with respect to the mass matrix and computed using an algorithm that factorizes this mass matrix. Hence, in this case, the user must ensure that the mass matrix of the finite element model is non singular.
 - A counterpart ROB whose columns span the same subspace, but which is orthonormalized with respect to the identity matrix.
- **ID:** In this case, a single ROB that is orthonormalized with respect to the identity matrix is computed.

SKIP	Sub-command keyword for specifying whether or not to skip some of the solution snapshots inputted using the SNAPFI sub-command keyword for the purpose of their clustering or the construction of the desired ROB (characters).
freq	Frequency (every so many) at which to skip a solution snapshot in any specified solution snapshot file (integer). The default value is 1.
offset	Offset in terms of the solution snapshot number to apply before starting the skipping process (integer). The default value is 0.
USE_NMF	Sub-command keyword that is relevant only for the solution of the class of constrained problems identified above, using the methods and algorithms also identified above. It activates the computation of a <i>dual</i> ROB using the non-negative matrix factorization approach based on the alternating Non-Negative Least-Squares (NNLS) method (characters).
maxnoi	Maximum number of outer-iterations for the non-negative matrix factorization algorithm (integer).
cvtolr	Relative convergence tolerance of the Frobenius norm of the dual basis increment computed at each iteration of the non-negative matrix factorization method (real). This parameter is relevant only for the solution of the class of constrained problems identified above, using the methods and algorithms also identified above.
numrob	Number of ROBs to be computed (integer). The default value is 1.
incsiz	Size increment to apply when computing multiple ROBs (integer).
numini	Number of random initializations to perform when computing each ROB (integer). The default value is 1.
USE_GREEDY	Sub-command keyword for activating the computation of a <i>dual</i> ROB using a Greedy method (characters). This parameter is relevant only for the solution of the class of constrained problems identified above, using the methods and algorithms also identified above.
USE_PQN	Sub-command keyword for activating the computation of a <i>dual</i> ROB using the non-negative matrix factorization approach based on a projected quasi-Newton method (characters). It is relevant only for the solution of the class of constrained problems identified above, using the methods and algorithms also identified above.
maxini	Maximum number of inner-iterations that can be performed by the projected quasi-Newton method for computing the non-negative matrix factorization (integer).
stepln	Step length parameter for the projected quasi-Newton method for computing the non-negative matrix factorization (real).
nblkcs	Number of column-wise blocks defining the parallelization strategy for the alternating NNLS method (integer). The default value is 1.
DO_CLUSTERING	Sub-command keyword for clustering a set of displacement solution snapshots using the k-means algorithm. In this case, the generated solution snapshot clusters and their centroids are saved in the binary files <pathandfilenameS1>.cluster _j and <pathandfilenameS1>.cluster _j .centroid, respectively, where j designates the _j -th cluster and <pathandfilenameS1> is the <pathandfilenameS1> specified after the SNAPFI sub-command keyword. If the optional VELSNAPFI sub-command keyword is also specified (see above), the inputted velocity solution snapshots are regrouped in the same clusters as the associated displacement solution snapshots and saved in the binary files <pathandfilenameV1>.cluster _j , where j is the cluster index. Similarly, if the optional ACCSNAPFI sub-command keyword is also specified (see above), the inputted acceleration solution snapshots are regrouped in the same clusters as the associated displacement solution snapshots and saved in the binary files <pathandfilenameA1>.cluster _j , where j is the cluster index.
nclusters	Desired number of clusters (integer).

Next: [COMPOSITE](#), Previous: [ROBC](#)

22 COMMENTS

The '*' at the beginning of a line indicates that the subsequent input on the same line is a comment. For FEM, it can be placed anywhere.

Next: [CFRAMES](#), Previous: [COMMENTS](#)

23 COMPOSITE (OR ORTHOTROPIC SHELL OR ANISOTROPIC SOLID)

Command Statement:	COMPOSITE
--------------------	------------------

The **COMPOSITE** command is used to specify the properties of a composite, anisotropic, or orthotropic solid, shell, or membrane element made of a *linear* material. It currently supports only the solid elements type 17, 23, 24, 25, 72, 91 92, and 97, the (Kirchhoff) shell elements type 15, 1515, 20, and 2020, and the membrane elements type 128 and 129. All information concerning constitutive coefficients or the layering of a composite is inputted using this command.

For the above solid elements, the constitutive matrix, the coefficients of thermal expansion, and the hygroelastic coupling coefficients are

inputted using the `COEF` sub-command. The constitutive matrix is defined in the local frame of the element specified in the [CFRAMES](#) command. Entries in the matrix that are not specified are determined from symmetry if appropriate, or are set to zero.

For the shell elements, the composite (anisotropic, or orthotropic) constitutive law can be prescribed either by entering the $C_{i,j}$

coefficients of the 6 by 6 constitutive matrix, or by defining the material properties and geometrical characteristics for each layer of the composite. In the first case, the `COEF` sub-command should be used. In the second case, either of the `LAYC` and `LAYN` sub-commands can be used, depending on whether the coupling between bending and membrane effects is to be enforced or not, respectively. When many layers of the composite are made of the same anisotropic material, the `LAYD` and `LAYMAT` sub-commands can be used instead of the `LAYC` sub-command to simplify the data entry process. Likewise, in similar circumstances, the `LAYO` and `LAYMAT` sub-commands can be used instead of the `LAYN` sub-command.

Note 1: If a laminate layup is unsymmetric (with respect to the mid-plane), the plies must be numbered from the top side to the bottom side, with the top side defined by the direction of its element normals that must point from bottom to top.

Note 2: In a nonlinear analysis, the `COEF` sub-command can currently be used only to define an anisotropic hyperelastic or hyperviscoelastic material. Hence, it can be specified together with the following material laws: Linear, LinearPlaneStress, HyperElasticPlaneStress, HenckyElastic, StVenantKirchhoff, ViscoLinearElastic, ViscoStVenantKirchhoff, ViscoLinearPlaneStress, and ViscoHyperElasticPlaneStress (see [MATERIAL](#)). In the case of StVenantKirchhoff and HyperElastic, the material laws relate the Green-Lagrange (engineering) strain and its conjugate stress, the second Piola-Kirchhoff stress. In the case of HenckyElastic, the material law relates the Lagrangian Hencky (engineering) strain and its conjugate stress, the rotated Kirchhoff stress.

Note 3: The case of a *linear* hyperelastic material includes the special case of a *linear* hygrothermoelastic material represented by the constitutive equation

$$\sigma = C_{ela} : [\epsilon - \alpha(T - T_{ref}) - \beta(C - C_{ref})]$$

where σ and ϵ denote the vectorized stress and strain tensors, respectively; T denotes the temperature field; C denotes the moisture concentration (mole number); C_{ela} denotes the standard constitutive matrix for elasticity; α and β denote the vectorized tensors of thermal expansion and hygroelastic coupling coefficients, respectively; and T_0 and C_0 denote the temperature and moisture concentration of the reference (stress-free) state of the material, respectively. For example, T_{ref} can be chosen to be the ambient temperature T_a specified in [MATERIAL](#).

The input format for the composite `COMPOSITE` is given below.

COMPOSITE

COEF	attribute#	w_1	w_2	w_3	w_4	w_5	w_6
ROW# (i)	COLUMN# (j)	VALUE ($C_{i,j}$)	—	—	—	—	—
.	.						
ROW# (i)	COLUMN# (j)	VALUE ($C_{i,j}$)					

COEF Sub-command keyword used to directly input the coefficients of the constitutive law. Since the 6 by 6 constitutive matrix is symmetric, up to 21 independent coefficients may follow.

attribute# Integer value that corresponds to the composite (or orthotropic or anisotropic) attribute of the element.

w_1 Thermal expansion coefficient associated with the strain ϵ_x . It can be omitted, in which case its default value is 0.

w_2 Thermal expansion coefficient associated with the strain ϵ_y . It can be omitted, in which case its default value is 0.

w_3 Thermal expansion coefficient associated with the strain ϵ_z . It can be omitted, in which case its default value is 0.

w_4 Thermal expansion coefficient associated with the strain γ_{xy} . It can be omitted, in which case its default value is 0.

w_5 Thermal expansion coefficient associated with the strain γ_{yz} . It can be omitted, in which case its default value is 0.

w_6 Thermal expansion coefficient associated with the strain γ_{xz} . It can be omitted, in which case its default value is 0.

ROW# Integer value that corresponds to the row i of the coefficient $C_{i,j}$.

A row can be skipped if all its entries are zero.

COLUMN# Integer value that corresponds to the column j of the coefficient $[C_{i,j}]$.

VALUE Real value of the coefficient $[C_{i,j}]$.

For a solid or anisotropic shell or membrane element equipped with a numerically-enforced plane stress counterpart of a 3D material law (see [MATLAW](#)), the constitutive matrix C relates the stresses to the engineering strains in the element's local frame system {x ; y ; z} as follows:

$$\begin{pmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{xz} \end{pmatrix} = \begin{pmatrix} C_{1,1} & C_{1,2} & C_{1,3} & C_{1,4} & C_{1,5} & C_{1,6} \\ C_{1,2} & C_{2,2} & C_{2,3} & C_{2,4} & C_{2,5} & C_{2,6} \\ C_{1,3} & C_{2,3} & C_{3,3} & C_{3,4} & C_{3,5} & C_{3,6} \\ C_{1,4} & C_{2,4} & C_{3,4} & C_{4,4} & C_{4,5} & C_{4,6} \\ C_{1,5} & C_{2,5} & C_{3,5} & C_{4,5} & C_{5,5} & C_{5,6} \\ C_{1,6} & C_{2,6} & C_{3,6} & C_{4,6} & C_{5,6} & C_{6,6} \end{pmatrix} \begin{pmatrix} \epsilon_x - w1(T - Ta) \\ \epsilon_y - w2(T - Ta) \\ \epsilon_z - w3(T - Ta) \\ \gamma_{xy} - w4(T - Ta) \\ \gamma_{yz} - w5(T - Ta) \\ \gamma_{xz} - w6(T - Ta) \end{pmatrix}$$

where Ta is the reference temperature specified in [MATERIAL](#).

For a (Kirchhoff) shell element that is *not* equipped with a numerically-enforced plane stress counterpart of a 3D material law (see [MATLAW](#)), the constitutive matrix C relates the forces and moments to the mid-surface strains and curvatures in the shell element's local frame system {x ; y ; z} as follows:

$$\begin{pmatrix} N_x \\ N_y \\ N_{xy} \\ M_x \\ M_y \\ M_{xy} \end{pmatrix} = \begin{pmatrix} C_{1,1} & C_{1,2} & C_{1,3} & C_{1,4} & C_{1,5} & C_{1,6} \\ C_{1,2} & C_{2,2} & C_{2,3} & C_{2,4} & C_{2,5} & C_{2,6} \\ C_{1,3} & C_{2,3} & C_{3,3} & C_{3,4} & C_{3,5} & C_{3,6} \\ C_{1,4} & C_{2,4} & C_{3,4} & C_{4,4} & C_{4,5} & C_{4,6} \\ C_{1,5} & C_{2,5} & C_{3,5} & C_{4,5} & C_{5,5} & C_{5,6} \\ C_{1,6} & C_{2,6} & C_{3,6} & C_{4,6} & C_{5,6} & C_{6,6} \end{pmatrix} \begin{pmatrix} \epsilon_x^0 - w1(T - Ta) \\ \epsilon_y^0 - w2(T - Ta) \\ \gamma_{xy}^0 - w3(T - Ta) \\ \kappa_x - w4(T - Ta) \\ \kappa_y - w5(T - Ta) \\ \kappa_{xy} - w6(T - Ta) \end{pmatrix}$$

where Ta is the reference temperature specified in [MATERIAL](#).

The element's local axes are defined with respect to the global reference frame by the three vectors defined in the [CFRAMES](#) command for the corresponding composite (orthotropic, or anisotropic) frame number that is specified in the [ATTRIBUTES](#) section. For shell elements, the constitutive matrix, C , can be decomposed into sub-matrices containing the bending and membrane properties of the shell element:

$$C = \begin{pmatrix} C_{mm} & C_{mb} \\ C_{bm} & C_{bb} \end{pmatrix}$$

Note the absence of the transverse shear behavior, as can be expected from a Kirchhoff type shell element. In the case of an isotropic material, or a single layer orthotropic or orthotropic material, there is no coupling between the bending and membrane behavior:

$$C_{mb} = C_{bm} = 0_{3 \times 3}$$

For an isotropic material, the membrane constitutive matrix can be defined in terms of the Young's modulus, E , Poisson's ratio, ν , and shell thickness, h as:

$$C_{mm} = \begin{pmatrix} \frac{Eh}{(1-\nu^2)} & \frac{\nu Eh}{(1-\nu^2)} & 0 \\ \frac{\nu Eh}{(1-\nu^2)} & \frac{Eh}{(1-\nu^2)} & 0 \\ 0 & 0 & \frac{Eh}{2(1+\nu)} \end{pmatrix}$$

while the bending constitutive matrix is given by:

$$C_{bb} = \begin{pmatrix} \frac{Eh^3}{12(1-\nu^2)} & \frac{\nu Eh^3}{12(1-\nu^2)} & 0 \\ \frac{\nu Eh^3}{12(1-\nu^2)} & \frac{Eh^3}{12(1-\nu^2)} & 0 \\ 0 & 0 & \frac{Eh^3}{24(1+\nu)} \end{pmatrix}$$

Using the [COEF](#) sub-command, different values of E , ν , and h can be utilized for building the membrane and bending components of the constitutive law, for example, if one wishes to adjust in some specific manner the bending and membrane behaviors of the shell element.

When the [COEF](#) sub-command keyword is used for a shell element that is *not* equipped with a numerically-enforced plane stress counterpart of a 3D material law (see [MATLAW](#)), the [MATERIAL](#) command must be used as follows: both structural and non-structural mass densities *per unit area* (see the section [Notes](#) in [MATERIAL](#) for the definition of a *non-structural* mass) must be inputted at the fourth position after the attribute number, and for shells, the total thickness must also be defined at the seventh position after the attribute number if a stress analysis is requested.

For shell elements, if the pre-integrated constitutive matrix C is not available, the [LAYC](#) or [LAYN](#) sub-command keywords (or their counterparts [LAYD](#), [LAYO](#) and [LAYMAT](#)) should be used to input the material properties and geometrical characteristics of each composite layer. If either [LAYN](#) or [LAYO](#) is used, no coupling between bending and membrane effects is enforced explicitly during integration through the

thickness — that is: $C_{1,4} = C_{1,5} = C_{1,6} = 0$, $C_{2,4} = C_{2,5} = C_{2,6} = 0$ and $C_{3,4} = C_{3,5} = C_{3,6} = 0$.

The input format given below is the same for both LAYC and LAYN sub-commands.

LAYC (or LAYN)		attribute#
k	$E_1^{(k)}$	$E_2^{(k)}$
	$\nu_{12}^{(k)}$	$G_{12}^{(k)}$
	$\mu_{1,12}^{(k)}$	$\mu_{2,12}^{(k)}$
	$\rho^{(k)}$	$h^{(k)}$
	$\theta_F^{(k)}$	$w_1^{(k)}$
	$w_2^{(k)}$	$w_{12}^{(k)}$
.	.	.
k	$E_1^{(k)}$	$E_2^{(k)}$
	$\nu_{12}^{(k)}$	$G_{12}^{(k)}$
	$\mu_{1,12}^{(k)}$	$\mu_{2,12}^{(k)}$
	$\rho^{(k)}$	$h^{(k)}$
	$\theta_F^{(k)}$	$w_1^{(k)}$
	$w_2^{(k)}$	$w_{12}^{(k)}$

The input format given below is the same for both LAYD and LAYO sub-commands.

LAYD (or LAYO)		attribute#
k	LAYER_MATERIAL_ID	$h^{(k)}$
		$\theta_F^{(k)}$
LAYMAT		
LAYER_MATERIAL_ID	$E_1^{(k)}$	$E_2^{(k)}$
	$\nu_{12}^{(k)}$	$G_{12}^{(k)}$
	$\mu_{1,12}^{(k)}$	$\mu_{2,12}^{(k)}$
	$\rho^{(k)}$	$w_1^{(k)}$
	$w_2^{(k)}$	$w_{12}^{(k)}$
or		
LAYER_MATERIAL_ID	$E_1^{(k)}$	$E_2^{(k)}$
	$\nu_{12}^{(k)}$	$G_{12}^{(k)}$
	$\rho^{(k)}$	
.	.	.
LAYER_MATERIAL_ID	$E_1^{(k)}$	$E_2^{(k)}$
	$\nu_{12}^{(k)}$	$G_{12}^{(k)}$
	$\mu_{1,12}^{(k)}$	$\mu_{2,12}^{(k)}$
	$\rho^{(k)}$	$w_1^{(k)}$
or		
LAYER_MATERIAL_ID	$E_1^{(k)}$	$E_2^{(k)}$
	$\nu_{12}^{(k)}$	$G_{12}^{(k)}$
	$\rho^{(k)}$	

LAYO	LAYC, LAYD, LAYN, or	Sub-command keyword used for inputting geometrical and material properties for each layer of a composite shell element (characters).
	attribute#	Integer value that corresponds to the composite (orthotropic) attribute of the element.
	k	Integer value that corresponds to the layer number.
	$E_1^{(k)}$	Young's modulus in the local direction $1^{(k)}$ (that is, in the direction of the fibers for the k^{th} layer).
	$E_2^{(k)}$	Young's modulus in the local direction $2^{(k)}$ (that is, in the direction orthogonal to the fibers for the k^{th} layer).
	$\nu_{12}^{(k)}$	Poisson's ratio for transverse strain in the local direction $2^{(k)}$ when stressed in the local direction $1^{(k)}$ for $\sigma_1^{(k)} = \text{constant}$ and all other stresses zero.
	$G_{12}^{(k)}$	Transverse shear modulus in the plane $1^{(k)} - 2^{(k)}$ of the layer.
	$\mu_{1,12}^{(k)}$	Coefficient of mutual influence of the first kind which characterizes stretching in the local direction $1^{(k)}$ caused by shear in the plane $1^{(k)} - 2^{(k)}$ of the layer for $\tau_{12}^{(k)} = \text{constant}$ and all other stresses zero.
	$\mu_{2,12}^{(k)}$	Coefficient of mutual influence of the first kind which characterizes stretching in the local direction $2^{(k)}$ caused by shear in the plane $1^{(k)} - 2^{(k)}$ of the layer for $\tau_{12}^{(k)} = \text{constant}$ and all other stresses zero.
	$\rho^{(k)}$	Density (mass per unit volume) of the structural mass type of the material of the k^{th} layer (real).
	$h^{(k)}$	Thickness of the k^{th} layer.
	$\theta_F^{(k)}$	Angle between a reference vector and the fibers in the layer that defines the orientation of these fibers. If CMP_FRM# is specified as an attribute for an element containing this layer under the ATTRIBUTES command, the reference vector is the projection onto the plane of that element of the first of the three vectors defining a local frame for this element and specified in the CFRAMES command. On the other hand, if a reference angle θ_{Ref} is specified as an attribute for an element containing this layer under the ATTRIBUTES command, the reference vector is the vector obtained by rotating the directional edge connecting local nodes 1 and 2 of this

$w_1^{(k)}$	element around its normal by an angle equal to θ_{Ref} . In both cases, this angle must be inputted in degrees.
$w_2^{(k)}$	Coefficient of thermal expansion in the direction $1^{(k)}$. It can be omitted, in which case its default value is 0.
$w_{12}^{(k)}$	Coefficient of thermal expansion in the direction $2^{(k)}$. It can be omitted, in which case its default value is 0.
LAYER_MATERIAL_ID	Coefficient of thermal expansion associated with the in-plane shear strain. It can be omitted, in which case its default value is 0.
LAYMAT	Identifier of a set of material properties (integer).
	Sub-command keyword that can be used for inputting the properties of a layer when it is made of a two-dimensional orthotropic material (characters).

The report number CU-CAS-94-16, "The 3-node Composite Shell and Isotropic Timoshenko Beam Elements" by Fran\c{c}ois M. Hemez, provides a detailed description of the theory and implementation for the type-20 composite (orthotropic) shell element.

Next: [CONDITION](#), Previous: [COMPOSITE](#)

24 COMPOSITE (ORTHOTROPIC SHELL OR ANISOTROPIC SOLID) ELEMENT FRAMES

Command Statement: **CFRAMES**

The CFRAMES command statement is used to specify the orientation of composite laminates and orthotropic or anisotropic elements with respect to the global reference frame. The input format of this command is given below with the number of lines equal to the number of different composite (or orthotropic or anisotropic) element frames referenced in the ATTRIBUTES command.

CFRAMES

CMP_FRM#	$S_{1x} S_{1y} S_{1z}$	$S_{2x} S_{2y} S_{2z}$	$S_{3x} S_{3y} S_{3z}$
----------	------------------------	------------------------	------------------------

CMP_FRM#	Identification of a composite frame (integer).
$S_{1x} S_{1y} S_{1z}$	The first axis, the local x-axis , expressed in the global frame. For LAYC and LAYN type composites, the orientation of the fibers of the layers should be defined with respect to the projection of this axis onto the plane of the element.
$S_{2x} S_{2y} S_{2z}$	The second axis of the frame expressed in the global frame (floats).
$S_{3x} S_{3y} S_{3z}$	The third axis of the frame expressed in the global frame (floats).

Next: [CONSTRAINTS](#), Previous: [CFRAMES](#)

25 CONDITION NUMBER

Command Statement: **CONDITION**

The CONDITION command statement is used to request the evaluation of the condition number of the system being solved by FEM. The input format is given below.

CONDITION

TOLERANCE	MAXITR
-----------	--------

TOLERANCE	Error tolerance for computing the lowest and highest eigenvalues of the finite element model using the inverse power and power methods, respectively (float). The default value is 10^{-3} .
MAXITR	Maximum number of iterations for computing the lowest and highest eigenvalues of the finite element model using the inverse power and power methods, respectively (integer). The default value is 100.

Next: [RMSHC](#), Previous: [CONDITION](#)

26 CONSTRAINTS

Command Statement: **CONSTRAINTS**

The **CONSTRAINTS** command statement is used to specify a default method for enforcing the constraints defined or associated with a problem. Its input format is given below.

Note 1: The Lagrange multiplier method for enforcing constraints associated with contact (see [CONTACTSURFACES](#)) or tied (see [TIEDSURFACES](#)) surfaces is currently available for all analyses. However for enforcing constraints of other origins, it is currently available for all but explicit dynamic analysis. Currently, only the `spooles` and `mumps` direct solvers with pivoting turned on, the `superlu` direct solver, and the `gmres` and `FETI DP` iterative solvers support this approach for solving systems of equations with *equality* constraints (see [STATICS](#)), but only the `FETI DP` iterative solver supports this approach for systems with *inequality* constraints.

Note 2: The augmented Lagrangian method for enforcing constraints is currently available only for [NONLINEAR static](#) and [NONLINEAR implicit](#) dynamic analyses.

Note 3: The elimination method is supported only by the following equation solvers (see [STATICS](#)): `skyline`, `sparse`, `mumps` in the context of a single domain, and `spooles`. Furthermore, this method is currently supported only when applied to all constraints: in other words, it cannot be combined with another method for enforcing constraints.

CONSTRAINTS

METHOD

METHOD

Specified default method for enforcing constraints (characters). This specified method can be overruled in the [CONTACTSURFACES](#), [MATERIAL](#), [LMPC](#), [NODALCONTACT](#), and [TIEDSURFACES](#) commands.

multipliers
In this case, which is also the default case, **AERO-S** uses the Lagrange multiplier method for enforcing all constraints associated with the [LMPC](#), [NODALCONTACT](#), [CONTACTSURFACES](#), and [TIEDSURFACES](#) commands, and with joint and rigid elements (see [TOPOLOGY](#)). **Warning:** before relying on this default value of **METHOD**, the user should read above the notes describing the scope of the Lagrange multiplier method.

elimination [factol [lhstol [rhstol]]] In this case, **AERO-S** uses an elimination method for enforcing all constraints associated with the commands [LMPC](#) and [TIEDSURFACES](#), and with joint and rigid elements (see [TOPOLOGY](#)). This method can be configured with the following three parameters:

- **factol**. This parameter is a tolerance used in the factorization of the constraint Jacobian matrix to define a zero pivot as a pivot whose absolute value is less or equal to **factol** $\times \epsilon$
[maxdiagcoeff], where ϵ is the machine precision and maxdiagcoeff is the largest diagonal coefficient of the matrix. Note that the factorization is performed only to compute the Reduced Row Echelon Form (RREF) of the linearized constraint equations if these equations cannot be arranged in such a form by simply permuting the rows and/or columns of the constraint Jacobian matrix. The factorization method is QR if **AERO-S** is configured with the C++ template library for linear algebra Eigen 3, or the Gauss-Jordan elimination method otherwise. The default value of this parameter is **10**.
- **lhstol**. This parameter is another tolerance used to set to zero any coefficient of the left hand side of the RREF of the linearized constraint equations that is smaller than **lhstol** $\times \epsilon$, where ϵ is the machine precision. The sparsity of this matrix is important for computational efficiency. The default value of this parameter is **10**.
- **rhstol**. This parameter is yet another tolerance used to set to zero any coefficient of the right hand side of the RREF of the linearized constraint equations that is smaller than **rhstol** $\times \epsilon$, where ϵ is the machine precision. The default value of this parameter is **0**.

This method is particularly efficient when **MORTAR_TYPE** in [TIEDSURFACES](#) is set to **1** (**AERO-S**'s dual mortar method) because in this case the constraint equations are constructed in RREF.

In this case, **AERO-S** uses the penalty method for enforcing all constraints associated with the [LMPC](#), [NODALCONTACT](#), [CONTACTSURFACES](#), and [TIEDSURFACES](#) commands, and with joint and rigid elements (see [TOPOLOGY](#)). The parameter **beta** should be a large positive number, typically of the order of **10⁸** (no default value is provided).

penalty beta
In this case, **AERO-S** uses the penalty method for enforcing all constraints associated with the [LMPC](#), [NODALCONTACT](#), [CONTACTSURFACES](#), and [TIEDSURFACES](#) commands, and with joint and rigid elements (see [TOPOLOGY](#)). The parameter **beta** should be a large positive number, typically of the order of **10⁸** (no default value is provided).

augmented beta

In this case, **AERO-S** uses the augmented Lagrangian method for enforcing all constraints associated with the [LMPC](#), [NODALCONTACT](#), [CONTACTSURFACES](#), and [TIEDSURFACES](#) commands, and with joint and rigid elements (see [TOPOLOGY](#)). The parameter **beta** should be a large positive number, typically of the order of **10⁸** (no default value is provided).

Next: [CONTACTSURFACES](#), Previous: [CONSTRAINTS](#)

27 CONSTRUCTING A REDUCED MESH

Command Statement: **RMSHC**

The **RMSHC** command can be used to:

- Construct a reduced mesh (or a series of them, when considering local Reduced-Order Bases (ROB)s for nonlinear model reduction)

that can be used for effectively hyper reducing a discrete reduced-order model, by sampling the mesh described in this ASCII Input Command Data file using:

- A single global ROB or a set of local ROBs.
- One or more specified dual ROBs for the solution of a class of constrained problems (see below for limitations).
- Training solutions specified on a single or multiple meshes from which training forces can be constructed.
- A non-negative least-squares algorithm.

After a reduced mesh is constructed, the trace on this mesh of the corresponding training ROB is computed and outputted, if `SAMPLMSH` is specified under [OUTPUT](#).

- Update the coordinates of previously constructed reduced meshes in order to enable their reuse in parametric shape studies when the topology of the high-dimensional meshes is maintained constant. In this case:
 - The coordinates specified in the ASCII Input Command Data file are the updated coordinates.
 - The element weights of a previously computed reduced mesh must be inputted in this file using [ATTRIBUTES](#) in order to specify the desired reduced mesh.
 - The sole purpose of this command becomes that of outputting the updated reduced mesh file and therefore many aspects of this command become irrelevant.
- Pre-compute any reduced-order terms and/or operators that are constant, and therefore do not necessitate hyper reduction, are required by a subsequent simulation based on a hyper reduced projection-based reduced order model, and whose associated **AERO-S** commands are included in the ASCII Input Command Data file. Specifically:
 - Reduced-order displacement and/or velocity initial conditions, in the presence of [IDISPLACEMENTS](#) and/or [IVELOCITIES](#) in the input file.
 - The reduced-order mass matrix, when the ROB is not mass-orthonormalized (see [READMODE](#)).
 - Reduced-order external, constant (non-follower) forces, in the presence of [FORCES](#) and/or [GRAVITY](#) in the input file.
 - Reduced-order linear multipoint constraints, in the presence of [LMPC](#) in the input file.
- Hyper reduce offline high-dimensional, non zero, initial displacement and/or velocity condition(s) specified in the same input file containing this command — which can be desired or required. The latter scenario occurs when only the trace of the ROB on the dofs of the reduced mesh is available - for example, when the ROB is randomized.

The computed or updated reduced meshes, the traces of the training ROBs on these meshes, the constant reduced-order information mentioned above, and the offline hyper reduced initial conditions can be outputted using `SAMPLMSH` in [OUTPUT](#).

Note 1: Currently, this command is not connected to [READROB](#) and consequently, it cannot select any ROB inputted via [READMODE](#) to perform any of the tasks it is designed to accomplish. Instead, it selects one of the two instances of a same ROB – namely, the orthonormalized instance or mass-orthonormalized instance – computed using the [ROBC](#) command and outputted when `ROBDATAF` is specified as a result in [OUTPUT](#), as follows:

- If the time-integration scheme specified in [DYNAMICS](#) is of the explicit type, this command selects the mass-orthonormalized ROB instance for its assigned task, regardless of the setting of the `flag_mnb` flag (see below the `USE_MNB` keyword).
- If the time-integration scheme is of the implicit type – which this command assumes by default – and the `flag_mnb` flag is set below to `on` (see below the `USE_MNB` keyword), this command also selects the mass-orthonormalized ROB instance for its assigned task.
- On the other hand, if the time-integration scheme is of the implicit type and `flag_mnb` is set to `off` (see below the `USE_MNB` keyword), this command selects orthonormalized ROB instance for its assigned task.

Note 2: By default:

- The reduced inertia forces associated with a linear structural dynamics problem and the component of the reduced inertia forces associated with a constant mass matrix in a nonlinear structural dynamics problem are not hyper reduced. This is because the part of these reduced forces whose computational complexity scales with the size of the high-dimensional model can be (and is automatically) precomputed efficiently.
- In a nonlinear structural dynamics problem, the component of the reduced inertia forces associated with a configuration-dependent mass matrix is automatically hyper reduced.

For eigen analysis, which can currently be performed using a reduced or hyper reduced computational model only external to **AERO-S**, the reduced inertia forces — and more specifically, the reduced mass matrix — may need to be hyper reduced for accuracy and/or computational efficiency reasons. In this case, the user should pay attention to the settings of the keywords `USE_CMF` and `USE_STR`.

Note 3: When the finite element model to be hyper reduced contains rotational degrees of freedom, the [DYNAMICS](#) command and the part of its input that identifies the time-integrator previously chosen for constructing the discrete reduced-order model to be hyper reduced must be included in the ASCII Input Command Data file.

Note 4: If the mesh sampling is to be trained also with the inertia forces, the velocity and acceleration snapshots must be provided using the `TRNSOL` sub-command keyword described below.

Note 5: Repeating within this command the *pair* of sub-command keywords (`PODR0B`, `TRNSOL`) `n` times, while using the same single instance of each other sub-command keyword, enables the construction of `n` hyper reduced meshes, one for each ROB specified in `PODR0B` — for example, for equipping a suite of `n` local ROBs with `n` associated reduced meshes.

Note 6: Currently, model reduction of linear or nonlinear problems with equality or inequality constraints is supported only for implicit dynamics, and for those cases where such constraints are linear — that is, for those cases where such constraints arise from:

- [NODALCONTACT](#) with `mode_v = 1` or `mode_v = 3`.
- [LMPC](#) with `mode_v = 1`.
- Constraint function elements with `mode_v = 1` (see [TOPOLOGY](#)).

Note 7: In such cases, snapshot collection for the construction of a dual basis can be performed only when the constraint method (see

[CONSTRAINTS](#)) is chosen to be:

- multipliers equipped with the FETI DP solver (see [STATICS](#)).
- augmented equipped with the parallel `mumps` solver (see [STATICS](#)).

Note 8: This command can be executed in parallel when the problem of interest is constraint-free, and model reduction is performed using a global ROB. In this case, the mesh sampling task is parallelized if the command [DECOMPOSE](#) is used to partition the mesh associated with the finite element model of interest, or executed from the command line (see [DECOMPOSE](#)) to request mesh partitioning or provide `aeros` a previously computed mesh partition. In particular, the `TRIVIAL` decomposition algorithm (see [DECOMPOSE](#)) is recommended for this purpose.

RMSHC

```

PODR0B    <pathandfilename1>      dimrb
DUALBASIS   <pathandfilename2>      dimdb
TRNSOL     <pathandfilename3>
TRNSOL     <pathandfilename3>      <pathandfilename4>
TRNSOL     <pathandfilename3>      <pathandfilename4>      <pathandfilename5>
DIMENS      dimens
SAMFRQ      samfrq
EXTFOL      flag_ext
SOLVER      solver_type
TOLERA      tolera
USE_MNB     flag_mnb
USE_CMF      flag_cmf
USE_IC0      flag_ico
USE_STR      flag_str
POSCFG

    x_scale_factor1    y_scale_factor1    z_scale_factor1
    [<pathandfilename1_1>    <pathandfilename1_2>    ...    <pathandfilename1_M>]
    x_scale_factor2    y_scale_factor2    z_scale_factor2
    [<pathandfilename2_1>    <pathandfilename2_2>    ...    <pathandfilename2_M>]
.

.

    x_scale_factorN    y_scale_factorN    z_scale_factorN
    [<pathandfilenameN_1>    <pathandfilenameN_2>    ...    <pathandfilenameN_M>]

NDSCFG
    nodalcoordinates1
    [<pathandfilename1_1>    <pathandfilename1_2>    ...    <pathandfilename1_M>]
    nodalcoordinates2
    [<pathandfilename2_1>    <pathandfilename2_2>    ...    <pathandfilename2_M>]
.

.

    nodalcoordinatesN
    [<pathandfilenameN_1>    <pathandfilenameN_2>    ...    <pathandfilenameN_M>]

```

PODR0B	Sub-command keyword to specify a path and filename for the binary file containing a local or global basis (see ROBC) V (characters).
<pathandfilename1>	Path and filename of the binary file containing the (primal) basis V (characters).
dimrb	Specifies the first <code>dimrb</code> columns of the basis stored in <code><pathandfilename1></code> as the ROB for training (integer). If the basis is not a <i>local</i> one (see ROBC), this parameter should be ignored and the dimension of the training ROB should be specified instead in <code>dimens</code> (see below).
DUALBASIS	Sub-command keyword for activating reading and using a precomputed dual basis W , in order to precompute the reduced constraint matrix $W^T C V$ for a class of constrained problems identified above, using the methods and algorithms also identified above, and outputting this reduced matrix in the reduced mesh file outputted itself in SAMPLMSH (see OUTPUT) (characters).
<pathandfilename2>	Name of the binary file (including path, if needed) containing the dual ROB W to be read (characters). This file is relevant only for the solution of the class of constrained problems identified above, using the methods and algorithms also identified above. It is generated by the command ROBC and is readable by this command. Note that currently, this ROB cannot be a local dual ROB, but only a global dual ROB.
dimdb	Specifies the first <code>dimdb</code> columns of the dual basis stored in <code><pathandfilename2></code> as the ROB for training (integer).
TRNSOL	Sub-command keyword to specify a path and filename for the binary file containing the training solutions needed for building the training force vectors (characters).
<pathandfilename3>	Path and filename of the binary file containing solution snapshots needed for building training force vectors (characters).
<pathandfilename4>	Path and filename of the binary file containing velocity solution snapshots for building training force vectors (characters). The corresponding option is recommended when the finite element model has rotational degrees of freedom, regardless of whether the time-integrator is explicit or implicit.
<pathandfilename5>	Path and filename of the binary file containing acceleration solution snapshots for building training force vectors (characters). The corresponding option is recommended when the finite element model has rotational degrees of

	freedom, and the time-integrator is implicit.
DIMENS	Sub-command keyword to specify the dimension of the basis stored in <pathandfilename>, when this basis is not a local one (see ROBC) (characters).
dimens	Specifies the first <code>dimens</code> columns of the non-local basis stored in <pathandfilename> as the ROB for training (integer).
SAMFRQ	Sub-command keyword to specify a sampling frequency for the inputted solution snapshots (characters).
samfrq	Specifies using every <code>samfrq</code> -th solution snapshot for building the training force vectors (integer).
EXTFOL	By default, the training force vectors are the internal force vectors. However, this sub-command keyword can be used to define the training force vectors as the union of the internal and external follower (see PRESSURE) force vectors.
flag_ext	
On	In this case, the training force vectors are defined as the union of the internal and external follower (see PRESSURE) force vectors.
Off	In this case, the training force vectors are defined as the internal force vectors only.
SOLVER	Sub-command keyword to specify the method for computing the reduced mesh and its elements weights (characters).
solver_type	Specifies the method for computing the reduced mesh and its elements weights (characters). Five choices are available:
nnlsqr	Specifies the Lawson and Hanson Non-Negative Least-Squares (NNLS) method based on the QR factorization algorithm. If a mesh partition is also specified (see DECOMPOSE), this method is executed in parallel using this mesh partition and the <code>ScalAPACK</code> and <code>MPI</code> libraries. This is the default method for computing the reduced mesh and its elements weights.
nnlscg	Specifies the Lawson and Hanson Non-Negative Least-Squares (NNLS) method based on the CG algorithm. If a mesh partition is also specified (see DECOMPOSE), this method is executed in parallel using this mesh partition and the <code>MPI</code> library. This method is also known in the literature as the Non-Negative Conjugate Gradient Pursuit method.
pfpq	Specifies the polytope faces pursuit method based on the QR factorization algorithm. If a mesh partition is also specified (see DECOMPOSE), this method is executed in parallel using this mesh partition and the <code>ScalAPACK</code> and <code>MPI</code> libraries.
pfpcg	Specifies the polytope faces pursuit method based on the CG algorithm. If a mesh partition is also specified (see DECOMPOSE), this method is executed in parallel using this mesh partition and the <code>MPI</code> library.
lassocg	Specifies the Least Absolute Shrinkage and Selection Operator (LASSO) method based on the CG algorithm. If a mesh partition is also specified (see DECOMPOSE), this method is executed in parallel using this mesh partition and the <code>MPI</code> library.
TOLERA	Sub-command keyword to specify a tolerance for controlling the error in energy conservation due to mesh sampling (characters).
tolera	Tolerance for controlling the error in energy conservation due to mesh sampling. For example, <code>tolera = 0.01</code> means that 99 % of the energy associated with the training force vectors and the ROB is conserved on the reduced mesh (real).
USE_MNB	Sub-command keyword to specify that the inputted ROB is mass-orthonormalized (characters).
flag_mnb	Flag for specifying which available ROB instance to select for the assigned task (characters).
On	In this case, this command selects the mass-orthonormalized ROB instance for the assigned task (characters).
Off	In this case, this command selects the orthonormalized ROB instance for the assigned task (characters).
USE_CMF	Sub-command keyword to specify whether to include or not the reduced inertia forces associated with a constant mass matrix in the training of ECSW (characters).
flag_cmf	Flag for specifying whether to include or not the reduced inertia forces associated with a constant mass matrix in the training of ECSW (characters).
On	In this case, the reduced inertia forces associated with a constant mass matrix are included in the ECSW training procedure, which implies the intent to hyperreduce the reduced-order mass matrix. Consequently, in this case, the reduced-order mass matrix is neither computed within the scope of this command nor stored in the file <PATHANDFILENAME>.elementmesh.inc if SAMPLMSH is specified as an output result under OUTPUT .
Off	In this case, the reduced inertia forces associated with a constant mass matrix are excluded from the ECSW training procedure, which implies the intent to pre-compute rather than hyperreduce the reduced-order mass matrix. Consequently and unless the ROB is mass-orthonormal, the reduced-order mass matrix is pre-computed in this case within the scope of this command and stored in the file <PATHANDFILENAME>.elementmesh.inc if SAMPLMSH is specified as an output result under OUTPUT .
USE_ICO	Warning: currently, AERO-S does not support a mechanism for suppressing the output of the reduced-order mass matrix when <code>flag_cmf</code> is set to off and the ROB is not mass-orthonormal.
flag_ico	Sub-command keyword to specify whether to hyper reduce or not any high-dimensional initial displacement and/or velocity condition(s) specified in the same ASCII Input Command Data file as this command - and therefore whether to include or not this/these initial condition(s) in the training of ECSW (characters).
On	In this case, the aforementioned initial condition(s) are included in the ECSW training procedure.
Off	In this case, the aforementioned initial condition(s) are not included in the ECSW training procedure.
USE_STR	This sub-command keyword is relevant only if <code>flag_cmf</code> and/or <code>flag_ico</code> are/is set to on. It should be used to specify how to train ECSW for the reduced inertia forces associated with a constant mass matrix and/or any specified high-dimensional initial conditions (characters).
flag_str	Flag for setting how to train ECSW for the reduced inertia forces associated with a constant mass matrix and/or any specified high-dimensional initial conditions (characters).
On	In this case, the contributions to the ECSW training procedure of the reduced inertia forces associated with a constant mass matrix and/or any specified high-dimensional initial conditions are stacked with those of all other reduced forces to be hyper reduced, in order to achieve accuracy for each individual hyper reduction. This setting is strongly recommended for EIGEN and DYNAMICS analyses in the context of the nonparametric probabilistic method for uncertainty quantification, because: in the first case, the reduced mass and stiffness matrices must be accurately hyper reduced; and in the second case, any specified non zero high-dimensional initial condition(s) must be accurately hyper reduced.

Off	In this case, the contributions to the ECSW training procedure of the reduced inertia forces associated with a constant mass matrix and/or any specified high-dimensional initial conditions are added to the contributions of all other reduced forces to be hyper reduced, in order to achieve accuracy for the hyper reduction of the balance of all aforementioned terms. This setting is recommended for transient DYNAMICS analysis, when no high-dimensional initial condition requires hyper reduction.
POSCFG	Sub-command keyword to specify that the collected training snapshots were computed on different meshes that share however the same connectivity — that is, on meshes that have different position configurations — and whose nodal coordinates differ only by scaling factors in the <i>x</i> , <i>y</i> , and <i>z</i> directions. These scaling factors are defined with respect to the nodal coordinates specified under the NODES command of this ASCII Input Command Data file. Note that even if a single common ROB is assigned to all specified different meshes, using the POSCFG sub-command keyword requires that this ROB go first through a clustering process using the ROBC command configured with the DO_CLUSTERING sub-command keyword and <i>nclusters</i> = 1, in order for AERO-S to build the association between a snapshot and the mesh on which it was computed.
x_scale_factor <i>i</i>	<i>x</i> -coordinate scale factor for the <i>i</i> -th mesh and the case where the collected training snapshots were computed on different meshes (real).
y_scale_factor <i>j</i>	<i>y</i> -coordinate scale factor for the <i>j</i> -th mesh and the case where the collected training snapshots were computed on different meshes (real).
z_scale_factor <i>k</i>	<i>z</i> -coordinate scale factor for the <i>k</i> -th mesh and the case where the collected training snapshots were computed on different meshes (real).
<pathandfilename <i>i_j</i> >	Name of the <i>optional</i> binary file (including path, if needed) containing (characters): <ul style="list-style-type: none"> • The <i>j</i>-th mass-orthonormalized local ROB computed on the <i>i</i>-th mesh, when model reduction is to be based on local ROBs. • The global mass-orthonormalized ROB computed on the <i>i</i>-th mesh, otherwise. Note that this optional entry should be omitted (for the sake of computational efficiency) when the training on different meshes is to be performed using orthonormal ROBs. Note also that the aforementioned ROBs can be efficiently constructed using the COMPRESS sub-command of the ROBC command (instead of performing SVD multiple times using different mass metrics).
NDSCFG	Sub-command keyword to specify that the collected training snapshots were computed on different meshes that share however the same connectivity — that is, on meshes that have different position configurations — and whose nodal coordinates differ by an arbitrary manner. Note that even if a single common ROB is assigned to all specified different meshes, using the NDSCFG sub-command keyword requires that this ROB go first through a clustering process using the ROBC command configured with the DO_CLUSTERING sub-command keyword and <i>nclusters</i> = 1, in order for AERO-S to build the association between a snapshot and the mesh on which it was computed.
nodalcoordinates <i>i</i>	Name of the binary file (including path, if needed) containing the nodal coordinates of the <i>i</i> -th mesh.

Next: [CONTROL](#), Previous: [RMSHC](#)

28 CONTACT SURFACES

Command Statement:	CONTACTSURFACES
--------------------	------------------------

The **CONTACTSURFACES** command can be used to enforce contact laws between pairs of surfaces defined using the command [SURFACETOPO](#), for static and dynamic analyses only. Surface interactions are detected using the search module of the library **ACME**. For explicit computations, the discrete kinematic constraint equations are defined and enforced as specified in the sub-command keyword [TDENFORCE](#) and its associated flag [FLAG_TDENFORCE](#) of the [DYNAMICS](#) object. For implicit computations, the discrete kinematic constraint equations are formulated using **AERO-S**'s mortar method and enforced using the method specified in [CONSTRAINTS](#) or in [CONSTRAINT_METHOD](#) below with the following limitation: in this case, the enforcement method can be only one of the following three methods: the penalty method, the Lagrange multiplier method, or the augmented Lagrange multiplier method.

Note 1: In general, the master surface should be chosen as that with the coarser discretization. Setting the master and slave contact surfaces to the same surface activates the numerical treatment of self-contact.

Note 2: For explicit dynamic computations performed with the flag [TDENFORCE](#) set to *On* (see [DYNAMICS](#)), the contact forces can be computed using one of the following four enforcement models: frictionless (default), constant friction, velocity-dependent friction, and pressure-dependent friction. For all other computations (for example, for explicit dynamic computations performed with the flag [TDENFORCE](#) set to *Off*, implicit dynamic or static computations) only the frictionless model is supported.

Note 3: The enforcement of contact surface constraints by the Lagrange multiplier method in static and implicit dynamic analyses is supported only by the FETI-DP family of solvers.

Note 4: The enforcement of contact surface constraints by the elimination method is not supported.

Note 5: For contact detection and enforcement, a surface is treated as 2-sided (or "shell" surface) when both of the following conditions are met: (a) the simulation is explicit dynamics with flag [TDENFORCE](#) set to *On* in [DYNAMICS](#) (or omitted since this is the default setting), and (b) a surface thickness attribute is specified under [SURFACETOPO](#). In all other cases, the contact detection and enforcement is 1-sided. For a 1-sided surface, the directions of the normals to the faces of the surface are important. Contact interactions between two 1-sided surfaces, or self-contact involving one 1-sided surface, can only be detected between two faces with normals in opposite directions. Furthermore, if an interaction is detected, the directions of the normals establish whether the configuration involves penetration or separation. For a 2-sided surface, the directions of the normals of the faces of the surface are not important. Contact interactions between two 2-sided surfaces, or self-contact involving one 2-sided surface, can be detected between two faces regardless of the directions of the normals. For contact between one 2-sided surface and one 1-sided surface, only the normal direction of the 1-sided surface is important. If an interaction is detected, the direction of the normal of the 1-sided surface establishes whether the configuration involves penetration or separation.

CONTACTSURFACES

For a static or implicit dynamic computation, or for an explicit dynamic computation with the flag `flagTDFENFORCE` set to off (see [DYNAMICS](#))

SURF_PAIR_ID#	MASTER	SLAVE	MORTAR_TYPE	NORMAL_TOL	TANGENTIAL_TOL	CONSTRAINT_METHOD
---------------	--------	-------	-------------	------------	----------------	-------------------

For an explicit dynamic computation with the flag `flagTDFENFORCE` set to on (see [DYNAMICS](#)) and a frictionless model

SURF_PAIR_ID#	MASTER	SLAVE	KPART_TYPE	NORMAL_TOL	TANGENTIAL_TOL	NUM_ITER	CONVERG_TOL	CONSTRAINT_METHOD
---------------	--------	-------	------------	------------	----------------	----------	-------------	-------------------

For an explicit dynamic computation with the flag `flagTDFENFORCE` set to on (see [DYNAMICS](#)) and a constant friction model

SURF_PAIR_ID#	MASTER	SLAVE	KPART_TYPE	NORMAL_TOL	TANGENTIAL_TOL	NUM_ITER	CONVERG_TOL	FRIC_COEF	CONSTRAINT_METHOD
---------------	--------	-------	------------	------------	----------------	----------	-------------	-----------	-------------------

For an explicit dynamic computation with the flag `flagTDFENFORCE` set to on (see [DYNAMICS](#)) and a velocity dependent friction contact model

SURF_PAIR_ID#	MASTER	SLAVE	KPART_TYPE	NORMAL_TOL	TANGENTIAL_TOL	NUM_ITER	CONVERG_TOL	STATIC_COEF	DYNAMIC_COEF
VELOCITY_DECAY									

For an explicit dynamic computation with the flag `flagTDFENFORCE` set to on (see [DYNAMICS](#)) and a pressure-dependent friction contact model

SURF_PAIR_ID#	MASTER	SLAVE	KPART_TYPE	NORMAL_TOL	TANGENTIAL_TOL	NUM_ITER	CONVERG_TOL	FRIC_COEF	REF_PRES	OFFSET_PRES
PRES_EXP										

SURF_PAIR_ID# Id number of the surface pair to be described (integer).
MASTER Identification of the master (mortar method) surface (see [SURFACETOPO](#)) (integer).
SLAVE Identification of the slave (mortar method) surface (see [SURFACETOPO](#)) (integer).
CONSTRAINT_METHOD Method for enforcing the associated constraints (characters). The default method is set in [CONSTRAINTS](#) and used whenever this entry is omitted.
multipliers
penalty **beta** The Lagrange multiplier method.
The penalty method. The parameter `beta` should be a large positive number, typically of the order of 10^8 (no default value is provided).
augmented **beta** The augmented Lagrangian method. The parameter `beta` should be a large positive number, typically of the order of 10^8 (no default value is provided). If penetration occurs, the value of `beta` should be increased and the value of the computational time-step specified in [DYNAMICS](#) decreased.
MORTAR_TYPE Mortar type: 0 = standard, 1 = dual, default value is 0 (integer).
NORMAL_TOL Normal search tolerance used by ACME to identify interactions. The default value is 0.1 (float) (see Figs. 1.2 and 1.3 in Section 1.3 of ACME's User Reference Manual). This tolerance should be larger than the maximum relative displacement occurred during one load step of a static simulation or one time-step of a dynamic one. If penetration occurs, the value of this parameter should be increased. For a surface with a surface thickness value, the normal tolerance is defined w.r.t. the position/s of the "lofted" surface/s. For a 1-sided surface, this is the position of the surface + 0.5*surface thickness in the direction of the normal. For a 2-sided surface, these are the positions of the surface +/- 0.5*surface thickness. The normal tolerance must be greater than the distance that a point on the contact surface can move in one time/load step. Therefore, a larger computational time/load step calls for a larger normal tolerance. For contact between two 2-sided surfaces, the normal tolerance must be less than the sum of the two surface thickness values. Also, for self-contact of a 2-sided surface, the normal tolerance must be less than twice the surface thickness value. Even for contact involving one or more 1-sided surfaces, there may be certain cases where it would be appropriate to limit the normal tolerance in order to prevent particular configurations from being treated as penetration. Increasing the normal tolerance will make the contact detection more robust, provided the aforementioned upper limits are respected. However, it also makes the simulation more expensive.
TANGENTIAL_TOL Tangential search tolerance used by ACME to identify interactions. The default value is 0.001 (float) (see Figs. 1.2 and 1.3 in Section 1.3 of ACME's User Reference Manual). If penetration occurs, this tolerance should be increased, in which case the simulation should be expected to become slower. The value of this parameter should be neither too small nor too large. A value corresponding to a small percentage (between 0.5% and 50%) of the minimum edge length of the face elements is suggested.

KPART_TYPE In ACME, kinematic partitioning pertains to determining the fraction $0 \leq f \leq 1$ of the total momentum to be absorbed by each surface of a pair of surfaces in contact. If surface 1 is in contact with surface 2 and the kinematic partitioning fraction for surface 1 is f_1 , it is $f_2 = 1 - f_1$ for surface 2. One specific value of the kinematic partitioning factor f can be inputted to ACME (see below), or this factor can be

automatically computed by ACME for each contact interaction as follows

$$f_1 = \rho_1 c_1 / (\rho_1 c_1 + \rho_2 c_2), \quad f_2 = \rho_2 c_2 / (\rho_1 c_1 + \rho_2 c_2)$$

where f_i , $i = 1, 2$, is the kinematic partitioning fraction for surface i , ρ_i is the density at a given node of this surface, and c_i is the speed of sound in the material of which this surface is made. Setting

KPART_TYPE to 0 specifies $f = 1$ for the slave surface (see SLAVE) and $f = 0$ for the master surface (see MASTER). Setting KPART_TYPE to 1 specifies that f_1 and f_2 are to be automatically computed by ACME as outlined above. The default value of KPART_TYPE is 0 (integer).

NUM_ITER	Maximum number of predictor-corrector iterations to be performed at each time step. The default value is 5 (integer).
CONVERG_TOL	Convergence tolerance of the predictor-corrector iteration loop. The default value is 1.0e-10 (float).
FRIC_COEF	Friction coefficient (float).
STATIC_COEF	Static friction coefficient (float).
DYNAMIC_COEF	Dynamic friction coefficient (float).
VELOCITY_DECAY	Velocity decay parameter (float).
REF_PRES	Reference pressure (float).
OFFSET_PRES	Offset pressure (float).
PRES_EXP	Pressure exponent (float).

Next: [CONWEP](#), Previous: [CONTACTSURFACES](#)

29 CONTROL STATEMENT

Command Statement:	CONTROL
--------------------	----------------

AERO-S always generates a performance file named `FNAME.timing` which reports on the complexity, memory, and CPU resources associated with the performed computation. By default, the prefix `FNAME` is set to the name of the **AERO-S** input file (**AERO-S** also outputs some of these performance results on the screen). This optional command statement can be used to reset the value of the prefix `FNAME`. This prefix is also used by **AERO-S**, when executed with the “-t” (or “-T”) option, to name the ASCII output file (`FNAME.top`) containing the geometry of the performed computation in the **XPost** format. It is also used by **AERO-S** for outputting, when requested, a number of domain decomposition (or mesh partitioning) files (see [DECOMPOSE](#)).

Currently, this command is also needed for specifying the type of analysis to be performed by **AERO-S**. For this reason, it features the entry `ANATYPE` described below. For parsing reasons, `ANATYPE` must be specified whenever the `CONTROL` command is included in the input file, even this entry is not needed.

CONTROL
FNAME
ANATYPE
NODESET
ELEMSET

FNAME	Prefix of the names of the files reporting on the performance, geometry, and mesh decomposition of the performed computation, when applicable. The default value, which is applicable when this optional command is entirely skipped, is the name of the AERO-S input file.
ANATYPE	Analysis type identifier (integer). It is needed for <i>any</i> thermal (heat conduction) analysis involving <i>radiation</i> , in which case it must be set to 2. Otherwise, this parameter should be set to 1.
NODESET	Name of the node set describing the grid points of the geometry of the computation in the file <code>FNAME.top</code> outputted in the XPost format, when AERO-S is executed with the “-t” (or “-T”) option. The default name is “nodes”.
ELEMSET	Name of the element set describing the connectivity of the geometry of the computation in the file <code>FNAME.top</code> outputted in the XPost format, when AERO-S is executed with the “-t” (or “-T”) option. The default name is “elems”.

Next: [DCTT](#), Previous: [CONTROL](#)

30 CONWEP

Command Statement:	CONWEP
--------------------	---------------

The `CONWEP` command statement is used to define, in conjunction with the `PRESSURE` command, a pressure load due to air blast in the free field using the software module `CONWEP`. More specifically, the `CONWEP` command is used to specify the position of the charge, its mass, and the detonation time. The `PRESSURE` command is used to specify those elements of the finite element model on which to apply the generated pressure load.

The input format of this command is given below.

CONWEP

XCHARGE	YCHARGE	ZCHARGE	MCHARGE	DTIME
---------	---------	---------	---------	-------

XCHARGE x-ordinate of the charge (real).
 YCHARGE y-ordinate of the charge (real).
 ZCHARGE z-ordinate of the charge (real).
 MCHARGE Mass of the explosive charge (real).
 DTIME Time at which the charge is detonated (real).

Next: [DIMASS](#), Previous: [CONWEP](#)

31 DENSITY-TEMPERATURE TABLE

Command Statement:	DCTT
--------------------	-------------

The **DCTT** command can be used to describe, for a given material, the evolution of the mass density with temperature. This evolution can be specified here in a curve (or one-dimensional table) defined by pairs of density and temperature values. Linear interpolation is used for "in between" points, and the extrema values are adopted for "outside" points. Several curves can be specified, one after the other. Each curve is identified by an ID number as described below.

Note 1: Currently, this command is supported only by the 4-noded tetrahedral termal element (type 50).

DCTT

CURVE curve_id
T_1 ρ_1
—
.
.
.
T_n ρ_n
—

CURVE Sub-command keyword (characters) that signals the definition of a density-temperature curve (or one-dimensional table).
 curve_id ID number for the following curve (or one-dimensional table) (integer).
 T_i A specified temperature value (real).
 ρ_i A specified mass density value at temperature T_i (real).

Next: [DYNAMICS](#), Previous: [DCTT](#)

32 DISCRETE NODAL MASS AND INERTIA

Command Statement:	DIMASS
--------------------	---------------

The command **DIMASS** is used to lump a discrete mass or inertia on a specified degree of freedom attached to a specified node, either for a diagonal contribution to the mass matrix or an off-diagonal one. If a node number appears more than once under this command with the same degree of freedom (or pair of degrees of freedom), **AERO-S** sums all the lumped masses at the implied entries of the mass matrix. If a gravity field is also specified in the input file using the command [GRAVITY](#), **AERO-S** generates at each specified translational degree of freedom at each specified node under this command a weight-force component equal to the product of the specified discrete mass and specified gravity acceleration in the corresponding direction.

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)), using the usual local numbering convention. By default, the nodal degree of freedom reference frames are the same as the global reference frame.

The input format is given below.

DIMASS

NODE#	DOF#	VALUE
-------	------	-------

or

NODE#	DOF1#	DOF2#	VALUE
-------	-------	-------	-------

NODE#	Node number where the mass or inertia will be added (integer).		
DOF#	Degree of freedom for which the mass or inertia will contribute a diagonal entry to the mass matrix (integer).		
VALUE	Value of the prescribed discrete mass or inertia (float).		
DOF1#	Degree of freedom for which the specified mass or inertia will contribute an off-diagonal entry (and its symmetric counterpart) to the mass matrix at the location implied by this degree of freedom and that specified in <code>DOF2#</code> (integer). Note that only the contribution to the lower triangular part of the mass matrix should be specified, and therefore <code>DOF1#</code> should be greater than <code>DOF2#</code> . Note also that for an explicit time-integration scheme or when the mass is requested to be LUMPED , this contribution will be part of the lumping process.		
DOF2#	Degree of freedom for which the mass or inertia will contribute an off-diagonal entry (and its symmetric counterpart) to the mass matrix at the location implied by this degree of freedom and that specified in <code>DOF1#</code> (integer). Note that only the contribution to the lower triangular part of the mass matrix should be specified, and therefore <code>DOF2#</code> should be smaller than <code>DOF1#</code> . Note also that for an explicit time-integration scheme or when the mass is requested to be LUMPED , this contribution will be part of the lumping process.		

If more than one line addressing the same degree of freedom of the same node are encountered under this command, the effect is the accumulation of the lumped masses or inertias at this degree of freedom.

Next: [EIGEN](#), Previous: [DIMASS](#)

33 DYNAMIC ANALYSIS

Command Statement:	DYNAMICS
--------------------	-----------------

The **DYNAMICS** command can be used to:

- Specify a damping model for a structural dynamics system and its parameters.
- Specify a time-integrator for a desired dynamic (transient, time-domain) structural, acoustic, or thermal (heat transfer) analysis and relevant parameters.
- Specify a linear or nonlinear (see also [FLOATINGFRAME](#)) modal superposition analysis method for a desired dynamic (transient, time-domain) structural or thermal (heat transfer) analysis and relevant parameters.
- Specify a computational approach for enforcing a class of constraints in explicit dynamic computations.
- Specify some parameters for managing dynamic fluid-structure interaction computations using the tandem **AERO-F-AERO-S**.

Currently, the following time-integrators are available in **AERO-S**:

1. The implicit generalized α method and the explicit central difference method for linear and nonlinear (in this case, the [NONLINEAR](#) command must also be present in the ASCII Input Command Data file) structural dynamic analyses.
2. The implicit generalized α method, the explicit central difference method, and the explicit modified wave equation algorithm for linear, time-domain acoustic analysis.
3. The generalized midpoint family of methods for linear thermal analysis.
4. The implicit midpoint rule for nonlinear thermal analysis.

Linear and Nonlinear Structural Dynamic Analyses and Linear Time-Domain Acoustic Analyses

- The implicit generalized α method for linear and nonlinear structural dynamic and linear time-domain acoustic analyses enforces the following equation of equilibrium

$$M\ddot{u}^{n+1-\alpha_f} + D\dot{u}^{n+1-\alpha_f} + F_{internal}(u^{n+1-\alpha_f}) = F_{external}^{n+1-\alpha_f}$$

where

$$u^{n+1-\alpha_f} = (1 - \alpha_f)u^{n+1} + \alpha_f u^n$$

$$\dot{u}^{n+1-\alpha_f} = (1 - \alpha_f)\dot{u}^{n+1} + \alpha_f \dot{u}^n$$

$$\ddot{u}^{n+1-\alpha_f} = (1 - \alpha_f)\ddot{u}^{n+1} + \alpha_f \ddot{u}^n$$

$$F_{external}^{n+1-\alpha_f} = F_{external}(t^{n+1-\alpha_f})$$

$$t^{n+1-\alpha_f} = (1 - \alpha_f)t^{n+1} + \alpha_f t^n$$

and for linear problems

$$F_{internal}(u^{n+1-\alpha_f}) = Ku^{n+1-\alpha_f}$$

Hence, the above equation of equilibrium can also be written as

$$M((1 - \alpha_m)\ddot{u}^{n+1} + \alpha_m\ddot{u}^n) + D((1 - \alpha_f)\dot{u}^{n+1} + \alpha_f\dot{u}^n) + F_{internal}((1 - \alpha_f)u^{n+1} + \alpha_f u^n) = F_{external}^{n+1 - \alpha_f}$$

Given $F_{external}(t)$, u^0 and \dot{u}^0 , the implicit generalized \square method solves the above equation as follows

$$\begin{aligned}\ddot{u}^0 &= M^{-1}(F_{external}^0 - F_{internal}(u^0) - Du^0) \\ u^{n+1} &= u^n + \Delta t \dot{u}^n + \Delta t^2 \left(\left(\frac{1}{2} - \beta\right) \ddot{u}^n + \beta \ddot{u}^{n+1} \right) \\ \dot{u}^{n+1} &= \dot{u}^n + \Delta t ((1 - \gamma) \ddot{u}^n + \gamma \ddot{u}^{n+1})\end{aligned}$$

where $\beta \neq 0$. The Newmark method is obtained by setting $\alpha_m = \alpha_f = 0$. The midpoint rule (also known as the trapezoidal rule for linear systems) is obtained by setting $\alpha_m = \alpha_f = 1/2$, $\beta = 1/4$ and $\gamma = 1/2$. The *classical* central difference method is obtained by setting $\alpha_m = \alpha_f = 0$, $\beta = 0$ and $\gamma = 1/2$.

The Newmark method ($\alpha_m = \alpha_f = 0$) is second-order time-accurate if and only if $\gamma = 1/2$. For undamped linear systems, it is unconditionally stable if and only if $\gamma \geq 1/2$ and $\beta \geq (\gamma + 1/2)^2/4$.

The generalized \square method is second-order time-accurate if and only if $\gamma = 1/2 - \alpha_m + \alpha_f$. For undamped systems, it is unconditionally stable for linear problems if and only if $\alpha_m \leq \alpha_f \leq 1/2$ and $\beta \geq 1/4 + (\alpha_f - \alpha_m)/2$. It maximizes high-frequency dissipation when

$$\beta = \frac{1}{4}(1 - \alpha_m + \alpha_f)^2, \quad \alpha_f = \frac{\rho_\infty}{\rho_\infty + 1}, \quad \alpha_m = \frac{2\rho_\infty - 1}{\rho_\infty + 1}$$

where ρ_∞ is the user-specified high-frequency dissipation ($0 \leq \rho_\infty \leq 1$).

- The central difference method for linear and nonlinear structural dynamic and time-domain acoustic analyses implemented in **AERO-S** enforces the following equation of equilibrium

$$M\ddot{u}^{n+1} + D\dot{u}^{n+\frac{1}{2}} + F_{internal}(u^{n+1}) = F_{external}^{n+1}$$

where

$$\dot{u}^{n+\frac{1}{2}} = \dot{u}^n + \frac{\Delta t}{2} \ddot{u}^n$$

Given $F_{external}(t)$, u^0 and \dot{u}^0 , it solves the above equation as follows

$$\begin{aligned}\ddot{u}^0 &= M^{-1}(F_{external}^0 - F_{internal}(u^0) - Du^0) \\ u^{n+1} &= u^n + \Delta t \dot{u}^n + \frac{\Delta t^2}{2} \ddot{u}^n \\ \dot{u}^{n+1} &= \dot{u}^n + \frac{\Delta t}{2} (\ddot{u}^n + \ddot{u}^{n+1})\end{aligned}$$

This central difference method for linear and nonlinear structural dynamic and linear time-domain acoustic analyses computes the acceleration as follows

$$\ddot{u}^{n+1} = M^{-1} \left(F_{external}^{n+1} - F_{internal}(u^{n+1}) - D(\dot{u}^n + \frac{\Delta t}{2} \ddot{u}^n) \right)$$

For this reason, **AERO-S** automatically lumps the mass matrix M when this method is selected so that it becomes explicit.

- The explicit modified wave equation method for linear time-domain acoustic analysis implemented in **AERO-S** enforces the following equation of equilibrium

$$M\ddot{u}^{n+1} + Du^{n+1} + \left(K - \frac{\Delta t^2}{q} KM^{-1} K\right) u^{n+1} = F^{n+1}$$

where D is the damping matrix associated with a non-absorbing boundary condition when such a condition is imposed, and q is a real-valued parameter.

Given $F(t)$, u^0 and \dot{u}^0 , it solves the above equation as follows

$$\begin{aligned}\ddot{u}^0 &= M^{-1} \left(F^0 - D\dot{u}^0 - K(u^0 - \frac{\Delta t^2}{q} M^{-1} K u^0) \right) \\ \dot{u}^{\frac{1}{2}} &= \dot{u}^0 + \frac{\Delta t}{2} \ddot{u}^0 \\ \overline{u^{n+1}} &= u^n + \Delta t \dot{u}^{\frac{1}{2}} \\ G^{n+1} &= K \left(I - \frac{\Delta t^2}{q} (M + \frac{\Delta t}{2} D)^{-1} K \right) u^{n+1} \\ \dot{u}^{n+\frac{3}{2}} &= (M + \frac{\Delta t}{2} D)^{-1} \left(M\dot{u}^{n+\frac{1}{2}} + \frac{\Delta t}{2} (2F^{n+1} - 2G^{n+1} - D\dot{u}^{n+\frac{1}{2}}) \right)\end{aligned}$$

For $D = 0$ and $q = 12$, the explicit modified wave equation method is fourth-order time-accurate. For $D = 0$ and $12 \leq q \leq 12.5$, experience reveals that for many applications, this method delivers an almost sixth-order time-accuracy. For $D \neq 0$, the explicit modified wave equation method is second-order time-accurate.

Linear Heat Transfer Analysis

- The generalized midpoint family of methods for linear thermal analysis enforces the equation of equilibrium

$$Q\dot{u}^{n+\frac{1}{2}} + Hu^{n+\frac{1}{2}} = F^{n+\frac{1}{2}}$$

where

$$u^{n+\frac{1}{2}} = \frac{1}{2}(u^{n+1} + u^n)$$

$$\dot{u}^{n+\frac{1}{2}} = \frac{1}{2}(\dot{u}^{n+1} + \dot{u}^n)$$

and solves it as follows

$$\begin{aligned}\dot{u}^0 &= Q^{-1}(F^0 - Hu^0) \\ \dot{u}^{n+\alpha} &= \alpha\dot{u}^{n+1} + (1-\alpha)\dot{u}^n \\ u^{n+1} &= u^n + \Delta t (\alpha\dot{u}^{n+1} + (1-\alpha)\dot{u}^n)\end{aligned}$$

The generalized midpoint family of methods outlined above is unconditionally unstable for $\alpha > 1/2$. Hence, only the choices satisfying $\alpha \leq 1/2$ are allowed. The forward Euler method is obtained for $\alpha = 0$. The backward Euler implicit method is obtained for $\alpha = 1$. The implicit trapezoidal rule is obtained for $\alpha = 1/2$.

Nonlinear Heat Transfer Analysis

- The implicit midpoint rule for nonlinear thermal analysis enforces the following equation of equilibrium

$$Q\dot{u}^{n+\frac{1}{2}} + F_{internal}(u^{n+\frac{1}{2}}) = F_{external}^{n+\frac{1}{2}}$$

where

$$u^{n+\frac{1}{2}} = \frac{1}{2}(u^{n+1} + u^n)$$

$$\dot{u}^{n+\frac{1}{2}} = \frac{1}{2}(\dot{u}^{n+1} + \dot{u}^n)$$

Given $F_{\text{external}}(t)$ and u^0 , it solves the above equation as follows

$$\dot{u}^0 = Q^{-1}(F_{\text{external}}^0 - F_{\text{internal}}(u^0))$$

$$u^{n+1} = u^n + \frac{\Delta t}{2}(\dot{u}^{n+1} + \dot{u}^n)$$

Structural Damping

Note that for the purpose of structural dynamic analysis, structural damping can be represented in **AERO-S** using two different approaches:

- The Rayleigh proportional damping, in which case the finite element damping matrix D takes the form

$$D = aK + bM$$

where a and b are two real scalars that may be specified in this **DYNAMICS** command for the entire structure, and/or in the **MATERIAL** command at the material- and therefore element-level (see the sub-command keyword **DAMPING_TYPE** in **MATERIAL**).

- Modal damping ratios that can be specified in this **DYNAMICS** command. However, as its name suggests, this approach is applicable only to modal dynamic analysis.

Note 1: Explicit time-integration algorithms are currently not supported for finite element models with rigid elements, massless and/or moment-of-inertia-less degrees of freedom, or linear multipoint constraints.

Note 2: Whenever an implicit time-integration algorithm is selected, an equation solver must be specified under the **STATICS** command.

Note 3: For nonlinear thermal analysis (see **NONLINEAR**), the appropriate midpoint rule algorithm is automatically selected by this command.

Note 4: Currently, the reduction of inequality constraints such as those associated with contact problems is supported only for nonlinear implicit structural dynamic computations, and only for those cases where the constraints are linear and enforced using the Lagrange multipliers method.

The input format of this command is given below.

DYNAMICS

IACC	flagIACC						
STABLE	option	coeff	tol	maxitr	freq	smsc	dadt
MODAL							
SROM	rob_id_1	[rob_id_2	...	rob_id_L]	[CONTACT rob_id_L+1]		
TROM							
MECH	β	γ			(midpoint implementation of Newmark — that is, $\alpha_m = \alpha_f = 1/2$)		
MECH	β	γ	α_f	α_m	(generalized α)		
MECH	ρ_∞				(generalized α)		
ACOU	β	γ			(midpoint implementation of Newmark — that is, $\alpha_m = \alpha_f = 1/2$)		
ACOU	β	γ	α_f	α_m	(generalized α)		
ACOU	ρ_∞				(generalized α)		
ACOU	β	γ	q		(modified wave equation method)		
HEAT	α						
TIME	th	tm	tt				
RAYDAMP	a	b					

```

MODDAMP
MODE#      MDV
.
.
.
MODE#      MDV
TDENFORCE flagTDENFORCE [maxItrTDENFORCE tolTDENFORCE    tiTDE    tfTDE]
IAEROF     flagIAEROF

```

IACC	For both linear and nonlinear explicit and implicit dynamic computations, the initial acceleration can be either set to zero, or computed by AERO-S to satisfy equilibrium at $t = 0$. The sub-command keyword IACC enables the user to make this choice (characters).
flagIACC	on/off flag (characters). The default value is On .
On	In this case, the acceleration is automatically initialized by AERO-S to satisfy equilibrium at $t = 0$. This initialization requires the solution of a linear system of equations where the left-hand side operator is the mass matrix. For implicit dynamic computations, the solver specified under STATICS command is used for this purpose.
Off	In this case, the acceleration is automatically initialized by AERO-S to zero.
STABLE	Sub-command keyword for managing the time-step of a linear or nonlinear, structural dynamic or acoustic time-integration using the explicit central difference method (characters).
option	This integer parameter specifies how to manage the computational time-step for an explicit simulation.
0	In this case, the time-step is not managed and the computations are performed using the specified computational time-step tm (see below).
1	In this case, which is also the default case, AERO-S estimates automatically the critical time-step — that is, the maximum stability time-step — and multiplies it by the specified coefficient coeff (see below). In the case of the linear explicit central difference method, AERO-S estimates the critical time-step by computing the largest eigenvalue of the finite element model using the power method. In the case of the nonlinear explicit central difference method, AERO-S estimates the critical time-step by an element-dependent method and updates this estimate every freq time-steps. If the specified computational time-step tm (see below) is larger than the product of the critical time-step and coeff , this product is used instead for the computation.
2	In this case, AERO-S computes automatically the critical time-step as described above for option = 1 . However in this case, the explicit computations are performed using a computational time-step equal to the product of the estimated critical time-step and coeff (see below), regardless of the value of the specified computational time-step tm (see below).
coeff	Real-valued coefficient for managing the computational time-step (see above). The default value is 0.8 .
tol	Error tolerance for computing the highest eigenvalue of the finite element model using the power method (real). The default value is 10^-3 .
maxitr	Maximum number of iterations for computing the highest eigenvalue of the finite element model using the power method (integer). The default value is 100 .
freq	Integer parameter specifying the frequency (every so many time-steps) at which the critical time-step is re-estimated during nonlinear explicit computations (see above). The default value is 1000 .
smsc	Integer-valued parameter that can be used to activate selective mass scaling. The default value is 0 .
0	In this case, no selective mass scaling is performed.
1	In this case, selective mass scaling is activated for nonlinear, explicit dynamic computations. Specifically, if at the start of the simulation the stability time-step of an element is less than tm (see below), the element mass matrix of that element is scaled so that its stability time-step is equal to tm .
dedt	Real-valued parameter that can be used to activate element deletion using a minimum stability time-step criterion. The default value is 0 . If at any time the stability time-step of an element is less than the specified value of this parameter, the element is deleted. This option is enabled only for nonlinear, explicit dynamic computations.
MODAL	Sub-command keyword (characters) for:
	<ul style="list-style-type: none"> • Requesting the construction of a <i>linear</i> structural dynamic ROM based on a Reduced-Order Basis (ROB) of the type eigen — and therefore consisting of mass-orthonormalized eigen vectors — which must be inputted using the first format of the READMODE command. • Performing the requested <i>linear</i> structural dynamic analysis using this structural dynamic ROM — which amounts to performing a linear modal superposition analysis. As in the case of a dynamic analysis using direct time-integration, the initial displacement and velocity conditions can be inputted in a generalized coordinates system, the finite element coordinates system, or both (see IDISPLACEMENTS and IVELOCITIES).
SROM	If tt (see below) is set to zero, AERO-S will exit after constructing the requested structural dynamic ROM.
	Sub-command keyword (characters) for:
	<ul style="list-style-type: none"> • Requesting the construction offline of a structural dynamic ROM based on: <ul style="list-style-type: none"> ◦ A primal ROB which must be inputted using the second format of the READMODE command and identified by rob_id_1 (see below). ◦ And when applicable, a dual ROB which must be inputted using the second format of the READMODE command and identified using rob_id_L+1 (see below). • Requesting the construction offline of a structural dynamic ROM for an unrestrained structural system, in a floating reference frame, based on (see FLOATINGFRAME): <ul style="list-style-type: none"> ◦ The rigid body modes of the system, which are automatically computed by AERO-S. ◦ A (partial) primal ROB associated with the flexible modes of the system, which must be inputted using the second format of the READMODE command and identified using rob_id_1 (see below). ◦ The modal superposition of all aforementioned modes, which is characterized <i>in general</i> by a nonlinear coupling between the rigid and flexible structural modes. • Performing the requested structural dynamic analysis online using the constructed structural dynamic ROM. In

the case of a standard ROM, the initial displacement and velocity conditions can be inputted in a generalized coordinates system, the finite element coordinates system, or both (see [IDISPLACEMENTS](#) and [IVELOCITIES](#)) — that is, as in the case of a dynamic analysis using direct time-integration. In the case of a structural dynamic ROM formulated in a floating reference frame, the initial values of the generalized velocity degrees of freedom (dofs) associated with the rigid body modes should be specified in [FLOATINGFRAME](#) using the [RMIV](#) sub-command keyword and those of the dofs associated with the structural deformation modes should be specified in [IVELOCITIES](#) using the [MODAL](#) sub-command keyword.

- Reconstructing the high-dimensional solution offline using [RODC](#), the ROB — or local ROBs — specified in this command via `rob_id_1` — or `rob_id_1, rob_id_2 ... rob_id_L` — and inputted in [READMODE](#).

If `tt` (see below) is set to zero, **AERO-S** will exit after constructing the requested structural dynamic ROM.

`rob_id_1` Integer number identifying the primal structural ROB inputted using [READMODE](#) (integer). Its default value is 0.

Note that whereas a mass-orthonormalized primal structural ROB can be used for both explicit and implicit dynamic computations using the corresponding SROM, an identity-orthonormalized primal structural ROB can be used only for implicit dynamic computations using the corresponding SROM.

`rob_id_2 ... rob_id_L` Optional suite of integer numbers identifying a suite of primal local ROBs, in which case model reduction based on the method of local ROBs is activated (integers). **Note that whereas a mass-orthonormalized primal structural ROB can be used for both explicit and implicit dynamic computations using the corresponding SROM, an identity-orthonormalized primal structural ROB can be used only for implicit dynamic computations using the corresponding SROM.**

`CONTACT` Optional sub-command keyword specifying that the following entry pertains to a dual ROB needed for the reduction of a contact problem (characters).

`rob_id_L+1` Optional integer number identifying the dual structural ROB inputted using [READMODE](#) (integer). It is needed only for the reduced-order solution of contact problems.

`TROM` Sub-command keyword (characters) for:

- Requesting the construction of a *linear* dynamic thermal (heat transfer) ROM based on an identity-orthogonal POD ROB inputted using [READMODE](#).
- Performing the requested *linear* dynamic thermal analysis using this *linear* thermal ROM.

If `tt` (see below) is set to zero, **AERO-S** will exit after constructing the requested linear dynamic thermal ROM.

`MECH OR ACOU OR HEAT` The sub-command keyword `MECH` can be used to request a structural dynamic analysis (characters). The sub-command keyword `ACOU` can be used to request a time-domain acoustic analysis (characters). The sub-command keyword `HEAT` can be used to request a thermal (heat transfer) analysis (characters).

`β` Newmark coefficient β for a second-order system (real). Set $\beta = 0$ ($\alpha_m = \alpha_f = 0$) and $\gamma = 1/2$ to specify the explicit central difference method.

`γ` Newmark coefficient γ for a second-order system (real).

`α_f` Generalized α_f coefficient α_f for a second-order system (real). To obtain the standard Newmark algorithm, set $\alpha_f = 0$. The default is $\alpha_f = 1/2$.

`α_m` Generalized α_m coefficient α_m for a second-order system (real). To obtain the standard Newmark algorithm, set $\alpha_m = 0$. The default is $\alpha_m = 1/2$.

`ρ_∞` Generalized ρ_∞ infinite frequency spectral radius for a second-order system (real). When this option is used and this coefficient is specified in the range $[0, 1]$, α_f , α_m , β and γ are automatically set to

$$\alpha_f = \frac{\rho_\infty}{\rho_\infty + 1}, \quad \alpha_m = \frac{2\rho_\infty - 1}{\rho_\infty + 1}, \quad \beta = \frac{1}{4}(1 - \alpha_m + \alpha_f)^2, \quad \gamma = \frac{1}{2} - \alpha_m + \alpha_f$$

which results in a second-order unconditionally stable scheme that minimizes low frequency dissipation. The parameter $0 \leq \rho_\infty \leq 1$ can be set to dissipate higher frequencies — which are usually spurious, numerical frequencies — while minimizing the dissipative side effect of the lower frequencies. For $\rho_\infty = 1$, the midpoint rule is recovered. Otherwise, the smaller is the value attributed to ρ_∞ within the range $[0, 1]$, the larger is the set of higher frequencies that are dissipated and the larger is the amount of dissipation that is introduced for these frequencies. *This option is particularly useful for stabilizing dynamic fluid-structure computations in the presence of constraints or rigid elements, as these introduce infinite frequencies in the numerical model. In this case, $\rho_\infty = 0.75$ is a recommended value, but the optimal value is problem dependent.*

`q` For $q = 12$, the explicit modified wave equation method is fourth-order time-accurate. Furthermore, for many applications, experience shows that for $12 \leq q \leq 12.5$, this method behaves almost like a sixth-order time-accurate scheme.

`HEAT` Sub-command keyword which precedes the parameter of the Newmark algorithm for a first-order time-dependent system (characters).

`α` Main parameter of the generalized trapezoidal family of methods for first-order systems (real). This parameter must be chosen within $\alpha \leq 1/2$.

`TIME` Sub-command keyword for specifying time-interval parameters (characters).

th	Integration time-step for thermal analysis (real). This time-step should be put to zero if a structural dynamic or time-domain acoustic analysis is to be performed.
tm	Integration time-step for a structural dynamic or time-domain acoustic analysis (real).
tt	End of time-interval for time-integration (real); beginning of time-interval for time-integration is zero for all types of time-dependent analyses.
RAYDAMP	This sub-command keyword is active only in the context of the option MECH . It is used for specifying <i>global</i> Rayleigh proportional damping coefficients — that is, Rayleigh proportional damping coefficients to be applied to the entire structure (characters). However, these coefficients are treated as default values as they can be overwritten by specifying counterpart material- or element-level coefficients in MATERIAL .
[a]	Rayleigh damping stiffness coefficient (real). In the nonlinear case, this coefficient is assigned at each iteration to the <i>initial</i> linearized stiffness matrix and the product of these two quantities, which remains constant through the nonlinear iterations, constitutes the contribution to the damping term.
[b]	Rayleigh damping mass coefficient (real).
MODDAMP	Sub-command keywords for signaling that the following lines specify modal damping ratios (characters). This option is active only when the sub-commands MECH and MODAL are also specified — that is, for a dynamic analysis of a mechanical system using modal superposition. If a mode is repeated, the modal damping ratio values are added for that mode. If both RAYDAMP and MODDAMP commands are specified, and the keyword MODAL is also specified, then modal damping takes precedence over Rayleigh damping.
MODE#	Mode ID number (integer).
MDV	Modal damping ratio (also known as modal damping factor and modal % of critical damping and usually denoted by [ξ]) for mode MODE# (real).
TDENFORCE	For implicit dynamic computations with TIEDSURFACES and/or CONTACTSURFACES constraints, the discrete kinematic constraint equations are automatically formulated by AERO-S using its mortar method. In this case, the user can specify in CONSTRAINTS , TIEDSURFACES , or CONTACTSURFACES the penalty, Lagrange multiplier, or augmented Lagrange multiplier method for enforcing these constraints.
	For explicit dynamic computations with TIEDSURFACES and/or CONTACTSURFACES constraints however, the discrete kinematic constraint equations can be either defined using a node-to-segment approach and enforced using ACME's enforcement module, or defined using AERO-S 's mortar method and enforced using its penalty method. TDENFORCE is the sub-command keyword which can be used to choose between either of these two options (characters). on/off flag (characters). The default value is on .
flagTDENFORCE	In this case, the discrete kinematic constraint equations are defined using a node-to-segment approach and the constraint forces are computed using ACME's enforcement module.
On	In this case, the discrete kinematic constraint equations are defined using AERO-S 's mortar method and the constraint forces are computed using its penalty method.
Off	
maxItrTDENFORCE	Optional parameter that can be used to specify the maximum number of outer iterations for contact enforcement when performed using the library ACME (integer). The default value is 1.
tolTDENFORCE	Optional parameter that can be used to specify the absolute tolerance for monitoring the convergence of the outer iterations of the contact enforcement performed by the library ACME (real). The default value is 0.
tiTDE	Optional parameter that can be used to specify the time at which contact detection and enforcement is activated (real). The default value is 0. This parameter can be used to speed up execution time when it can be determined a priori that no contact will occur during the first part of a simulation.
tfTDE	Optional parameter that can be used to specify the time at which contact detection and enforcement is deactivated (real). The default value is infinity. This parameter can be used to speed up execution time when it can be determined a priori that no contact will occur during the final part of a simulation.
IAEROF	For linear and nonlinear implicit dynamic computations, when flagACC is set to on and the AERO command is specified in the AERO-S ASCII Input Command Data file to perform a fluid-structure computation using the tandem AERO-F-AERO-S , the user can request to either ignore or account for the contribution of the initial aeroelastic forces in the computation of the initial acceleration of the structural subsystem. The sub-command keyword IAEROF enables the user to make this choice (characters).
flagIAEROF	on/off flag (characters). The default value is off .
On	In this case, <i>SendInitialForce</i> must be set to true in the ASCII Input Command Data File of AERO-F , in which case the aeroelastic forces are accounted for in the computation of the initial acceleration of the structural subsystem.
Off	In this case, <i>SendInitialForce</i> must be set to false (or omitted) in the ASCII Input Command Data File of AERO-F and the contribution of the initial aeroelastic forces is neglected in the computation of the initial acceleration of the structural subsystem.

Next: [TOPOLOGY](#), Previous: [DYNAMICS](#)

34 EIGENVALUE PROBLEMS

Command Statement:	EIGEN
--------------------	--------------

The **EIGEN** command statement is used to request the construction and solution of a generalized eigenvalue problem of the form

$$AX = BX\Omega^2$$

where **A** and **B** are two finite element square matrices, **X** is the rectangular matrix of generalized eigenvectors, and **Ω²** is the diagonal matrix of corresponding eigenvalues. Currently, only the subspace iteration method (the default choice) and the ARPACK package are available in **AERO-S** for this purpose.

The input format of this command is given below. An example input file can be found in FEM.d/fem_examples/Eigen.d.

Note 1: The usage of the **EIGEN** command requires the simultaneous usage of the **STATICS** command to specify an equation solver. This solver must be chosen according to the properties of A and B (most importantly, watch out for the case where A — or in the shifted case $A - \sigma B$ — is singular).

Note 2: When A is singular and no shift is specified:

- If the **GRBM** command is specified in the input file:
 - If the **sparse** or **skyline** equation solver is specified under the **STATICS** command, the set of zero energy modes returned by the **EIGEN** command is the union of the set of rigid body modes computed by **GRBM**, the set of zero energy modes (mechanisms) determined by the chosen equation solver, and any other zero energy mode captured only by the eigen solver.
 - If a different equation solver is specified under the **STATICS** command, the set of zero energy modes returned by the **EIGEN** command is the union of the set of rigid body modes computed by **GRBM** and any other zero energy mode captured only by the eigen solver.
- On the other hand, if only the **TRBM** command is specified in the input file, the set of zero energy modes returned by the **EIGEN** command is the set of zero energy modes computed by the chosen eigen solver.

EIGEN

VERSION	
SHIFT	σ
NSBSPV	nsbspv
NEIGPA	neigpa
TOLEIG	toleig
TOLJAC	toljac
ARPACK	which mode
ARPACK	lbound nshifts
ARPACK	lbound ubound neigps
MAXITR	maxitr

VERSION

explicit

In this case, the A matrix is used explicitly when constructing the reduced generalized eigen problem in the subspace iteration method. Otherwise (default), the reduced generalized eigen problem is constructed by exploiting some mathematical identities to avoid using explicitly the A matrix which has been factored by then. Note that specifying the **explicit** version of the subspace iteration algorithm incurs additional memory storage.

SHIFT

σ

NSBSPV

nsbspv

In the case of the subspace iteration method (default eigensolver), **nsbspv** denotes the number of subspace iteration vectors and should be set to the minimum between twice the number of requested eigen pairs, $[2 \times$

neigpa , and $\text{neigpa} + 8$. If rigid body modes are expected, the previous formula should be increased by the total number of rigid body modes. In the case of ARPACK, **nsbspv** denotes the number of Krylov vectors (per shift, if a shifting strategy is used).

NEIGPA

neigpa

Number of requested eigen pairs. If rigid body modes are found, their number is included in the specified value of **neigpa**.

TOLEIG

toleig

Tolerance for the convergence of the subspace iteration method.

TOLJAC

toljac

Tolerance for the Jacobi Determinant algorithm used in the subspace iteration method.

ARPACK

The presence of this keyword under the **EIGEN** command specifies the usage of the ARPACK eigensolver in the specified mode (see **mode** below). Otherwise, the default choice is the subspace iteration method. The ARPACK choice is required whenever one of the two matrices A or B is indefinite, or one or both of them are singular. Indefinite systems arise: (1) if the analysis involves the **HELMHOLTZ**, **IMPEDANCE**, or **EIGEN** command with a positive shift (see **SHIFT** in **EIGEN**) or for buckling analysis (see ARPACK in **EIGEN**), or (2) the structural model includes rigid and/or joint elements (see **TOPOLOGY**), linear multi-point constraints (see **LMPC**), or tied surfaces (see **TIEDSURFACES**), and the Lagrange multiplier method is chosen for enforcing the associated constraints (see **CONSTRAINTS**). In either case, the equation solver must be properly chosen in **STATICS**.

which

Keyword (characters) to specify which eigenpairs to be computed by ARPACK. This keyword can take one of the following values.

LA

In this case, the **neigpa** eigenpairs whose eigenvalues are just to the right of the shift σ are computed. This is the default value of **which** when the shift is zero ($\sigma = 0$).

SA

In this case, the **neigpa** eigenpairs whose eigenvalues are just to the left of the shift σ are computed.

BE

In this case, **neigpa** eigenpairs with eigenvalues on either side of the shift σ are computed. This is the default value of **which** when the shift is non zero ($\sigma \neq 0$).

mode

Integer identifier to specify the mode in which to run ARPACK. It can take one of the following values.

3	This default mode, which is the shift-invert mode, is recommended for all generalized eigenvalue problems except those for which the matrix B is indefinite. Hence, this mode is particularly not recommended for buckling analysis.
4	This mode is recommended for generalized eigenvalue problems where the matrix B is indefinite. Hence, this mode is particularly recommended for buckling analysis. In this case, a nonzero shift should be specified in SHIFT (see above).
lbound	Lower bound of a set or range of eigenvalues to be computed by ARPACK (real).
nshifts	Number of shifts to be used by ARPACK when computing the neigpa eigenvalues that are greater than lbound (integer). The specific values of the shifts are automatically selected by ARPACK, and neigpa/nshifts eigenpairs are computed per shift.
ubound	Upper bound of a range of eigenvalues to be computed by ARPACK (real).
neigps	When a range of eigenvalues [lbound, ubound] is specified, ARPACK computes all eigenpairs whose eigenvalues lie within this range. In this case, a first shift is set to lbound and neigps consecutive eigenvalues within the range [lbound, ubound] are computed. Then, a recursive procedure in which the largest previously computed eigenvalue is chosen as a new shift and neigps new eigenvalues within the range [lbound, ubound] are computed using this shift is applied until all eigenvalues within the specified range are captured. The default value for neigps is 50.
MAXITR	
maxitr	Maximum number of iterations for the eigensolver. The default value is <input type="text" value="10"/> nsbspv.

Next: [LOCROB](#), Previous: [EIGEN](#)

35 ELEMENT TOPOLOGY *S*

Command Statement:	TOPOLOGY
--------------------	-----------------

The TOPOLOGY command is used to signal that the following data lines correspond to the connectivity and type of each element.

Note 1: Different structural and solid mechanics elements (M = Mechanic) support different material laws (see [MATLAW](#)). In general, all such elements support the standard linear elastic constitutive equation that can be specified in [MATERIAL](#). In addition:

- The three-dimensional solid elements (type 17, 23, 24, 25, 72, 91, 92, and 97) support the pre-defined material laws Linear, HenckyElastic, MooneyRivlin, NeoHookean, Ogden, StVenantKirchhoff, BilinearPlastic, FiniteStrainPlastic, LogStrainPlastic, SimoPlastic, ViscoLinearElastic, ViscoMooneyRivlin, ViscoNeoHookean, and ViscoStVenantKirchhoff (see [MATLAW](#)).
- The shell elements 15 and 1515 support the pre-defined material laws J2Plasticity, LinearPlaneStress, PlaneStressBilinearPlastic, ViscoLinearPlaneStress, ShellFabricMap, ShelfFabricMat, ShellViscoFabricMap, and ShellViscoFabricMat (see [MATLAW](#)).
- The shell element 16 supports the pre-defined material laws HypoElastic, J2Plasticity, KK1, and KK2 (see [MATLAW](#)).
- The membrane elements 128 and 129 support the pre-defined material laws LinearPlaneStress, HyperElasticPlaneStress, ViscoLinearPlaneStress, ViscoHyperElasticPlaneStress, PlaneStressMooneyRivlin, PlaneStressNeoHookean, PlaneStressBilinearPlastic, PlaneStressFiniteStrainPlastic, PlaneStressViscoMooneyRivlin, PlaneStressViscoNeoHookean, FabricMap, FabricMat, ViscoFabricMap, and ViscoFabricMat (see [MATLAW](#)).

Note 2: For flexible beam elements (type 6, 7, or 14), AERO-S supports a run-time generation of local frames that is activated either when the target flexible beam element is identified under the [EFFRAMES](#) command and a third node is specified to generate its local frame, or when a third node is found in the definition of a flexible beam element within *this* command for the same purpose. The only requirement for the third node is that it does not be colinear with the other two nodes defining the flexible beam element and its local *x*-axis. Under the [EFFRAMES](#) command, the third node defines the local *xy* plane. Under *this* command, it defines the local *xz* plane. Using the third

node option under *this* command for a flexible beam element relieves the user from specifying a local frame for this element under the [EFFRAMES](#) command. An example illustrating the third node option in the [TOPOLOGY](#) command can be found in [APPENDIX 3](#).

Note 3: In a linear dynamic analysis, the freeplay-induced forces and moments are interpreted as configuration-dependent and automatically treated as piecewise constant (see [PIECEWISE](#) in [STATICS](#)).

Note 4: Currently, only the SPOOLES and MUMPS sparse direct solvers and the FETI-DP(H) solvers can handle rigid elements.

Note 5: Thermal loads (see [TEMPERATURES](#)) are currently implemented only for the truss, Euler beam, Timoshenko beam, 4-noded plane stress/strain, 4-noded as well as 10-noded tetrahedron, 8-noded as well as 20-noded and 32-noded brick, 6-noded as well as 15-noded and 26-noded penta elements, and all 3- and 4-noded shell elements.

Note 6: The heat radiation elements 56, 57, and 58 are primarily *nonlinear* thermal elements. Hence, the presence of any of these elements in a thermal model usually implies a nonlinear analysis and therefore requires specifying the [NONLINEAR](#) command in the ASCII Input Command Data file. However, in the presence of such elements in the thermal model but absence of the [NONLINEAR](#) command in the aforementioned input file, AERO-S performs a linearized thermal (perturbation) analysis where the conductivity matrix is adjusted by the Jacobian of the finite element treatment of the radiation boundary condition evaluated at an equilibrium temperature field that must be specified in this case using the [ETEMP](#) command. In principle, the equilibrium temperature field is the solution of a related *nonlinear* steady thermal analysis problem with radiation boundary conditions; therefore, it captures the effect of the ambient (reference) temperature of the enclosure receiving the radiation which is specified in the parameter *Tr* of the [MATERIAL](#) command. However, the linearized thermal (perturbation) problem solved by AERO-S in the presence of a heat radiation element in the thermal model but absence of the [NONLINEAR](#) command is not affected by this reference temperature.

Note 7: For DEM (Discontinuous Enrichment Method) and DGM (Discontinuous Galerkin Method) elements, Q, T, and H designate a quadrilateral, a triangle in two dimensions and a tetrahedron in three dimensions, and an hexahedron, respectively. In the notation Q-X-Y, T-X-Y, and H-X-Y, X denotes the number of enrichment functions in the element and Y denotes the number of Lagrange multiplier degrees of freedom per edge or face of the element. In the notation X1xX2-Y for elastodynamics, X1 denotes the number of directions of the plane wave enrichment functions, X2 denotes the number of enrichment functions per wave direction (X2 = 2 (one pressure and one shear wave) in two dimensions and X2 = 3 (one pressure and two shear waves) in three dimensions), and Y denotes the number of Lagrange multiplier

degrees of freedom per edge or face of the element. The connectivity (geometry and local node numbering) of all Q, T, and H DEM and DGM elements is that of Q, T, and H higher-order isoparametric elements, respectively. Hence, for each DEM element, the degree of the polynomial field can be deduced from the number of nodes of that element.

Note 8: The fluid elements (type 301, 302, 311, 312, 321, and 331) are active only in the following cases: (1) a mass computation using the [MASS](#) command, (2) a sloshing eigen computation using the [SLOSH](#) and related commands, (3) a hydroelastic eigenvalue computation using the [EIGEN](#), [HEFRS](#), [HEFSB](#) and related commands.

The input format of this command is given below.

ELEMENT#	ELEMENT_TYPE	CONNECTIVITY_NODES
ELEMENT#	Element identification number whose type and connectivity are to be specified (integer).	
ELEMENT_TYPE:	M=Mechanic - A=Acoustic (Helmholtz or Time-Domain) - H=Heat- C=Coupled Thermoelastic - F=Fluid	
1	M: 3d truss (bar) with 3 dof/node (Figure 1).	
2	M: 2d 4-node plane stress/plane strain quadrilateral with 2 dof/node (Figure 2). Currently, a consistent mass matrix is not available for this element.	
3	H: 3d 4-node quadrilateral with 1 dof/node (Figure 2).	
4	M: 2d 3-node triangle with 2 dof/node (Figure 3). Currently, a consistent mass matrix is not available for this element.	
6	M: 3d 2-node Bernoulli beam with 6 dof/node (Figure 1).	
7	M: 3d 2-node Timoshenko beam with 6 dof/node (Figure 1). Currently, a consistent mass matrix is not available for this element.	
8	M: 3d 3-node triangular AQR shell with 6 dof/node (Figure 3). Currently, a consistent mass matrix is not available for this element.	
88	M: 3d 4-node shell with 6 dof/node that splits into 2 triangular shells of type 8 (Figure 2). Currently, a consistent mass matrix is not available for this element.	
9	H: 3d 2-node line with 1 dof/node (Figure 1).	
10	H: 2d 4-node quadrilateral with 1 dof/node (Figure 2).	
11	M: 3d 1-node (lumped) torsional spring with 3 dof/node for linear analysis (Figure 4).	
12	M: 3d 1-node (lumped) translational spring with 3 dof/node for linear analysis.	
15	M: 3d 3-node triangular AQR shell with 6 dof/node and composite as well as nonlinear material capability (Figure 3). Note that only a lumped mass matrix is available for this element. Hence, this element effectively replaces the elements 8 and 20 even though both are still supported.	
1515	M: 3d 4-node quadrilateral AQR shell with 6 dof/node and composite as well as nonlinear material capability (Figure 2). Note that only a lumped mass matrix is available for this element. Hence, this element effectively replaces the elements 88 and 2020 even though both are still supported.	
16	M: 3d Belytschko-Tsay 4-node quadrilateral or degenerated quadrilateral (Figure 2), or 3-node triangular shell with 6 dof/node and 1-point quadrature rule (Figure 3), and nonlinear material capability. Currently, this element is available only for explicit dynamic analyses.	
17	M: 3d 8-node brick with 3 dof/node (Figure 5).	
18	M: 3d 4-node shear panel with 3 dof/node (Figure 2). Currently, a consistent mass matrix is not available for this element.	
19	M: 3d 3-node triangular membrane with 6 dof/node (Figure 3 ; only in-plane and drilling stiffnesses). Currently, a consistent mass matrix is not available for this element.	
20	M: 3d 3-node triangular composite or orthotropic shell with 6 dof/node (Figure 3). Currently, a consistent mass matrix is not available for this element.	
2020	M: 3d 4-node composite or orthotropic shell with 6 dof/node (Figure 2). Currently, a consistent mass matrix is not available for this element.	
21	M: 3d translational spring-link with 3 dof/node for linear analysis (Figure 1).	
22	M: 3d torsional spring-link with 3 dof/node for linear analysis (Figure 1).	
23	M: 3d 4-node tetrahedron with 3 dof/node (Figure 6) and nonlinear material capability.	
24	M: 3d 6-node pentahedron with 3 dof/node (Figure 7) and nonlinear material capability.	
25	M: 3d 10-node tetrahedron with 3 dof/node (Figure 8) and nonlinear material capability.	
26	M: 3d 2-node damper element with 3 dof/node (Figure 1).	
27	M: 3d 2-node or curved 3-node cable with 3 dof/node (Figure 26). In general, this element is designed for nonlinear analysis (NONLINEAR), in which case its constitutive law is	
	$\sigma(\epsilon) = E(1 - \nu\epsilon)^2\epsilon \quad \text{if } \epsilon > 0$	
	$\sigma(\epsilon) = 0 \quad \text{if } \epsilon \leq 0$	
	However, this element can also be used in a linear analysis, in which case its constitutive law becomes	
	$\sigma(\epsilon) = E\epsilon$	
	Hence, for a linear analysis, this element reduces to a two-node (or curved three-node), bar-like element.	
30	A: 2d 4-node quadrilateral with 1 dof/node (Figure 2).	
31	A: 2d 4-node quadrilateral GLS element with 1 dof/node (Figure 2).	
32*	A: 2d 8-node quadrilateral with 1 dof/node (Figure 9).	
33*	A: 2d 4-node quadrilateral bubble element with 1 dof/node (Figure 2).	

34* A: 2d 4-node quadrilateral two-level bubble element with 1 dof/node ([Figure 2](#)).
 35 A: 2d 3-node triangle with 1 dof/node ([Figure 3](#)).
 36 A: 2d 3-node triangular GLS element with 1 dof/node ([Figure 3](#)).
 38* A: 2d 6-node triangle with 1 dof/node ([Figure 10](#)).
 40 A: 3d 4-node tetrahedron with 1 dof/node ([Figure 6](#)).
 41 A: 3d 4-node tetrahedral GLS element with 1 dof/node ([Figure 6](#)).
 42 A: 3d 10-node tetrahedron with 1 dof/node ([Figure 8](#)).
 44 A: 3d 8-node brick GLS element with 1 dof/node ([Figure 5](#)).
 45* A: 3d 8-node brick with 1 dof/node ([Figure 5](#)).
 46 H: 3d 3-node triangular thermal (heat conduction) element ([Figure 3](#)).
 4646 H: 3d 4-node quadrilateral thermal (heat conduction) element that splits into 2 triangular elements of type 46 ([Figure 2](#)).
 47 H: 3d 2-node line with 1 dof/node for boundary convection ([Figure 1](#)).
 48 H: 3d 4-node quadrilateral with 1 dof/node for boundary convection ([Figure 2](#)).
 49 H: 3d 3-node triangle with 1 dof/node for boundary convection ([Figure 3](#)).
 50 H: 3d 4-node, 10-node, 20-node, or 35-node tetrahedron with 1 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line ([Figure 11](#))**.
 51 H: 3d 8-node brick with 1 dof/node ([Figure 5](#)).
 52* M: 3d 6-node triangular shell ([Figure 10](#)).
 53 H: 2d 3-node triangular thermal (heat conduction) element ([Figure 3](#)).
 56 H: 3d 2-node heat radiation element ([Figure 1](#)).
 57 H: 3d 3-node triangular heat radiation element ([Figure 3](#)).
 58 H: 3d 4-node quadrilateral heat radiation element ([Figure 2](#)).
 59 A: 2d 6-node axisymmetric triangular element with 1 dof/node ([Figure 10](#)).
 60 A: 2d 4-node axisymmetric quadrilateral with 1 dof/node ([Figure 2](#)).
 61 A: 2d 3-node axisymmetric triangle with 1 dof/node ([Figure 3](#)).
 62 A: 2d 8-node axisymmetric quadrilateral with 1 dof/node ([Figure 9](#)).
 65 M: 3d 2-node rigid truss (bar) ([Figure 1](#)); enforces constant length of the element.
 66 M: 3d 2-node rigid beam of *non zero length* ([Figure 1](#)); enforces constant length of the element, equal rotations of the cross sections at its two nodes, and other constraints between its rotational and translational dofs to simulate a genuinely rigid beam.
 67 M: 3d 2-node rigid link (translational and rotational, [Figure 1](#)); for each specified dof at one node, it enforces equality to the corresponding dof at the other node. Therefore, this element enforces constant length like element type 66; however, it also enforces additional constraints and therefore is different from element type 66.
 68 M: 3d 2-node rigid translational link ([Figure 1](#)); for each specified translational dof at one node, it enforces equality to the corresponding dof at the other node. Therefore, this element enforces constant length like element type 65; however, it also enforces additional constraints and therefore is different from element type 65.
 69 M: 3d 2-node rigid rotation link ([Figure 1](#)); for each rotational dof at one node, enforces equality to the corresponding dof at the other node.
 70 M: 3d 8-node rigid brick with 3 dof/node ([Figure 5](#)); enforces constant distance between each pair of its nodes (special case of element type 71).
 71 M: 3d rigid line, plane, or solid element with 3 dof/node and anywhere from 3 to 32 nodes per element ([Figure 12](#)); enforces constant distance between each pair of its nodes.
 72 M: 3d 20-node brick ([Figure 12](#)) with 3 dof/node and nonlinear material capability.
 73 M: 3d 3-node rigid shell ([Figure 3](#)); equivalent to two rigid beam, each defined by an edge of the element (special case of type 74).
 74 M: 3d rigid line, plane, or solid element with 6 dof/node and anywhere from 3 to 32 nodes per element ([Figure 13](#)); enforces constant distance between each pair of its nodes, equal values of the rotational dofs at all nodes, and other constraints to simulate a genuinely rigid element.
 76 M: 3d 4-node rigid shell ([Figure 2](#)); equivalent to three rigid beam, each defined by an edge of the element (special case of type 74).
 77 M: 3d 1-node point-to-point constraint element with 3 dof/node.
 78 M: 3d 1-node point-to-line constraint element with 3 dof/node.
 79 M: 3d 1-node point-to-plane constraint element with 3 dof/node.
 81 H: 2d 4-node contact resistance thermal element with 1 dof/node that can be inserted between two thermal elements of type 10 ([Figure 2](#)).
 82 H: 3d 8-node contact resistance thermal element with 1 dof/node that can be inserted between two thermal elements of type 51 ([Figure 5](#)).
 83 H: 3d 6-node contact resistance thermal element with 1 dof/node that can be inserted between two thermal elements of type 50 with 4 nodes each ([Figure 7](#)).
 84 H: 2d 3-node triangular bulk fluid (thermal) element with 1 dof/node. The first node appearing in the connectivity list of this element must be inside the bulk fluid, and the other two nodes must be located at the interface boundary between the bulk fluid and the surrounding material. The temperature at the first node is by definition the average temperature of the bulk fluid ([Figure 14](#)).
 85 H: 3d 4-node tetrahedron bulk fluid (thermal) element with 1 dof/node. The first node appearing in the connectivity list of this element must be inside the bulk fluid, and the other three nodes must be located at the interface boundary between the bulk fluid and the surrounding material. The temperature at the first node is by definition the average temperature of the bulk fluid ([Figure 15](#)).
 86 H: 3d 5-node pyramidal bulk fluid (thermal) element with 1 dof/node. The first node appearing in the connectivity list of this element must be inside the bulk fluid, and the other four nodes must be located at the interface boundary between the bulk fluid and the surrounding material. The temperature at the first node is by definition the average temperature of the bulk fluid ([Figure 16](#)).

87 M: 3d 4-node quadrilateral membrane with 6 dof/node ([Figure 2](#); only in-plane and drilling stiffnesses). **Currently, a consistent mass matrix is not available for this element.**
 89 A: 3d 6-node wedge with 1 dof/node ([Figure 7](#)).
 90 M: 3d 32-node serendipity brick with 3 dof/node ([Figure 13](#)) and nonlinear material capability.
 91 M: 3d 26-node serendipity wedge with 3 dof/node ([Figure 17](#)) and nonlinear material capability.
 92 A: 3d 32-node serendipity brick with 1 dof/node ([Figure 13](#)).
 93 A: 3d 26-node serendipity wedge with 1 dof/node ([Figure 17](#)).
 94 A: 3d 8-node, 27-node, 64-node, or 125-node hexahedron with 1 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 18](#)). They also support the PML (Perfectly Matching Layer) computational technology.
 95 A: 3d 4-node, 10-node, 20-node, or 35-node tetrahedron with 1 dof/node ([Figure 11](#)). **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** They also support the PML (Perfectly Matching Layer) computational technology.
 96 M: 3d 15-node wedge with 3 dof/node and nonlinear material capability ([Figure 31](#)).
 97 A: 2d 4-node, 9-node, 16-node, or 25-node quadrilateral with 1 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 19](#)). They also support the PML (Perfectly Matching Layer) computational technology.
 98 A: 2d 3-node, 6-node, or 10-node triangle with 1 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 20](#)). They also support the PML (Perfectly Matching Layer) computational technology.
 99 M: 2d 4-node, 9-node, 16-node, or 25-node quadrilateral with 3 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 19](#)).
 100 M: 2d 3-node, 6-node, or 10-node triangular element with 3 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 20](#)).
 101 M: 3d 8-node, 27-node, 64-node, or 125-node hexahedron with 3 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 18](#)).
 102 M: 3d 4-node, 10-node, 20-node, or 35-node tetrahedron with 3 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 11](#)).
 103 A: 3d 8-node, 27-node, 64-node, or 125-node spectral hexahedron with 1 dof/node. **These spectral elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 18](#)).
 104 M: 3d 2-node rigid beam of arbitrary length, including zero length ([Figure 1](#)); enforces constant length of the element, equal rotations of the cross sections at its two nodes, and other constraints between its rotational and translational dofs to simulate a genuinely rigid beam.
 105 A: 2d 4-node, 9-node, 16-node, or 25-node spectral quadrilateral element with 1 dof/node. **These spectral elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 19](#)).
 106 H: 3d 8-node, 27-node, 64-node, or 125-node hexahedron with 1 dof/node. **These "p-type" elements use a special local node-numbering for connectivity which goes line-by-line** ([Figure 18](#)).
 107 M: 3d fabric truss with 3 dof/node ([Figure 1](#)). **Currently, a consistent mass matrix is not available for this element.**
 108 M: 3d 2-node planar joint with 6 dof/node ([Figure 27](#)). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed, as far as input is concerned, as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. In this format, the *x* and *y* axes of the local frame correspond to the *x* and *y* axes of the co-rotating reference frame at node 1 in its undeformed configuration. These axes should be defined so that the position of node 2 is constrained to lie in the plane they define.
 109 M: 3d 2-node welded joint with 6 dof/node ([Figure 30](#)). This element constrains the relative translations and rotations between two nodes.
 110 M: 3d 2-node spherical joint with 3 dof/node. This element constrains the relative translations between two nodes ([Figure 21](#)).
 111 M: 3d 2-node translational joint with 3 rotational dof/node ([Figure 28](#)). This element constrains the relative rotations between two nodes.
 112 M: 3d 2-node universal joint with 6 dof/node ([Figure 22](#)). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed, as far as input is concerned, as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. In this format, the *y* axis of the local frame corresponds to the orientation in the undeformed configuration of the *y* axis of the co-rotating reference frame at node 2, and the *z* axis of the local frame corresponds to the orientation in the undeformed configuration of the *z* axis of the co-rotating reference frame at node 1. These two axes should be defined so that they remain orthogonal during the deformations.
 113 M: 3d 2-node revolute joint with 6 dof/node ([Figure 23](#)). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed, as far as input is concerned, as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. In this format, the *x* axis of the local frame corresponds to the orientation in the undeformed configuration of the axis of free relative rotation.
 114 M: 3d 2-node cylindrical joint with 6 dof/node ([Figure 24](#)). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed, as far as input is concerned, as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. In this format, the *x* axis of the local frame corresponds to the orientation in the undeformed configuration of the axis of free relative displacement and rotation.
 115 M: 3d 2-node prismatic joint with 6 dof/node ([Figure 25](#)). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed, as far as input is concerned, as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. In this format, the *x* axis of the local frame defines the

- orientation in the undeformed configuration of the axis of free relative translation.
- 126 M: 3d 2-node revolute joint-with-driver and 6 dof/node ([Figure 23](#)) and a relative rotation (between node 2 and node 1, defined as rotation of node 2 - rotation of node 1) around the joint axis of rotation that can be prescribed using a time-dependent law specified in [MATERIAL](#) (for static analysis, the relative rotation is set to the initial value of the chosen law). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed, as far as input is concerned, as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. In this format, the x axis of the local frame corresponds to the orientation in the undeformed configuration of the axis of forced relative rotation.
- 127 M: 3d 2-node pin-in-slot joint with 6 dof/node ([Figure 29](#)). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed, as far as input is concerned, as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. In this format, the x -axis of the local frame corresponds to the orientation in the undeformed configuration of the axis of free relative translation, and its y -axis corresponds to the orientation in the undeformed configuration of the axis of free relative rotation.
- 128 M: 3d 4-node plane stress quadrilateral membrane with 3 dof/node ([Figure 2](#)).
- 129 M: 3d 3-node plane stress triangular membrane with 3 dof/node ([Figure 3](#)).
- 131 M: 3d 1-node discrete mass and inertia element with 6 dof/node. It accepts an offset as an attribute (see [MATERIAL](#)) and therefore offers a functionality not provided by the [DIMASS](#) command.
- 132 M: 3d 2-node *massless* rigid beam of *non zero* length ([Figure 1](#)); enforces constant length of the element, equal rotations of the cross sections at its two nodes, and other constraints between its rotational and translational dofs to simulate a genuinely rigid beam. Because this element is designed to be always massless, it is computationally more efficient than its counterpart, element type 66, equipped with a zero density.
- 133 M: 3d 2-node *massless* rigid beam ([Figure 1](#)) of arbitrary length, including zero length; enforces constant length of the element, equal rotations of the cross sections at its two nodes, and other constraints between its rotational and translational dofs to simulate a genuinely rigid beam. Because this element is designed to be always massless, it is computationally more efficient than its counterpart, element type 106, equipped with a zero density.
- 134 M: 3d 2-node prismatic joint-with-driver and 6 dof/node ([Figure 25](#)) and a relative displacement (between node 2 and node 1) along the joint axis of displacement that is prescribed using a time-dependent law specified in [MATERIAL](#) (for static analysis, the relative displacement is set to the initial value of the chosen law). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed as far as input is concerned as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. Its x axis defines the orientation in the undeformed configuration of the axis of prescribed relative translation.
- 150 M: 3d 4-node rigid tetrahedron with 3 dof/node ([Figure 6](#)).
- 151 M: 3d 6-node rigid pentahedron with 3 dof/node ([Figure 7](#)).
- 177 M: 3d 2-node point-to-moving-point constraint element with 3 dof/node.
- 178 M: 3d 3-node point-to-moving-line constraint element with 3 dof/node.
- 179 M: 3d 4-node point-to-moving-plane constraint element with 3 dof/node.
- 180 M: 3d 8-node Q1-P0 hexahedron with 3 dof/node ([Figure 5](#)) with nonlinear material capability. This element, which uses a 3-field mixed formulation, is suitable for modeling incompressible and nearly incompressible materials with large displacements.
- 181 M: 3d 20-node Q2-P0 hexahedron with 3 dof/node ([Figure 5](#)) with nonlinear material capability. This element, which uses a 3-field mixed formulation, is suitable for modeling incompressible and nearly incompressible materials with large displacements.
- 182 M: 3d 20-node Q2-P1 hexahedron with 3 dof/node ([Figure 5](#)) with nonlinear material capability. This element, which uses a 3-field mixed formulation, is suitable for modeling incompressible and nearly incompressible materials with large displacements.
- 183 M: 3d 4-node P1-P0 tetrahedron with 3 dof/node ([Figure 6](#)) with nonlinear material capability. This element, which uses a 3-field mixed formulation, is suitable for modeling incompressible and nearly incompressible materials with large displacements.
- 200 M: 3d 2-node uniaxial translational spring with 3 dof/node for linear or nonlinear analyses. The initial orientation of the spring axis is aligned with that of the element defined by its two nodes: hence, this element cannot have a zero length.
- 201 M: 3d 2-node uniaxial translational spring with 6 dof/node for linear or nonlinear analyses. The initial orientation of the spring axis is the local x -axis of the element frame. It must be specified under [EFRAMES](#), using the same format as for a beam. This translational spring can connect with any other element with 6 dof/node without raising any mechanism issue. Furthermore, it distinguishes itself from element type 200 in that it can have a zero length.
- 202 M: 3d 2-node uniaxial torsional spring with 3 rotational dof/node for linear or nonlinear analyses. The initial orientation of the spring axis is the local x -axis of the element frame. It must be specified under [EFRAMES](#), using the same format as for a beam.
- 203 M: 3d 2-node uniaxial tension-only translational spring with 3 dof/node and an *optional* freeplay model, for linear or nonlinear analyses. The parameters of the optional freeplay model can be specified in [MATERIAL](#). The initial orientation of the spring axis is aligned with that of the element defined by its two nodes: hence, this element cannot have a zero length.
- 204 M: 3d 2-node uniaxial tension-only or compression-only translational spring with 6 dof/node and an *optional* freeplay model, for linear or nonlinear analyses. The parameters of the optional freeplay model can be specified in [MATERIAL](#). The initial orientation of the spring axis is the local x -axis of the element frame. It must be specified under [EFRAMES](#), using the same format as for a beam. The element acts as a tension-only translational spring if the dot product of the local x -axis specified under [EFRAMES](#) and the axis implied by the ordering of the two nodes defining this element is positive. Otherwise, it acts as a compression-only translational spring.
- 205 M: 3d 2-node uniaxial tension-only or compression-only torsional spring with 3 rotational dof/node and an *optional* freeplay model, for linear or nonlinear analyses. The parameters of the optional freeplay model can be specified in [MATERIAL](#). The initial orientation of the spring axis is the local x -axis of the element frame. It must be specified under [EFRAMES](#), using the same format as for a beam. The element acts as a tension-only rotational spring if the dot product of the local x -axis specified under [EFRAMES](#) and the axis implied by the ordering of the two nodes defining this element is positive.

- positive. Otherwise, it acts as a compression-only rotational spring.
- 220 M: 3d 2-node spherical joint spring combination with 6 dof/node ([Figure 21](#)). This element is equivalent to one spherical joint (type 120) and three embedded torsional springs (type 202). The initial orientation of the first embedded spring's axis is the local x -axis of the element frame. The initial orientation of the second embedded spring's axis is the local y -axis of the element frame. The initial orientation of the third embedded spring's axis is the local z -axis of the element frame. This information must be specified under [EFRAMES](#), using the same format as for a beam element.
- 221 M: 3d 2-node translational joint spring combination element with 6 dof/node ([Figure 28](#)). This element is equivalent to one translational joint (type 121) and three embedded translational springs (type 201). The initial orientation of the first embedded spring's axis is the local x -axis of the element frame. The initial orientation of the second embedded spring's axis is the local y -axis of the element frame. The initial orientation of the third embedded spring's axis is the local z -axis of the element frame. This information must be specified under [EFRAMES](#), using the same format as for a beam element.
- 222 M: 3d 2-node universal joint spring combination element with 6 dof/node ([Figure 22](#)). This element is equivalent to one universal joint (type 122) and two embedded torsional spring elements (type 202). The initial orientation of the first embedded spring's axis is the local y -axis of the element frame. The initial orientation of the second embedded spring's axis is the local z -axis of the element frame. This information must be specified under [EFRAMES](#), using the same format as for a beam element.
- 223 M: 3d 2-node revolute joint spring combination element with 6 dof/node ([Figure 23](#)). This element is equivalent to one revolute joint (type 123) and one embedded torsional spring element (type 202). The initial orientation of the embedded spring's axis is the local x -axis of the element frame. This information must be specified under [EFRAMES](#), using the same format as for a beam element.
- 224 M: 3d 2-node cylindrical joint spring combination element with 6 dof/node ([Figure 24](#)). This element is equivalent to one cylindrical joint (type 124), one embedded translational spring (type 201), and one embedded torsional spring (type 202). The initial orientation of the first (translational) embedded spring's axis is the local x -axis of the element frame. The initial orientation of the second (torsional) embedded spring's axis is also the local x -axis of the element frame. This information must be specified under [EFRAMES](#), using the same format as for a beam element.
- 225 M: 3d 2-node prismatic joint spring combination element with 6 dof/node ([Figure 25](#)). This element is equivalent to one prismatic joint (type 125) and one embedded translational spring (type 201). The initial orientation of the embedded spring's axis is the local x -axis of the element frame. This information must be specified under [EFRAMES](#), using the same format as for a beam element.
- 226 M: 3d 2-node revolute joint-with-actuator with 6 dof/node ([Figure 23](#)). This element is equivalent to one revolute joint spring combination element (type 223) and two equal and opposite embedded follower (see [FORCES](#)) moments applied to the two nodes of the element. The magnitude of these moments is prescribed using a time-dependent law specified in [MATERIAL](#) (for static analysis, the magnitude of the moments is set to the initial value of the chosen law). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed as far as input is concerned as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. Its x -axis defines the orientation in the undeformed configuration of the axis about which the embedded moment applied to the second node of the element acts, and also the initial orientation of the embedded spring's axis.
- 227 M: 3d 2-node pin-in-slot joint spring combination element with 6 dof/node ([Figure 29](#)). This element is equivalent to one pin-in-slot joint (type 127), one embedded translational spring element (type 201), and one embedded torsional spring (type 202). The initial orientation of the first embedded (translational) spring's axis is the local x -axis of the element frame. The initial orientation of the second embedded (torsional) spring's axis is the local y -axis of the element frame.
- This information must be specified under [EFRAMES](#), using the same format as for a beam.
- 234 M: 3d 2-node prismatic joint-with-actuator element with 6 dof/node ([Figure 25](#)). This element is equivalent to one prismatic joint spring combination element (type 225) and two equal and opposite embedded follower (see [FORCES](#)) forces applied to the two nodes of the element. The magnitude of these forces is prescribed using a time-dependent law specified in [MATERIAL](#) (for static analysis, the magnitude of the forces is set to the initial value of the chosen law). This element has a co-rotating reference frame attached to each of its two nodes. Both frames have the same orientation in the undeformed configuration and therefore can be viewed as far as input is concerned as the same frame. The orientation of this frame in the undeformed configuration must be specified under [EFRAMES](#), using the same format as for a beam. Its x -axis defines the orientation in the undeformed configuration of the direction in which the embedded force applied to the second node of the element acts, and also the initial orientation of the embedded spring's axis.
- 280 M: 3d 8-node Q1-P0 incompressible hexahedron with 3 dof/node ([Figure 5](#)) with nonlinear material capability. This element, which uses a 4-field mixed formulation, is suitable for modeling incompressible and nearly incompressible materials with large displacements.
- 301 F: 2d 4-node quadrilateral with 1 dof/node ([Figure 2](#)) for modeling an incompressible, inviscid fluid — in the presence of gravity — using a displacement-potential formulation, and computing sloshing modes.
- 302 F: 2d 2-node element with 1 dof/node for modeling the free-surface of an incompressible, inviscid fluid using a displacement-potential formulation ([Figure 1](#)). For two-dimensional sloshing or hydroelastic mode computations, it is to be used with element type 301.
- 311 F: 3d 4-node tetrahedron with 1 dof/node ([Figure 6](#)) for modeling an incompressible, inviscid fluid — in the presence of gravity — using a displacement-potential formulation, and computing sloshing modes.
- 312 F: 3d 3-node triangle with 1 dof/node for modeling the free-surface of an incompressible, inviscid fluid using a displacement-potential formulation ([Figure 3](#)). For three-dimensional sloshing or hydroelastic mode computations, it is to be used with element type 311.
- 321 F: 2d 4-node quadrilateral with 1 dof/node ([Figure 2](#)) for modeling an incompressible, weightless fluid using a displacement-potential formulation, and computing hydroelastic vibration modes.
- 323 M: This element is a variant of element type 223 that is equipped with a freepay model ([Figure 23](#)). Hence, it is a 3d 2-node revolute joint spring combination element with a nonlinear (or piecewise linear, see [PIECEWISE](#) in [STATICS](#)) torsional spring and 6 dof/node. The parameters of its freepay model can be specified in [MATERIAL](#). The initial orientation of the embedded spring's axis is the local x -axis of the element frame. This information must be specified under [EFRAMES](#), using the same format as for a beam.

- 325 M: This element is a variant of element type 225 that is equipped with a freeplay model ([Figure 25](#)). Hence, it is a 3d 2-node prismatic joint spring combination element with a nonlinear (or piecewise linear, see [PIECEWISE](#) in [STATICS](#)) translational spring and 6 dof/node. The parameters of its freeplay model can be specified in [MATERIAL](#). The initial orientation of the embedded spring's axis is the local x -axis of the element frame. This information must be specified under [EFRAMES](#), using the same format as for a beam.
- 331 F: 3d 4-node tetrahedron with 1 dof/node ([Figure 6](#)) for modeling an incompressible, weightless fluid using a displacement-potential formulation, and computing hydroelastic vibration modes.
- 333 H: 3d 2-node line with 1 dof/node for ablation associated with the Q^* ablation model ([Figure 1](#)).
- 334 H: 3d 3-node triangle 1 dof/node for ablation associated with the Q^* ablation model ([Figure 3](#)).
- 335 H: 3d 4-node quadrilateral with 1 dof/node for ablation associated with the Q^* ablation model ([Figure 2](#)).
- 1100 A: 2d 4-node Helmholtz DGM element Q-4-1 ([Figure 19](#) with $p = 2$).
- 1101 A: 2d 4-node Helmholtz DGM element Q-8-2 ([Figure 19](#) with $p = 2$).
- 1102 A: 2d 4-node Helmholtz DGM element Q-16-4 ([Figure 19](#) with $p = 2$).
- 1103 A: 2d 4-node Helmholtz DGM element Q-32-8 ([Figure 19](#) with $p = 2$).
- 1110 A: 2d 3-node Helmholtz DGM element T4-1 ([Figure 20](#) with $p = 2$, or [Figure 3](#)).
- 1111 A: 2d 3-node Helmholtz DGM element T8-2 ([Figure 20](#) with $p = 2$, or [Figure 3](#)).
- 1120 A: 2d 4-node Helmholtz DEM element Q-4-1 ([Figure 19](#) with $p = 2$).
- 1121 A: 2d 4-node Helmholtz DEM element Q-8-2 ([Figure 19](#) with $p = 2$).
- 1122 A: 2d 4-node Helmholtz DEM element Q-16-4 ([Figure 19](#) with $p = 2$).
- 1123 A: 2d 4-node Helmholtz DEM element Q-32-8 ([Figure 19](#) with $p = 2$).
- 1130 A: 2d 3-node Helmholtz DEM element T4-1 ([Figure 20](#) with $p = 2$, or [Figure 3](#)).
- 1131 A: 2d 3-node Helmholtz DEM element T8-2 ([Figure 20](#) with $p = 2$, or [Figure 3](#)).
- 1150 A: 3d 8-node Helmholtz DGM element H-6-1 ([Figure 18](#) with $p = 2$).
- 1151 A: 3d 8-node Helmholtz DGM element H-26-4 ([Figure 18](#) with $p = 2$).
- 1152 A: 3d 8-node Helmholtz DGM element H-56-8 ([Figure 18](#) with $p = 2$).
- 1153 A: 3d 8-node Helmholtz DGM element H-98-12 ([Figure 18](#) with $p = 2$).
- 1160 A: 3d 4-node Helmholtz DGM element T-6-1 ([Figure 11](#) with $p = 2$, or [Figure 6](#)).
- 1161 A: 3d 4-node Helmholtz DGM element T-26-4 ([Figure 11](#) with $p = 2$, or [Figure 6](#)).
- 1162 A: 3d 4-node Helmholtz DGM element T-56-8 ([Figure 11](#) with $p = 2$, or [Figure 6](#)).
- 1170 A: 3d 8-node Helmholtz DEM element H-6-1 ([Figure 18](#) with $p = 2$).
- 1171 A: 3d 8-node Helmholtz DEM element H-26-4 ([Figure 18](#) with $p = 2$).
- 1172 A: 3d 8-node Helmholtz DEM element H-56-8 ([Figure 18](#) with $p = 2$).
- 1173 A: 3d 8-node Helmholtz DEM element H-98-12 ([Figure 18](#) with $p = 2$).
- 1200 M: 2d 4-node Elastodynamic DGM element Q-4x2-2 ([Figure 19](#) with $p = 2$).
- 1201 M: 2d 4-node Elastodynamic DGM element Q-16x2-8 ([Figure 19](#) with $p = 2$).
- 1220 M: 2d 4-node Elastodynamic DEM element Q-4x2-2 ([Figure 19](#) with $p = 2$).
- 1250 M: 3d 8-node Elastodynamic DGM element H-6x3-3 ([Figure 18](#) with $p = 2$).
- 1251 M: 3d 8-node Elastodynamic DGM element H-26x3-15 ([Figure 18](#) with $p = 2$).
- 1252 M: 3d 8-node Elastodynamic DGM element H-50x3-28 ([Figure 18](#) with $p = 2$).

CONNECTIVITY_NODES These should be listed in a stacked fashion on a single line. The various orderings are graphically depicted in the figures below.

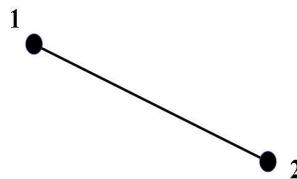


Figure 1: local node numbering for element types 1, 6, 7, 9, 21, 22, 26, 47, 56, 65--69, 111, 302, 333

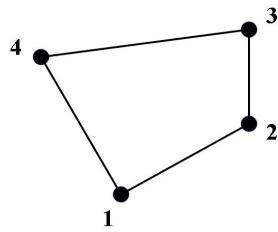


Figure 2: local node numbering for element types 2, 3, 88, 10, 1515, 16, 18, 2020, 30, 31, 33, 34, 4646, 48, 58, 76, 81, 87, 128, 301, 321, 335

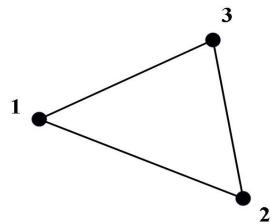


Figure 3: local node numbering for element types 4, 8, 15, 16, 19, 20, 35, 36, 46, 49, 53, 57, 73, 129, 312, 334, 1110, 1111, 1130, 1131

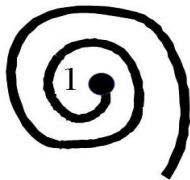


Figure 4: local node numbering for element type 11

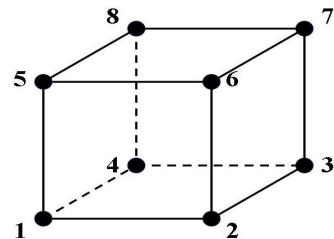


Figure 5: local node numbering for element types 17, 44, 45, 51, 70, 82, 180, 181, 182, 280

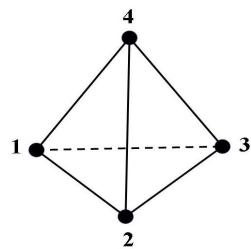


Figure 6: local node numbering for element types 23, 40, 41, 150, 183, 311, 331, 1160, 1161, 1162

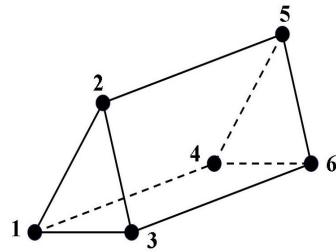


Figure 7: local node numbering for element types 24, 83, 90, 151

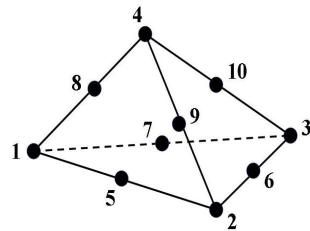


Figure 8: local node numbering for element types 25, 42

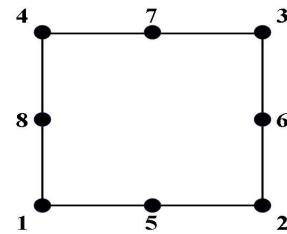


Figure 9: local node numbering for element type 32

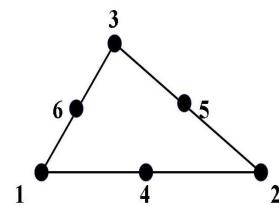


Figure 10: local node numbering for element types 38, 52

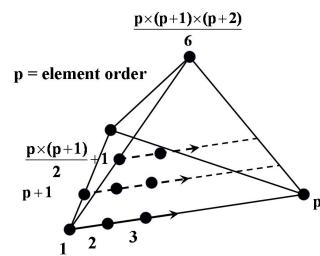


Figure 11: local node numbering for element types 50, 96, 103, 1160, 1161, 1162

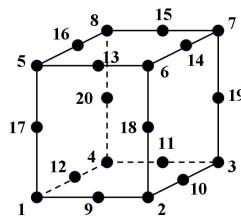


Figure 12: local node numbering for element types 71, 72

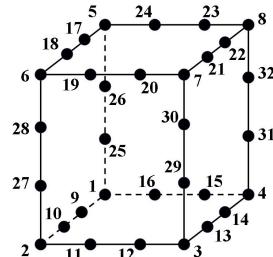


Figure 13: local node numbering for element types 74, 91, 93

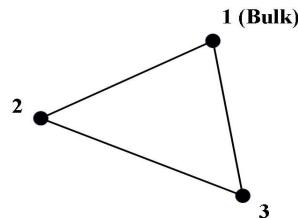


Figure 14: local node numbering for element type 84

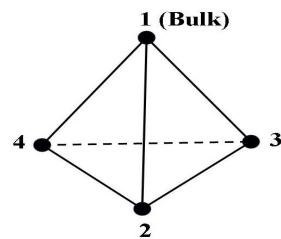


Figure 15: local node numbering for element type 85

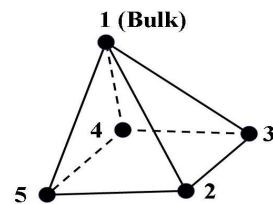


Figure 16: local node numbering for element type 86

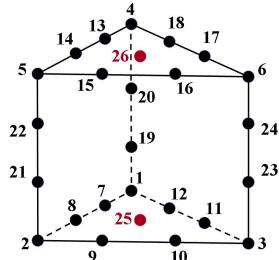


Figure 17: local node numbering for element types 92, 94

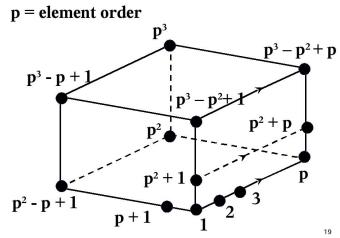


Figure 18: local node numbering for element types 95, 102, 105, 109, 1150, 1151, 1152, 1153, 1170, 1171, 1172, 1173, 1250, 1251, 1252

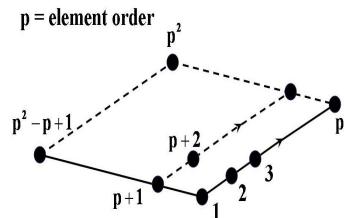


Figure 19: local node numbering for element types 98, 100, 108, 1100, 1101, 1102, 1103, 1120, 1121, 1122, 1123, 1200, 1201, 1220

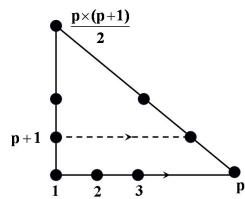


Figure 20: local node numbering for element types 99, 101, 1110, 1111, 1130, 1131

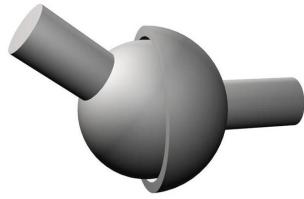


Figure 21: element types 120, 220



Figure 22: element types 122, 222

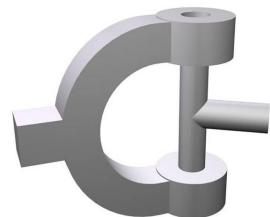


Figure 23: element types 123, 126, 223, 226, 323



Figure 24: element types 124, 224



Figure 25: element types 125, 134, 225, 234, 325

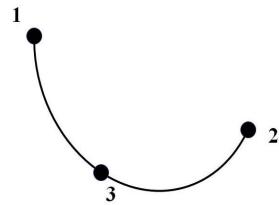


Figure 26: element type 27

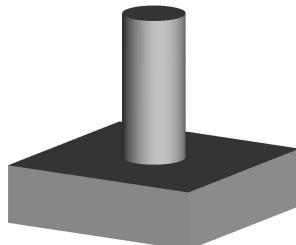


Figure 27: element type 118

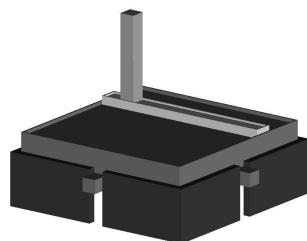


Figure 28: element types 121, 221

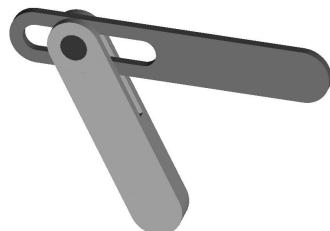


Figure 29: element types 127, 227

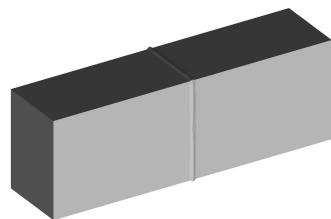


Figure 30: element type 119

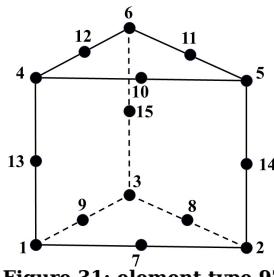


Figure 31: element type 97

Next: [END](#), Previous: [TOPOLOGY](#)

36 ENABLING REDUCTION AND HYPER REDUCTION USING LOCAL REDUCED-ORDER BASES

Command Statement: **LOCROB**

The LOCROB command should be used to enable the reduction or hyper reduction of nonlinear structural dynamic computation using local Reduced-Order Bases (ROBs), following the theory described in *D. Amsallem, M. Zahr and C. Farhat, "Nonlinear Model Order Reduction Based on Local Reduced-Order Bases," International Journal for Numerical Methods in Engineering, Vol. 92, pp. 891-916 (2012)*.

Note 1: Currently, **AERO-S** supports reduced-order computations using the method of local ROBs only for *implicit* dynamic problems.

Note 2: Using a set of local ROBs instead of a single (global) ROB requires repeating in the same ASCII Input Command Data file and for each local ROB, the commands [READMODE](#), [ATTRIBUTES](#), the sub-command keywords [PODROB](#) and [TRNSOL](#) of the command [RMSHC](#), and the sub-command keywords [CENTROID](#) and [AUXI](#) of this command. For example, the j -th occurrence of the command [READMODE](#) specifies the filename and size of the j -th local ROB, the j -th occurrence of the command [ATTRIBUTES](#) specifies the element weights of the reduced mesh associated with the j -th local ROB, etc.

The input format of this command is given below.

LOCROB
CENTROID <code><pathandfilename1></code> AUXI <code><pathandfilename2></code>

CENTROID	Sub-command keyword for specifying a file containing the centroid of a snapshot cluster generated by ROBC (characters). This sub-command and its input are needed when sampling a mesh using RMSHC and local bases, or when performing a nonlinear structural dynamic computation using local ROBs without hyper reduction.
<code><pathandfilename1></code>	Path and name of a file containing the centroid of a snapshot cluster (characters).
AUXI	Sub-command keyword for specifying a file containing auxiliary quantities generated by RMSHC that are required for enabling a fast <i>online</i> selection of the appropriate local ROB during a nonlinear structural dynamic computation performed using a hyper reduced computational model (characters).
<code><pathandfilename2></code>	Path and name of a file containing the auxiliary quantities required for enabling a fast <i>online</i> selection of the appropriate local ROB during a computation performed using a hyper reduced nonlinear computational model (characters).

Next: [SOLVERCNTL](#), Previous: [LOCROB](#)

37 END

The **END** statement is used to indicate the end-of-file. It should always be the last statement.

Next: [FWEIGHTS](#), Previous: [END](#)

38 EQUATION SOLVER CONFIGURATION

Command Statement: **SOLVERCNTL SOLVER_ID**

The main purpose of the SOLVERCNTL command statement is to support the `solverhandle` solver of the [STATICS](#) command, which is nothing but a

properly configured equation solver (here, configuring an equation solver refers to setting its parameters, if it has any). Its main benefits are two-fold:

- To enable the configuration of a multi-level FETI-DP solver, which cannot be performed within the [STATICS](#) command, and then its selection in the [STATICS](#) command as a solverhandle solver.
- To configure outside the [STATICS](#) command, in a manner that is clearer and less prone to errors than within the [STATICS](#) command, one or multiple other equation solvers and attribute to each of them a `SOLVER_ID`; then, at each simulation of interest, one of these solverhandle solvers can be selected as needed by specifying in the [STATICS](#) command the solver solverhandle `SOLVER_ID`. Hence, unlike the above benefit, this one is purely for convenience.

The input format of this command is given below.

Note 1: All `METHODS` and `PARAMETERS` specifiable in [STATICS](#) can also be specified in this command.

Note 2: Specifying a FETI-DP solver as a coarse solver of a global FETI-DP solver, including recursively, corresponds to configuring a multi-level FETI-DP solver. In this case:

- The original subdomains are treated as superelements and these are rearranged recursively into coarse-level-subdomains at each coarse level where the coarse solver is specified to be `FETI DP`.
- For a two-level FETI-DP solver, the number of coarse-level-subdomains is by default the number of MPI processes, and each coarse-level-subdomain is constructed as the agglomeration of the original subdomains assigned to an MPI process. Alternatively, a coarse-level mesh decomposition can be specified for each coarse level by including in the directory where the simulation is launched a file named `decomposition.N`, with the integer `N` being the number of subdomains in the previous level mesh decomposition. The format of this file is the same as that of the original decomposition file.
- For a three-or-higher-level FETI-DP solver, a coarse-level mesh decomposition must be specified for each coarse level except the last one. At the last coarse level, the default setting described above applies, but the user can also specify an alternative mesh partition even at this coarse level by inputting the corresponding file as described above.
- At each coarse level, an agglomerated coarse subdomain must consist of subdomains of a previous decompositon that were assigned to the same CPU. Hence, a user who wishes to input a specific "decomposition.N" file must have knowledge of the content of the `CPUMAP` file described in [BINARY](#) in order to satisfy this requirement (which is otherwise automatically satisfied by each default mesh decomposition).
- Only primal augmentation is supported (see [STATICS](#)).

<code>SOLVERCNTL</code>	<code>SOLVER_ID</code>
-------------------------	------------------------

<code>METHOD</code>
<code>PARAMETERS</code>

<code>SOLVER_ID</code>	Non-negative integer which identifies explicitly a configured equation solver – that is, an equation solver and a set of values for its parameters, when applicable (integer). This equation solver can be selected in STATICS as the solver solverhandle <code>SOLVER_ID</code> .
<code>METHOD</code>	See STATICS . Any <code>METHOD</code> that can be specified and configured in STATICS can also be specified and configured here.
<code>PARAMETERS</code>	See STATICS . Any solver parameter that can be specified in STATICS can also be specified here. However, only the parameters relevant to the equation solver specified in <code>METHOD</code> above should be specified here.

Next: [FSINTERFACE](#), Previous: [SOLVERCNTL](#)

39 FIELD WEIGHTS FOR MESH DECOMPOSITION

Command Statement:	FWEIGHTS
--------------------	-----------------

In order to achieve load balancing when generating a mesh partition, the second-step of the non-trivial mesh partitioning strategy (see [DECOMPOSE](#)) accounts for the different weights (or weighting coefficients, see [WEIGHTS](#)) attributed to the elements of the given mesh according to their type (see [TOPOLOGY](#)). The default values attributed by **AERO-S** to these elements types are based on their relative computational complexity. For each different field — that is, Mechanic, Heat, Coupled Thermoelastic, Fluid, or Acoustic — these weights range between 1 and some higher value that is field-dependent. The command [WEIGHTS](#) can be used to reset, if desired, some or all of these weights to some user-specified values.

For a multidisciplinary simulation such as fluid-structure using **AERO-S** alone, the command [FWEIGHTS](#) can be used to attribute to each involved field (for example, Mechanic and Acoustic) a weight to allow [DECOMPOSE](#) to account additionally for the relative computational complexity of a typical element of this field to those of the typical elements of the other fields involved in the simulation. In this case, the weight w_e of each element type (whether its default value or that set using [WEIGHTS](#)) is automatically reset by **AERO-S** to the ratio

$$\frac{w_e \times w_{field_of_e}}{\sum_{involvedfields} w_{field}}$$

where $w_{field_of_e}$ is the weight of the field of element type e , the scope of *involvedfields* is defined by all fields specified using this command.

The input format of this command is given below.

FWEIGHTS	
ACOU	Sub-command keyword for specifying a weight to the field Acoustic (see TOPOLOGY).
acou_weight	Weight to be attributed to the field Acoustic (real).
FLUI	Sub-command keyword for specifying a weight to the field Fluid (see TOPOLOGY).
flui_weight	Weight to be attributed to the field Fluid (real).
MECH	Sub-command keyword for specifying a weight to the field Mechanic (see TOPOLOGY).
mech_weight	Weight to be attributed to the field Mechanic (real).
HEAT	Sub-command keyword for specifying a weight to the field Heat (see TOPOLOGY).
heat_weight	Weight to be attributed to the field Heat (real).

Next: [MFTT](#), Previous: [FWEIGHTS](#)

40 FLUID/STRUCTURE INTERFACE

Command Statement:	FSINTERFACE
--------------------	-------------

The FSINTERFACE command statement can be used to define the fluid/structure interface in a coupled elastoacoustic frequency response problem where: (1) the computational acoustic fluid and structural domains are discretized using either a single mesh, or two different meshes with matching or non-matching discrete interfaces, and (2) in either case, each of these two computational domains has its own representation of the fluid/structure interface in the form of a discrete surface defined using [SURFACETOPO](#). In this scenario, the command statement FSINTERFACE is used to define more specifically the fluid/structure interface as the pairing of these two surfaces.

The input format of this command is given below.

Note 1: In this case, the capabilities of the ACME library are used to generate the fluid-structure coupling coefficients.

FSINTERFACE		
-------------	--	--

SURF_PAIR_ID#	FLUID_SURF	STR_SURF
---------------	------------	----------

or

SURF_PAIR_ID#	FLUID_SURF	STR_SURF	NORMAL_TOL	TANGENTIAL_TOL
---------------	------------	----------	------------	----------------

SURF_PAIR_ID#	Id number of the surface pair to be described (integer).			
FLUID_SURF	Identification of the fluid surface (see SURFACETOPO) (integer).			
STR_SURF	Identification of the structure surface (see SURFACETOPO) (integer).			
NORMAL_TOL	Normal search tolerance used by ACME to identify interactions, default value is 0.1 (float) (see Figs. 1.2 and 1.3 in Section 1.3 of ACME's User Reference Manual).			
TANGENTIAL_TOL	Tangential search tolerance used by ACME to identify interactions, default value is 0.001 (float) (see Figs. 1.2 and 1.3 in Section 1.3 of ACME's User Reference Manual).			

Next: [GRBM](#), Previous: [FSINTERFACE](#)

41 FORCE TIME TABLE-MECHANICS AND ACOUSTICS

Command Statement:	MFTT [TABLE_ID]
--------------------	-----------------

The MFTT command statement can be used to implement time-dependent tensor-product forms of the boundary conditions enforced by the

commands **FORCE**, **ATDDNB**, **ATDROB**, and **PRESSURE**. Pairs of time and amplification values are input. Linear interpolation is also used for "in between" points.

When applied with the **FORCE** command, the result is the amplification of the force value. When applied with the **ATDDNB** command, the result is the amplification of the resulting distributed Neumann boundary condition. When applied with the **ATDROB** command, the result is the amplification of the right hand-side of the distributed Robin boundary condition.

MFTT [TABLE_ID]

TIME_1	AMP_1
.	.
.	.
.	.
TIME_n	AMP_n

TABLE_ID

Optional non-negative integer which uniquely identifies a force-time table so that it can be associated with a "load" set to define the "load" case for a dynamic analysis using the **LOADCASE** command. The default value is 0. Hence, the **MFTT** command can be repeated as many times as desired within the same input file using each time a different value for TABLE_ID and different data.

TIME_1
AMP_1

A specified time point (float).

A specified amplification value at time point TIME_1 (float). This amplification factor is automatically set to zero for all times prior to the earliest specified time point and all times later than the latest specified time point.

Next: [GEPS](#), Previous: [MFTT](#)

42 GEOMETRIC RIGID BODY MODES

Command Statement: GRBM [X Y Z]

A singular finite element stiffness matrix arises in a linear or nonlinear static or quasistatic analysis, or in an eigen analysis, if, for example, any of the following conditions are encountered:

- The underlying problem is formulated without sufficient Dirichlet boundary conditions.
- The finite element model contains mechanisms or other oddities that generate singularities.
- The underlying problem contains redundant constraints and the Lagrange multiplier method is chosen for enforcing them (see [CONSTRAINTS](#)).

In this case:

- In the context of a *linear* static or quasistatic analysis, the **GRBM** command can be used to:
 - Determine the rigid body modes of the prevailing stiffness matrix that are due to a lack of sufficient Dirichlet boundary conditions or [LMPCs](#) to guarantee its invertibility.
 - Assist, using the above information, the direct equation solvers **sparse** and **skyline**, and, to some extent, the iterative solver **FETI-DP** (see [STATICS](#) and [Note 4](#) below) in solving the admissible system of linear equations governed by this singular matrix. For an **EIGEN** analysis however, this assistance is performed only when the shift is zero.
- In the context of an **EIGEN** (structural) analysis, the **GRBM** command can be used to:
 - Determine the rigid body modes of the prevailing stiffness matrix that are due to a lack of sufficient Dirichlet boundary conditions or [LMPCs](#) to guarantee its invertibility.
 - Use the above information to assist, if no shifting is requested (see [EIGEN](#)), the direct equation solvers **sparse** and **skyline** and, to some extent, the iterative solver **FETI-DP** (see [STATICS](#) and [Note 4](#) below), in solving the admissible system of linear equations governed by this singular matrix each time they are called by the chosen eigensolver.

Specifically, this command has two *distinct* effects:

1. It requests the computation of the rigid body modes of an unrestrained, partially restrained, or even restrained finite element model using the hybrid geometric-algebraic method published in *C. Farhat and M. Geradin, "On the General Solution by a Direct Method of a Large-Scale Singular System of Linear Equations: Application to the Analysis of Floating Structures," International Journal for Numerical Methods in Engineering, Vol. 41, pp. 675-696 (1998)*. This hybrid method combines a geometry-based algorithm and the SVD factorization of the matrix of constraints associated with any specified boundary conditions and/or [LMPCs](#). The SVD factorization relies on the first tolerance specified under this command for identifying the deemed singular values associated with the rigid body modes due to insufficient Dirichlet boundary conditions or [LMPCs](#). In this case, **AERO-S** prints on the screen the total number of rigid body modes discovered by **GRBM**. Then, the rigid body modes themselves are used, for example, to assist the **RBMFILTER** command, or support the **EIGEN** command. This first aspect of the **GRBM** command is independent however of the solution method specified in [STATICS](#).
2. When the direct method **sparse** or **skyline** is selected under the **STATICS** command as the equation solver, this command triggers the computation of the generalized inverse of the singular finite element stiffness matrix in factored form using the algorithm also published in *C. Farhat and M. Geradin, "On the General Solution by a Direct Method of a Large-Scale Singular System of Linear Equations: Application to the Analysis of Floating Structures," International Journal for Numerical Methods in Engineering, Vol. 41, pp. 675-696 (1998)*. This algorithm relies on the information computed by the hybrid geometric-algebraic method outlined above — that is, the number of rigid body modes, and in some cases, the rigid body modes themselves, discovered and constructed by **GRBM**.

In some cases, a finite element matrix can have a larger number of singularities (also known as nullities) than that due to the presence of

rigid body modes. This is true, for example, when the finite element model contains mechanisms. To address this issue, when the `GRBM` command is specified and `skyline` or `sparse` is chosen in [STATICS](#) as the equation solver, the generalized inverse of the stiffness matrix is computed in factored form using the aforementioned hybrid geometric-algebraic algorithm, and its pivots are monitored for small values using the second tolerance specified under this command. When these small values are correctly deemed to be zero pivot values, the solvers `skyline` and `sparse` capture the additional singular modes due to mechanisms or other model oddities.

Note 1: See [TRBM](#) for an alternative method for analyzing singular systems. Note however that whereas the rigid body modes computed by the `GRBM` command are "pure" translational and rotational modes, those computed using the `TRBM` command may be linear combinations of translational and rotational modes.

Note 2: This command can be used together with the [TRBM](#) command. If both `GRBM` and [TRBM](#) commands are specified in the same **AERO-S** ASCII Input Command Data file, then:

- If the analysis to be performed is a *linear* static or quasistatic analysis and the equation solver specified in [STATICS](#) is the `sparse` or `skyline` solver, or if this analysis is an eigen (structural) analysis, `GRBM` can be used to (see also **Note 4** below regarding the case of the iterative solver `FETI-DP`):
 - Determine the rigid body modes of the prevailing stiffness matrix that are due to a lack of sufficient Dirichlet boundary conditions or [LMPCs](#) to guarantee the invertibility of this matrix.
 - Assist the direct equation solvers `sparse` and `skyline` (see [STATICS](#)) in solving the admissible system of linear equations governed by this singular matrix. For an [EIGEN](#) analysis, this assistance is performed however only when the shift is zero. Therefore the [TRBM](#) command is ignored in this case.
- Otherwise:
 - `GRBM` is used to determine, *for the purpose of information and only for this purpose*, the rigid body modes of the prevailing stiffness matrix that are due to a lack of sufficient Dirichlet boundary conditions or [LMPCs](#) to guarantee the invertibility of this matrix.
 - [TRBM](#) is used to assist the direct equation solver specified in [STATICS](#) in solving the admissible system of linear equations governed by this singular matrix.

Note 3: In some cases, the tangent stiffness matrix associated with an unrestrained or partially restrained *nonlinear* finite element model can have fewer rigid body modes than that which can be predicted for its linear finite element model counterpart. For this reason, the `GRBM` command cannot reliably assist the direct equation solvers `sparse` and `skyline` (see [STATICS](#)) in solving singular system of linearized equations associated with a nonlinear finite element analysis. In this case, the alternative [TRBM](#) command should be used for this purpose.

Note 4: If a FETI-1 or FETI-2 solver is specified under [STATICS](#), the rigid body modes of the floating subdomains are computed using the same hybrid geometric-algebraic method and their singular matrices are factored using the same aforementioned algorithm (see the subsection of [STATICS](#) focused on FETI parameters). If a FETI-DP solver is specified under [STATICS](#), the first tolerance specified under this command is used for detecting and computing the rigid body modes of the global stiffness matrix.

<code>GRBM</code>	<code>[X Y Z]</code>
<code>VALUE_1</code>	<code>VALUE_2</code>
<code>VALUE_3</code>	
or	
<code>GRBM</code>	<code>[X Y Z]</code>
<code>VALUE_2</code>	

X Y Z By default, the implementation of the hybrid geometric-algebraic method underlying this command computes the rotational rigid body modes by rotating the structure about the node of the mesh with the smallest ID number. However, this option can be used to specify a different point about which to rotate this model for the aforementioned purpose (for example, its center of gravity), by inputting the `X`, `Y` and `Z` coordinates of this point.

- | | |
|----------------------|---|
| <code>VALUE_1</code> | Tolerance for identifying the zero singular values of the constraint matrix analyzed by the hybrid geometric-algebraic method for finding the rigid body modes of a finite element model (real). Its default value is 1.0e-6. |
| <code>VALUE_2</code> | Tolerance for capturing algebraically the mechanisms and other model oddities that cause a finite element stiffness matrix to be singular, and which cannot be detected by the geometric-algebraic method alone (real). Its default value is 1.0e-16. |
| <code>VALUE_3</code> | Optional integer parameter for the case where the problem contains LMPCs . It can take either of the following two values: <ul style="list-style-type: none"> • <code>VALUE_3 = 0</code>. In this case, the nodes involved in each LMPC are assumed to be fully connected in the same way, for example, as nodes which are connected by beam elements. In this case: <ul style="list-style-type: none"> ◦ The <code>GRBM</code> command can be processed much faster than otherwise. ◦ However, it may also underestimate or overestimate the total number of rigid body modes. If it underestimates it, the direct equation solvers <code>sparse</code> and <code>skyline</code> (see STATICS) can be expected to recover nevertheless the correct total number of rigid body modes of the singular system they are applied to, because they automatically monitor the small pivots during the factorization of the governing matrix using the tolerance <code>VALUE_2</code>. On the other hand, if <code>GRBM</code> overestimates the total number of rigid body modes, these direct solvers will recover in general the wrong rigid body modes and therefore deliver an incorrect solution of the linear singular system of equations to which they are applied. For this reason, this setting of <code>VALUE_3</code> should be used with care — that is, when the user has sufficient insight in the problem to be solved to conclude that <code>GRBM</code> may at worst underestimate the correct number of rigid body modes. For example, this setting is perfectly safe for a problem where two bodies are connected with tied surfaces and therefore is |

recommended for this case.

- **VALUE_3 = 1** (default value). In this case, the above assumption about **GRBM** and **LMPCs** is not made. For this reason, **GRBM** is in general more reliable when **VALUE_3** is set to 1, but its processing may be computationally more intensive in this case.

Next: [PARAMETERS](#), Previous: [GRBM](#)

43 GEOMETRIC STIFFENING DUE TO PRESTRESS

Command Statement: **GEPS**

The **GEPS** command statement is essentially a switch that turns on the accounting of pre-stress effects in the form of a geometric stiffness matrix $[K_G]$. This geometric stiffness matrix is computed around a displacement field specified under the [IDISP6](#) command which must be present in the input file. A sample input file illustrating the use of **GEPS** can be found in FEM.d/fem_examples/Buckle.d/

The effect of this switch on the [EIGEN](#) command statement is to replace the stiffness matrix by the tangent stiffness matrix and therefore compute the modes of the structure around the deformed position implied by the displacement field specified under the [IDISP6](#) command.

Its effect on the [DYNAMICS](#) and [STATICS](#) commands is to replace the stiffness matrix by the tangent stiffness matrix around the displacement field specified under the [IDISP6](#) command, including in aeroelastic applications.

Note 1: The **GEPS** command can be used in any linear analysis that is not a modal analysis, and in an [EIGEN](#) analysis.

Note 2: The theory underlying the effect of initial stress involves three different configurations: the pre-stress-free configuration C_0 , the pre-stressed configuration C^* , and the deformed configuration C obtained after applying a loading to C^* . In principle, the passage from C_0 to C^* could involve large displacements, rotations, and/or deformations, but the passage from C^* to C is restricted to small displacements and rotations. This principle defines the scope for the **GEPS** command. In **AERO-S**, an analysis which accounts for pre-stress effects using this command is performed using the geometry of C_0 , the displacement field (including rotations, when applicable) that defines the passage from C_0 to C^* , and any set of external loads that does not include the loads responsible for the passage from C_0 to C^* . Therefore, any outcome of an analysis using the **GEPS** command — including that outputted in [OUTPUT](#) — should be interpreted as an incremental outcome which can be superposed to the outcome associated with or responsible for the passage from C_0 to C^* .

Note 3: The [PRELOAD](#) command defines a related capability which nevertheless differs in two important aspects. Firstly, an analysis using the [PRELOAD](#) command is performed on the geometry of C^* . Secondly and more importantly, the [PRELOAD](#) command can be used in all of linear, nonlinear, static, transient, and eigen analyses. However, preloads can be currently inputted to truss and membrane elements only.

Note 4: If an aeroelastic analysis is requested with all of the **GEPS**, [IDISP6](#), and [IDISPLACEMENTS](#) commands present in the input file, **AERO-S** interprets the [IDISPLACEMENTS](#) command and its content as the initialization of the incremental displacement field from the configuration C^* to the configuration C . In this case, it sends to the fluid code at each time-step the sum of the updated incremental displacement and the displacement inputted under the [IDISP6](#) command. Hence, this scenario is particularly suitable for the case where the fluid code is started from an updated CFD mesh position. If on the other hand an aeroelastic analysis is requested with only the **GEPS** and [IDISP6](#) commands present in the input file, **AERO-S** understands that the incremental displacement field is initialized to zero. However, it communicates in this case with the fluid code in a very special manner: at the first time-step, it sends to the fluid code the initial value of the incremental displacement field (which in this case is zero), and at each subsequent time-step, the sum of the updated incremental displacement and the displacement specified under the [IDISP6](#) command. Hence, the latter scenario is particularly suitable for the case where the fluid code is started from an undeformed CFD mesh.

GEPS

Next: [GRAVITY](#), Previous: [GEPS](#)

44 GLOBAL SIMULATION PARAMETERS

Command Statement: **PARAMETERS**

The **PARAMETERS** command can be used to set some global parameters of a simulation of interest. Its input format is given below.

PARAMETERS

ACME	flagACME
------	----------

AGGRESSIVE_TOLERANCES	
ANDES_ALPHA	α
ANDES_BETAB	β_b
ANDES_BETAM	β_m
ANDES_CLR	clr
ANDES_CQR	cqr
AUTO_TOL	
DEFAULT_PENALTY	
GLOBAL_SEARCH_CULL	
MORTAR_INTEGRATION	numPTS
MORTAR_SCALING	scalingFCT
NLMMEM_PTYPE	flagPRS
NO_GHOSTING	
NO_MULTIPLE_INTERACTIONS	
NO_NORMAL_SMOOTHING	
NO_SECONDARY	
NO_WARPED_VOLUME	
OLD_DYNAMIC_SEARCH	
PARTITION_GAP	
RESOLUTION_TYPE	flagRT
SHARP_NON_SHARP_ANGLE	theta
SHELL_SIMPLE_LOFTING	
SKIP_PHYSICAL_FACES	

ACME	Sub-command keyword for specifying the mode of execution of the contact search algorithm of the ACME library (characters).
flagACME	Specifies the mode of execution of the contact search algorithm of the ACME library (integer). Three choices are available: 0 Sequential (default). Currently, this is the only choice available for the mortar method. 1 Parallel with centralized input on the host machine. 2 Parallel with distributed input. This option requires a specialized decomposition where a pair of master and slave entities must be in the same subdomain.
AGGRESSIVE_TOLERANCES	Sub-command keyword for activating an <i>aggressive</i> automatic determination of the contact search tolerances, when this search is performed using the ACME library (characters). In this context, the term aggressive implies that the tolerances are set less conservatively in an attempt to reduce the cost of the contact search, possibly sacrificing some robustness.
ANDES_ALPHA	Sub-command keyword for specifying the algorithmic parameter α for shell elements 15 and 1515 (characters). α Specifies the value of the algorithmic parameter α for shell elements 15 and 1515 (real). The default value is 1.5. For thin shells with high membrane stresses, the 0 value is recommended.
ANDES_BETAB	Sub-command keyword for specifying the algorithmic parameter β_b for shell elements 15 and 1515 (characters). β_b Specifies the value of the algorithmic parameter β_b for shell elements 15 and 1515 (real). The default value is 1.
ANDES_BETAM	Sub-command keyword for specifying the algorithmic parameter β_m for shell elements 15 and 1515 (characters). β_m Specifies the value of the algorithmic parameter β_m for shell elements 15 and 1515 (real). The default value is 0.32. For very thin shells with high membrane stresses, the 0 value is recommended.
ANDES_CLR	Sub-command keyword for specifying the algorithmic parameter clr for shell elements 15 and 1515 (characters). clr Specifies the value of the algorithmic parameter clr for shell elements 15 and 1515 (real). The default value is 0.
ANDES_CQR	Sub-command keyword for specifying the algorithmic parameter cqr for shell elements 15 and 1515 (characters). cqr Specifies the value of the algorithmic parameter cqr for shell elements 15 and 1515 (real). The default value is 1.
AUTO_TOL	Sub-command keyword for activating an automatic determination of the contact tolerances, when contact search is performed using the ACME library (characters).
DEFAULT_PENALTY	Sub-command keyword for activating the default penalty parameter when contact enforcement is performed using the ACME library (characters). In this case, the penalty parameter specified in the AERO-S Input Command Data file is ignored.
GLOBAL_SEARCH_CULL	Sub-command keyword for activating the following contact search option when contact search is performed using the ACME library (characters). When this option is activated, a full contact search is performed only once; subsequent searches can update existing interactions but cannot find new ones.

MORTAR_INTEGRATION	Sub-command keyword for specifying the number of integration points to be used for the mortar method (characters).
numPTS	Specifies the number of integration points to be used in the mortar method (integer). Six choices are available: 1, 3, 6 (default value), 7, 12 and 16.
MORTAR_SCALING	Sub-command keyword for specifying a scaling of the constraint function in the mortar method (characters).
scalingFCT	Scaling factor to be applied to the constraint function in the mortar method (real). The default value is 1.
NLMEM_PTYPE	Sub-command keyword for specifying how the pressure load is to be defined for nonlinear membrane elements 128 and 129 (characters).
flagPRS	Specifies how the pressure load is to be defined for nonlinear membrane elements 128 and 129 (integer). Two choices are available:
0	In this case (which is the default case), the pressure is defined as the force per unit area A , where A is the area in the current/updated configuration.
1	In this case, the pressure is defined as the force per unit area A_0 , where A_0 is the area in the reference/undeformed configuration. Note that for the pressure defined on element faces identified using the SURFACETOPO command, or on elements other than elements 128 and 129, this is the only definition of pressure currently available.
NO_GHOSTING	Sub-command keyword for deactivating ghosting in a contact search performed using the ACME library (characters).
NO_MULTIPLE_INTERACTIONS	Sub-command keyword for disabling multiple interactions in the contact search, when this search is performed using the ACME library (characters).
NO_NORMAL_SMOOTHING	Sub-command keyword for deactivating normal smoothing in the contact search, when this search is performed using the ACME library (characters).
NO_SECONDARY	Sub-command keyword for deactivating the secondary decomposition in the contact search of the ACME library (characters).
NO_WARPED_VOLUME	Sub-command keyword for activating an option to accelerate contact search when using the ACME library (characters). This option should be activated only when none of the quadrilateral faces of the contact or tied surfaces are initially warped or can become warped during the simulation.
OLD_DYNAMIC_SEARCH	Sub-command keyword for disabling the so-called new augmented method for contact search when this search is performed using the ACME library (characters).
PARTITION_GAP	Sub-command keyword for activating the following option in the contact search, when this search is performed using the ACME library (characters). The purpose of this option is to partition a gap into the part that occurs during the current computational time-step and that which pre-existed at the previous computational time-step.
RESOLUTION_TYPE	Sub-command keyword for specifying the type of normal smoothing in contact when the contact search is performed using the ACME library (characters).
flagRT	Specifies the type of normal smoothing (integer): 0 specifies that a node-based normal is to be used, while 1 (which is the default value) specifies that an edge-based normal is to be used.
SHARP_NON_SHARP_ANGLE	Sub-command keyword for specifying the minimum angle parameter used for multiple interactions and normal smoothing when contact search is performed using the ACME library (characters).
theta	Specifies the value of the minimum angle parameter in degrees (float). The default value is 30 degrees.
SHELL_SIMPLE_LOFTING	Sub-command keyword for activating an alternative simplified lofting algorithm in the contact search of the ACME library (characters).
SKIP_PHYSICAL_FACES	Sub-command keyword for activating the following option in the contact search, when this search is performed using the ACME library (characters). This option disables a feature of the ACME library designed to identify certain corner cases that may be difficult for the search algorithm to otherwise handle.

Next: [GROUPS](#), Previous: [PARAMETERS](#)

45 GRAVITATIONAL ACCELERATION

Command Statement:	GRAVITY
--------------------	----------------

The GRAVITY command is used to specify directional gravitational acceleration constants. The input format is given below.

Note 1: By default, **AERO-S** computes all element gravity loads by a consistent approach, **except for explicit dynamic computations** (see [DYNAMICS](#)), in which case **AERO-S** always uses a lumped approach for this purpose. If a consistent mass matrix is not available for a particular element (see [TOPOLOGY](#)), then **AERO-S** uses in all cases a lumped gravity load for that element. Alternatively, the [LUMPED](#) command can be used to instruct **AERO-S** to compute all element mass matrices and gravity loads by a lumped approach.

GRAVITY

COEFF_X	COEFF_Y	COEFF_Z
---------	---------	---------

COEFF_X	Gravitational Acceleration Constant in the X direction.
COEFF_Y	Gravitational Acceleration Constant in the Y direction.
COEFF_Z	Gravitational Acceleration Constant in the Z direction.

Next: [CCTT](#) Previous: [GRAVITY](#)

46 GROUPS

Command Statement:	GROUPS
--------------------	---------------

The GROUPS command can be used to organize elements into groups according to their attribute number (see [ATTRIBUTES](#)), and nodes into groups according to the integer number identifying them or the surface to which they belong (see [SURFACETOPO](#)). For example, the notion of a group of elements is used by the [RANDOM](#) command to attribute a defined random material property to a group of elements. That of a group of nodes is used by the [OUTPUT](#) command to output computed results for groups of nodes.

Note 1: Group identifiers must be contiguous integers starting from 1.

GROUPS

A	ATTRIBUTE#	GROUP#
---	------------	--------

or

A	STARTING_ATTRIBUTE#	ENDING_ATTRIBUTE#	GROUP#
---	---------------------	-------------------	--------

or

N	NODE#	GROUP#
---	-------	--------

or

N	STARTING_NODE#	ENDING_NODE#	GROUP#
---	----------------	--------------	--------

or

N SURFACE	SURFACE#	GROUP#
-----------	----------	--------

A

Sub-command keyword (character) signaling that elements are to be grouped according to their attribute number — that is, all elements whose attribute is the same as ATTRIBUTE# or within the range [STARTING_ATTRIBUTE#, ENDING_ATTRIBUTE#] are to be put in the same group GROUP#.

ATTRIBUTE#

Attribute ID number of the elements to be placed in the same group GROUP# (integer).

STARTING_ATTRIBUTE#

First attribute ID number of a sequence of attribute ID numbers that define the group GROUP# (integer).

ENDING_ATTRIBUTE#

Last attribute ID number of a sequence of attribute ID numbers that define the group GROUP# (integer).

N

Sub-command keyword (character) signaling that nodes are to be grouped according to their node identifying number — that is, the node whose identifying number is NODE# or all nodes whose identifying numbers are within the range [STARTING_NODE#, ENDING_NODE#] are to be put in the same group GROUP#.

N SURFACE

Sub-command keywords (characters) signaling that nodes are to be grouped according to the surface to which they belong, and therefore according to the integer number identifying that surface (see [SURFACETOPO](#)).

NODE#

ID number of the node to be placed in the group GROUP# (integer).

STARTING_NODE#

First node ID number of a sequence of node ID numbers that define the group GROUP# (integer).

ENDING_ATTRIBUTE#

Last node ID number of a sequence of node ID numbers that define the group GROUP# (integer).

GROUP#

Group ID number of the constructed group of elements or nodes (integer).

Next: [HZEM](#) Previous: [GROUPS](#)

47 HEAT CONDUCTION COEFFICIENT-TEMPERATURE TABLE

Command Statement:	CCTT
--------------------	-------------

The CCTT command can be used to describe, for a given material, the evolution of the heat conduction coefficient with temperature. This evolution can be specified here in a curve (or one-dimensional table) defined by pairs of heat conduction coefficient and temperature values. Linear interpolation is used for "in between" points, and the extrema values are adopted for "outside" points. Several curves can be specified, one after the other. Each curve is identified by an ID number as described below.

Note 1: Currently, this command is supported only by the 4-noded tetrahedral thermal element (type 50).

CCTT

CURVE curve_id
T_1 k_1
.
.
.
T_n k_n

CURVE	Sub-command keyword (characters) that signals the definition of a heat conduction coefficient-temperature curve (or one-dimensional table).
curve_id	ID number for the following curve (or one-dimensional table) (integer).
T_i	A specified temperature value (real).
YM_i	A specified heat conduction coefficient value at temperature T_i (real).

Next: [HZEMFILTER](#), Previous: [CCTT](#)

48 HEAT ZERO ENERGY MODE

Command Statement:	HZEM
--------------------	-------------

This command is effective only for quasistatic and transient thermal, thermoelastic, and aerothermal analyses involving thermal zero energy modes — that is, a singular conductivity matrix — when the equation solver specified under the [STATICS](#) command is the [sparse](#) or [skyline](#) solver. In this case, the constant temperature mode and the generalized inverse of the conductivity matrix are computed using a hybrid physics-algebraic algorithm, rather than the tolerance-based algorithm associated with the [TRBM](#) command.

HZEMNext: [HELMHOLTZ](#), Previous: [HZEM](#)

49 HEAT ZERO ENERGY MODES FILTER

Command Statement:	HZEMFILTER
--------------------	-------------------

This command is effective for thermal simulations involving thermal zero energy modes. When this command is specified in the input file and the thermal problem contains zero energy modes, a non-trivial projector is constructed. In a static or quasistatic analysis, the transpose of this projector is applied to the (possibly variable) right hand-side to make sure that the external load is self-equilibrated and therefore admissible. In a dynamic analysis: a) this projector is applied to the initial solution (initial displacement *and* velocity fields), and b) at each time-step, the transpose of this projector is applied to the (time-dependent) right-hand side of the equation being solved. In all cases, the net effect is to work with (and output) a solution where the thermal zero energy modes have been filtered out.

The constructed projector can be written as $P = I - Z(Z^T Q Z)^{-1} Z^T Q$ where I is the identity matrix, Z is the matrix of the zero energy modes of the thermal system (always computed with the HZEM method), and Q is either the identity matrix for static analysis, or the capacity matrix for quasistatic and dynamic analyses. Hence, in a quasistatic analysis, this command requires the specific heat to be present in the material properties specified under the [MATERIAL](#) command.

HZEMFILTERNext: [AXIHELMHOLTZ](#), Previous: [HZEMFILTER](#)

50 HELMHOLTZ

Command Statement:	HELMHOLTZ
--------------------	------------------

The command statement [HELM](#) can be used to specify an absorbing boundary condition, and/or a set of incident plane waves *or* point source terms associated with spherical waves for a frequency-domain acoustic or elastoacoustic (Helmholtz) problem. Its input format is given below.

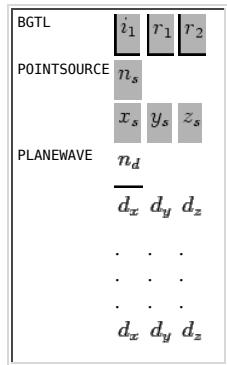
Note 1: For **AERO-S**, the time-harmonic form of the solution of a formulated Helmholtz problem is $P = p e^{-i\omega t}$, where P denotes the scalar field of interest — for example, the pressure perturbation — p is its amplitude, and ω denotes its circular frequency. When the formulated Helmholtz problem includes an incident wave of amplitude p^i (which can also be specified as a source term), **AERO-S** solves

this problem for the *scattered* amplitude $p^s = p - p^i$.

Note 2: If an incident plane wave or a point source term is specified under this command (currently, only one or the other can be specified under **HELM**), and other Dirichlet boundary conditions are also specified under **HDIR**, only the Dirichlet boundary condition $p^s = -p^i$

associated with the amplitude of the incident wave or point source terms specified under **HELM** are applied to the formulated Helmholtz problem.

HELM



BGTL

[i_1]
[r_1]

Order (0, 1, or 2) of the generalized Bayliss-Gunburger-Turkel non reflecting condition to be applied on the artificial boundary Σ specified under the **HARB** command (integer).

This real number (float) is to be used only when the artificial boundary Σ (see **HARB**) is supposed to be either a sphere or an ellipsoid, and some geometric approximations generated by **AERO-S** are to be overwritten by values that can in that case be evaluated exactly. Otherwise, r_1 should simply not be

inputted. For a sphere, the curvature of Σ , which is otherwise automatically approximated by **AERO-S** is overwritten by $1/r_1$. Similarly, for an ellipsoid of the form $(x/a)^2 + (y/b)^2 + (z/c)^2 = 1$,

[$r_1 = a$].

[r_2]

This real number (float) is to be used only when the artificial boundary Σ (see **HARB**) is supposed to be an ellipsoid, and some geometric approximations generated by **AERO-S** are to be overwritten by values that can in that case be evaluated exactly. Otherwise, r_2 should simply not be inputted. For an ellipsoid

of the form $(x/a)^2 + (y/b)^2 + (z/c)^2 = 1$, [$r_2 = b = c$].

POINTSOURCE

[n_s]

Keyword indicating that the Dirichlet and/or Neumann boundary conditions (see **HDIR** and **HDBN**) of the acoustic (Helmholtz) or elastoacoustic problem defined by this command are associated with a time-harmonic spherical wave of the form $\frac{1}{4\pi} \frac{e^{jkr}}{r}$, where $r = |X - X_s|$, and propagated by a point

[x_s]
[y_s]
[z_s]

source located at the point X_s (character). Multiple spherical waves and point sources can be specified under this command leading to multiple acoustic or elastoacoustic analyses. Hence, this line should be followed by a line specifying the number of point sources and spherical waves, n_s , and n_s lines specifying for each wave its point source X_s . When used to define Neumann boundary conditions (see **HDBN**), this option is available only for the faces of type 10 and 11 (see **HDBN**). Number of point sources propagating spherical waves (integer).

x coordinate of the location of a point source (float).

y coordinate of the location of a point source (float).

z coordinate of the location of a point source (float).

PLANEWAVE

Keyword indicating that the Dirichlet and/or Neumann boundary conditions (see **HDIR** and **HDBN**) of the acoustic (Helmholtz) or elastoacoustic problem defined by this command are associated with an incident time-harmonic plane wave of the form e^{jkr} (character). Multiple such waves can be specified under this command leading to multiple acoustic or elastoacoustic analyses. Hence, this line should be

followed by a line specifying the number of plane waves to be considered, n_d , and n_d lines specifying for each wave its direction of propagation d_i .
 n_d
 $\overline{d_x}$ Number of incident planar waves (integer).
 d_y Normalized x component of the direction of a planar wave (float).
 d_z Normalized y component of the direction of a planar wave (float).
 d_z Normalized z component of the direction of a planar wave (float).

Next: [HARB](#), Previous: [HELMHOLTZ](#)

51 HELMHOLTZ ANALYSIS (AXISYMMETRIC/PARTIALLY AXISYMMETRIC)

Command Statement: **AXIHELMHOLTZ**

The command statement AXIHELMHOLTZ can be used to:

- Activate the acoustic analysis in the frequency domain (Helmholtz) of an axisymmetric or partially axisymmetric scatterer.
- Specify the acoustic analysis parameters including the directions of the incident planar wave.

The computational technology underlying this command is that described in:

- U. Hetmaniuk and C. Farhat, "A Fictitious Domain Decomposition Method for the Solution of Partially Axisymmetric Acoustic Scattering Problems - Part II: Neumann Boundary Conditions," *International Journal for Numerical Methods in Engineering*, Vol. 58, pp. 63-81 (2003)
- U. Hetmaniuk and C. Farhat, "A Finite Element-Based Fictitious Domain Decomposition Method for the Fast Solution of Partially Axisymmetric Sound-Hard Acoustic Scattering Problems," *Finite Elements in Analysis and Design*, Vol. 39, pp. 707-725 (2003)
- C. Farhat and U. Hetmaniuk, "A Fictitious Domain Decomposition Method for the Solution of Partially Axisymmetric Acoustic Scattering Problems - Part I: Dirichlet Boundary Conditions," *International Journal for Numerical Methods in Engineering*, Vol. 54, pp. 1309-1332 (2002)

Note 1: This command should be used together with the following commands, as needed: [AXIHARB](#), [AXIHASFFP](#), [AXIHCFCNO](#), [AXIHDIR](#), [AXIHFFP](#), [AXIHLMP](#), [AXIHNEUFD](#), [AXIHNEURD](#), [AXIHNOBAS](#), [AXIHWN](#), and [DECOMPOSE](#).

Note 2: This command is currently supported by the following two-dimensional, axisymmetric element types (see [TOPOLOGY](#)): type 59 (6-node triangle), type 60 (4-node quadrilateral), type 61 (3-node triangle), and type 62 (8-node quadrilateral).

Note 3: This command is currently supported only by the FETI solver (see [STATICS](#)), with the following restrictions:

- For local and coarse solvers: only `sparse` and `skyline` may be requested.
- For the interface equation solver: only `gcr` and `agmres` may be requested.
- For the solution of multiple systems: acceleration is available only for systems with multiple right sides arising from multiple planar wave incident directions, using for `gcr` the pair of sub-command keyword and corresponding numerical value `MAXORTH maxorth_v`, and for `agmres` the pair of sub-command keyword and corresponding numerical value `DEFLECTION_size deflation_size_v`.

Note 4: In the case of a partially axisymmetric scatterer and Neumann (sound-hard) boundary conditions, the user must:

- Specify, in addition to the two-dimensional mesh associated with the computational domain of the axisymmetric component of the scatterer, a three-dimensional mesh discretizing the fictitious computational domain associated with its non-axisymmetric component.
- Use the command [AXIHNEUFD](#) to specify the nodal Neumann (sound-hard) boundary conditions and the command [AXIHCFCNO](#) to specify the set of constrained fictitious nodes.
- Use a special mesh decomposition where:
 - In the header of the decomposition file, the number of subdomains is set to `-(NSUB_AXI + 1)`, where `NSUB_AXI` is the number of subdomains into which the two-dimensional mesh associated with the computational domain of the axisymmetric component of the scatter is decomposed.
 - All elements discretizing the fictitious domain are placed in the last subdomain.

Note 5: When one or more planar wave incident directions is/are specified using this command, the directions specified using the commands [AXIHDIR](#), [AXIHLMP](#), and/or [AXIHNEURD](#) are ignored.

The input format of this command is given below.

AXIHELMHOLTZ
PLANEWAVE n_d
$\overline{d_x}$ d_y d_z
.
.
.

d_x	d_y	d_z
NMODES	nmodes_v	
NTSLICES	ntslices_v	

PLANEWAVE	Sub-command keyword to specify one or more incident planar waves (characters).
n_d	Number of incident planar waves (integer).
$\overline{d_x}$	Normalized x component of the planar wave incident direction (real).
d_y	Normalized y component of the planar wave incident direction (real).
d_z	Normalized z component of the planar wave incident direction (real).
NMODES nmodes_v	This pair of sub-command keyword (characters) and corresponding numerical value (integer) specifies the number of Fourier modes to be used in the expansion of the solution in the two-dimensional computational domain associated with the axisymmetric component of the scatterer.
NTSLICES ntslices_v	This pair of sub-command keyword (characters) and corresponding numerical value (integer) is relevant only for postprocessing the three-dimensional solution. It specifies the number of angular (θ) slices ($\theta \in [0, 2\pi]$) of the overall computational domain where to postprocess the solution at the locations obtained by rotating the two-dimensional mesh associated with the axisymmetric component of the scatterer (integer).

Next: [AXIHARB](#), Previous: [AXIHELMHOLTZ](#)

52 HELMHOLTZ ARTIFICIAL ABSORBING BOUNDARY CONDITIONS *S*

Command Statement:	HARB
--------------------	-------------

The HARB command statement is used to specify the artificial boundary Σ on which the absorbing condition specified in the HELMHOLTZ command (see [HELMHOLTZ](#)) is to be applied. The input format is given below.

Note 1: In frequency-domain computations, the absorbing condition is applied in general to the solution variable. Because it does not make sense to absorb the incident field, **AERO-S** works in this case with the scattered field as the solution variable.

Note 2: The BGTL (Bayliss-Gunzberger-Turkel-Like) absorbing boundary condition (see [HELMHOLTZ](#)) of order 0 can be applied to any face type described below. The BGTL of order 1 or 2 can be applied only to the face types 1, 2, 3, and 6 described below.

Note 3: When using a BGTL absorbing boundary condition, the face types chosen for constructing the artificial boundary Σ must be compatible with the faces of the Helmholtz elements (see [TOPOLOGY](#)) they overlay (see summary table given below).

FACE_TYPE	Problem Dimension	BGTL Order	ELEMENT_TYPE
1	2D	0,1,2	30,31,33,34,35,36
2	2D	0,1,2	32,38
3	3D	0,1,2	40,41
4	3D	0	44,45
6	3D	0,1,2	42
10	3D	0	95
11	3D	0	96
12	2D	0	98
13	2D	0	99

HARB

FACE#	FACE TYPE	CONNECTIVITY_NODES
-------	-----------	--------------------

FACE#	Face (or edge in two dimensions) id number whose type and connectivity are to be specified (integer). In practice, this id number is ignored by AERO-S .
FACE_TYPE	
1	2-node line segment. To be used with two-dimensional linear elements.
2	3-node line segment. To be used with two-dimensional quadratic elements.
3	3-node triangular face. To be used with three-dimensional linear tetrahedral element.

4	4-node quad face. To be used with three-dimensional linear hexahedral element.
6	6-node triangular face. To be used with three-dimensional quadratic tetrahedral element.
10	4, 9, 16 or 25-node quadrilateral face. To be used with three-dimensional full isoparametric hexahedral elements.
11	3, 6, 10 or 15-node triangular face. To be used with three-dimensional full isoparametric tetrahedral elements.
12	2, 3, 4 or 5-node line segment. To be used with two-dimensional full isoparametric quadrilateral elements.
13	2, 3 or 4-node line segment. To be used with two-dimensional full isoparametric triangular elements.
CONNECTIVITY_NODES	These should be listed in a stacked fashion on a single line, and numbered clockwise (when looking from outside in three dimensions).

Next: [AXIHASFFP](#) Previous: [HARB](#)

53 HELMHOLTZ ARTIFICIAL ABSORBING BOUNDARY CONDITIONS (AXISYMMETRIC/PARTIALLY AXISYMMETRIC) *S*

Command Statement: **AXIHARB**

The command **AXIHARB** is relevant only to the frequency-domain acoustic (Helmholtz) analysis of an axisymmetric or partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)), where it can be used to specify an axisymmetric BGTL (Bayliss-Gunzberger-Turkel-Like) absorbing boundary condition on the chosen artificial boundary of the two-dimensional mesh associated with the axisymmetric component of the scatterer. Its input format is given below.

```
AXIHARB
BGTL
SURFR SURFZ
NODE1# NODE2# [NODE3#]
...
...
NODE1# NODE2# [NODE3#]
```

BGTL Specifies the order (0, 1, or 2) of the generalized Bayliss-Gunzburger-Turkel non reflecting condition as well as the shape of the artificial boundary Σ on which this condition is applied (integer). Note that the shape of Σ must be consistent with that implied by the list of nodes inputted below. The following values of this parameter are supported:

- 0: basic Sommerfeld condition.
- -1: first-order BGTL condition applied on an artificial boundary Σ corresponding to the projection on the two-dimensional mesh associated with the axisymmetric component of the scatterer of a cylinder of revolution parameterized in three parts by:
 - $r(t) = \{t \times \text{SURFR}\}, \{\text{SURFR}\}, \{(1-t) \times \text{SURFR}\}$
 - $z(t) = \{-\text{SURFZ}\}, \{-\text{SURFZ}\}, \{-2\tau \times \text{SURFZ}\}, \{\text{SURFZ}\}$

On the two-dimensional mesh, a corner of Σ is replaced by a circle and the two edges connecting at a corner are assumed to have the same size.

- 1: first-order BGTL condition applied on an artificial boundary Σ corresponding to the projection on the two-dimensional mesh associated with the axisymmetric component of the scatterer of an ellipsoid of revolution parameterized by:

$$\circ r(t) = \text{SURFR} \times \cos(t), (\text{SURFR} > 0.0)$$

$$\circ z(t) = -\text{SURFZ} \times \sin(t), (\text{SURFZ} > 0.0)$$

- -2: second-order BGTL condition applied on an artificial boundary Σ corresponding to the projection on the two-dimensional mesh associated with the axisymmetric component of the scatterer of a cylinder of revolution parameterized in three parts by:

$$\circ r(t) = \{t \times \text{SURFR}\}, \{\text{SURFR}\}, \{(1-t) \times \text{SURFR}\}$$

$$\circ z(t) = \{-\text{SURFZ}\}, \{-\text{SURFZ}\}, \{-2\tau \times \text{SURFZ}\}, \{\text{SURFZ}\}$$

On the two-dimensional mesh, a corner of Σ is replaced by a circle and the two edges connecting at a corner are assumed to have the same size.

- 2: second-order BGTL condition applied on an artificial boundary Σ corresponding to the projection on the two-dimensional mesh associated with the axisymmetric component of the scatterer of an ellipsoid of revolution parameterized by:

$$\circ r(t) = \text{SURFR} \times \cos(t), (\text{SURFR} > 0.0)$$

$$\circ z(t) = -\text{SURFZ} \times \sin(t), (\text{SURFZ} > 0.0)$$

SURFR	Real-valued coefficient contributing to the parameterization of the artificial boundary Σ (real). See above.
SURFZ	Real-valued coefficient contributing to the parameterization of the artificial boundary Σ y (real). See above.
NODE1#	ID number of the first node of a line element of the two-dimensional mesh associated with the axisymmetric component of the scatterer, contributing to the geometrical definition of the artificial boundary Σ (integer).
NODE2#	ID number of the second node of the same line element of the two-dimensional mesh associated with the axisymmetric component of the scatterer, contributing to the geometrical definition of the artificial boundary Σ (integer).
NODE3#	ID number of the optional third node of the same line element of the two-dimensional mesh associated with the axisymmetric component of the scatterer, contributing to the geometrical definition of the artificial boundary Σ (integer).

Next: [AXIHCFCNO](#), Previous: [AXIHARB](#)

54 HELMHOLTZ AUXILIARY SURFACE FAR-FIELD PATTERN (AXISYMMETRIC/PARTIALLY AXISYMMETRIC) *S*

Command Statement: **AXIHASFFF**

The command **AXIHASFFF** is relevant only to the frequency-domain acoustic (Helmholtz) analysis of an axisymmetric or partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)). It can be used to describe, using line elements that are edges of the two-dimensional mesh associated with the axisymmetric component of the scatterer, a *closed* auxiliary surface that can be used to compute the far-field pattern of the solution: this makes this command (and the related command [AXIHNOBAS](#)) necessary for computing (correctly) the far-field pattern (see [AXIHFPP](#)). The closed auxiliary surface should not intersect any component of the scatterer.

Note 1: Currently, the description by line elements of the closed auxiliary surface must refer to nodes of the two-dimensional mesh associated with the axisymmetric component of the scatterer defined in [NODES](#).

The input format of this command is given below.

```
AXIHASFFF
LINE_ELEMENT# LINE_ELEMENT_TYPE SC CONNECTIVITY_NODES
...
...
LINE_ELEMENT# LINE_ELEMENT_TYPE SC CONNECTIVITY_NODES
```

LINE_ELEMENT#	ID number of a line element corresponding to an edge of the two-dimensional mesh and contributing to the description of the auxiliary surface defined above (integer). The type and connectivity of this edge should be specified as described below (integer). Currently, this parameter is ignored by AERO-S .
LINE_ELEMENT_TYPE	Line element type (integer).
11	2-node line segment, to be used with two-dimensional axisymmetric linear elements.
12	3-node line segment, to be used with two-dimensional axisymmetric quadratic elements.
SC	Sommerfeld constant (real). Currently this value is ignored by AERO-S .
CONNECTIVITY_NODES	List of node ID numbers describing the connectivity of a line segment (integers).

Next: [HDIR](#), Previous: [AXIHASFFF](#)

55 HELMHOLTZ CONSTRAINED FICTITIOUS NODES (PARTIALLY AXISYMMETRIC) *S*

Command Statement: **AXIHCFCNO**

The command **AXIHCFCNO** is relevant only to the frequency-domain acoustic (Helmholtz) analysis of a partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)) with Neumann (sound-hard) boundary conditions. It can be used to identify the list of nodes in the fictitious domain located on the intersection between the axisymmetric and non-axisymmetric surfaces of the scatterer, where [AXIHNEUFD](#) can then compute the appropriate source term. Its input format is given below.

```
AXIHCFCNO
NODE#
...
...
NODE#
```

NODE# ID number of a node of the three-dimensional mesh located on the intersection between the axisymmetric and non-axisymmetric surfaces of the scatterer (integer).

Next: [AXIHDIR](#), Previous: [AXIHCFCNO](#)

56 HELMHOLTZ DIRICHLET BOUNDARY CONDITIONS *S*

Command Statement: **HDIR**

The HDIR command statement is used to specify *nodal* Dirichlet boundary conditions for a frequency-domain acoustic scattering (Helmholtz) problem. The input format is given below.

Note 1: When the keyword PLANEWAVE or POINTSOURCE is specified under [HELMHOLTZ](#), it overwrites the REAL_PART_VALUE and IMAGINARY_PART_VALUE (see below) by $p^s = -p^i$, where p^i is the amplitude of an incident time-harmonic plane wave specified by PLANEWAVE, or an that of an incident time-harmonic spherical wave propagating from a source located at a point specified by POINTSOURCE.

HDIR

NODE#	DOF#	REAL_PART_VALUE	IMAGINARY_PART_VALUE
-------	------	-----------------	----------------------

NODE#	Node number where the Dirichlet boundary condition is specified (integer).
DOF#	Degree of freedom local number where the boundary condition is specified (integer). This number should be set to 8.
REAL_PART_VALUE	Real part of the value of the specified boundary condition (float).
IMAGINARY_PART_VALUE	Imaginary part of the value of the specified boundary condition (float).

Next: [HDNB](#), Previous: [HDIR](#)

57 HELMHOLTZ DIRICHLET BOUNDARY CONDITIONS (AXISYMMETRIC/PARTIALLY AXISYMMETRIC) *S*

Command Statement: **AXIHDIR**

The command AXIHDIR is relevant only to the frequency-domain acoustic (Helmholtz) analysis of an axisymmetric or partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)). It can be used to specify *nodal* Dirichlet (sound-soft) boundary conditions on the two-dimensional mesh associated with the axisymmetric component of the scatter of the form $p^s = p^i$, where p^s is the scattered wave, $p^i = Ce^{jkX \cdot d}$ is the amplitude of the incident time-harmonic planar wave specified in [AXIHELMHOLTZ](#) or below (see also [HELMHOLTZ](#)), j is the pure imaginary number satisfying $j^2 = -1$, k is the wavenumber specified in [AXIHWN](#), $X \in R^3$ is the vector position of a point in space, d is the normalized direction of the incident planar wave, and C is in general a complex-valued constant.

Note 1: When a planar wave incident direction is specified in [AXIHELMHOLTZ](#), the direction specified in this command is ignored.

Note 2: In this context, the classical Dirichlet (sound-soft) boundary condition corresponds to $C = -1.0$.

The input format of this command is given below.

AXIHDIR

CR	CI
d_x	d_y d_z
NODE#	
...	
...	
NODE#	

CR	Real part of the constant C of the Dirichlet (sound-soft) boundary condition $p^s = Ce^{jkX.d}$ (real).
CI	Imaginary part of the constant C of the Dirichlet (sound-soft) boundary condition $p^s = Ce^{jkX.d}$ (real).
d_x	Normalized x component of the planar wave incident direction (real).
d_y	Normalized y component of the planar wave incident direction (real).
d_z	Normalized z component of the planar wave incident direction (real).
NODE#	ID number of a node of the two-dimensional mesh where the nodal Dirichlet (sound-soft) boundary condition is specified (integer).

Next: [AXIHFFP](#), Previous: [AXIHDIR](#)

58 HELMHOLTZ DISTRIBUTED NEUMANN BOUNDARY CONDITION *S*

Command Statement: **HDNB**

The **HDNB** command statement can be used to specify on all or segments of the surface of a scatterer a *distributed* Neumann boundary condition of the form $\frac{\partial p^s}{\partial n} = -\frac{\partial p^i}{\partial n}$, where the superscripts s and i designate scattered and incident quantities, respectively. The incident

time-harmonic wave can be a time-harmonic plane wave specified by **PLANEWAVE**, or a time-harmonic spherical wave propagating from a source located at a point specified by **POINTSOURCE** in **HELMHOLTZ**. Note that for problems involving a nonhomogenous fluid, **HDNB** can only be used to define a surface that is submerged entirely in one and only one fluid.

The input format of this command is given below.

HDNB

FACE#	FACE_TYPE	CONNECTIVITY_NODES
-------	-----------	--------------------

FACE#	Face (or edge in two dimensions) id number whose type and connectivity are to be specified (integer). In practice, this id number is ignored by FEM.	
FACE_TYPE		
1	2-node line segment. To be used with two-dimensional linear elements.	
2	3-node line segment. To be used with two-dimensional quadratic elements.	
3	3-node triangular face. To be used with three-dimensional linear tetrahedral elements.	
4	4-node quad face. To be used with three-dimensional linear hexahedral elements.	
6	6-node triangular face. To be used with three-dimensional quadratic tetrahedral elements.	
10	4, 9, 16 or 25-node quadrilateral face. To be used with three-dimensional full isoparametric hexahedral elements.	
11	3, 6, 10 or 15-node triangular face. To be used with three-dimensional full isoparametric tetrahedral elements.	
12	2, 3, 4 or 5-node line segment. To be used with two-dimensional full isoparametric quadrilateral elements.	
13	2, 3 or 4-node line segment. To be used with two-dimensional full isoparametric triangular elements.	
CONNECTIVITY_NODES	These should be listed in a stacked fashion on a single line, and numbered counter clockwise (when looking from infinity in three dimensions).	

Next: [HLMPC](#), Previous: [HDNB](#)

59 HELMHOLTZ FAR-FIELD PATTERN (AXISYMMETRIC/PARTIALLY AXISYMMETRIC) *S*

Command Statement: **AXIHFP** **INCREMENT**

The command **AXIHFP** is relevant only to the frequency-domain acoustic (Helmholtz) analysis of an axisymmetric or partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)), where it can be used to activate the outputting to a file of the far-field pattern of the acoustic pressure field. However: this field is computed only if the user also specifies in the **AERO-S** ASCII Input Command Data file the command [AXIHASFFF](#); and it is correctly computed only if the user also specifies in the aforementioned file the command [AXIHNOBAS](#).

The output file will have the prefix specified in **OUTPUT** for the result **GHELMHOL** and the extension ".ffp". It will contain on each line four values " α ", " β ", real-part(ffp), imaginary-part(ffp)". The first two quantities are the spherical angles (expressed in radians) that determine the direction $d = [\cos\beta * \cos\alpha, \cos\beta * \sin\alpha, \sin\beta]$. The number of output lines depends on the parameter **INCREMENT** (see below).

The input format of this command is given below.

AXIHFFP	INCREMENT
---------	-----------

INCREMENT Specifies the number of longitudinal directions where to evaluate the far-field pattern: (**INCREMENT**/2)+1) latitudinal directions are also considered and therefore the far-field pattern is evaluated at ((**INCREMENT**/2)+1)***INCREMENT** points uniformly distributed in spherical coordinates.

Next: [AXIHLMP](#), Previous: [AXIHFFP](#)

60 HELMHOLTZ LINEAR MULTIPoint CONSTRAINTS

Command Statement:	HLMP
--------------------	-------------

The **HLMP** command statement is used to specify a set of *complex* linear multipoint constraint equations of the form

$$\sum_{j=1}^{j=n} c_{ij} u_j = r_i, \quad i = 1, 2, \dots$$

in a frequency-domain acoustic model. There is no limitation on the number of multipoint constraints, or number of degrees of freedom related by the same constraint equation. The format of this command statement is as follows.

Note 1: The Lagrange multiplier method for enforcing the constraints associated with this command is supported only by the FETI-DP family of solvers, the GMRES solver, and the SPOOLES and MUMPS direct sparse solvers (see [STATICS](#)).

HLMP

CONSTRAINT#	RHS	CONSTRAINT_METHOD
NODE#	DOF#	REAL-PART-COEFF IMAGINARY-PART-COEFF
.		
NODE#	DOF#	REAL-PART-COEFF IMAGINARY-PART-COEFF

CONSTRAINT#	This corresponds to the constraint equation number <i>i</i> (integer).	
RHS	This is the right-hand side r_i of the <i>i</i> -th constraint equation (float). It can have both a real and an imaginary part, in which case the real part is specified first and followed by the imaginary part.	
CONSTRAINT_METHOD	This is the method for enforcing the constraint (characters). The default method is set in CONSTRAINTS and used whenever this entry is omitted.	
multipliers	The Lagrange multiplier method.	
elimination	The elimination method.	
[beta]	penalty	The penalty method. The parameter <i>beta</i> should be a large positive number, typically of the order of 10^8 (no default value is provided).
NODE#	DOF#	This is the number of the node contributing the coefficient c_{ij} of the <i>i</i> -th constraint equation (integer).
DOF#		This is the local number of the degree of freedom at the node specified above contributing the coefficient c_{ij} of the <i>i</i> -th constraint equation (integer).
REAL_PART_VALUE		This is the real part of the coefficient c_{ij} of the <i>i</i> -th constraint equation (float).
IMAGINARY_PART_VALUE		This is the imaginary part of the coefficient c_{ij} of the <i>i</i> -th constraint equation (float).

Next: [KIRLOC](#), Previous: [HLMP](#)

61 HELMHOLTZ LINEAR MULTIPPoint CONSTRAINTS (PARTIALLY AXISYMMETRIC) *S*

Command Statement:	AXIHLMP
--------------------	----------------

The command **AXIHLMP** is relevant only to the frequency-domain acoustic (Helmholtz) analysis of a partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)). It specifies the (weak) enforcing of a set of complex-valued linear multipoint constraint equations corresponding to:

- $p^s = Ce^{jkX.d}$ (see [AXIHDIR](#)) at the "rotated" nodes of the two-dimensional mesh associated with the axisymmetric component of the scatterer that lie on the external surface of the non-axisymmetric component, when the analysis is performed with Dirichlet (sound-soft) boundary conditions.
- The continuity of the solution across the interface between the three-dimensional mesh associated with the non-axisymmetric component of the scatterer and the "rotated" nodes of the two-dimensional mesh associated with its axisymmetric component, when the analysis is performed with Neumann (sound-hard) boundary conditions.

Currently, the axisymmetric and non-axisymmetric components of the scatterer are assumed to have matching discrete interfaces, which simplifies the expression, input, and processing of these constraints.

Note 1: When a planar wave incident direction is specified in [AXIHELMHOLTZ](#), the direction specified in this command is ignored.

Note 2: In this context, the classical Dirichlet (sound-soft) boundary condition corresponds to $C = -1.0$.

The input format of this command is given below.

```
AXIHLMPC
CR   CI
dx  dy  dz
NODE# ANGLE1 ANGLE2 NWEDGES
...
...
NODE# ANGLE1 ANGLE2 NWEDGES
```

CR The setting of this real-valued parameter (real) depends on the boundary conditions of the analysis:

- In the case of Dirichlet (sound-soft) boundary conditions, **cr** is the real part of the constant C in the boundary condition $p^s = Ce^{jkX.d}$.
- In the case of Neumann (sound-hard) boundary conditions, **cr** should be set to 0.

CI The setting of this real-valued parameter (real) depends on the boundary conditions of the analysis:

- In the case of Dirichlet (sound-soft) boundary conditions, **ci** is the imaginary part of the constant C in the boundary condition $p^s = Ce^{jkX.d}$ (real).
- In the case of Neumann (sound-hard) boundary conditions, **ci** should be set to 0.

d_x Normalized **x** component of the planar wave incident direction (real).

d_y Normalized **y** component of the planar wave incident direction (real).

d_z Normalized **z** component of the planar wave incident direction (real).

NODE# ID number of a node of the two-dimensional mesh associated with the axisymmetric component of the scatterer, where any of the constraints outlined above is to be enforced (integer).

ANGLE1 Lower bound θ_1 of the angular interval $[\theta_1, \theta_2]$ in which the two-dimensional mesh associated with the axisymmetric component of the scatterer has to be rotated to match the three-dimensional mesh associated with the non-axisymmetric component of the scatterer at the interface between both components (real).

ANGLE2 Upper bound θ_2 of the angular interval $[\theta_1, \theta_2]$ in which the two-dimensional mesh associated with the axisymmetric component of the scatterer has to be rotated to match the three-dimensional mesh associated with the non-axisymmetric component of the scatterer at the interface between both components (real).

NWEDGES Number of uniform wedges in the interval $[\theta_1, \theta_2]$ (integer) that:

- In the case of Dirichlet (sound-soft) boundary conditions, generate the locations of the rotated nodes of the two-dimensional mesh associated with the axisymmetric component of the scatterer, where to apply the linear multipoint constraints.
- In the case of Neumann (sound-hard) boundary conditions, recover the locations of the nodes of the three-dimensional mesh associated with the non-axisymmetric component of the scatterer.

62 HELMHOLTZ LOCATIONS WHERE TO COMPUTE THE KIRCHHOFF INTEGRAL

Command Statement: **KIRLOC**

When **KIRCHHOFF** is selected as an [OUTPUT](#) result and the surface of the scatterer is defined using the command [HSCB](#), this command should be used to specify the locations of the points where the (far-field) solution of a frequency-domain acoustic or elastoacoustic (Helmholtz) problem is to be evaluated using the Kirchhoff integral and outputted. These points do not need to be nodes of the mesh. Usually, but not necessarily, they are outside the computational domain, in the far-field. The keyword **KIRLOC** should be followed by as many lines as there are points where it is desired to evaluate the solution of the aforementioned Helmholtz problem. Each line contains the coordinates of such a point.

The syntax of this command is given below.

KIRLOC

X-ORDINATE	Y-ORDINATE	Z-ORDINATE
------------	------------	------------

X-ORDINATE	X-ordinate of a point where to evaluate the solution of the Helmholtz problem to be solved using the Kirchhoff integral (real).
Y-ORDINATE	Y-ordinate of a point where to evaluate the solution of the Helmholtz problem to be solved using the Kirchhoff integral (real).
Z-ORDINATE	Z-ordinate of a point where to evaluate the solution of the Helmholtz problem to be solved using the Kirchhoff integral (real).

Next: [AXIHNEUFD](#), Previous: [KIRLOC](#)

63 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS *S*

Command Statement: **HNEU [LOADSET_ID]**

The **HNEU** command statement is used to specify the *nodal* Neumann boundary conditions for a frequency-domain acoustic scattering (Helmholtz) problem. The input format is given below.

HNEU [LOADSET_ID]

NODE#	REAL_PART_VALUE	IMAGINARY_PART_VALUE
-------	-----------------	----------------------

LOADSET_ID	Optional non-negative integer which identifies explicitly the "load" set to which the source term generated by this command belongs to (integer). The default value is 0. Hence, the HNEU command can be repeated as many times as desired within the same input file using each time a different value for LOADSET_ID and different data. The LOADCASE command can refer to LOADSET_ID to define one or multiple "load" cases for static analysis (see the STATICS command and the explanation of its sub-command CASES), and/or the "load" case for dynamic analysis.
NODE#	Node number where the Neumann boundary condition is specified (integer).
REAL_PART_VALUE	Real part of the value of the specified boundary condition (float).
IMAGINARY_PART_VALUE	Imaginary part of the value of the specified boundary condition (float).

Next: [AXIHNEURD](#), Previous: [HNEU](#)

64 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS - FICTITIOUS DOMAIN (PARTIALLY AXISYMMETRIC) *S*

Command Statement: **AXIHNEUFD**

The command **AXIHNEUFD** is only relevant to the frequency-domain acoustic (Helmholtz) analysis of a partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)) with Neumann (sound-hard) boundary conditions. It can be used to activate the computation of the source term acting on the external surfaces of the non-axisymmetric component of the scatterer and that acting on the intersection between its axisymmetric and non-axisymmetric surfaces. Its input format is given below.

AXIHNEUFD

ELEMENT_TYPE	INT	NYSLICES	Y1	Y2	CONNECTIVITY_NODES
--------------	-----	----------	----	----	--------------------

EL_TYPE		Line element or surface element type (integer).			
4		4-node quadrilateral element, to be used with three-dimensional linear hexahedral elements, to compute the source term acting on the intersection between the axisymmetric and non-axisymmetric surfaces of the scatterer.			
8		8-node quadrilateral, to be used with three-dimensional quadratic hexahedral elements, to compute the source term acting on the intersection between the axisymmetric and non-axisymmetric surfaces of the scatterer.			
11		2-node line segment, to be used with two-dimensional axisymmetric linear elements, to compute the source term acting on the external surfaces of the non-axisymmetric component of the scatterer.			
12		3-node line segment, to be used with two-dimensional axisymmetric quadratic elements, to compute the source term acting on the external surfaces of the non-axisymmetric component of the scatterer.			
14		4-node quadrilateral, to be used with three-dimensional linear hexahedral elements, to compute the source term acting on the external surfaces of the non-axisymmetric component of the scatterer.			
18		8-node quadrilateral, to be used with three-dimensional quadratic hexahedral elements, to compute the source term acting on the external surfaces of the non-axisymmetric component of the scatterer.			
INT		Integer number that must be set as follows:			
-1		for EL_TYPE = 4 and 8.			
0		for el_TYPE = 11, 12, 14, and 18.			
NYSLICES		Number of slices in the <i>y</i> (vertical) direction, to compute the source term acting on the external surfaces of the non-axisymmetric component of the scatterer, or that acting on the intersection between its axisymmetric and non-axisymmetric surfaces (integer).			
Y1		Lower bound <i>y</i> ₁ (real) of the interval $[y_1, y_2]$ (see NYSLICES above).			
Y2		Upper bound <i>y</i> ₂ (real) of the interval $[y_1, y_2]$ (see NYSLICES above).			
CONNECTIVITY_NODES		List of node ID numbers describing the connectivity of the line or surface element (integers).			

Next: [AXIHNOBAS](#), Previous: [AXIHNEUFD](#)

65 HELMHOLTZ NEUMANN BOUNDARY CONDITIONS - REAL DOMAIN (AXISYMMETRIC/PARTIALLY AXISYMMETRIC) *S*

Command Statement:	AXIHNEURD
--------------------	-----------

The command AXIHNEURD is relevant only to the frequency-domain acoustic (Helmholtz) analysis of an axisymmetric or partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)). It can be used to specify, in the real part of the computational domain, *nodal* boundary conditions corresponding to the Neumann (sound-hard) boundary condition of the form $\partial p^s / \partial n = \partial p^i / \partial n$, where the superscripts *s* and *i* designate scattered and incident waves, respectively, *n* denotes the normalized outward normal to the surface of the scatterer, $p^i = Ce^{jkX.d}$ is the amplitude of the incident time-harmonic planar wave specified in [AXIHELMHOLTZ](#) or below (see also [HELMHOLTZ](#)), *j* is the pure imaginary number satisfying $j^2 = -1$, *k* is the wavenumber specified in [AXIHWAN](#), $X \in R^3$ is the vector position of a point in space, *d* is the normalized direction of the incident planar wave, and *C* is in general a complex-valued constant.

Note 1: When a planar wave incident direction is specified in [AXIHELMHOLTZ](#), the direction specified in this command is ignored.

Note 2: In this context, the classical Dirichlet (sound-soft) boundary condition corresponds to $C = -1.0$.

The input format of this command is given below.

AXIHNEURD

CR CI
<i>d_x</i> <i>d_y</i> <i>d_z</i>
NODE1# NODE2# [NODE3#]
...
...
NODE1# NODE2# [NODE3#]

CR	Real part of the constant C of the specified Neumann (sound-hard) boundary condition associated with $p^i = Ce^{jkX.d}$ (real).
CI	Imaginary part of the constant C of the specified Neumann (sound-hard) boundary condition associated with $p^i = Ce^{jkX.d}$ (real).
d_x	Normalized x component of the planar wave incident direction (real).
d_y	Normalized y component of the planar wave incident direction (real).
d_z	Normalized z component of the planar wave incident direction (real).
NODE1#	ID number of the first node of an element of the two-dimensional mesh associated with the axisymmetric component of the scatterer where the Neumann (sound-hard) boundary condition is specified (integer).
NODE2#	ID number of the second node of an element of the two-dimensional mesh associated with the axisymmetric component of the scatterer where the Neumann (sound-hard) boundary condition is specified (integer).
NODE3#	ID number of the optional third node of an element of the two-dimensional mesh associated with the axisymmetric component of the scatterer where the Neumann (sound-hard) boundary condition is specified (integer).

Next: [HSCB](#), Previous: [AXIHNEURD](#)

66 HELMHOLTZ NODES BELOW AUXILIARY SURFACE (AXISYMMETRIC/PARTIALLY AXISYMMETRIC) *S*

Command Statement: **AXIHNOBAS**

The command **AXIHNOBAS** is relevant only to the frequency-domain acoustic (Helmholtz) analysis of an axisymmetric or partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)). It can be used to list all nodes of the two-dimensional mesh associated with the axisymmetric component of the scatterer that are enclosed by the auxiliary surface defined in [AXIHASFPP](#) for the purpose of computing the far-field pattern of the solution, so that these nodes can be skipped during the computation of the far-field pattern. Hence, this command is necessary for correctly computing the far-field pattern (see [AXIHFFP](#)).

The input format of this command is given below.

AXIHNOBAS
NODE#
...
...
NODE#

NODE# ID number of a node of the two-dimensional mesh associated with the axisymmetric component of the scatterer that is enclosed by the auxiliary surface defined in [AXIHASFPP](#) (integer).

Next: [AXIHWAN](#), Previous: [AXIHNOBAS](#)

67 HELMHOLTZ SCATTERER BOUNDARY *S*

Command Statement: **HSCB**

The **HSCB** command statement is used to specify the surface of a scatterer. It is used to compute the farfield pattern of a frequency-domain acoustic solution, and therefore is necessary for the output of that farfield pattern (see [OUTPUT](#)).

The input format of this command is given below.

HSCB

FACE#	FACE_TYPE	CONNECTIVITY_NODES
-------	-----------	--------------------

FACE#	Face (or edge in two dimensions) id number whose type and connectivity are to be specified (integer). In practice, this id number is ignored by FEM.	
FACE_TYPE	1	2-node line segment. To be used with two-dimensional linear elements.
	2	3-node line segment. To be used with two-dimensional quadratic elements.

3	3-node triangular face. To be used with three-dimensional linear tetrahedral elements.
4	4-node quad face. To be used with three-dimensional linear hexahedral elements.
6	6-node triangular face. To be used with three-dimensional quadratic tetrahedral elements.
10	4, 9, 16 or 25-node quadrilateral face. To be used with three-dimensional full isoparametric hexahedral elements.
11	3, 6, 10 or 15-node triangular. To be used with three-dimensional full isoparametric tetrahedral elements.
12	2, 3, 4 or 5-node line segment. To be used with two-dimensional full isoparametric quadrilateral elements.
13	2, 3 or 4-node line segment. To be used with two-dimensional full isoparametric triangular elements.
CONNECTIVITY_NODES	These should be listed in a stacked fashion on a single line, and numbered counter clockwise (when looking from outside in three dimensions).

Next: [HWIB](#), Previous: [HSCB](#)

68 HELMHOLTZ WAVE NUMBER (AXISYMMETRIC/PARTIALLY AXISYMMETRIC) *S*

Command Statement: **AXIHWAN**

The command **AXIHWAN** is relevant only to the frequency-domain acoustic (Helmholtz) analysis of an axisymmetric or partially axisymmetric scatterer (see [AXIHELMHOLTZ](#)), where it can be used to specify the wavenumber for such an analysis. Its input format is given below.

AXIHWAN

KAPPA

KAPPA Wavenumber at which the frequency-domain acoustic analysis is to be performed (real).

Next: [HEFSB](#), Previous: [AXIHWAN](#)

69 HELMHOLTZ WET INTERFACE BOUNDARY *S*

Command Statement: **HWIB**

The **HWIB** command statement can be used to describe the wet interface boundary of the structure in a coupled frequency response elastoacoustic (fluid-structure) problem where a single mesh is used to discretize both of the computational acoustic fluid and structural domains. Its input format is given below.

Note 1: This command uses the same input format as **HARB** and **HDNB** but is supported only for face types 10, 11, 12 and 13.

Note 2: In this case, the capabilities of the ACME library are not used to generate the fluid-structure coupling coefficients.

HWIB

FACE#	FACETYPE	CONNECTIVITY_NODES
-------	----------	--------------------

FACE#	Face (or edge in two dimensions) id number whose type and connectivity are to be specified (integer). In practice, this id number is ignored by FEM.	
FACETYPE		
10	4, 9, 16 or 25-node quadrilateral face (to be used appropriately with three-dimensional full isoparametric hexahedral elements).	
11	3, 6, 10 or 15-node triangular (to be used appropriately with three-dimensional full isoparametric tetrahedral elements).	
12	2, 3, 4 or 5-node line segment (to be used appropriately with two-dimensional full isoparametric quadrilateral elements).	
13	2, 3 or 4-node line segment (to be used appropriately with two-dimensional full isoparametric triangular elements).	
CONNECTIVITY_NODES	These should be listed in a stacked fashion on a single line, and numbered as shown in the following examples: 9-noded quad: 7 8 9 4 5 6 1 2 3	

6-noded triangle:
 6
 4 5
 1 2 3
 etc.

Next: [HEFRS](#), Previous: [HWIB](#)

70 HYDROELASTIC FLUID/STRUCTURE BOUNDARY

Command Statement: **HEFSB**

The **HEFSB** command is used to specify the fluid-structure interface boundary in a hydroelastic eigenvalue problem. The underlying discrete model assumes that each node on this interface is shared by fluid and structural elements. The input format of this command is given below.

Note 1: AERO-S automatically determines that a hydroelastic eigenvalue problem is to be solved once it finds in the input file the **HEFSB** command, or the [HEFRS](#) command which specifies its free-surface boundary. This problem is solved using the added mass approach. The computed frequencies and structural mode shapes can be output as in a regular structural eigen computation.

HEFSB

FACE#	FACE_TYPE	CONNECTIVITY_NODES
FACE#		Face (or edge in two dimensions) identification number whose type and connectivity are to be specified (integer).
FACE_TYPE		
1		2-node line segment (to be used appropriately with two-dimensional linear elements).
2		3-node line segment (to be used appropriately with two-dimensional quadratic elements).
3		3-node triangular face (to be used appropriately with three-dimensional linear tetrahedral and prismatic elements).
4		4-node quadrilateral face (to be used appropriately with three-dimensional linear hexahedral and prismatic elements).
6		6-node triangular face (to be used appropriately with three-dimensional quadratic tetrahedral elements).
10		Quadrilateral face (to be used appropriately with higher-order iso-parametric hexahedral elements).
11		Triangular face (to be used appropriately with higher-order iso-parametric tetrahedral elements).
12		Line segment (to be used appropriately with higher-order iso-parametric quadrilateral elements).
13		Line segment (to be used appropriately with higher-order iso-parametric triangular elements).
CONNECTIVITY_NODES		These should be listed in a stacked fashion on a single line. The nodes of low-order faces should be ordered counter-clockwise (when looking from infinity in three dimensions). Those of higher-order faces (type 10-13) should be ordered from left to right and bottom to top using any convenient axis system (when viewed from infinity in three dimensions).

Next: [WEIGHTMASS](#), Previous: [HEFSB](#)

71 HYDROELASTIC FREE SURFACE BOUNDARY

Command Statement: **HEFRS**

The **HEFRS** command can be used to specify the free surface boundary either in a hydroelastic eigenvalue problem, or in an acoustic or elastoacoustic (fluid-structure interaction) problem where the fluid is assumed to be weightless. At each node located on this free surface, AERO-S automatically applies a zero Dirichlet boundary condition for the fluid potential equation in the first case, and for the scattered pressure in the second case. For this reason, this command can be also used to apply these specific Dirichlet boundary conditions wherever desired in the fluid mesh.

The input format of this command is given below.

Note 1: AERO-S automatically determines that a hydroelastic eigenvalue problem is to be solved once it finds in the input file the **HEFRS** command, or the [HEFSB](#) command which specifies its hydroelastic fluid/structure interface boundary. This problem is solved using the added mass approach. The computed frequencies and structural mode shapes can be output as in a regular structural eigen computation.

HEFRS

FACE#	FACE_TYPE	CONNECTIVITY_NODES
FACE#		Face (or edge in two dimensions) identification number whose type and connectivity are to be specified (integer).
FACE_TYPE		Currently, this entry is ignored by AERO-S . Still, it must be assigned a dummy integer value (integer).
CONNECTIVITY_NODES		These should be listed in a stacked fashion on a single line.

Next: [IMPEDANCE](#), Previous: [HEFRS](#)

72 HYPER REDUCTION OF THE INERTIA FORCES ASSOCIATED WITH A CONSTANT MASS MATRIX

Command Statement:	WEIGHTMASS
--------------------	------------

The command **WEIGHTMASS** can be used to request during an online computation the hyper reduction of the inertia forces associated with a constant mass matrix using an inputted reduced mesh and associated weights, computed using the command [RMSHC](#) and the hyper reduction method ECSW.

Next: [ACCELERATIONS](#), Previous: [WEIGHTMASS](#)

73 IMPEDANCE ANALYSIS

Command Statement:	IMPEDANCE	[SWEEP_ID]
--------------------	-----------	------------

The **IMPEDANCE** command can be used for two different purposes:

1. Signal that the problem to be solved is a forced frequency response vibro-acoustic (aka elastoacoustic and fluid-structure), structural dynamic, or acoustic problem possibly formulated in an f or $f \times \varphi$ parameter domain and of the form

$$Z((2\pi f_j))q_{jk} = r((2\pi f_j), \varphi_k)$$

or more specifically,

$$\begin{bmatrix} K_s - i(2\pi f_j)D_s - (2\pi f_j)^2 M_s \\ (2\pi f_j)^2 C \end{bmatrix} \begin{bmatrix} u_s \\ p_f \end{bmatrix} = \begin{bmatrix} g_s((2\pi f_j), \varphi_k) \\ \frac{1}{\rho_f} g_f((2\pi f_j), \varphi_k) \end{bmatrix}, \quad j = 1, 2, \dots, \quad k = 1, 2, \dots$$

and specify a fast solution algorithm for this problem. In the above block matrix equation, i is the pure imaginary number satisfying $i^2 = -1$, f_j is a frequency whose corresponding *circular* frequency is denoted by ω_j and related to it by $\omega_j = 2\pi f_j$, $\varphi \in [\varphi_l, \varphi_u]$ is either a load case (see [LOADCASE](#)) or an angle of incidence (see [PLANEWAVE](#) in [HELMHOLTZ](#)), K_s , M_s and D_s are the structural stiffness, mass, and damping matrices, respectively, K_f and M_f are the acoustic fluid stiffness and mass matrices arising from the discretization of the Helmholtz operator, respectively, the matrix S_a arises when the formulation of the acoustic subproblem includes an absorbing boundary condition, ρ_f is the acoustic fluid density specified in [MATERIAL](#), c_f is the speed of sound in the acoustic fluid and is also specified in [MATERIAL](#), C is a fluid-structure coupling matrix, u_s , p_f , g_s , and g_f denote the amplitude vectors of the time-harmonic structural displacement ($u_s e^{-i\omega_j t}$), pressure fluctuation field ($u_f e^{-i\omega_j t}$), structural external forcing input ($g_s e^{-i\omega_j t}$), and acoustic fluid external forcing input ($g_f e^{-i\omega_j t}$), respectively, and the superscript T designates the transpose operation.

Specifically, this command can be used to perform: (1) a single frequency/single source analysis, (2) a single frequency/multiple sources analysis, (3) a frequency sweep/single source analysis, and (4) a frequency sweep/multiple source analysis, where a source refers here to a source term (or right hand-side) associated with a [LOADCASE](#) or an angle of incidence generated using [PLANEWAVE](#) in [HELMHOLTZ](#).

2. To identify within a specified range of interest the eigenvalues missed by a previous eigenvalue computation and their corresponding eigenvectors.

For the purpose of frequency response analysis, structural damping can be represented in **AERO-S** using two different approaches:

- The Rayleigh proportional damping, in which case the matrix D_s takes the form

$$D_s = aK_s + bM_s$$

where a and b are two real scalars that may be specified in this **IMPEDANCE** command for the entire structure, and/or in the [MATERIAL](#) command at the material and therefore element level (see the sub-command keyword [DAMPING_TYPE](#) in [MATERIAL](#)).

- The concept of loss factor introduced in the Young modulus of a given material — and thererore at the element level of a finite element model — as follows

$$E^{e*} = E^e(1 - i\eta_E)$$

where E^e denotes the element-level Young modulus, the star superscript designates the "damped" Young modulus, i denotes the pure

imaginary number satisfying $i^2 = -1$, the minus sign in $-i$ is due to the convention $u_s e^{-i\omega_j t}$, and η_E is the Young modulus loss factor. In general, η_E is a function of the frequency. This function may be specified in the [MATERIAL](#) command (see the sub-command keyword [DAMPING_TYPE](#) in [MATERIAL](#)), or more generally in the form of one or multiple lookup tables (or curves) in [SDETAFT](#).

If the physical domain of the acoustic fluid is unbounded but its computational support is truncated by an artificial boundary surface, S_a is a complex sparse matrix that arises from the discretization on this surface of the absorbing boundary specified in [HELMHOLTZ](#). Alternatively, if the computational support of an infinite acoustic fluid domain is truncated using a perfectly matched layer (PML) whose properties are specified in [MATERIAL](#), S_a is a complex, non-Hermitian matrix with non-zero entries only at the degrees of freedom inside the PML. If the parameters of the PML are kept constant when the frequency is varied, or the physical acoustic fluid domain is bounded and delimited by boundary surfaces of the structural domain, the above block matrix equation can be re-written in terms of complex non-Hermitian matrices K_f and M_f so that $S_a = 0$.

The coupling matrix C is automatically constructed by **AERO-S** when the [HWIB](#) or [FSINTERFACE](#) command is present in the input file, and set to zero otherwise. The amplitude vector of the time-harmonic structural external forcing input, g_s , can be modeled using either the [FORCES](#) or [PRESSURE](#) command, as appropriate. The amplitude vector of the time-harmonic acoustic fluid external forcing input, g_f , can be modeled using the [HNEU](#) command, or a combination of the [HELMHOLTZ](#), [HDIR](#), and [HDNB](#) commands, as appropriate. The case of a purely acoustic problem is governed by the second row of the above block matrix equation after setting $u_s = 0$. That of a purely structural dynamic problem is given by the first row of the above block matrix equation after setting $p_f = 0$.

Note 1: When structural damping is modeled using the concept of the Young modulus loss factor η_E or the rubber material damping approach, the variations of E , η_E , μ , and η_μ with the frequency f are described using curves (or lookup tables) that are defined in [SDETAFT](#) or [RUBDAFT](#), and the frequency sweep analysis is to be performed using a fast reconstruction algorithm, special attention should be paid to ensure that:

- In the frequency band of interest — that is, in the frequency band where a sweep is requested — E , η_E , μ , and η_μ are defined, as needed.
- The frequency band of interest defined as above does not contain within it, except perhaps at its end points, any of the frequency points specified in [SDETAFT](#) or [RUBDAFT](#) to define a curve. This is because the fast frequency response algorithms implemented in **AERO-S** assume that E , η_E , μ , and η_μ are affine functions of f within the frequency band where the sweep is to be performed.

Note 2: Because of the limitation noted in the second bullet above, the [IMPEDANCE](#) command can be repeated multiple times in the same input file with different parameters to enable a convenient approach for performing a frequency sweep analysis in a large frequency band where E , η_E , μ , or η_μ is not an affine function of the frequency f , but can be well approximated by a piece-wise linear function of f : in this case, the frequency band of interest can be partitioned into multiple contiguous frequency bands within each E , η_E , μ , and η_μ can be assumed to be affine functions of f , and a separate [IMPEDANCE](#) command can be designed and included in the same input file for performing a frequency sweep analysis in each frequency band.

Note 3: The brute force approach for performing a frequency sweep analysis consists of rebuilding and solving the problem $Z((2\pi f_j)) q_{jk} = r((2\pi f_j), \varphi_k)$ for each sampled parameter point (f_j, φ_k) . It is obtained by choosing Taylor 0 for RECONS.

The input format of this command is given below.

IMPEDANCE	
FREQ	f
RAYDAMP	a b
FREQSWEEP1	f_0 Δf $n_{\Delta f}$
FREQSWEEP	f_l f_u n_s^* n_s
FREQSWEEP	f_l f_j^* n_j ⋮

FREQSWEPA	f_l	f_u	n_s	IMOR	$[\epsilon \ n_s^{*max} \ n_v^{min} \ n_v^{max} \ \Delta n_v]$
FREQSWEPAW	f_l	f_u	n_s	IMOR	$[\epsilon \ localflag \ [n_v] \ tol \ tolref]$
RECONS	alg para_1	para_2	para_3		
PADEPOLES	σ_l	σ_u			

SWEEP_ID Optional non-negative integer which identifies explicitly a sweep case (integer). The default value is 0. Hence, the **IMPEDANCE** command can be repeated as many times as desired within the same input file using each time a different value for **SWEEP_ID** and different parameters. Doing so defines a multiple sweep case that is managed by the **IMPEDANCE** command itself.

FREQ Sub-command keyword for specifying a forced frequency in the case of a single frequency response analysis (characters).

f Value of the forced frequency (real). The corresponding value of ω is $2\pi f$.

RAYDAMP Sub-command keyword for specifying Rayleigh proportional damping coefficients for the entire structure (characters).

a Rayleigh damping stiffness coefficient (real).

b Rayleigh damping mass coefficient (real).

FREQSWEET1 Sub-command keyword for requesting a frequency sweep analysis using a one-point configuration of the reconstruction algorithm specified after the sub-command keyword **RECONS** (characters). In this case, the following ($2n_{\Delta_f} + 1$) frequencies f_j are swept in the frequency band $[f_0 - n_{\Delta_f}\Delta_f, f_0 + n_{\Delta_f}\Delta_f]$

$$f_j = f_0 + j\Delta_f, \quad -n_{\Delta_f} \leq j \leq n_{\Delta_f}$$

This sub-command does not support frequency sweep problems with multiple right hand-sides.

Frequency defining the center of the frequency band $[f_0 - n_{\Delta_f}\Delta_f, f_0 + n_{\Delta_f}\Delta_f]$ (real).

f_0 Frequency sweep increment (real).

Together with f_0 and Δ_f , this parameter defines the frequency band $[f_0 - n_{\Delta_f}\Delta_f, f_0 + n_{\Delta_f}\Delta_f]$ (integer).

FREQSWEET Sub-command keyword for requesting a frequency sweep analysis in a frequency band $[f_l, f_u]$ of interest using a reconstruction algorithm specified after the sub-command keyword **RECONS** (characters). In this case, the user can choose between two different schemes for specifying a set of interpolation frequency points within $[f_l, f_u]$ — that is, a set of frequencies where the response and its first few consecutive frequency derivatives are to be computed using the brute force approach — and then sampling frequencies in $[f_l, f_u]$ and rapidly computing the frequency response function at these points using the reconstruction algorithm **alg** specified after the sub-command keyword **RECONS** discussed below.

The first scheme, referred to here as the "regular" sampling scheme, introduces $n_s^* \geq 2$ equally-spaced interpolation frequencies f_j^* in $[f_l, f_u]$, including f_l and f_u . Hence, these frequencies are given by

$$f_j^* = f_l + (j-1)\frac{(f_u - f_l)}{(n_s^* - 1)}, \quad j = 1, \dots, n_s^*$$

Then, the regular scheme samples each interval $[f_{j-1}^*, f_j^*]$ into $n_s \geq 1$ equal frequency increments, and therefore into $(n_s + 1)$ frequency points including f_{j-1}^* and f_j^* , where it rapidly reconstructs the frequency response function using the algorithm **alg**. Hence, the first scheme computes the frequency response at $((n_s^* - 1)n_s + 1)$ frequencies in the frequency band $[f_l, f_u]$ (counting f_l and f_u).

The second scheme, referred to here as the "irregular" sampling scheme, requires the user to specify first on the first line after the keyword **FREQSWEET** the lower end of the frequency spectrum, f_l , and then every other desired interpolation frequency f_j^* (in ascending order) on a separate line together with an integer number n_j specifying the sampling of the interval $[f_{j-1}^*, f_j^*]$ into n_j equal frequency increments. Hence, it defines $(n_j + 1)$ sampling frequencies in $[f_{j-1}^*, f_j^*]$ including f_{j-1}^* and f_j^* . The upper end of the frequency spectrum, f_u , is in this case the last inputted interpolation frequency. It follows that if, for example, n_s^* interpolation frequencies are inputted by the user, and n_j is set to $n_j = n_s$ for all j (again, for example), the second scheme computes the frequency response also at $((n_s^* - 1)n_s + 1)$ frequencies in the frequency band $[f_l, f_u]$.

including f_l and f_u .

This sub-command does not support frequency sweep problems with multiple right hand-sides, except when performing the brute force approach (see `a1g` below).

Lower end of the frequency band of interest $[f_l, f_u]$ (real).

Upper end of the frequency band of interest $[f_l, f_u]$ (real).

Number of equally-spaced interpolation frequencies f_j^* to introduce in the frequency band of interest $[f_l, f_u]$

, including f_l and f_u (integer). Hence, these frequencies are given by

$$f_j^* = f_l + (j - 1) \frac{(f_u - f_l)}{(n_s^* - 1)}, \quad j = 1, \dots n_s^*$$

When part of the definition of `FREQSWEEP`, this parameter specifies the number of equally-spaced frequency intervals in which a frequency "sub-band" $[f_{j-1}^*, f_j^*]$ — where f_{j-1}^* and f_j^* are interpolating frequencies (see above) — is to be sampled (integer). In this case, $[f_{j-1}^*, f_j^*]$ is sampled into $(n_s + 1)$ frequencies, including f_{j-1}^* and f_j^* . When part of the definition of `FREQSWEPPA` or `FREQSWEPPAW`, this parameter specifies the number of equal size frequency intervals in which the frequency band of interest $[f_l, f_u]$ is to be sampled (integer). In this case, $[f_l, f_u]$ is sampled into $(n_s + 1)$ frequencies, including f_l and f_u . When part of the definition of `FREQSWEEP`, the inputted value of this parameter must satisfy $n_s \geq 1$. When part of the definition of `FREQSWEPPA` or `FREQSWEPPAW`, it must satisfy $n_s \geq 2$. In both cases, the frequency response function is computed at all sampled frequency points.

Interpolation frequency (real).

Number of equal size frequency increments in which a frequency interval of the form $[f_{j-1}^*, f_j^*]$ is to be sampled (integer). In this case, $[f_{j-1}^*, f_j^*]$ is sampled into $(n_j + 1)$ frequency points.

Sub-command keyword requesting an *adaptive* frequency sweep analysis in a frequency band $[f_l, f_u]$ of

interest using a reconstruction algorithm specified after the sub-command keyword `RECONS` (characters). In this case, the user can set the maximum number of interpolation frequencies, specify the number of frequencies to sample and at which to reconstruct the frequency response function, specify the reconstruction algorithm, and tune some of the parameters of the automatic adaptation procedure. This sub-command supports frequency sweep problems with multiple right hand-sides.

Sub-command keyword requesting an *adaptive* sweep analysis in the f or $f \times \varphi$ domain using a windowing strategy with a rotating buffer along the frequency axis, and a reconstruction algorithm specified after the sub-command keyword `RECONS` (characters). The frequency f is varied in the band $[f_l, f_u]$ of interest, and the parameter φ , when applicable, is varied in another band of interest $[\varphi_l, \varphi_u]$ or in a set of values φ_k . This sub-command supports frequency sweep problems with multiple right hand-sides.

Specifies an interpolatory model order reduction algorithm for the adaptive frequency sweep procedures associated with the sub-command keywords `FREQSWEPPA` and `FREQSWEPPAW` (characters). Currently, three options are available: (1) `KrylovGalProjection`, (2) `GalProjection`, and (3) `WCAGalProjection`. These three algorithms are the same as those associated with the sub-command keyword `RECONS` and are described below. It is recommended to use `KrylovGalProjection` as much as possible, except for acoustic scattering problems and coupled fluid-structure interaction problems with structural damping and/or absorbing boundary conditions where `KrylovGalProjection` is not valid.

Specifies the tolerance level for assessing the convergence of the adaptive frequency sweep procedures associated with the sub-command keyword `FREQSWEPPA` and `FREQSWEPPAW` using the criterion

$$\frac{\|Z(f_i)\tilde{q} - r(f_i)\|_2}{\|r(f_i)\|_2} \leq \epsilon$$

where \tilde{q} is the reconstructed value of q using the chosen interpolatory model order reduction algorithm (real). The recommended setting for this parameter is $10^{-6} \leq \epsilon \leq 10^{-2}$, and the default setting is $\epsilon = 10^{-2}$.

Specifies the maximum number of interpolation frequencies for the adaptive frequency sweep procedure associated with the sub-command keyword `FREQSWEPPA`. The recommended practice is $6 \leq n_s^{*\max} \leq 8$ (integer). If

for such a setting of this parameter the value of the tolerance level ϵ (see below) is not reached, a higher value of $n_s^{*\max}$ should be used, or the frequency band of interest should be split in two smaller bands and a

	separate frequency sweep should be performed in each one of them instead. The default value of $n_s^{*\max}$ is 6.
n_v^{\min}	Specifies the minimum number of solution vectors to be computed per interpolation frequency for the adaptive frequency sweep procedure associated with the sub-command keyword FREQSWEPA (integer). The recommended value is the default value <u>8</u> .
n_v^{\max}	Specifies the maximum number of solution vectors to be computed per interpolation frequency for the adaptive frequency sweep procedure associated with the sub-command keyword FREQSWEPA (integer). The recommended values are <u>48</u> for the KrylovGalProjection algorithm, and <u>16 - 24</u> for the GalProjection algorithm. The default values for this parameter are 16 when IMOR = GalProjection, and 48 when IMOR = KrylovGalProjection.
Δn_v	Specifies the increment number of solution vectors per interpolation frequency to be considered for computation for the adaptive frequency sweep procedure associated with the sub-command keyword FREQSWEPA (integer). The recommended value is the default value <u>4</u> .
localflag	This integer parameter can be set either to 0 or to 1. It controls the functioning of the adaptive frequency sweep with a windowing strategy command FREQSWEPAW for problems with multiple right hand-sides specified using the command LOADCASE or the sub-command PLANEWAVE in HELMHOLTZ. If localflag = 0, a global Interpolatory Reduced-Order Model (IROM) is constructed for the strip $[f_{j-1}, f_j] \times [\varphi_l, \varphi_u]$ or $[f_{j-1}, f_j] \times \{\varphi_k\}$ using the information computed at the interpolatory parameter points determined at the boundaries of this strip, and applied to the reconstruction of the solution at every sampled point inside this strip. This setting is suitable for frequency sweep problems with multiple right hand-sides specified using the sub-command PLANEWAVE in HELMHOLTZ because the underlying computational strategy relies on the describability of the multiple right hand-sides by a continuous function of a single parameter. If localflag = 1, the reduced-order basis V_j is computed during the windowing strategy using the same interpolatory parameters φ_k as those determined for the computation of V_{j-1} , and the solutions in the strip $[f_{j-1}, f_j] \times [\varphi_l, \varphi_u]$ or the set $[f_{j-1}, f_j] \times \{\varphi_k\}$ are computed using a set of local IROMs. Each local IROM is constructed for an interval $[f_{j-1}, f_j] \times \varphi_k$ using information computed at the two points (f_{j-1}, φ_k) and (f_j, φ_k) , and used to reconstruct the solutions at all parameter points sampled inside the interval $[(f_{j-1}, \varphi_k), (f_j, \varphi_k)]$. Hence, the setting localflag = 1 is suitable for frequency sweep problems with multiple right hand-sides specified using the command LOADCASE as the underlying computational approach builds independent projections for each right hand-side.
n_v	Specifies the total number of snapshot vectors to be computed per interpolation frequency or interpolation point (f_j, φ_k) for the adaptive sweep procedure with a windowing strategy FREQSWEPAW (integer).
tol	Tolerance for determining the placement of the next interpolation frequency f_j in the context of the adaptive sweep procedure with a windowing strategy FREQSWEPAW. A suitable value is such that $\text{tol}^*\epsilon \leq 10^{-1}$ (real).
tol1f	Tolerance for controlling the global-within-one-frequency algorithm (localflag = 0) that attempts to ensure that residual at the interpolation frequency is less than $\text{tol}^*\text{tol1f}$ (real). A suitable value is $\text{tol1f} = 10^{-2}$.
RECONS	Sub-command keyword for defining a reconstruction algorithm in the case of a frequency sweep analysis and setting its parameters (characters).
alg	Name of the reconstruction algorithm (characters). Six such algorithms are available and listed below. The default value is Taylor with 8 derivatives. The brute force reconstruction algorithm corresponds to Taylor with 0 derivative.
Taylor	Taylor series expansion algorithm (characters).
Pade	Conventional multipoint Pade series expansion algorithm (characters).
PadeLanczos	Multipoint Pade series expansion algorithm based on a Lanczos procedure (characters). This algorithm is less prone to ill-conditioning and therefore better performing than the Pade algorithm. However, it is available only for purely structural or acoustic <i>undamped</i> frequency response problems where furthermore K and M are symmetric.
KrylovGalProjection	Interpolatory model order reduction algorithm based on a Galerkin projection and a Krylov subspace (characters). This algorithm, which cannot be used for acoustic scattering problems and coupled fluid-structure problems with structural damping and/or absorbing boundary conditions, delivers nevertheless a much better performance than both of the Pade and GalProjection algorithms for all other frequency response problems of the form given at the beginning of the description of the IMPEDANCE command. It is also known as the KGP algorithm.
GalProjection	Interpolatory model order reduction algorithm based on a Galerkin projection and orthogonalized frequency derivatives (characters). This algorithm is also known as the DGP algorithm.
WCASEGalProjection	Interpolatory model order reduction algorithm based on the combination of the Well-Conditioned Asymptotic Waveform Evaluation (WCASE) scheme, and the multipoint Galerkin projection interpolatory scheme with orthogonalized vectors. This algorithm is also known as the WCASE+DGP algorithm. It is preferred to GalProjection for acoustic scattering problems and coupled fluid-structure interaction problems with structural damping or absorbing boundary conditions.
para_1	For the Taylor algorithm, this parameter specifies the order of the Taylor series (integer). For the Pade, PadeLanczos, GalProjection, and KrylovGalProjection algorithms, it specifies how many of the n_s^* interpolation frequencies to use at a time to apply a multipoint version of the chosen reconstruction algorithm. (Note however that for GalProjection and KrylovGalProjection, para_1 = 1 and para_1 = n_s^* are currently the only options).

para_2	This parameter is relevant only for the <code>Pade</code> , <code>GalProjection</code> , and <code>KrylovGalProjection</code> reconstruction algorithms (integer). For <code>Pade</code> , it specifies the order of the numerator of the rational function (<code>L</code> of $[L/M]$) (integer). For <code>GalProjection</code> , it specifies the number of frequency derivatives of the response to compute at each interpolation frequency. For <code>KrylovGalProjection</code> , it specifies the number of Krylov vectors to compute at each interpolation frequency.
para_3	This parameter is relevant only for the <code>Pade</code> and <code>PadeLanczos</code> reconstruction algorithms. It specifies the order of the denominator of the rational function (<code>M</code> of $[L/M]$) (integer).
PADEPOLES	This sub-command keyword is active only with <code>alg = PadeLanczos</code> and needs input from the sub-command <code>FREQSWEEP</code> (see above) (characters). It also requires the presence in the input file of the <code>MODE</code> command to retrieve the EIGENMODES file or files associated with a previous eigen computation. This (these) file(s) should be placed in the execution path. In this case, this sub-command instructs AERO-S to: (1) compute the approximation by a multipoint Pade expansion of a rational transfer function whose poles are exactly the eigenvalues of the symmetric pencil of interest (K, M), and (2) exploit these poles to identify, in a specified range of interest $[\sigma_l, \sigma_u]$, the eigenvalues that may have been missed by a previous eigen computation in which the <code>MODE</code> command was used to store the results in the EIGENMODES file or files (characters). The multipoint Pade expansion is constructed using points $\sigma_j \in [\sigma_l, \sigma_u]$ that should be generated by the sub-command <code>FREQSWEEP</code> with σ_j playing the role of f_j , $f_l = \sigma_l$ and $f_u = \sigma_u$ (and for example, $n_s = 0$). The parameters of this expansion should be specified in <code>para_1</code> , <code>para_2</code> and <code>para_3</code> (see above). At the end of the computation, AERO-S outputs on the screen the poles of the Pade rational function in the specified range $[\sigma_l, \sigma_u]$, excluding those poles corresponding to the modes read from the EIGENMODES file or files. The specified range $[\sigma_l, \sigma_u]$ can be narrower than that of the eigenvalues read in the EIGENMODES file or files. However, if it is wider, some of the poles of the Pade rational function may, in some cases, not correspond to missed eigenvalues. In any case, the output of the eigenvectors associated with the poles or missed eigenvalues is not currently implemented in AERO-S .
σ_l	Lower end of an eigenvalue interval of interest (real).
σ_u	Upper end of an eigenvalue interval of interest (real).

Next: [IDISPLACEMENTS](#), Previous: [IMPEDANCE](#)

74 INITIAL ACCELERATIONS (Not Supported Yet)

Command Statement: **IACCELERATIONS**

The **IACCELERATIONS** command statement is used to specify a nodal acceleration type of initial conditions. Each node can have up to six degrees of freedom.

IACCELERATIONS

NODE#	DOF#	VALUE
-------	------	-------

NODE#	Node number where the initial acceleration is specified (integer).
DOF#	Degree of freedom local number where the initial acceleration is specified (integer).
VALUE	Value of the specified initial acceleration (float).

Next: [IDISP6](#), Previous: [IACCELERATIONS](#)

75 INITIAL DISPLACEMENTS *S*

Command Statement: **IDISPLACEMENTS**

For a structural analysis, the **IDISPLACEMENTS** command can be used to specify initial conditions for the nodal displacement degrees of freedom (dofs), or the generalized coordinates of a reduced-order representation of the displacement field. For a time-domain acoustic analysis, this command can be used to specify nodal values of the initial solution by setting `dof#` to 8 (see below).

Note 1: All nodal degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: In the context of a linearized (perturbation) analysis, the initial displacements specified under this command are interpreted as initial displacement perturbations.

Note 3: If both `IDISPLACEMENTS` and `IDISP6` commands are specified in the ASCII Input Command Data file, then:

- If the structural dynamics analysis to be performed is not based on a projection-based Reduced-Order Model (ROM), or if it is based on a projection-based ROM as in the sub-command keyword `MODAL` or `SROM` of `DYNAMICS` but the initial displacement inputted using this command is specified for the nodal displacement dofs (and not generalized coordinates), then:
 - If the `GEPS` command is not specified in this file, **AERO-S** selects the content of `IDISP6` to initialize the displacement field.
 - If the `GEPS` command is also specified in this file, **AERO-S** uses the content of the `IDISP6` command to construct the geometric stiffness, and that of `IDISPLACEMENTS` to initialize the displacement field.
- If the structural dynamics analysis to be performed is based on a projection-based ROM, whether a modal one or not, and the initial displacement inputted using this command is specified for the generalized coordinates using the sub-command keyword `MODAL`, then:
 - **AERO-S** initializes the displacement field using the superposition of the projection onto the Reduced-Order Basis (ROB) of the input to the `IDISP6` command and the initial displacement inputted using the sub-command keyword `MODAL` of this command (`IDISPLACEMENTS`).

Note 4: If both of the `GEPS` and `IDISP6` commands are present in the ASCII Input Command Data file but the `IDISPLACEMENTS` command is not specified in this file, **AERO-S** uses the content of `IDISP6` to construct the geometric stiffness and initializes the incremental displacement field to zero.

Note 5: Even though it is intended primarily for structural dynamic computations, this command is also enabled for quasistatic fluid-structure (aeroelastic) and fluid-thermal-structure (aerothermoelastic) steady-state (static) computations (see `QSTATICS`). In such cases, it allows the transmittal to the flow solver **AERO-F** of the initial structural displacement inputted using this command. Then, **AERO-F**:

- Computes the increment fluid-structure interface displacement defined by the traces on this interface of this inputted displacement and that corresponding to the difference between the initial position of the CFD grid specified in `Input.Position` within the **AERO-S** ASCII Input Command Data file, and the original position of this CFD mesh.
- Prescribes this increment interface displacement and computes the corresponding motion of the interior CFD grid points.

Note 6: For a dynamic simulation based on a hyper reduced projection-based reduced-order model and its associated reduced mesh, a high-dimensional initial displacement condition — from which the corresponding low-dimensional initial condition can be inferred by projection onto the Reduced-Order Basis (ROB) inputted using `READMODE` — can be specified using this command: only on the nodes attached to the elements of the inputted reduced mesh; and if the high-dimensional initial displacement field is zero at all other nodes of the original mesh. On the other hand, such an initial condition can be specified on all or an arbitrary subset of nodes of the high-dimensional mesh in the same ASCII Input Command Data file containing the `RMSHC` command, and therefore in the same simulation performed to construct the reduced mesh: in this case, the initial condition for the vector of generalized coordinates is computed offline and outputted in the same output file containing the reduced mesh (see `SAMPLMSH` in `OUTPUT`). Furthermore, if for some reason only the trace of the ROB on the degrees of freedom of the reduced mesh is available — for example, when the ROB is randomized — a specified high-dimensional initial displacement condition must be hyper reduced using the `RMSHC` command.

For nodal displacement dofs, the format of this command is as follows.

`[IDISPLACEMENTS]`

or

`[IDISPLACEMENTS ZERO]`

`NODE# DOF# VALUE`

<code>NODE#</code>	Node number where the initial displacement or acoustic field is specified (integer).
<code>DOF#</code>	Local identification number of the degree of freedom where the initial displacement or acoustic field is specified (integer).
<code>VALUE</code>	Value of the specified initial displacement or acoustic field (real).
<code>ZERO</code>	This initializes the entire initial displacement or acoustic field to zero. No other <code>IDISPLACEMENTS</code> data has to be entered when this option is selected (characters).

For displacement generalized coordinates, the format of this command is as follows.

`[IDISPLACEMENTS]`

`[MODAL rob_id]`
`[MODE# VALUE]`

`MODAL`

This sub-command keyword (characters) can be used to:

- Input the initial conditions for the generalized coordinates associated with the ROB chosen for initializing the displacement field.
- Superpose to the initial conditions specified in the finite element coordinates system additional initial conditions formulated in the generalized coordinates system associated with the ROB chosen for this purpose, subject to any restriction implied by the content of **Note 3**. An example where no restriction applies is one where `IDISPLACEMENTS` is used to input both modal and non-modal initial displacement conditions, where modal means in this context an initial condition specified for the generalized coordinates associated with a chosen ROB, and non-modal means an initial condition specified for the nodal displacement dofs. An example where a restriction applies, is that where `IDISPLACEMENTS` is used to input modal initial displacement conditions, and `IDISP6` is used to input non-modal initial displacement conditions.

In the second case, **IDISPLACEMENTS** should be specified only once and the **MODAL** sub-command and its associated data can be inserted either before or after the data set associated with the initial conditions specified in the finite element coordinates system. This option can also be used to perform a "Ping-Pong" analysis (**AERO**). In all cases, the primal displacement ROB must be inputted in **READMODE**.

rob_id	This parameter is relevant only when the <i>second format</i> of the READMODE command is used to input one or multiple ROBs. In this case, it is an integer number identifying the primal ROB inputted in READMODE to be used to initialize the displacement field (integer).
MODE#	Mode ID number (or column number of the ROB identified above) whose generalized coordinate is to be initialized below (integer).
VALUE	Value of the specified initial condition (real).

Next: [IDISP6PITA](#), Previous: [IDISPLACEMENTS](#)

76 INITIAL DISPLACEMENT 6 COLUMNS (IDISP6 completely spelled out) *S*

Command Statement: **IDISP6 []**

The **IDISP6** command statement is used to specify a nodal displacement type of initial conditions in a different format than that of the **IDISPLACEMENTS** command. This format is specified below.

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: If both this command and the command [LMPC](#) are specified in the input file, the content of **IDISP6** must satisfy the linear multipoint constraints described in [LMPC](#).

Note 3: If the [GEPS](#) command is not specified, and both [IDISPLACEMENTS](#) and **IDISP6** commands are present in the input file, **AERO-S** selects the **IDISP6** command to initialize the displacement field.

Note 4: If the [GEPS](#) command is present in the input file, then **AERO-S** uses the content of the command **IDISP6** to construct the geometric stiffness, and the content of [IDISPLACEMENTS](#), if available, to initialize the incremental displacement field.

Note 5: Because a generalized coordinate initial condition is independent of the number of degrees of freedom at a node, its specification in **AERO-S** is done in [IDISPLACEMENTS](#) and therefore not supported by this command.

Note 6: In the context of a linearized (perturbation) analysis, the initial displacements specified under this command are interpreted as initial displacement perturbations.

Note 7: Even though it is intended primarily for structural dynamic computations, this command is also enabled for quasistatic fluid-structure (aeroelastic) and fluid-thermal-structure (aerothermoelastic) steady-state (static) computations (see [OSTATICS](#)). In such cases, it allows the transmittal to the flow solver **AERO-F** of the initial structural displacement inputted using this command. Then, **AERO-F**:

- Computes the increment fluid-structure interface displacement defined by the traces on this interface of this inputted displacement and that corresponding to the difference between the initial position of the CFD grid specified in [Input.Position](#) within the **AERO-S** ASCII Input Command Data file, and the original position of this CFD mesh.
- Prescribes this increment interface displacement and computes the corresponding motion of the interior CFD grid points.

IDISP6 []

NODE#	VAL_DOF1	VAL_DOF2	VAL_DOF3	VAL_DOF4	VAL_DOF5	VAL_DOF6
-------	----------	----------	----------	----------	----------	----------

	Amplification factor that multiplies each VAL_DOF <i>i</i> value for each node (real).
NODE#	Node number where the initial displacement is specified (integer).
VAL_DOF<i>i</i>#	Value of the specified initial displacement for the <i>i</i> -th degree of freedom (real).

Next: [IVEL6PITA](#), Previous: [IDISP6](#)

77 INITIAL OR SEED DISPLACEMENT FOR PITA (IDISP6PITA completely spelled out)

Command Statement: **IDISP6PITA**

The **IDISP6PITA** command statement can be used to specify the time-slices' initial or seed displacements for the PITA methodology (see [PITA](#)), thus overriding the alternative approach consisting of generating this information by applying the basic time-integrator on the coarse time-grid. It is applicable in both linear and nonlinear settings. Its syntax is similar to that of the **IDISP6** (see [IDISP6](#)) command except that it can introduce up to as many sets of initial displacements as there are time-slices, one set per time-slice. Each set of initial displacements must be input after the previous one, starting with that corresponding to the first time-slice and continuing in consecutive order. All those time-slices for which a set of seed displacement values is not specified are initialized by the aforementioned alternative approach.

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

IDISP6PITA	NUMSLICES
------------	-----------

NODE#	VAL_DOF1	VAL_DOF2	VAL_DOF3	VAL_DOF4	VAL_DOF5	VAL_DOF6
-------	----------	----------	----------	----------	----------	----------

NUMSLICES Number of consecutive time-slices for which seed displacements are specified (integer).
 NODE# Node number where the seed displacement is specified (integer).
 VAL_DOFi# Value of the seed displacement for the i-th degree of freedom at the beginning of a time-slice (float).

Next: [ITEMPERATURES](#), Previous: [IDISP6PITA](#)

78 INITIAL OR SEED VELOCITY FOR PITA (IVEL6PITA completely spelled out)

Command Statement:	IVEL6PITA
--------------------	------------------

The **IVEL6PITA** command statement can be used to specify the time-slices' initial or seed velocities for the PITA methodology (see [PITA](#)), thus overriding the alternative approach consisting of generating this information by applying the basic time-integrator on the coarse time-grid. It is applicable in both linear and nonlinear settings. Its syntax is similar to that of the **IDISP6** (see [IDISP6](#)) command except that it can introduce up to as many sets of initial velocities as there are time-slices, one set per time-slice. Each set of initial velocities must be input after the previous one, starting with that corresponding to the first time-slice and continuing in consecutive order. All those time-slices for which a set of seed velocity values is not specified are initialized by the aforementioned alternative approach.

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

IVEL6PITA	NUMSLICES
-----------	-----------

NODE#	VAL_DOF1	VAL_DOF2	VAL_DOF3	VAL_DOF4	VAL_DOF5	VAL_DOF6
-------	----------	----------	----------	----------	----------	----------

NUMSLICES Number of consecutive time-slices for which seed velocities are specified (integer).
 NODE# Node number where the seed velocity is specified (integer).
 VAL_DOFi# Value of the seed velocity for the i-th degree of freedom at the beginning of a time-slice (float).

Next: [IVELOCITIES](#), Previous: [IVEL6PITA](#)

79 INITIAL TEMPERATURES *S*

Command Statement:	ITEMPERATURES
--------------------	----------------------

The **ITEMPERATURES** command can be used to specify initial conditions for the nodal temperature degrees of freedom (dofs) or temperature generalized coordinates of a thermal model.

For nodal temperature dofs, the format of this command is as follows.

ITEMPERATURES

NODE	VALUE
------	-------

NODE# Node number where the initial temperature is specified (integer).
 VALUE Value of the specified initial temperature (real).

For temperature generalized coordinates, the format of this command is as follows.

ITEMPERATURES	
MODAL	rob_id
MODE#	VALUE

- MODAL This sub-command keyword can be used to input the initial conditions for the generalized coordinates associated with the Reduced-Order Basis (ROB) chosen for initializing the temperature field. When this option is used, it must also be specified in [DYNAMICS](#), and the primal temperature Reduced-Order Basis (ROB) must be inputted using [READMODE](#).
- rob_id This parameter is relevant only when the *second format* of the [READMODE](#) command is used to input one or multiple ROBs. In this case, it is an integer number identifying the primal ROB inputted in [READMODE](#) to be used to initialize the temperature field (integer).
- MODE# Mode ID number (or column number of the ROB identified above) whose generalized coordinate is to be initialized below (integer).
- VALUE Value of the specified initial condition (real).

Next: [INPC](#), Previous: [ITEMPERATURES](#)

80 INITIAL VELOCITIES *S*

Command Statement: IVELOCITIES	
--------------------------------	--

For a structural analysis, the **IVELOCITIES** command can be used to specify initial conditions for the nodal velocity degrees of freedom (dofs), or the generalized coordinates of a reduced-order representation of the velocity field. For a time-domain acoustic analysis, this command can be used to specify nodal values of the initial time-derivative of the solution by setting dof# to 8 (see below).

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: For a dynamic simulation based on a hyper reduced projection-based reduced-order model and its associated reduced mesh, a high-dimensional initial velocity condition — from which the corresponding low-dimensional initial condition can be inferred by projection onto the Reduced-Order Basis (ROB) inputted using [READMODE](#) — can be specified using this command: only on the nodes attached to the elements of the inputted reduced mesh; and if the high-dimensional initial velocity field is zero at all other nodes of the original mesh. On the other hand, such an initial condition can be specified on all or an arbitrary subset of nodes of the original mesh in the same ASCII Input Command Data file containing the [RMSHC](#) command, and therefore in the same simulation performed to construct the reduced mesh: in this case, the initial condition for the vector of generalized coordinates is computed offline and outputted in the same output file containing the reduced mesh (see [SAMPLMSH](#) in [OUTPUT](#)). Furthermore, if for some reason only the trace of the ROB on the degrees of freedom of the reduced mesh is available — for example, when the ROB is randomized — a specified high-dimensional initial velocity condition must be hyper reduced using the [RMSHC](#) command.

For nodal velocity dofs, the format of this command is as follows.

IVELOCITIES		
NODE#	DOF#	VALUE

- NODE# Node number where the initial velocity is specified (integer).
 DOF# Degree of freedom local number where the initial velocity is specified (integer).
 VALUE Value of the specified initial temperature (real).

For velocity generalized coordinates, the format of this command is as follows.

IVELOCITIES	
MODAL	rob_id
MODE#	VALUE

MODAL This sub-command keyword (characters) can be used to:

- Input the initial conditions for the generalized coordinates associated with the Reduced-Order Basis (ROB) chosen for initializing the velocity field.
- Superpose to the initial conditions specified in the finite element coordinates system additional initial conditions formulated in the generalized coordinates system associated with the ROB chosen for this purpose.

In the second case, **IVELOCITIES** should be specified only once, and the **MODAL** sub-command and its associated data can be inserted either before or after the data set associated with the initial conditions specified in the finite element coordinates system. This option can also be used to perform a "Ping-Pong" analysis ([AERO](#)). In all cases, the primal velocity ROB must be inputted using [READMODE](#).

rob_id This parameter is relevant only when the *second format* of the [READMODE](#) command is used to input one or multiple ROBs. In this case, it is an integer number identifying the primal ROB inputted in [READMODE](#) to be used to initialize the velocity field (integer).

MODE# Mode ID number (or column number of the ROB identified above) whose generalized coordinate is to be initialized below (integer).

VALUE	Value of the specified initial condition (real).
-------	--

Next: [MATERIAL](#), Previous: [IVELocities](#)

81 INTRUSIVE POLYNOMIAL CHAOS

Command Statement:	INPC
--------------------	-------------

The **INPC** command signals to the **AERO-S** code to perform an intrusive non deterministic analysis using the Polynomial Chaos representation of the solution. It requires that the input file also contains the [GROUPS](#), [RANDOM](#), and [STATICS](#) commands and their respective data.

Note 1: Currently, this command is limited to linear, static, structural analysis.

Note 2: Currently, this command is supported only by the PCG, BCG, and CR iterative solvers (see [STATICS](#)).

The input format of this command is given below.

INPC

DEG_OUT

DEG_OUT	Degree of the polynomial chaos representation of the solution (integer).
---------	--

Next: [LMPC](#), Previous: [INPC](#)

82 LINEAR MATERIAL PROPERTIES *S*

Command Statement:	MATERIAL
--------------------	-----------------

The **MATERIAL** command should be used for any of the following purposes:

- For structural analysis problems:
 - Define one or several material ID numbers to be assigned to elements of the computational model using the command [ATTRIBUTES](#), and for each defined material ID number:
 - Specify the parameters of the linear elastic material law for a linear analysis and those of the default material law (see below) for a nonlinear analysis.
 - Specify the geometrical properties of the elements of the computational model assigned this material ID number.
- To this end, the user should know that **AERO-S** assigns default material laws to the elements of a computational structural model as follows:
- For a bar, beam, or shell element, it assigns the linear elastic material law. In this case, this command should be used to specify the properties of this material law for each defined material ID number. In the presence of the command [NONLINEAR](#) in the ASCII Input Command Data file, **AERO-S** activates the geometrically nonlinear formulation of the internal force vector and tangent stiffness matrix using the corotational approach.
 - For a 3D solid element or a membrane element type 128 or 129:
 - In the absence of the command [NONLINEAR](#) in the ASCII Input Command Data file, **AERO-S** assigns the linear elastic material law. Again, this command should be used to specify the properties of this material law for each defined material ID number.
 - In the presence of the command [NONLINEAR](#) in the ASCII Input Command Data file, **AERO-S** assigns the Saint Venant-Kirchhoff hyperelastic law and activates the geometrically nonlinear formulation of the internal force vector and tangent stiffness matrix using the total Lagrangian method. The Saint Venant-Kirchhoff hyperelastic material law is an extension of the linear elastic material model to the nonlinear regime. It has the form

$$S = \lambda \operatorname{tr}(\epsilon) I + 2\mu \epsilon$$

where **S** is the second Piola-Kirchhoff stress tensor,

$$\epsilon = \frac{1}{2} [(\nabla u)^T + \nabla u + (\nabla u)^T \cdot \nabla u]$$

is the Green-Lagrange strain tensor, **λ** and **μ** are the Lame constants, and **I** is the second-order unit tensor. **AERO-S** deduces the Lame properties of this material law from the values of the Young modulus **E** and Poisson ratio **ν** - which should be specified for each defined material ID number using this command - as follows

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \quad \text{and} \quad \mu = \frac{E}{2(1+\nu)}$$

The strain-energy density function for the Saint Venant-Kirchhoff model is defined as

$$W(\epsilon) = \frac{\lambda}{2} [tr(\epsilon)]^2 + \mu tr(\epsilon^2)$$

and the second Piola-Kirchhoff stress is given by

$$S = \frac{\partial W}{\partial \epsilon}$$

Otherwise, the user can use the command [MATLAW](#) to define or select among a large list of pre-defined more general material laws, and the command [MATUSAGE](#) to attribute them to the elements of the computational domain.

- Specify the method to use to enforce constraints associated with joint and rigid elements assigned the aforementioned material ID number.
- For linear acoustic and acoustic scattering analysis problems in the frequency or time domain, for linear and nonlinear thermal (heat transfer) problems, and for linear hygro (moisture transfer) problems:
 - Define a material ID number to be assigned to elements of the computational model using the command [ATTRIBUTES](#), and:
 - Specify the physical properties of the governing equations for the elements of the computational model assigned the aforementioned material ID number.
 - Specify the geometrical properties of the elements of the computational model assigned the aforementioned material ID number.

Several input formats are possible for this command and are described below.

Note 1: This command is required in any **AERO-S** ASCII Input Command Data file, even when such a file contains the commands [MATLAW](#) and [MATUSAGE](#).

Note 2: Even for a given input format, the interpretation of a data entry can change from one element type to another. For example for the first input format, the main interpretation of some data entries changes when the element type is that of a lumped torsional spring, a torsional spring connector, a linear spring connector, a Timoshenko beam, a thermal element, a rigid translational link, or a rigid rotational link.

Note 3: A *non-structural* mass is defined here and throughout this User's Reference Manual as a discrete mass that is attributed to an element, and which therefore affects the mass matrix of this element and its contribution to a gravity load. Specifically, it affects only the translational degrees of freedom attached to this element, in the same manner as a discrete mass that is specified for a translational degree of freedom attached to a specified node (see the command [DIMASS](#)).

Note 4: For composite elements, this command is not required unless stress and/or strain output is requested. Even in this case, only the total thickness of the element must be specified under this command. Dummy or zero values can be entered for the other properties as their real values should be specified under the [COMPOSITE](#) command.

Note 5: The Perfectly Matched Layer (PML) method is supported only by the elements type 95, 96, 98, and 99.

Note 6: Specifying under this command a method for enforcing the constraints associated with joint and/or rigid elements sharing a certain material ID number overrides for these elements the specification under [CONSTRAINTS](#) of a default method for enforcing constraints.

MATERIAL

MID	A	E	<u><i>μ</i></u>	<u><i>ρ</i></u>	h	k	t	P	Ta	<u><i>c_p</i></u>	<u><i>w</i></u>	I _{xx}	I _{yy}	I _{zz}	ymin	ymax	zmin	zmax	DAMPING_TYPE
a	b																		

MID

The material id number.

A

Cross sectional area for trusses, beams, and lineal thermal elements (float).

E

Young's Modulus for all element types (float). However, this field can also be filled by a **negative integer** in which case it means that Young's Modulus is a function of temperature and is to be determined from the **YMTT** and **TEMPERATURES** commands. In the latter case, the integer specified here points to the id of the Young's modulus-temperature table in the **YMTT** command (negative integer).

μ

Poisson's ratio for all element types except trusses (float).

ρ

Mass density per unit volume accounting for both structural and non-structural types of mass (see definition in the **Notes** section above), except in the following situations (float):

- For a composite or orthotropic shell element (type 15, 1515, 20, and 2020) and when the composite properties are defined using the **COEF** option (see [COMPOSITE](#)), this parameter denotes the mass density per unit area and accounts for both structural and non-structural types of mass.
- For a composite or orthotropic shell element (type 15, 1515, 20, and 2020) and when the composite properties are defined using the **LAYC** option (or a combination of **LAYO** and **LAYMAT**, see [COMPOSITE](#)) or the **LAYN** option (or a combination of **LAYO** and **LAYMAT**, see [COMPOSITE](#)), and for any other composite shell element (type 8, 88, 16, 73, and 76), this parameter denotes the mass density per unit area of the non-structural type of mass only. In this case, the density associated with the structural type of mass should

	be specified for each individual material layer using the command COMPOSITE .
h	Heat convection coefficient (only for boundary convection elements: type 47, 48, and 49 (float)).
k	Heat conduction coefficient (float). For nonlinear thermal and fluid-thermal (aerothermal) analysis, this coefficient may be a function of temperature. In this case, this parameter should be set to a negative number, the temperature-dependent mass density should be inputted using the command DCTT , and the absolute value of the negative integer inputted here is interpreted by AERO-S to be the ID of the mass density-temperature table inputted using the command DCTT (negative integer).
t	Element thickness (float).
P	Perimeter/circumference area for thermal elements (float), or the depth of the boundary where convection occurs when element type 47 is used to model non-lateral boundary convection (float).
Ta	Reference temperature (Absolute) of the element (float). WARNING: if this value is different from the nodal temperature of that element (see TEMPERATURES), it creates a thermal loading even in a pure structural analysis.
C_p	Specific heat coefficient in a thermal analysis (float). For nonlinear thermal and fluid-thermal (aerothermal) analysis, this coefficient may be a function of temperature. In this case, this parameter should be set to a negative number, the temperature-dependent specific heat coefficient should be inputted using the command SCTT , and the absolute value of the negative integer inputted here is interpreted by AERO-S to be the ID of the specific heat coefficient-temperature table inputted using the command SCTT (negative integer).
w	Coefficient of thermal expansion in either a thermal analysis, or a structural analysis if the specified reference temperature τ_a is different from the specified nodal temperatures (see TEMPERATURES and THERMOE) (float). In a thermal analysis, this field can also be filled by a negative integer meaning that the coefficient of thermal expansion is a function of temperature and is to be determined from the TETT and TEMPERATURES commands. In this case, the integer specified here points to the id of the coefficient of thermal expansion-temperature table in the TETT command (negative integer).
Ixx/ss	For a mechanical analysis (MECH under the DYNAMICS command), this is the cross-sectional moment of inertia about the local and centroidal x-axis. For an acoustic analysis in the time-domain (ACOU under the DYNAMICS command), this is the speed of sound in the medium represented by the attribute number.
Iyy	Cross-sectional moment of inertia about the local and centroidal <i>principal</i> y-axis.
Izz	Cross-sectional moment of inertia about the local and centroidal <i>principal</i> z-axis.
ymin	Negative local y-coordinate of the bottom fiber of a beam cross section.
ymax	Positive local y-coordinate of the top fiber of a beam cross section.
zmin	Negative local z-coordinate of the top fiber of a beam cross section.
zmax	Positive local z-coordinate of the top fiber of a beam cross section.
DAMPING_TYPE	Optional sub-command keyword to specify a type of damping (characters). Two options are currently available:
RAYDAMP	This option specifies a Rayleigh proportional type of damping for structural dynamic (see DYNAMICS) or frequency response (see IMPEDANCE) analysis. In this case, damping is introduced at the element level in the form of a finite element damping matrix K^e that is constructed as $D^e = aK^e + bM^e$ <p>where M^e and K^e denote the element-level mass and stiffness matrices, respectively, and a and b are two real coefficients that are specified below.</p>
STRDAMP	This option specifies a structural type of damping for frequency response analysis (see IMPEDANCE). In this case, damping is introduced in the form of the following modified local (element-level) Young modulus
	$\underline{E^* = E(1 - i\eta_E)}$
	where E denotes here the element-level Young modulus, the star superscript designates the modified Young modulus, i denotes the pure imaginary number satisfying $i^2 = -1$, the minus sign in $-i$ is due to the convention $u_s e^{-i\omega_j t}$ (see IMPEDANCE), and η_E is the Young modulus <i>loss factor</i> . In general, η_E is a function of the forced frequency $f = \omega/2\pi$. If in the frequency range of interest η_E can be represented as a linear function of f — that is,
	$\underline{\eta_E(f) = \eta_{E_1} + \eta_{E_2}f}$
	this linear representation can be specified in the MATERIAL command by inputting $a = \eta_{E_1}$ and $b = \eta_{E_2}$ below and specifying f in IMPEDANCE . Otherwise, an arbitrary variation of the Young modulus loss factor with the frequency can be inputted in the form a lookup table (or curve) using SDETAFT , in which case given a frequency f specified in IMPEDANCE , η_E is determined by local interpolation.
RUBDAMP	This option is supported only for the tetrahedral and hexahedral (brick) solid elements (type = 17, 23, 25, 72, 91, 102, or 103). It specifies a rubber material type of damping for frequency response analysis (see IMPEDANCE). In this case, damping is introduced in the form of the following modified local (element-level)

Young modulus and shear modulus

$$E^* = E(1 - i\eta_E)$$

$$\mu^* = \mu(1 - i\eta_\mu)$$

where E and μ denote here the element-level Young modulus and shear modulus, respectively, the star superscript designates the modified moduli, i denotes the pure imaginary number satisfying $i^2 = -1$, the minus sign in $-i$ is due to the convention $u_s e^{-i\omega_j t}$ (see [IMPEDANCE](#)), and η_E and η_μ are the Young modulus *loss factor* and shear modulus *loss factor*, respectively. To specify E , η_E , μ , and η_μ , a distinction should be made between two cases:

- The particular case where all of E , η_E , μ , and η_μ are frequency independent. In this case, and only in this case, E should be specified as usual, Poisson's ratio ν should be also specified as usual, μ is deduced by **AERO-S** from E and ν , and η_E and η_μ should be inputted in a and b , respectively (see below).
- The general case where some or all of the parameters E , η_E , μ , and η_μ are functions of the forced frequency $f = \omega/2\pi$. In this case, arbitrary variations of these properties with the frequency (including a constant behavior, if any) must be inputted in the form of a lookup table that is defined in [RUBDAFT](#) and assigned using the parameter a (see below); then, given a frequency f specified under [IMPEDANCE](#), E , η_E , μ , η_μ are determined by **AERO-S** by local interpolation.

a

If [DAMPING_TYPE](#) is set to [RAYDAMP](#), a specifies the Rayleigh damping stiffness coefficient (real) — that is, the stiffness coefficient in the Rayleigh proportional damping matrix D^e for the material identified by [MID](#). In the nonlinear case, this coefficient is assigned at each iteration to the linearized stiffness matrix and the product of these two quantities, which varies with the nonlinear iterations, constitutes the stiffness contribution to the damping term. In a structural dynamic analysis, this value overrides for the material identified by [MID](#) any value of the Rayleigh damping stiffness coefficient specified under [DYNAMICS](#). In any analysis performed using the [IMPEDANCE](#) command, this value overrides for the material identified by [MID](#) any value of the Rayleigh damping stiffness coefficient specified under [IMPEDANCE](#).

On the other hand, if [DAMPING_TYPE](#) is set to [STRDAMP](#), a has a different meaning that pertains to the loss factor η_E briefly discussed above. In this case, if the loss factor of the material identified by [MID](#) is to be represented as a linear function of the frequency f of the form $\eta_E(f) = \eta_{E_1} + \eta_{E_2}f$ (see above), then $a = \eta_{E_1}$ and should be specified as such. However, if the loss factor is to be interpolated using an arbitrary lookup table defined in [SDETAFT](#), a should be set to the negative integer value $-curve_id$ (integer), where $curve_id$ is the "id number" of the relevant lookup table defined in [SDETAFT](#) using the sub-command keyword [CURVE](#). In that case, b (see below) should be simply ignored.

If [DAMPING_TYPE](#) is set to [RUBDAMP](#) however, then:

- If all of the parameters E , η_E , μ , and η_μ are frequency independent, $a = \eta_E$.
- Otherwise — that is, if E , η_E , μ , or η_μ is a frequency dependent parameter — a should be set to the negative integer value $-table_id$ (integer), where $table_id$ is the "id number" of the relevant lookup table defined in [RUBDAFT](#) using the sub-command keyword [TABLE](#) and containing the arbitrary variations of E , η_E , μ , and η_μ with the frequency f (including a constant behavior of some of these parameters, if any).

b

If [DAMPING_TYPE](#) is set to [RAYDAMP](#), b specifies the Rayleigh damping mass coefficient (real) — that is, the mass coefficient in the Rayleigh proportional damping matrix D^e (real) for the material identified by [MID](#) (see above). In a structural dynamic analysis, this value overrides for the material identified by [MID](#) any value of the Rayleigh damping mass coefficient specified under [DYNAMICS](#). In any analysis performed using the [IMPEDANCE](#) command, this value overrides for the material identified by [MID](#) any value of the Rayleigh damping mass coefficient specified under [IMPEDANCE](#).

On the other hand, if [DAMPING_TYPE](#) is set to [STRDAMP](#), b has a different meaning that pertains to the loss factor η_E briefly discussed above. In this case, if the loss factor of the material identified by [MID](#) is to be represented as a linear function of the frequency f of the form $\eta_E(f) = \eta_{E_1} + \eta_{E_2}f$ (see above), then $b = \eta_{E_2}$ and should be specified as such. However, if the loss factor is to be interpolated using an arbitrary lookup table defined in [SDETAFT](#), a (see above) should be set to the negative integer value $-curve_id$ (integer), where $curve_id$ is the "id

"number" of the *relevant* lookup table defined in [SDETAFT](#) using the sub-command keyword **CURVE**, and **b** should be simply ignored.

If **DAMPING_TYPE** is set to **RUBDAMP** however, then:

- If all of the parameters **E**, **η_E** , **μ** , and **η_μ** are frequency independent, **b** = **η_μ** .
- Otherwise — that is, if **E**, **η_E** , **μ** , or **η_μ** is a frequency dependent parameter — **b** is ignored.

If the element is a lumped torsional spring (**eltyp** = 11), a lumped linear spring (**eltyp** = 12), a linear spring connector (**eltyp** = 21), or a torsional spring connector (**eltyp** = 22), the material properties are defined as follows.

MID	Kx	Ky	Kz	lx1	ly1	lz1	lx2	ly2	lz2	lx3	ly3	lz3	DAMPING	c
MID														The material id number from element attribute table.
Kx														Torsional/Translational spring constant along local x-axis (float).
Ky														Torsional/Translational spring constant along local y-axis (float).
Kz														Torsional/Translational spring constant along local z-axis (float).
lx1	ly1	lz1												The first axis of the local frame expressed in the global frame (floats).
lx2	ly2	lz2												The first axis of the local frame expressed in the global frame (floats).
lx3	ly3	lz3												The first axis of the local frame expressed in the global frame (floats).
DAMPING														Sub-command keyword for specifying the Rayleigh damping <i>stiffness</i> coefficient (characters).
c														Rayleigh damping stiffness coefficient (real).

If the element is a discrete mass and inertia (type 131), the material properties are defined as follows.

MID	MASS	m	Ixx	Iyy	Izz	Ixy	Iyz	Ixz	cx	cy	cz	
MID												The material ID number from the element attribute table (integer).
MASS												Keyword indicating that the following data entries specify the properties of a discrete mass and inertia.
m												Discrete mass (real).
Ixx												Ixx component of the discrete inertia tensor (real). This tensor is defined in the element frame if one is specified under EFRAMES . Otherwise, it is defined in the global frame.
Iyy												Iyy component of the discrete inertia tensor (real). This tensor is defined in the element frame if one is specified under EFRAMES . Otherwise, it is defined in the global frame.
Izz												Izz component of the discrete inertia tensor (real). This tensor is defined in the element frame if one is specified under EFRAMES . Otherwise, it is defined in the global frame.
Ixy												Ixy component of the discrete inertia tensor (real). This tensor is defined in the element frame if one is specified under EFRAMES . Otherwise, it is defined in the global frame.
Iyz												Iyz component of the discrete inertia tensor (real). This tensor is defined in the element frame if one is specified under EFRAMES . Otherwise, it is defined in the global frame.
Ixz												Ixz component of the discrete inertia tensor (real). This tensor is defined in the element frame if one is specified under EFRAMES . Otherwise, it is defined in the global frame.
cx												x component of the offset vector from the node to the center of the discrete mass (real).
cy												y component of the offset vector from the node to the center of the discrete mass (real).
cz												z component of the offset vector from the node to the center of the discrete mass (real).

If the element is a Timoshenko beam (**eltyp** = 7), the material properties are defined as follows.

MID	A	E	ν	ρ	α_Y	α_Z	C1	P	Ta	c_p	w	Ixx	Iyy	Izz	ymin	ymax	zmin	zmax
DAMPING_TYPE	a	b																
MID																		
A																		
E																		
ν																		

The material id number from element attribute table.
Cross sectional area (float).
Young's Modulus (float).
Poisson's ratio (float).

ρ
 α_y

Mass density per unit volume (float).

α_z

Shear deflection constant associated with I_{yy} (float). This shear deflection constant is defined as the ratio of the cross-sectional area and the effective shear area associated with the local y axis specified in [EFRAMES](#).

C1
P
Ta
 c_p

This effective shear area is defined in chapter 12 of *Timoshenko and Goodier, "Theory of Elasticity"*. Note that this shear deflection constant is the inverse of the shear coefficient defined in https://en.wikipedia.org/w/index.php?title=Timoshenko_beam_theory&oldid=10000000 and used by other finite element codes such as NASTRAN.

Shear deflection constant associated with I_{zz} (float). This shear deflection constant is defined as the ratio of

the cross-sectional area and the effective shear area associated with the local z axis specified in [EFRAMES](#). This effective shear area is defined in chapter 12 of *Timoshenko and Goodier, "Theory of Elasticity"*. Note that this shear deflection constant is the inverse of the shear coefficient defined in https://en.wikipedia.org/w/index.php?title=Timoshenko_beam_theory&oldid=10000000 and used by other finite element codes such as NASTRAN.

Non-uniform torsion constant (float).

Not Applicable
Not Applicable
Not Applicable

w

Not Applicable

Ixx

Cross-sectional moment of inertia about the local and centroidal x-axis

Iyy

Cross-sectional moment of inertia about the local and centroidal *principal* y-axis.

Izz

Cross-sectional moment of inertia about the local and centroidal *principal* z-axis.

ymin

Negative local y-coordinate of the bottom fiber of a beam cross section.

ymax

Positive local y-coordinate of the top fiber of a beam cross section.

zmin

Negative local z-coordinate of the top fiber of a beam cross section.

zmax

Positive local z-coordinate of the top fiber of a beam cross section.

DAMPING_TYPE

Optional sub-command keyword to specify a type of damping (characters). Two options are currently available:

RAYDAMP

This option specifies a Rayleigh proportional type of damping for structural dynamic (see [DYNAMICS](#)) or frequency response (see [IMPEDANCE](#)) analysis (see above for further details).

STRDAMP

This option specifies a structural type of damping for frequency response analysis (see [IMPEDANCE](#)) (see above for further details).

a

If DAMPING_TYPE is set to RAYDAMP, a specifies the Rayleigh damping stiffness coefficient (real). On the other hand, if DAMPING_TYPE is set to STRDAMP, then:

- If the loss factor of the material identified by MID is to be represented as a linear function of the frequency f of the form $\eta_E(f) = \eta_{E_1} + \eta_{E_2}f$ (see above), then $a = \eta_{E_1}$ and should be specified as such.
- However, if the loss factor is to be interpolated using an arbitrary lookup table defined in [SDETAFT](#), a should be set to the *negative integer* value -curve_id (integer), where curve_id is the "id number" of the *relevant* lookup table defined in [SDETAFT](#) using the sub-command keyword CURVE. In that case, b (see below) should be simply ignored.

b

If DAMPING_TYPE is set to RAYDAMP, b specifies the Rayleigh damping mass coefficient (real). On the other hand, if DAMPING_TYPE is set to STRDAMP, then:

- If the loss factor of the material identified by MID is to be represented as a linear function of the frequency f of the form $\eta_E(f) = \eta_{E_1} + \eta_{E_2}f$ (see above), then $b = \eta_{E_2}$ and should be specified as such.
- However, if the loss factor is to be interpolated using an arbitrary lookup table defined in [SDETAFT](#), a (see above) should be set to the *negative integer* value -curve_id (integer), where curve_id is the "id number" of the *relevant* lookup table defined in [SDETAFT](#) using the sub-command keyword CURVE, and b should be simply ignored.

If the element is a rigid translational or rotational link (eltyp = 68 or 69), the material properties are defined as follows.

MID	x	y	z	dum											
-----	---	---	---	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

MID
x
y
z
dum

The material id number from element attribute table.
Any nonzero value implies a rigid motion in this direction (float).
Any nonzero value implies a rigid motion in this direction (float).
Any nonzero value implies a rigid motion in this direction (float).
Any dummy value (float).

If the element is a constraint function element (type = 77, 78, 79, 177, 178, 179), the method for enforcing the associated constraints (see [CONSTRAINTS](#)) can be specified as follows.

MID	CONMAT	[CONSTRAINT_METHOD]	[MODE]	[mode_v]
-----	--------	---------------------	--------	----------

MID	The material ID number from the element attribute table (integer).			
CONMAT	Keyword indicating that the following data entries specify the method chosen for enforcing the constraints associated with the joint, rigid, or constraint function elements sharing the material id number MID (characters).			
MODE	Optional sub-command keyword to specify whether a constraint is of the equality or inequality type (characters).			
mode_v	Setting this parameter to 0, which is its default value, specifies that the constraint CONSTRAINT# is of the equality type. Setting it to 1 specifies that it is of the inequality type (integer).			
CONSTRAINT_METHOD	Optional parameter which specifies the method for enforcing the constraints associated with the joint, rigid, or constraint function elements sharing the material ID number MID (characters or characters and real). The default method is set in CONSTRAINTS and used whenever this entry is omitted.			
multipliers	The Lagrange multiplier method.			
elimination	The elimination method.			
penalty beta	The penalty method. The real-valued parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).			
augmented beta	The augmented Lagrangian method. The real-valued parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).			

If the element is a joint (eltyp = 119, 120, 121, 122, 123, 124, 125, or 127), rigid (eltyp = 65, 66, 67, 68, 69, 70, 71, 73, 74, or 76), or constraint function element (eltyp = 77, 78, 79, 177, 178, 179), the method for enforcing the associated constraints (see [CONSTRAINTS](#)) and any applicable mass property can be specified as follows.

MID	CONMAT	[CONSTRAINT_METHOD]	[MASS]	DENSITY	[GEOPARA]
-----	--------	---------------------	--------	---------	-----------

MID	The material id number from element attribute table.						
CONMAT	Keyword indicating that the following data entries specify the method chosen for enforcing the constraints associated with the joint, rigid, or constraint function elements sharing the material id number MID (characters).						
CONSTRAINT_METHOD	Optional parameter which specifies the method for enforcing the constraints associated with the joint, rigid, or constraint function elements sharing the material id number MID (characters or characters and real). The default method is set in CONSTRAINTS and used whenever this entry is omitted.						
multipliers	The Lagrange multiplier method.						
elimination	The elimination method.						
beta	penalty	The penalty method. The real-valued parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).					
beta	augmented	The augmented Lagrangian method. The real-valued parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).					
MASS	Optional keyword indicating that the rigid elements with eltyp = 65, 66, 70, 73, or 76 sharing the material id number MID are to be attributed an element level (lumped or consistent) mass matrix computed as in the case of flexible elements, using the following data entries (characters).						
DENSITY	Optional parameter which specifies the density for the rigid elements with eltyp = 65, 66, 70, 73, or 76 sharing the material id number MID (real).						
GEOPARA	Optional parameter which specifies the cross sectional area for those rigid truss (eltyp = 65) and beam (eltyp = 66) elements, or the thickness for those rigid shell (eltyp = 73 or 76) elements sharing the material id number MID (real).						

If the element is a nonlinear spring (eltyp = 200, 201, 202, 203, 204, or 205), the definition of its material properties can be specified as follows.

MID	SPRINGMAT	k	[FREEPLAY]	[freeplay_ul]
-----	-----------	---	------------	---------------

MID	The material id number from element attribute table.			
SPRINGMAT	Keyword indicating that the following data entry specifies the stiffness constant of the nonlinear spring elements sharing the material id number MID (characters).			
k	Stiffness constant of the nonlinear spring elements sharing the material id number MID (real).			

FREEPLAY	Optional keyword indicating that the following data entry specifies the freeplay parameter for the spring elements of type 203, 204, or 205 sharing the material ID number MID (characters). For each such spring element, freeplay is modeled using this parameter as follows:
	$k^{eff} = 0 \quad \text{for } -\infty \leq \delta \leq freeplay_ul$
	$k^{eff} = k \quad \text{for } freeplay_ul \leq \delta$

where k is the stiffness coefficient k of the spring, and δ is its displacement. In a linear dynamic analysis, the freeplay-induced forces and moments are interpreted as configuration-dependent and automatically treated as piecewise constant (see [PIECEWISE](#) in [STATICS](#)).

freeplay_ul Freeplay upper displacement limit which should be specified in the unit of length chosen for the computational model.

If the element is a joint spring combination (eltyp = 220, 221, 222, 223, 225, 227, or 323), the definition of its material properties and the method for enforcing the associated constraints (see [CONSTRAINTS](#)) can be specified as follows.

MID	CONMAT	[CONSTRAINT_METHOD]	SPRINGMAT	k1	k2	k3	FREEPLAY	freeplay_ll1	freeplay_ul1	factor_lz1	factor_dz1
factor_uz1	freeplay_ll2	freeplay_ul2	factor_lz2	factor_dz2	factor_uz2	freeplay_ll3	freeplay_ul3	factor_lz3	factor_dz3		

MID	The material id number from element attribute table.											
CONMAT	Keyword indicating that the following data entries specify the method chosen for enforcing the constraints associated with the joint spring combination elements sharing the material id number MID (characters).											
CONSTRAINT_METHOD	Optional parameter which specifies the method for enforcing the constraints associated with the joint spring combination elements sharing the material id number MID (characters or characters and real). The default method is set in CONSTRAINTS and used whenever this entry is omitted.											
multipliers	The Lagrange multiplier method.											
elimination	The elimination method.											
beta	penalty	The penalty method. The real-valued parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).										
beta	augmented	The augmented Lagrangian method. The real-valued parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).										
SPRINGMAT	Keyword indicating that the following data entry specifies the stiffness coefficient of the joint spring combination elements sharing the material ID number MID (characters).											
k1 k2 k3	Stiffness constants of the joint spring combination elements sharing the material ID number MID (real). More specifically k_1 is the stiffness of the first embedded spring, k_2 that of the second embedded spring when applicable, and k_3 that of the third embedded spring when applicable.											
FREEPLAY	Keyword indicating that the following data entries specify the freeplay parameters for the springs embedded in the joint spring combination elements sharing the material ID number MID (characters). For each embedded spring i ($i = 1, 2, \text{ or } 3$), freeplay is modeled using 5 parameters:											
	<ul style="list-style-type: none"> A pair of lower and upper deflection limits $freeplay_ll_i$ and $freeplay_ul_i$ for defining the "dead zone" $[freeplay_ll_i, freeplay_ul_i]$. Three factors $factor_lzi$, $factor_dzi$, and $factor_uzi$ for defining the effective stiffness coefficient k_i^{eff} as follows 											

$$k_i^{eff} = factor_lzi \times k_i \quad \text{for } \delta_i \leq freeplay_ll_i$$

$$k_i^{eff} = factor_dzi \times k_i \quad \text{for } freeplay_ll_i \leq \delta_i \leq freeplay_ul_i$$

$$k_i^{eff} = factor_uzi \times k_i \quad \text{for } freeplay_ul_i \leq \delta_i$$

where k_i is the stiffness coefficient k_i of the embedded spring i , the deflection δ_i of this spring is a rotation in the case of a rotational spring, and a displacement in the case of a translational spring.

Note however that:

- Currently, freeplay is supported only for the element type 323 which has a single embedded spring with a stiffness coefficient k_1 .
- In a linear dynamic analysis, the freeplay-induced forces and moments are interpreted as configuration-dependent and automatically treated as piecewise constant (see [PIECEWISE](#) in [STATICS](#)).

freeplay_ll1 (freeplay_ll2, or freeplay_ll3) Freeplay deflection lower limit for the first (second, or third) embedded spring. For a torsional spring, the deflection corresponds to the rotation associated with this spring, in which case this parameter should be specified in radians. For a translational spring, the deflection corresponds to the displacement associated with this spring, in which case this parameter should be specified in the unit of length chosen for the computational model.
 freeplay_ul1 (freeplay_ul2, or freeplay_ul3) Freeplay deflection upper limit for the first (second, or third) embedded spring. For a torsional spring, the deflection corresponds to the rotation associated with this spring, in which case this parameter should be specified in radians (real). For a translational spring, the deflection corresponds to the displacement associated with this spring, in which case this parameter should be specified in the unit of length chosen for the computational model.
 factor_lz1 (factor_lz2, or factor_lz3) Freeplay correction factor for the stiffness coefficient k_1 (k_2 , or k_3) for $\delta_1 \leq \text{freeplay_ll}_1$ ($\delta_2 \leq \text{freeplay_ll}_2$, or $\delta_3 \leq \text{freeplay_ll}_3$), where the deflection δ_1 (δ_2 , or δ_3) is a rotation in the case of a rotational spring, and a displacement in the case of a translational spring.
 factor_dz1 (factor_dz2, or factor_dz3) Freeplay correction factor for the stiffness coefficient k_1 (k_2 , or k_3) for $\text{freeplay_ll}_1 \leq \delta_1 \leq \text{freeplay_ul}_1$ ($\text{freeplay_ll}_2 \leq \delta_2 \leq \text{freeplay_ul}_2$, or $\text{freeplay_ll}_3 \leq \delta_3 \leq \text{freeplay_ul}_3$), where the deflection δ_1 (δ_2 , or δ_3) is a rotation in the case of a rotational spring, and a displacement in the case of a translational spring.
 factor_uz1 (factor_uz2, or factor_uz3) Freeplay correction factor for the stiffness coefficient k_1 (k_2 , or k_3) for $\text{freeplay_ul}_1 \leq \delta_1 \leq \text{freeplay_ul}_2$ ($\text{freeplay_ul}_3 \leq \delta_3$), where the deflection δ_1 (δ_2 , or δ_3) is a rotation in the case of a rotational spring, and a displacement in the case of a translational spring.

If the element is a revolute joint-with-driver element (eltyp = 126), or a revolute joint-with-actuator element (eltyp = 226), or a prismatic joint-with-driver element (eltyp = 134), or a prismatic joint-with-actuator element (eltyp = 234), its description includes the prescription of the relative rotation (eltyp = 126), applied moment (eltyp = 226), relative displacement (eltyp = 134), applied force (eltyp = 234), in the form $g(t) = \text{SCALE_FACTOR} * f(t) + \text{SHIFT}$ (eltyp = 126, 134, 226, 234), where SCALE_FACTOR is an amplification factor and $f(t)$ is a time-dependent function governed by FUNCTION_TYPE and up to four parameters a , b , c , and d . The properties of these four elements can be specified as follows.

MID	CONMAT	[CONSTRAINT_METHOD]	FUNCTION_TYPE	SCALE_FACTOR SHIFT	a	b	c	d	SPRINGMAT	k
-----	--------	---------------------	---------------	--------------------	-----	-----	-----	-----	-----------	-----

MID	The material id number from element attribute table.																			
CONMAT	Keyword indicating that the following data entries specify the method chosen for enforcing the constraints also fixed the items below associated with the revolute or prismatic joint-with-driver or joint-with-actuator elements sharing the material id number MID (characters).																			
CONSTRAINT_METHOD	Optional parameter which specifies the method for enforcing the constraints associated with the revolute or prismatic joint-with-driver or joint-with-actuator elements sharing the material id number MID (character or characters and real). The default method is set in CONSTRAINTS and used whenever this entry is omitted.																			
multipliers	The Lagrange multiplier method.																			
elimination	The elimination method.																			
FUNCTION_TYPE	penalty beta	The penalty method. The real-valued parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).																		
augmented	RAMP	The augmented Lagrangian method. The real-valued parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).																		
SINE	Type of the relative rotation/applied moment/relative translation/applied force prescribed by the revolute joint-with-driver/revolute joint-with-actuator/prismatic joint-with-driver/prismatic joint-with-actuator elements sharing the material id number MID (characters). There are five types to choose from:																			
TRIA	Sets $\theta(t) / M(t) / u(t) / F(t)$ to a sinusoidal relative rotation/applied moment/relative translation/applied force partially described by $f(t) = \sin(at + b)$, where a and b are two parameters specified below.																			
$f(t) = 0 \text{ for } t \leq a, \quad f(t) = \frac{t-a}{b-a} \text{ for } a \leq t \leq b, \quad \text{and} \quad f(t) = 1 \text{ for } t \geq b$																				
where a and b are two parameters specified below.																				
Sets $\theta(t) / M(t) / u(t) / F(t)$ to a time-dependent relative rotation/applied moment/relative translation/applied force partially described by a triangular (hat) function of the form																				
$f(t) = 0 \text{ for } t \leq a, \quad f(t) = \frac{t-a}{b-a} \text{ for } a \leq t \leq b, \quad f(t) = \frac{c-t}{c-b} \text{ for } b \leq t \leq c, \quad \text{and} \quad f(t) = 0 \text{ for } t \geq c$																				
where a , b and c are three parameters specified below.																				

TRAP	Sets $\theta(t)/M(t)$ / $u(t)$ / $F(t)$ to a time-dependent relative rotation/applied moment/relative translation/applied force partially described by a trapezoidal function of the form $f(t) = 0$ for $t \leq a$, $f(t) = \frac{t-a}{b-a}$ for $a \leq t \leq b$, $f(t) = 1$ for $b \leq t \leq c$, $f(t) = \frac{d-t}{d-c}$ for $c \leq t \leq d$, and $f(t) = 0$ for $t \geq d$ where a , b , c and d are four parameters specified below.
SSHA	Sets $\theta(t)$ / $M(t)$ / $u(t)$ / $F(t)$ to a time-dependent relative rotation/applied moment/relative translation/applied force partially described by an S-shaped function of the form $f(t) = 0$ for $t \leq a$, $f(t) = \frac{(t-a)^2}{2(b-a)^2}$ for $a \leq t \leq b$, $f(t) = 1 - \frac{(c-t)^2}{2(c-b)^2}$ for $b \leq t \leq c$, and $f(t) = 1$ for $t \geq c$ where a , b and c are three parameters specified below.
SCALE_FACTOR	Constant amplification factor (real).
SHIFT	Constant shift (real).
a	Time parameter (real).
b	Time parameter (real).
c	Time parameter (real).
d	Time parameter (real).
SPRINGMAT	Keyword indicating that the following data entry specifies the stiffness constant of the revolute or prismatic joint-with-actuator elements sharing the material id number MID (characters).
k	Stiffness constant of the revolute or prismatic joint-with-actuator elements sharing the material id number MID (real).

If the element is a thermal element (eltyp = 3, 10, 46-51, 53, 56-58, 81-86, 4646), the material properties are defined as follows.

MID	THRMAT	A	ρ	c_p	h/ϵ	α	k	t	P	Tr
-----	--------	---	--------	-------	--------------	----------	---	---	---	----

MID	The material id number from element attribute table.
THRMAT	keyword that specifies that the following data entries are associated with thermal elements.
A	Cross sectional area for lineal thermal elements (float).
ρ	Mass density per unit volume (float).
c_p	Specific heat coefficient (float).
h/ϵ	Heat convection coefficient for a heat convection or bulk fluid thermal element, or transfer factor for a heat radiation element (float). When radiation is exchanged between two bodies Ω_1 (identified here by this value of MID) and Ω_2 (any other body), the transfer factor \mathcal{F}_{1-2} depends on the emittances of both bodies as well as the geometrical view. In the special case of a gray object in a large environment — that is, when Ω_1 represents a smaller body and Ω_2 a larger isothermal environment (for example, the atmosphere at some temperature) — \mathcal{F}_{1-2} becomes the emissivity ϵ_1 of the first body. In this case, $\mathcal{F}_{1-2} = \epsilon_1 = 1$ if Ω_1 is furthermore a black body.
α	Stefan's constant, also known as the Stefan-Boltzmann constant (in SI units, $\sigma = 5.670400 \times 10^{-8} \text{ Js}^{-1} \text{ m}^{-2} \text{ K}^{-4}$) (float).
k	Heat conduction coefficient (float).
t	Thickness of a (two-dimensional) thermal element (float).
P	Perimeter/circumference area for thermal elements (float), or depth of the boundary where convection (radiation) occurs when element type 47 (56) is used to model non-lateral boundary convection (radiation) (float).
Tr	For a heat radiation element, reference temperature of the enclosure receiving the radiation (float).

If the element is an acoustic element (eltyp = 31-45, 63, 90, 93-96, 98-108), the material properties may be defined as follows.

MID	AMAT	c	ρ
-----	------	---	--------

or

MID	AMAT	c_R	c_I	ρ
-----	------	-------	-------	--------

or

MID	AMAT	<i>c</i>	ρ	pml_type	γ	pmlx1	pmlx2	pmly1	pmly2	pmlz1	pmlz2
-----	------	----------	--------	----------	----------	-------	-------	-------	-------	-------	-------

or

MID	AMAT	<i>c_R</i>	<i>c_I</i>	ρ	pml_type	γ	pmlx1	pmlx2	pmly1	pmly2	pmlz1	pmlz2
-----	------	----------------------	----------------------	--------	----------	----------	-------	-------	-------	-------	-------	-------

*MID**AMAT**c**c_R**c_I* ρ

pml_type

The material id number from element attribute table.

Keyword that specifies that the following data entries are associated with acoustic fluid elements.

Speed of sound in the material identified by *MID* (float).Real part of the speed of sound in the material identified by *MID* (float).Imaginary part of the speed of sound in the material identified by *MID* (float).Density (mass per unit volume) of the material identified by *MID* (float).

Type of PML (Perfectly Matched Layer) (integer).

pml_type = 1 designates a box PML. In this case the PML elements are assumed to be in the region defined by $pmlx1 < x < pmlx2$, $-pmlx2 < x < -pmlx1$, $pmly1 < y < pmly2$, $-pmly2 < y < -pmly1$, $pmlz1 < z < pmlz2$, and $-pmlz2 < z < -pmlz1$.
pml_type = 2 designates a spherical PML. In this case the PML elements are assumed to be in the region defined by $pmlx1 < r < pmlx2$, where $r = \sqrt{x^2 + y^2 + z^2}$.

pml_type = 3 designates a cylindrical PML. In this case the PML elements are assumed to be in the region defined by $pmlx1 < r < pmlx2$, where $r = \sqrt{x^2 + y^2}$, $pmlz1 < z < pmlz2$, and $-pmlz2 < z < -pmlz1$.

PML attenuation parameter (real). Given the speed of sound *c* specified above, the circular frequency $\omega = 2\pi f$ where *f* is the frequency specified in [IMPEDANCE](#) — and therefore the wavenumber $k = \frac{\omega}{c}$ — the

thickness of the PML layer *t*, and this attenuation parameter γ , the PML damps the outgoing waves by the

factor $e^{-ktC(\gamma)}$ where $C(\gamma) = (1 - \frac{1}{\gamma})\frac{5\gamma}{7} + \frac{1}{7^2} - \frac{1}{2}$. The recommended value for γ is $1.5 \leq \gamma \leq 4.0$. Note that

the mesh within the PML must be designed so that it properly resolves the decaying solution. To this effect, also note that in practice, using two cubic elements through the thickness of the PML is sufficient to resolve a decaying solution characterized by a damping factor $e^{-ktC(\gamma)} = 0.001$ — that is, a solution that decays exponentially within the PML, from a relative amplitude of 1 to a relative amplitude of 0.001.

*pmlx1**pmlx2**pmly1**pmly2**pmlz1**pmlz2*

PML geometrical parameter (see above) (real).

If the element is a fabric truss element (*eltyp* = 111), the material properties are defined as follows (see Powell, D.A. and Zohdi, T.I. Attachment mode performance of network-modeled ballistic fabric shielding. *Composites: Part B* 2009; 40: 451-460).

MID	FABMAT	type	<i>E</i>	ρ	<i>A</i>	<i>U_c</i>	<i>U_f</i>	λ	<i>h</i>	<i>d</i>	$\delta\lambda$	<i>n_p</i>	<i>N_f</i>
-----	--------	------	----------	--------	----------	----------------------	----------------------	-----------	----------	----------	-----------------	----------------------	----------------------

*MID**FABMAT**type**E* ρ *A**U_c**U_f* λ *h**d*

The material id number from element attribute table (integer).

Keyword specifying that the following data entries are associated with fabric elements (characters).

Type of fabric material (integer). If *type* = 1, the fabric properties are automatically determined by **AERO-S** using a micro-scale computation. If *type* = 2, they are determined using Gaussian distribution.

If *type* = 1, this is Young's modulus for the fibrils that make up the yarn (float). If *type* = 2, it is the mean value of Young's modulus for the yarn.

Mass density per unit volume of the fibril/yarn (float).

Cross sectional area of the entire yarn element (float).

If *type* = 1, this is the breaking stretch (l/l_0) of the fibrils (float). If *type* = 2, it is the stretch level at which the yarn begins to damage.

Stretch level at which the entire yarn has failed (i.e. the stress response is approximately zero) (float).

If *type* = 1, this is the initial guess for the parameter controlling the damage rate of the yarn; its actual value is positive and determined by Newton's method (float). If *type* = 2, it is the slope of the assumed linear variation of the damage parameter with Young's modulus.

If *type* = 1, this is the length of an unstretched yarn (initial length of the truss element) (float). If *type* = 2, it is the value of the damage parameter for a zero Young's modulus.

If *type* = 1, this is the standard deviation for the inclination distance of the fibrils (misalignment of the fibrils) (float). If *type* = 2, it is the standard deviation associated with the assumed Gaussian distribution of Young's modulus.

$\delta\lambda$	If type = 1, this parameter is set to 0 and ignored (float). If type = 2, it is the standard deviation associated with the assumed Gaussian distribution of the damage parameter.
n_p	If type = 1, this is the maximum number of Newton iterations for fitting the damage parameter λ to micro-scale data (integer). If type = 2, this parameter is ignored.
N_f	If type = 1, this is the number of fibers in a typical yarn (integer). If type = 2, this parameter is ignored.

Next: [LOAD](#), Previous: [MATERIAL](#)

83 LINEAR MULTIPONT CONSTRAINTS FOR MECHANICAL ANALYSIS

Command Statement: **LMPC**

The **LMPC** command can be used to specify a set of *linear* multipoint constraint equations for an otherwise linear or nonlinear (see [NONLINEAR](#)) analysis. These constraints equations can be of the form

$$\sum_{j=1}^{j=n} c_{ij} u_j = r_i, \quad i = 1, 2, \dots$$

or

$$\sum_{j=1}^{j=n} c_{ij} u_j \leq r_i, \quad i = 1, 2, \dots$$

where c_{ij} is a constant coefficient and u_j denotes a degree of freedom of the structural model. There is no limitation on the number of multipoint constraints, or number of degrees of freedom related by the same constraint equation.

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: Whereas the constraints specified under this command must be linear, the degrees of freedom u_j can be governed by nonlinear equations of equilibrium. Therefore, this command is also supported for linear analysis with geometric stiffening due to prestress (see [GEPS](#)), and [NONLINEAR](#) analysis. It is noted however that for linear analysis using [GEPS](#), linear multipoint constraints involving rotational degrees of freedom may be violated by the solution computed by **AERO-S** if the initial displacement field underlying the prestress contains large rotations.

Note 3: The Lagrange multiplier method for enforcing the constraints associated with this command is supported only by the FETI-DP family of iterative solvers, the GMRES solver, and the sparse direct solvers SPOOLES and MUMPS (with pivoting enabled), and in all but explicit dynamic analyses (see [STATICS](#)).

Note 4: For time-dependent problems, the specified initial conditions must verify the specified linear multipoint constraints.

The format of this command statement is as follows.

```
[LMPC]
CONRAINT# RHS [MODE] [mode_v] CONSTRAINT_METHOD
NODE#      DOF# COEFF
.
.
NODE#      DOF# COEFF
```

CONSTRAINT#	This corresponds to the constraint equation number i (integer).
RHS	This is the right-hand side r_i of the i -th constraint equation (real).
MODE	Optional sub-command keyword to specify whether a constraint is of the equality or inequality type (characters).
mode_v	Setting this parameter to 0, which is its default value, specifies that the constraint CONSTRAINT# is of the equality type. Setting it to 1 specifies that it is of the inequality type (integer).
CONSTRAINT_METHOD	This is the method for enforcing the constraint (characters). The default method is set in CONSTRAINTS and used whenever this entry is omitted.
multipliers	The Lagrange multiplier method.
elimination	The elimination method.

[beta]	penalty	The penalty method. The parameter <code>beta</code> should be a large positive number, typically of the order of 10^8 (no default value is provided).
NODE#		This is the number of the node contributing the coefficient c_{ij} of the i -th constraint equation (integer).
DOF#		This is the local number of the degree of freedom at the node specified above contributing the coefficient c_{ij} of the i -th constraint equation (integer).
COEFF		This is the coefficient c_{ij} of the i -th constraint equation (real).

Next: [LOADCASE](#), Previous: [LMPC](#)

84 LOAD

Command Statement:	LOAD
--------------------	-------------

The `LOAD` command statement is used to inform **AERO-S** where the user defined subroutines for user defined forces and/or displacements and/or control are located. An example input file using the `LOAD` command can be found in `FEM.d/fem_examples/Control.d`

LOAD pathandfilename

pathandfilename Specifies **between quotes " "** the path and filename of the `LOAD` file. The extension of this file must be `".so"`.

Next: [LUMPED](#), Previous: [LOAD](#)

85 LOADCASE DEFINITION

Command Statement:	LOADCASE [LOADCASE_ID]
--------------------	-------------------------------

The `LOADCASE` command can be used to define one or more "load" cases. Each "load" case is defined as a linear combination of "load" sets. Defining no "load" case is equivalent to defining a load case 0 which combines the "load" set 0 and all "loads" generated by the commands which do not support the "load" set construct (`LOADSET_ID`). Note also that any "load" generated by a command which does not support the "load" set construct (`LOADSET_ID`) is automatically added to every "load" case defined herein.

The input format of this command is given below.

LOADCASE [LOADCASE_ID]

LOADSET_ID_1 COEFF_1
.
.
.
LOADSET_ID_n COEFF_n

LOADCASE_ID

Optional non-negative integer which uniquely identifies a "load" case. The default value is 0. For static analysis, multiple "load" cases can be defined by repeating this command in the same input file using different values for `LOADCASE_ID` and different data. For dynamic analysis, only one "load" case can be defined and must be attributed the identifier 0.

Integer identifying a "load" set.

LOADSET_ID_1

For static and dynamic analyses, this parameter is a real coefficient that can be used to amplify the "load" set identified by `LOADSET_ID_1` (real). For dynamic analysis, this parameter can also consist of the keyword `MFTT` or `HFTT` followed by an integer identifying a `TABLE_ID` defined in [MFTT](#) or [HFTT](#). In this case, the amplification factor for the "load" set identified by `LOADSET_ID_1` is the time-dependent function defined by `MFTT[TABLE_ID]` or `HFTT[TABLE_ID]`.

COEFF_1

Next: [MASS](#), Previous: [LOADCASE](#)

86 LUMPED

Command Statement:	LUMPED
--------------------	---------------

By default, **AERO-S** computes all element mass matrices and gravity (see [GRAVITY](#)) loads by a consistent approach, **except for explicit dynamic computations** (see [DYNAMICS](#)), in which case **AERO-S** always uses a lumped approach for this purpose. If a consistent mass matrix is not available for a particular element (see [TOPOLOGY](#)), then **AERO-S** uses in all cases a lumped mass matrix and gravity load for

that element.

Alternatively, this command can be used to instruct **AERO-S** to compute all element mass matrices and gravity loads by a lumping method.

LUMPED

Next: [MATLAW](#), Previous: [LUMPED](#)

87 MASS EVALUATION

Command Statement: **MASS**

The **MASS** command statement is used to signal that the user would like that the total mass of the structural system be computed. The result is output on the screen by FEM. In addition to the total mass of the structure, the center of gravity (cg), the center of volume, and the closest node to the cg are also computed and printed on the screen.

Next: [MATUSAGE](#), Previous: [MASS](#)

88 MATERIAL LAW

Command Statement: **MATLAW**

The **MATLAW** command can be used for two different purposes:

- Specify one or several pre-defined material laws that can be:
 - Linear based on linear kinematics.
 - Nonlinear based on linear kinematics.
 - Or nonlinear based on nonlinear kinematics.
- Define and label a material law that can fit any of the above three descriptions and is supported by a user-defined subroutine to be linked with **aeros** via packaging in a library with a ".so" extension. For this purpose, the user should also grab from AERO-S.d/Matlaw.d a special makefile and a few include files. In this case, this command is also needed to pass to the user-defined library the parameters it expects.

Because the **MATERIAL** command can be used to input the parameters of a linear elastic material law, the **MATLAW** and **MATUSAGE** commands should be used primarily for defining and assigning nonlinear material laws. To this end, it is noted that by default, the following material law is assigned by **AERO-S** to an element of a computational structural model:

- For a bar, beam, or shell element, the linear elastic material law. In this case, the elastic properties of this material law should be specified in **MATERIAL** for the material ID of the aforementioned element. In the presence of the **NONLINEAR** command in the ASCII Input Command Data file, the geometrically nonlinear formulation of the internal force vector and tangent stiffness matrix is activated using the corotational approach.
- For a 3D solid element or a membrane element type 128 or 129:
 - In the absence of the **NONLINEAR** command in the ASCII Input Command Data file, the linear elastic material law. Again, the elastic properties of this material law should be specified in this case in **MATERIAL** for the material ID of the aforementioned element.
 - In the presence of the **NONLINEAR** command in the ASCII Input Command Data file, the Saint Venant-Kirchhoff hyperelastic law. This material law is an extension of the linear elastic material model to the nonlinear regime. It has the form

$$\underline{S = \lambda \operatorname{tr}(\epsilon)I + 2\mu \epsilon}$$

where **S** is the second Piola-Kirchhoff stress tensor,

$$\epsilon = \frac{1}{2} [(\nabla u)^T + \nabla u + (\nabla u)^T \cdot \nabla u]$$

is the Green-Lagrange strain tensor, **λ** and **μ** are the Lame constants, and **I** is the second-order unit tensor. The Lame properties of this material law are deduced from the Young modulus **E** and Poisson ratio **ν** specified in **MATERIAL** for the material ID of the aforementioned element as follows

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \quad \text{and} \quad \mu = \frac{E}{2(1+\nu)}$$

The strain-energy density function for the Saint Venant-Kirchhoff model is defined as

$$W(\epsilon) = \frac{\lambda}{2} [\operatorname{tr}(\epsilon)]^2 + \mu \operatorname{tr}(\epsilon^2)$$

and the second Piola-Kirchhoff stress is given by

$$S = \frac{\partial W}{\partial \epsilon}$$

In this case, the geometrically nonlinear formulation of the internal force vector and tangent stiffness matrix is activated using the total Lagrangian method.

Note 1: To be active, this command requires the presence of the [NONLINEAR](#) command in the same ASCII Input Command Data file (in this case, if this command is used to define or specify one or several material laws that are linear and based on linear kinematics, and provided that the computational model does not contain any other source of nonlinearity such as, for example, a follower force, the nonlinear solution strategy specified in [NONLINEAR](#) is guaranteed to converge in a single iteration). Otherwise, this command and its content are ignored, the linear elastic material law is attributed to all elements of the computational structural model, and the properties of this material law should be specified in [MATERIAL](#).

Note 2: In the presence of the [NONLINEAR](#) command in the ASCII Input Command Data file, the material laws and properties specified under this command and assigned using [MATUSAGE](#) take precedence over those inputted using [MATERIAL](#), when a conflict arises.

Note 3: In the absence of the [MATLAW](#) command but presence of the [NONLINEAR](#) command in the ASCII Input Command Data file:

- The linear elastic material law with infinitesimal strains is assigned by default to all beam and shell elements, and the nonlinear geometric effects of these elements are accounted for using the corotational method.
- The Saint Venant-Kirchhoff hyperelastic material law is assigned by default to all 3D solid elements and the membrane elements type 128 and 129, and the nonlinear geometric effects are accounted for using the total Lagrangian method.

Note 4: In the absence of both [NONLINEAR](#) and [MATLAW](#) commands in the ASCII Input Command Data file, the linear elastic material law with infinitesimal strains is assigned by default to all elements of the computational structural model.

Note 5: The various pre-defined material laws that can be specified using this command are supported by different structural and solid mechanics elements (see [TOPOLOGY](#)) as follows:

- The three-dimensional solid elements (type 17, 23, 24, 25, 72, 91, 92, and 97) support the pre-defined material laws [Linear](#), [HenckyElastic](#), [MooneyRivlin](#), [NeoHookean](#), [Ogden](#), [StVenantKirchhoff](#), [BilinearPlastic](#), [FiniteStrainPlastic](#), [LogStrainPlastic](#), [SimoPlastic](#), [ViscoLinearElastic](#), [ViscoMooneyRivlin](#), [ViscoNeoHookean](#), and [ViscoStVenantKirchhoff](#).
- The shell elements 15 and 1515 support the pre-defined material laws [J2Plasticity](#), [PlaneStressBilinearPlastic](#), [ShellFabricMap](#), [ShellFabricMat](#), [ShellViscoFabricMap](#), and [ShellViscoFabricMat](#).
- The shell element 16 supports the pre-defined material laws [HypoElastic](#), [J2Plasticity](#), [KK1](#), and [KK2](#).
- The membrane elements 128 and 129 support the pre-defined material laws [LinearPlaneStress](#), [HyperElasticPlaneStress](#), [PlaneStressMooneyRivlin](#), [PlaneStressNeoHookean](#), [PlaneStressBilinearPlastic](#), [PlaneStressFiniteStrainPlastic](#), [ViscoHyperElasticPlaneStress](#), [ViscoLinearPlaneStress](#), [PlaneStressViscoMooneyRivlin](#), [PlaneStressViscoNeoHookean](#), [FabricMap](#), [FabricMat](#), [ViscoFabricMap](#), and [ViscoFabricMat](#).

Note 6: The element deletion method is supported only for nonlinear explicit dynamic computations. In such computations, the mass matrix is lumped. For this reason, and because no node is deleted when an element is deleted from the finite element model, neither the mass matrix nor a gravity load is affected by such an event, and mass is conserved.

Note 7: Anisotropic material laws defined using this command (including user-defined and pre-defined materials such as [FabricMat](#), [FabricMap](#), and [TorchMat](#)) are typically defined using a local material frame of reference specified using the [CFRAMES](#) command.

The input format of this command is given below.

WARNING: for formatting issues pertaining to this User's Reference Manual only, the data associated with some pre-defined material laws has been described below on separate lines; when preparing an AERO-S ASCII Input Command Data file however, this data should be specified on a single line.

MATLAW			
READ	MATERIAL_NAME	<pathandfilename>	
MATERIAL_ID	MATERIAL_NAME	PARA#1	...
MATERIAL_ID	Linear	ρ	E
		T_a	w
			TulerButcher
		σ_0	β_{TB}
			K_f
MATERIAL_ID	LinearPlaneStress	p	E
		t	$[T_a \quad w]$
MATERIAL_ID	HyperElasticPlaneStress	p	E
		t	$[T_a \quad w]$
MATERIAL_ID	HenckyElastic	ρ	E
		T_a	w
			TulerButcher
		σ_0	β_{TB}
			K_f
MATERIAL_ID	HypoElastic	E	ν
			ρ

MATERIAL_ID MooneyRivlin	ρ	c_1	c_2	
	c	[TulerButcher	σ_0	
	β_{TB}	K_f]	
MATERIAL_ID NeoHookean	ρ	E	ν	
	[TulerButcher	σ_0	β_{TB}	
		K_f]	
MATERIAL_ID Ogden	ρ	μ_1	μ_2	
	[$\mu_3 \dots \mu_9$]	α_1	α_2	
	[$\alpha_3 \dots \alpha_9$]	K_1	[K_2]	
MATERIAL_ID StVenantKirchhoff	ρ	E	ν	
	[T_a]	w	TulerButcher	
	σ_0	β_{TB}	K_f]
MATERIAL_ID BilinearPlastic	ρ	E	ν	
	E_T	σ_{yield}	β	
	[T_a]	w	MAXEPS	
MATERIAL_ID FiniteStrainPlastic	ρ	E	ν	
	E_T	σ_{yield}	β	
	[T_a]	w	MAXEPS	
MATERIAL_ID J2Plasticity	ρ	E	ν	
	σ_{yield}	K	ρ	
			H	
MATERIAL_ID KK1	ρ	E	ν	
	σ_{yield}	K	ρ	
			H	
MATERIAL_ID KK2	ρ	E	ν	
	σ_{yield}	K	ρ	
MATERIAL_ID LogStrainPlastic	ρ	E	ν	
	E_T	σ_{yield}	β	
	[T_a]	w	MAXEPS	
MATERIAL_ID SimoPlastic	ρ	E	ν	
	E_T	σ_{yield}	β	
MATERIAL_ID ViscoLinearElastic	ρ	E	ν	
	[T_a]	w	g_1	
	τ_1	g_2	τ_2	
	g_3	τ_3	TulerButcher	
	σ_0	β_{TB}	K_f]
MATERIAL_ID ViscoLinearPlaneStress	ρ	E	ν	

	t	[T_a]	\underline{w}	
	$\underline{g_1}$	τ_1	$\underline{g_2}$	
	$\underline{\tau_2}$	—	—	
MATERIAL_ID ViscoHyperElasticPlaneStress	$\underline{\rho}$	\underline{E}	$\underline{\nu}$	
	t	[T_a]	\underline{w}	
	$\underline{g_1}$	τ_1	$\underline{g_2}$	
	$\underline{\tau_2}$	$\underline{g_3}$	$\underline{\tau_3}$	
MATERIAL_ID ViscoMooneyRivlin	$\underline{\rho}$	c_1	c_2	
	c	$\underline{g_1}$	τ_1	
	$\underline{g_2}$	τ_2	$\underline{g_3}$	
	$\underline{\tau_3}$	[TulerButcher	σ_0	
	β_{TB}	K_f	—	
MATERIAL_ID ViscoNeoHookean	$\underline{\rho}$	\underline{E}	$\underline{\nu}$	
	$\underline{g_1}$	τ_1	$\underline{g_2}$	
	$\underline{\tau_2}$	$\underline{g_3}$	$\underline{\tau_3}$	
	[TulerButcher	σ_0	β_{TB}	
		—		
		K_f	—	
MATERIAL_ID ViscoStVenantKirchhoff	$\underline{\rho}$	\underline{E}	$\underline{\nu}$	
	$\underline{[T_a]}$	\underline{w}	g_1	
	$\underline{\tau_1}$	$\underline{g_2}$	$\underline{\tau_2}$	
	$\underline{g_3}$	$\underline{\tau_3}$	[TulerButcher	
	$\underline{\sigma_0}$	β_{TB}	K_f	
MATERIAL_ID PlaneStressMooneyRivlin	$\underline{\rho}$	c_1	c_2	
	c	t	[TulerButcher	
	σ_0	β_{TB}	K_f	
MATERIAL_ID PlaneStressNeoHookean	$\underline{\rho}$	\underline{E}	$\underline{\nu}$	
	t	[TulerButcher	σ_0	
	β_{TB}	K_f	—	
MATERIAL_ID PlaneStressBilinearPlastic	$\underline{\rho}$	\underline{E}	$\underline{\nu}$	
	E_T	σ_{yield}	β	
	$\underline{[T_a]}$	\underline{w}	t	
MATERIAL_ID PlaneStressFiniteStrainPlastic	$\underline{\rho}$	\underline{E}	$\underline{\nu}$	YSSFSRT-ID
	E_T	σ_{yield}	β	
	$\underline{[T_a]}$	\underline{w}	t	
MATERIAL_ID PlaneStressViscoMooneyRivlin	$\underline{\rho}$	$\underline{c_1}$	c_2	YSSFSRT-ID
	c	$\underline{g_1}$	τ_1	
	$\underline{g_2}$	τ_2	$\underline{g_3}$	

	τ_3	t	[TulerButcher]
	σ_0	β_{TB}	K_f]
MATERIAL_ID PlaneStressViscoNeoHookean	ρ	E	ν
	g_1	τ_1	g_2
	τ_2	g_3	τ_3
	t	[TulerButcher	σ_0]
	β_{TB}	K_f]	
MATERIAL_ID FabricMap	ρ	XX-SS2DT-ID	YY-SS2DT-ID
	XY-SS1DT-ID	t	[T_a
MATERIAL_ID ShellFabricMap	ρ	XX-SS2DT-ID	YY-SS2DT-ID
	XY-SS1DT-ID	t	[T_a
MATERIAL_ID FabricMat	ρ	XX-YMST-ID	YY-YMST-ID
	G_{xy}	ν_{xy}	ν_{yx}
	t	[T_a	w]
MATERIAL_ID ShellFabricMat	ρ	XX-YMST-ID	YY-YMST-ID
	G_{xy}	ν_{xy}	ν_{yx}
	t	[T_a	w]
MATERIAL_ID ViscoFabricMap	ρ	XX-SS2DT-ID	YY-SS2DT-ID
	XY-SS1DT-ID	t	[T_a
	w]	g_1	τ_1
	g_2	τ_2	g_3
	τ_3		
MATERIAL_ID ShellViscoFabricMap	ρ	XX-SS2DT-ID	YY-SS2DT-ID
	XY-SS1DT-ID	t	[T_a
	w]	g_1	τ_1
	g_2	τ_2	g_3
	τ_3		
MATERIAL_ID ViscoFabricMat	ρ	XX-YMST-ID	YY-YMST-ID
	G_{xy}	ν_{xy}	ν_{yx}
	t	[T_a	w]
	g_1	τ_1	g_2
	τ_2	g_3	τ_3
MATERIAL_ID ShellViscoFabricMat	ρ	XX-SS2DT-ID	YY-SS2DT-ID
	XY-SS1DT-ID	t	[T_a
	w]	g_1	τ_1
	g_2	τ_2	g_3
	τ_3		

MATERIAL_ID TorchMat	ρ	module	t
	a_{cf}	a_{xx}	a_{yy}
	\underline{a}_{xy}	$\underline{\underline{T}}_a$	$\underline{\underline{w}}$
	c_{00}	c_{01}	c_{02}
	c_{11}	c_{12}	c_{22}
	b_0	b_1	b_2
	b_3	b_4	b_5

READ Sub-command keyword used to introduce a material name (characters) and associate it with the library located and stored in <pathandfilename.so> (characters) and linked with **aeros**.

MATERIAL_ID ID number of the material law (integer).

MATERIAL_NAME Name of the material law (characters).

<pathandfilename> Name of the file (including path, if needed) containing the material law library. This entry must be specified between quotes (" "), and the file it specifies must have the extension .so.

PARA#1 First parameter expected by the library <pathandfilename.so>.

PARA#N Last parameter expected by the library <pathandfilename.so>.

Linear Name of the pre-defined standard linear elastic material law with linear kinematics (characters). Its conjugate strain and stress are the infinitesimal strain and the first Piola-Kirchhoff stress. This material law has an anisotropic variant which can be specified using the sub-command COEF of **COMPOSITE**: in this case, the constitutive matrix and coefficients of thermal expansion are defined using the information inputted under COEF, and the parameters ρ , E , ν , and w inputted under this command are ignored.

LinearPlaneStress Name of the pre-defined material law that is the plane stress counterpart of the linear elastic material law Linear described above (characters).

HyperElasticPlaneStress Name of the pre-defined finite strain hyperelastic material law that is the plane stress counterpart of the material law stvenantKirchhoff described above (characters).

HenckyElastic Name of another pre-defined finite strain hyperelastic material law (appropriate for larger strains) (characters). Its conjugate strain and stress are the Lagrangian Hencky (engineering) strain and the rotated Kirchhoff stress. This material law has an anisotropic variant which can be specified using the COEF sub-command of **COMPOSITE**, in which case the parameters ρ , ν , and E inputted under this command are

ignored.

HypoElastic Name of the pre-defined plane stress version of the hypoelastic material law (characters).

MooneyRivlin Name of the pre-defined compressible version of the finite strain Mooney-Rivlin hyperelastic material law (characters).

NeoHookean Name of the pre-defined compressible version of the finite strain hyperelastic material law (characters).

Ogden (with K_1 only) Name of the pre-defined Ogden hyperelastic material law with 1 to 9 terms in the Ogden series that defines the strain energy density in terms of the deviatoric principal stretches $(\bar{\lambda}_1, \bar{\lambda}_2, \bar{\lambda}_3)$ characterizing the distortional response, and a 1-term bulk volumetric response function

$$\Psi(\bar{\lambda}_1, \bar{\lambda}_2, \bar{\lambda}_3, J) = \sum_{p=1}^N \frac{\mu_p}{\alpha_p} (\bar{\lambda}_1^{\alpha_p} + \bar{\lambda}_2^{\alpha_p} + \bar{\lambda}_3^{\alpha_p} - 3) + \frac{K_1}{2} (J - 1)^2$$

Here, J denotes the determinant of the deformation gradient, μ_1 to μ_9 and α_1 to α_9 are material properties that characterize the distortional response of the material, and K_1 is its bulk modulus.

Ogden (with K_1 and K_2) Name of the pre-defined Ogden hyperelastic material law with 2 to 9 terms in the Ogden series that defines the strain energy density in terms of the deviatoric principal stretches $(\bar{\lambda}_1, \bar{\lambda}_2, \bar{\lambda}_3)$ characterizing the distortional response, and a 2-term bulk volumetric response function

$$\Psi(\bar{\lambda}_1, \bar{\lambda}_2, \bar{\lambda}_3, J) = \sum_{p=1}^N \frac{\mu_p}{\alpha_p} (\bar{\lambda}_1^{\alpha_p} + \bar{\lambda}_2^{\alpha_p} + \bar{\lambda}_3^{\alpha_p} - 3) + \frac{K_1}{2} (J - 1)^2 + \frac{K_2}{2} (J - 1)^4$$

Here, J denotes the determinant of the deformation gradient, μ_1 to μ_9 and α_1 to α_9 are material properties that characterize the distortional response of the material, and K_1 and K_2 are material properties that characterize its volumetric response.

StVenantKirchhoff Name of a pre-defined finite strain hyperelastic material law (appropriate for metals and moderate strains) (characters). Its conjugate strain and stress are the Green-Lagrange (engineering) strain - which introduces in the problem to be solved a geometric nonlinearity - and the second Piola-Kirchhoff stress. This material law has

an anisotropic variant which can be specified using the `coef` sub-command of [COMPOSITE](#), in which case the parameters `[P]`, `[V]`, and `[E]` inputted under this command are ignored.

BilinearPlastic	Name of a pre-defined infinitesimal strain bilinear plastic material law featuring kinematic hardening (characters). Its conjugate strain and stress are the infinitesimal strain and the first Piola-Kirchhoff stress.
FiniteStrainPlastic	Name of a pre-defined finite strain plastic material law featuring kinematic hardening (characters). Its conjugate strain and stress are the Green-Lagrange (engineering) strain - which introduces in the problem to be solved a geometric nonlinearity - and the second Piola-Kirchhoff stress.
J2Plasticity	Name of the pre-defined plane stress version of the von Mises plasticity material law based on the second stress invariant featuring kinematic and/or isotropic hardening (characters). Its conjugate strain and stress are the infinitesimal strain and the first Piola-Kirchhoff stress.
KK1	Name of the pre-defined plane stress plasticity material law due to Korkolis and Kydiakides featuring kinematic and/or isotropic hardening and an advanced yield function including deformation-induced anisotropy (characters). Its conjugate strain and stress are the infinitesimal strain and the first Piola-Kirchhoff stress.
KK2	Name of the pre-defined plane stress plasticity material law for aluminum Al-6260-T4 due to Korkolis and Kydiakides featuring an advanced yield function including deformation-induced anisotropy and an experimental-based stress-strain curve (characters). Its conjugate strain and stress are the infinitesimal strain and the first Piola-Kirchhoff stress.
LogStrainPlastic	Name of another pre-defined finite strain plastic material law featuring logarithmic strains and kinematic hardening (characters). Its conjugate strain and stress are the Lagrangian Hencky (engineering) strain and the rotated Kirchhoff stress.
SimoPlastic	Name of a pre-defined finite strain elasto-plastic material law formulated in the logarithmic principal stretches and based on a multiplicative decomposition of the deformation gradient (see <i>J. C. Simo, "Algorithms for Static and Dynamic Multiplicative Plasticity that Preserve the Classical Return Mapping Schemes of the Infinitesimal Theory," Computer Methods in Applied Mechanics and Engineering, Vol. 99, pp. 61-112 (1992)</i>)
ViscoLinearElastic	Name of a pre-defined infinitesimal strain linear elastic material law equipped with a 3-term Prony series viscoelastic model (see, for example, <i>S.M. Goh, M.N. Charalambides, J.G. Williams, "Determination of the Constitutive Constants of Non-Linear Viscoelastic Materials," Mechanics of Time-Dependent Materials, Vol. 8, pp 255-268 (2004)</i>).
ViscoLinearPlaneStress	Name of a pre-defined infinitesimal strain plane-stress material law for membrane elements (types 128 and 129, only), based on the material law <code>LinearPlaneStress</code> with a Prony series viscoelastic model (characters).
ViscoHyperElasticPlaneStress	Name of a pre-defined finite strain plane-stress material law for membrane elements (types 128 and 129, only), based on the material law <code>HyperElasticPlaneStress</code> with a Prony series viscoelastic model (characters).
ViscoMooneyRivlin	Name of a pre-defined finite strain Mooney-Rivlin material law equipped with a 3-term Prony series viscoelastic model (see, for example, <i>S.M. Goh, M.N. Charalambides, J.G. Williams, "Determination of the Constitutive Constants of Non-Linear Viscoelastic Materials," Mechanics of Time-Dependent Materials, Vol. 8, pp 255-268 (2004)</i>).
ViscoNeoHookean	Name of a pre-defined finite strain Neo-Hookean material law equipped with a 3-term Prony series viscoelastic model (see, for example, <i>S.M. Goh, M.N. Charalambides, J.G. Williams, "Determination of the Constitutive Constants of Non-Linear Viscoelastic Materials," Mechanics of Time-Dependent Materials, Vol. 8, pp 255-268 (2004)</i>).
ViscoStVenantKirchhoff	Name of a pre-defined finite strain hyperelastic material law equipped with a 3-term Prony series viscoelastic model (see, for example, <i>S.M. Goh, M.N. Charalambides, J.G. Williams, "Determination of the Constitutive Constants of Non-Linear Viscoelastic Materials," Mechanics of Time-Dependent Materials, Vol. 8, pp 255-268 (2004)</i>).
PlaneStressMooneyRivlin	Name of the pre-defined material law that is the plane stress counterpart of the material law <code>MooneyRivlin</code> described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in <i>S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)</i> , which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressNeoHookean	Name of the pre-defined material law that is the plane stress counterpart of the material law <code>NeoHookean</code> described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in <i>S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)</i> , which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressBilinearPlastic	Name of the pre-defined material law that is the plane stress counterpart of the material law <code>BilinearPlastic</code> described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in <i>S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)</i> , which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressFiniteStrainPlastic	Name of the pre-defined material law that is the plane stress counterpart of the material law <code>FiniteStrainPlastic</code> described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in <i>S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)</i> , which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressViscoMooneyRivlin	Name of the pre-defined finite strain material law that is the plane stress counterpart of the material law <code>ViscoMooneyRivlin</code> described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in <i>S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)</i> , which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressViscoNeoHookean	Name of the pre-defined finite strain material law that is the plane stress counterpart of the material law <code>ViscoNeoHookean</code> described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in <i>S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)</i> , which involves solving a nonlinear equation at each Gauss point of an element.

FabricMap	Name of a pre-defined finite strain plane-stress material law for membrane elements (types 128 and 129, only) in which the stress response is given exclusively by tables (characters). This material law is equivalent to material 034M (MAT_FABRIC_MAP) in LS-DYNA.
ShellFabricMap	Name of a pre-defined infinitesimal strain plane-stress material law for shell elements (types 15 and 1515, only) in which the stress response is given exclusively by tables (characters). This material law is equivalent to material 034M (MAT_FABRIC_MAP) in LS-DYNA.
FabricMat	Name of a pre-defined finite strain orthotropic plane-stress material law for membrane elements (types 128 and 129, only) in which the material Young moduli are specified in tables (characters). This material law is equivalent to material 034 (MAT_FABRIC) in LS-DYNA.
ShellFabricMat	Name of a pre-defined infinitesimal strain orthotropic plane-stress material law for shell elements (types 15 and 1515, only) in which the material Young moduli are specified in tables (characters). This material law is equivalent to material 034 (MAT_FABRIC) in LS-DYNA.
ViscoFabricMap	Name of a pre-defined finite strain plane-stress material law for membrane elements (types 128 and 129, only), based on the material law FabricMap with a Prony series viscoelastic model (characters).
ShellViscoFabricMap	Name of a pre-defined infinitesimal strain plane-stress material law for shell elements (types 15 and 1515, only), based on the material law ShellFabricMap with a Prony series viscoelastic model (characters).
ViscoFabricMat	Name of a pre-defined finite strain plane-stress material law for membrane elements (types 128 and 129, only), based on the material law FabricMat with a Prony series viscoelastic model (characters).
ShellViscoFabricMat	Name of a pre-defined infinitesimal strain plane-stress material law for shell elements (types 15 and 1515, only), based on the material law ShellFabricMat with a Prony series viscoelastic model (characters).
TorchMat	Name of a pre-defined plane-stress material law for membrane elements (types 128 and 129, only), in which the stress response is given by a regression artificial neural network function (characters). The constitutive function associated with this material law can be written as

$$[S_{xx} \ S_{yy} \ S_{xy}]^T = C [E_{xx} \ E_{yy} \ E_{xy}]^T + a_{cf} N (E_{xx}/a_{xx}, E_{yy}/a_{yy}, 2E_{xy}/a_{xy})$$

where C is a 3×3 symmetric matrix, N is the regression artificial neural network function, S is the second Piola-Kirchhoff stress tensor, E is the Green-Lagrange strain, and c_f is a user-specified scaling factor for the outputs of the regression artificial neural network function.

Mass density per unit volume (real).

ρ

Material Young modulus (real).

E

Material Poisson ratio (real).

ν

Ambient (reference) temperature of the element assigned this material law (real). The default value is 0.

WARNING: if this value is different from the nodal temperature (see TEMPERATURES) of that element, it causes the generation of a thermal load even in a pure structural analysis. This parameter can be skipped (and therefore is optional) if the parameter w is simultaneously skipped (see below).

Coefficient of thermal expansion (real). The default value is 0. This parameter can be skipped (and therefore is optional) if the parameter T_a is simultaneously skipped (see above).

T_a

w

Optional sub-command keyword (characters) for specifying the Tuler-Butcher brittle failure criterion and its parameters. When a material is subjected to dynamic loading, crack propagation strongly depends on the strain rate, stress wave amplitude, and the exposure time. In this case, the Tuler-Butcher failure criterion can be used as a material brittle failure criterion. It can be expressed as

$$\int_0^{t_f} (\sigma_1 - \sigma_0)^{\beta_{TB}} dt \geq K_f$$

for $\sigma_1 \geq \sigma_0 \geq 0$, where σ_1 denotes the maximum principal stress, σ_0 is a specific threshold stress, t_f is time for the fracture and K_f is the stress impulse for failure. Note that this brittle damage model degenerates to the maximum principle stress model when K_f is set to zero. Specifying this sub-command keyword and inputting after it the parameters of the Tuler-Butcher failure criterion activates the method of element deletion during a nonlinear dynamic computation. Specifically, the stress at a Gauss point is set in this case to zero when the damage accumulation function described above becomes greater than or equal to K_f at this point.

When this criterion is met at all Gauss points of an element, this element is deleted from the finite element model.

σ_0

Specific threshold stress parameter of the Tuler-Butcher brittle failure criterion (real).

β_{TB}

Exponent in the damage accumulation function of the Tuler-Butcher brittle failure criterion (real).

K_f

Stress impulse failure (or macroscopic fracture threshold) parameter of the Tuler-Butcher brittle failure criterion (real).

c_1, c_2, c

Coefficients of the compressible version of the Mooney-Rivlin model (real) where the strain energy density function can be written as

$$\Psi(J, I_1, I_2) = c(J-1)^2 - 2(c_1 + 2c_2)\ln J + c_1(I_1 - 3) + c_2(I_2 - 3)$$

and J denotes the determinant of the deformation gradient and I_1 , I_2 and I_3 are its three invariants.

$\mu_1 - \mu_9$	Material properties of the pre-defined hyperelastic material law named <code>ogden</code> that characterize the distortional response of the material (real).
$\alpha_1 - \alpha_9$	Material properties of the pre-defined hyperelastic material law named <code>ogden</code> that characterize the distortional response of the material (real).
K_1	Material property (or Bulk modulus) of the pre-defined hyperelastic material law named <code>ogden</code> that characterizes the volumetric response of the material (real).
K_2	Material property of the pre-defined hyperelastic material law named <code>ogden</code> that characterizes the volumetric response of the material (real).
E_T	Tangent modulus (for any material with linear hardening and requiring the explicit input of this parameter) representing the slope of the inelastic portion of uniaxial stress vs uniaxial strain curve (real). Note that if a YSST curve is specified for the material, this parameter is used only to define the kinematic hardening modulus and the curve is used to obtain the isotropic hardening modulus..
σ_{yield}	Yield stress for the bilinear, finite strain, J2, and KK1 plastic material laws (real). If a negative integer value <code>-INT</code> (note the minus sign) is specified for this otherwise positive real parameter, the yield stress is understood to be that function of the effective plastic strain that is tabulated in YSST and identified by <code>curve_id = INT</code> .
β	Free-parameter of the hardening model (real): $\beta = 0$ corresponds to pure kinematic hardening, $\beta = 1$ to pure isotropic hardening, and $0 \leq \beta \leq 1$ to a combination of kinematic and isotropic hardening.
<code>MAXEPS</code>	Effective plastic strain at failure (default value is infinity). Specifying a finite and strictly positive value for this parameter activates the method of element deletion during a nonlinear explicit dynamic computation. Specifically, the stress at a Gauss point is set in this case to zero when the effective plastic strain becomes greater than or equal to <code>MAXEPS</code> at this point. When this criterion is met at all Gauss points of an element, this element is deleted from the finite element model.
ϵ	Relative tolerance for the convergence of the nonlinear material solver at each material point (real). The default value is 10^{-6} .
<code>YSSFSRT-ID</code>	Integer ID (integer) of the yield stress scaling factor vs effective plastic strain rate curve (see YSSFSRT). Omitting this parameter sets the default mode of computations where the yield stress scaling factor is set to <code>1</code> and therefore its dependence on the effective plastic strain rate is ignored.
K	Isotropic hardening modulus for the J2 plasticity and KK1 material laws (real). Note that in theory, $K = \beta * E * E_T / (E - E_T)$, but in AERO-S , K is specified directly for these material laws. Note also that if a YSST curve is specified for the material, then this parameter is ignored.
H	Kinematic hardening modulus for the J2 plasticity and KK1 material laws (real). Note that in theory, $H = (1 - \beta) * E * E_T / (E - E_T)$, but in AERO-S , H is specified directly for these material laws.
$g_1 - g_3$	Characteristic amplitudes for the 3-term Prony series viscoelastic model used in the pre-defined material laws <code>ViscoLinearElastic</code> , <code>ViscoNeoHookean</code> , and <code>ViscoMooneyRivlin</code> (real).
$\tau_1 - \tau_3$	Relaxation times for the 3-term Prony series viscoelastic model used in the pre-defined material laws <code>ViscoLinearElastic</code> , <code>ViscoNeoHookean</code> , and <code>ViscoMooneyRivlin</code> (real).
t	Element material thickness (real).
<code>XY-SS1DT-ID</code>	Integer ID (integer) identifying the one-dimensional table (i.e. curve) giving the local second Piola-Kirchhoff xy -stress as a function of the local Green-Lagrange xy -strain (see SS1DT).
<code>XX-SS2DT-ID</code>	Integer ID (integer) identifying the two-dimensional table (i.e. surface) giving the local xx -stress as a function of the local xx -strain and yy -strain (see SS2DT).
<code>YY-SS2DT-ID</code>	Integer ID (integer) identifying the two-dimensional table (i.e. surface) giving the local yy -stress as a function of the engineering local yy -strain and xx -strain (see SS2DT).
<code>XX-YMST-ID</code>	Integer ID (integer) identifying the one-dimensional table (i.e. curve) giving the material Young modulus E_{xx} as a function of the local xx -strain (see YMST).
<code>YY-YMST-ID</code>	Integer ID (integer) of the one-dimensional table (i.e. curve) giving the material Young modulus E_{yy} as a function of the local yy -strain (see YMST).
G_{xy}	Material shear modulus (real).
ν_{xy}	Material Poisson ratio (real).
ν_{yx}	Material Poisson ratio (real).
<code>module</code>	Path and filename of a <code>PyTorch</code> module encoding the regression artificial neural network function underlying the material law <code>TorchMat</code> (characters).
c_f	Scaling factor for the outputs of the regression artificial neural network function underlying the material law <code>TorchMat</code> (float). It can be used, for example, to allow using different systems of units for the the second Piola-Kirchhoff stress tensor S when training offline the regression artificial neural network and exploiting it online in the constitutive model shown above, in which case c_f should be set to the appropriate unit conversion factor. The default value is <code>1</code> .
a_{xx}	Scaling factor for the E_{xx} component of the input of the regression artificial neural network function underlying the material law <code>TorchMat</code> (float). For consistency, this factor must be the same as that used during the offline training of the regression artificial neural network for the purpose of optimal convergence.

a_{yy}	Scaling factor for the E_{yy} component of the input of the regression artificial neural network function underlying the material law <code>TorchMat(float)</code> . For consistency, this factor must be the same as that used during the offline training of the regression artificial neural network for the purpose of optimal convergence.
a_{xy}	Scaling factor for the E_{xy} component of the input of the regression artificial neural network function underlying the material law <code>TorchMat(float)</code> . For consistency, this factor must be the same as that used during the offline training of the regression artificial neural network for the purpose of optimal convergence.
c_{ij}	Coefficient located at the i -th row and j -th column of the matrix C defining the linear term of the regression artificial neural network function underlying the material law <code>TorchMat</code> (see above), $i = 1, \dots, 3, j = 1, \dots, 3$ (float). The default value is $C = 0$. When a non-zero matrix C is specified, the regression artificial neural network function can be interpreted as a correction to a linear stress-strain model (see above).
b_i	i -th coordinate of a strain-space bounding box used by the material law <code>TorchMat</code> as follows (float). The contribution of the regression artificial neural network function to the linear stress-strain model (see above) is added only when the components of the strain tensor satisfy the following conditions:

$$b_0 \leq E_{xx} \leq b_3; \quad b_1 \leq E_{yy} \leq b_4; \quad \text{and} \quad b_2 \leq 2E_{xy} \leq b_5$$

The default values are as follows: $b_i = -\infty, i = 0, \dots, 2$; and $b_i = +\infty, i = 3, \dots, 5$.

Next: [DECOMPOSE](#), Previous: [MATLAW](#)

89 MATERIAL LAW USAGE

Command Statement: **MATUSAGE**

The `MATUSAGE` command can be used to assign a material law specified under [MATLAW](#) to one or several elements.

Note 1: Because the [MATERIAL](#) command is used to input the parameters of the linear elastic material law, the [MATLAW](#) and [MATUSAGE](#) commands should be used primarily for defining and assigning nonlinear material laws. To this end, it is noted that by default, the following material law is assigned by **AERO-S** to an element of a computational structural model:

- For a bar, beam, or shell element, the linear elastic material law. In this case, the elastic properties of this material law should be specified in [MATERIAL](#), for the material ID of the aforementioned element. In the presence of the [NONLINEAR](#) command in the ASCII Input Command Data file, the geometrically nonlinear formulation of the internal force vector and tangent stiffness matrix is activated using the corotational approach.
- For a 3D solid element or a membrane element type 128 or 129:
 - In the absence of the [NONLINEAR](#) command in the ASCII Input Command Data file, the linear elastic material law. Again, the elastic properties of this material law should be specified in [MATERIAL](#), for the material ID of the aforementioned element.
 - In the presence of the [NONLINEAR](#) command in the ASCII Input Command Data file, the Saint Venant-Kirchhoff hyperelastic law. In this case, the Young modulus and Poisson ratio should be specified in [MATERIAL](#), for the material ID of the aforementioned element (see [MATLAW](#) for further details); and the geometrically nonlinear formulation of the internal force vector and tangent stiffness matrix is activated using the total Lagrangian method.

Note 2: To be active, this command requires the presence of the [NONLINEAR](#) command in the same ASCII Input Command Data file (in this case, if this command is used to assign one or several material laws defined in [MATLAW](#) that are linear and based on linear kinematics, and provided that the computational model does not contain any other source of nonlinearity such as, for example, a follower force, the nonlinear solution strategy specified in [NONLINEAR](#) is guaranteed to converge in a single iteration). Otherwise, this command is ignored, the linear elastic material law is assigned to all elements of the computational structural model, and the properties of this material law should be set in [MATERIAL](#).

Note 3: In the presence of the [NONLINEAR](#) command in the ASCII Input Command Data file, the material laws and properties specified under the [MATLAW](#) command and assigned using this command take precedence over those inputted using [MATERIAL](#), when a conflict arises.

Note 4: In the absence of the [MATLAW](#) command but presence of the command [NONLINEAR](#) in the ASCII Input Command Data file:

- The linear elastic material law with infinitesimal strains is assigned by default to all beam and shell elements, and the nonlinear geometric effects of these elements are accounted for using the corotational method.
- The Saint Venant-Kirchhoff hyperelastic material law is assigned by default to all 3D solid elements and the membrane elements type 128 and 129, and the nonlinear geometric effects are accounted for using the total Lagrangian method.

Note 5: In the absence of both the [NONLINEAR](#) and [MATLAW](#) commands in the ASCII Input Command Data file, the linear elastic material law with infinitesimal strains is assigned by default to all elements of the computational structural model.

The input format of this command is given below.

MATUSAGE

ELEMENT#	MATLAW_LABEL#
----------	---------------

or

STARTING_ELEMENT#	ENDING_ELEMENT#	MATLAW_LABEL#
-------------------	-----------------	---------------

ELEMENT#	Element number whose material law number is to be specified (integer).
MATLAW_LABEL	Material law identification number (integer).
MATERIAL_LABEL	First element of a sequence of elements that have the same MATLAW_LABEL (integer).
MATERIAL_NAME	Last element of a sequence of elements that have the same MATLAW_LABEL (integer).

Next: [NODALCONTACT](#), Previous: [MATUSAGE](#)

90 MESH DECOMPOSITION

Command Statement:	DECOMPOSE
--------------------	------------------

The command **DECOMPOSE** can be used to perform an element-based mesh partitioning (or domain decomposition) of the computational domain, and exit or continue with a finite element analysis. For this purpose, four different mesh partitioning strategies suitable for different objectives are available:

- **STRAT1**, which is a trivial, one-step strategy intended for parallel element sampling (see [RMSHC](#)). This strategy partitions the given mesh as follows:
 - Let **NELES** denote the number of elements in the mesh of interest, and let **NSUBS** denote the desired number of subdomains. If $NELES\%NSUBS = 0$, then elements 1 to $NELES/NSUBS$ are assigned to subdomain 1, elements $NELES/NSUB + 1$ to $2NELES/NSUBS$ are assigned to subdomain 2, ... Because **AERO-S** allows gaps in the element numbering, element 1 refers here to the element of the mesh with the lowest ID number, element 2 to that with the second lowest ID number, ...
 - If $NELES\%NSUBS > 0$ however, the first $NELES\%NSUBS$ subdomains get each one more element.
 - It should be noted that:
 - This mesh partitioning strategy is not suitable for the iterative FETI solvers (see [STATICS](#)).
 - For applications where the finite element model contains multiple disconnected components, it allows a subdomain to contain elements from different components.
 - It delivers subdomains that are as close as possible to uniform in size, where size refers here to the number of elements of a subdomain. Hence, this strategy does not account for the [WEIGHTS](#) of the elements of the mesh.
 - It guarantees that the generated mesh partition contains exactly the requested number of subdomains, **NSUBS**.
 - **STRAT2**, which is a two-step strategy intended for the family of domain-decomposition-based iterative FETI solvers (see [STATICS](#)). In the first step, an initial mesh partition is generated using the greedy algorithm. During the second step, the subdomain aspect ratios of this partition are optimized for the benefit of the optimal convergence of the chosen iterative FETI solver (see [STATICS](#)). **This strategy is the default mesh partitioning strategy**. Unlike **STRAT1**:
 - For applications where the finite element model contains multiple disconnected components, each subdomain is allowed to contain elements from the same component only.
 - This mesh partitioning strategy accounts for the [WEIGHTS](#) of the elements of the mesh and therefore performs load balancing according to these weights.
 - It does not guarantee that the generated mesh partition will contain exactly the requested number of subdomains, **NSUBS**, because it performs some necessary optimizations that may result in the generation of a slightly different number of subdomains.
 - Two alternative strategies intended for use with the parallel direct solver **mumps**, and for use in parallel explicit dynamic computations:
 - **STRAT3**, which is based on a greedy algorithm similar to that used in **STRAT2**, but without *some* of the mechanism treatments intended for an iterative FETI solver.
 - **STRAT4**, which is based on SCOTCH, the software package and libraries for sequential and parallel graph partitioning, static mapping and clustering, sequential mesh and hypergraph partitioning, and sequential and parallel sparse matrix block ordering.
- Note however that neither **STRAT3** nor **STRAT4** is suitable for any iterative FETI solver, and neither of them accounts for the [WEIGHTS](#) of the elements of the mesh and therefore neither of them performs load balancing according to these weights.

The syntax of this input command is given below.

Note 1: This command can also be executed as a command line when running **AERO-S**, in which case the syntax is as follows

```
aeros --dec --nsub <number_of_subdomains> [--trivial --deter --load --mem --exit --allow-mechanisms --use-scotch] <filename.aeros.aicdf>
```

where:

- The argument **--dec** requests a mesh decomposition.
- The argument **--nsub** specifies the number of subdomains.
- The argument **--trivial** specifies the trivial mesh partitioning strategy **STRAT1**.
- The argument **--allow-mechanisms** requests the deactivation of the special treatment of spring, truss and membrane elements in **STRAT2**, where the greedy-based algorithm is post-processed to produce a mechanism-free mesh partition usable by an iterative FETI solver. Hence, this argument has no effect when used together with **--trivial** or **--use-scotch**. Since **STRAT2** is the default mesh partitioning strategy, executing this command line without specifying a mesh partitioning strategy activates the mesh partitioning strategy **STRAT3**.
- The argument **--use-scotch** requests the mesh decomposition strategy **STRAT4**, which is based on the SCOTCH software package and libraries.

The meaning of the other arguments can be easily determined from the explanations of the syntax of the **DECOMPOSE** input command given

below. In this case, the generated mesh decomposition file and associated analysis files (predicted load distribution and memory consumption statistics) are outputted using the same prefix which is given by the [FNAME](#) entry of the command [CONTROL](#).

Note 2: When this command is executed from the command line (see above) and specified in the **AERO-S** ASCII Input Command Data file, the value of any argument specified on the command line overrides that specified in the ASCII Input Command Data file.

Note 3: When a decomposition file is already available, the syntax of the command line for running **AERO-S** with this decomposition file on a shared memory is

```
aeros -d <decomposition_pathandfilename> [-v <verbose_frequency>] -n <number_of_processors> <filename.aeros.aicdf>
```

In this case, the above command can also be combined with some of the other features of the command **DECOMPOSE** as in the following syntax

```
aeros -d <decomposition_pathandfilename> [--load --mem] [-v <verbose_frequency>] [-n <number_of_processors>] <filename.aeros.aicdf>
```

Note 4: Using this command to generate an element-based mesh partition (or domain decomposition) of the computational domain does not imply that the *global* finite element model data will be distributed across all (subdomain-based) computational processes. By default, the entire finite element model data is duplicated in each MPI process, which is not a memory-efficient strategy. To distribute the global finite element model data across the MPI processes according to the generated domain decomposition — that is, to store in each MPI process only the part of the global data pertaining to the subdomains assigned to this MPI process — the following additional steps should also be performed. First, the finite element model data should be re-arranged into a set of binary distributed input files associated with the generated domain decomposition using **SOWER** (see **SOWER**'s User's Reference Manual). Then, the aforementioned binary distributed input files should be inputted to the desired simulation via the ASCII Input Command Data file using the command **BINARY** and its sub-commands. In the case of a parallel system with a hybrid memory, if the user does not wish to use **SOWER** for the aforementioned purpose, the user should at least use the command `mpirun` with the "`--bynode`" option in order to limit the number of MPI processes — and therefore memory duplications — per shared memory sub-system.

DECOMPOSE

TRIVIAL OR ALLOW_MECHANISMS OR USE_SCOTCH	
NSUBS	nsubs
DETER	
FSGL	
OUTFILE	decomposition_pathandfilename
OUTLOAD	
OUTMEM	
EXIT	
SKIP	

TRIVIAL	The two-step strategy based on the greedy algorithm (<i>STRAT2</i>) is the default mesh partitioning strategy (characters). This sub-command keyword can be used to request instead the trivial mesh partitioning strategy (<i>STRAT1</i>).
ALLOW_MECHANISMS	The two-step strategy based on the greedy algorithm (<i>STRAT2</i>) is the default mesh partitioning strategy (characters). This sub-command keyword can be used to request instead a modified version of this strategy (<i>STRAT3</i>) where mechanisms due to various arrangements of spring, truss, and membrane elements are allowed in order to produce superior mesh partitions where the number of generated subdomains is as close as possible to the requested number of subdomains <code>NSUBS</code> (see below).
USE_SCOTCH	The two-step strategy based on the greedy algorithm (<i>STRAT2</i>) is the default mesh partitioning strategy (characters). This sub-command keyword can be used to request instead the mesh partitioning strategy based on SCOTCH (<i>STRAT4</i>), the software package and libraries for sequential and parallel graph partitioning, static mapping and clustering, sequential mesh and hypergraph partitioning, and sequential and parallel sparse matrix block ordering.
NSUBS nsubs	Sub-command keyword for specifying the desired number of subdomains (characters). Number of subdomains (integer). The generated mesh partition is guaranteed to contain <code>nsubs</code> subdomains only if the TRIVIAL mesh partitioning strategy is chosen (see above). Otherwise, the generated mesh partition may contain a different or slightly different number of subdomains, because of some optimizations that could be performed by the other mesh partitioning strategies. The default value is 1.
DETER	This sub-command keyword is relevant only for the two-step mesh partitioning strategy based on the greedy algorithm (<i>STRAT2</i>), whose second step is performed by default using a simulated annealing procedure (characters). It requests using instead a deterministic approach in order to reduce the CPU time of the second step.
FSGL	This sub-command keyword is relevant only for the two-step mesh partitioning strategy based on the greedy algorithm (<i>STRAT2</i>) and vibro-acoustic (aka elastodynamic and fluid-structure) analysis (for example, see IMPEDANCE) (characters). It requests performing the mesh decomposition in such a way that the fluid-structure interface is embedded within the subdomains as much as possible.
OUTFILE <decomposition_pathandfilename>	Sub-command keyword to request outputting the decomposition in <code><decomposition_pathandfilename></code> (characters). Name of the mesh decomposition file (characters). The format of this file is essentially the number and list of the elements in each subdomain. The default is <code>FNAME.optDec</code> where <code>FNAME</code> is the prefix specified under the command CONTROL or its own default value.
OUTLOAD	This sub-command keyword is relevant only for the two-step mesh partitioning strategy based on the greedy algorithm (<i>STRAT2</i>) (characters). It requests outputting the load distribution in the file <code><decomposition_pathandfilename>.load</code> . The load distribution is based on the weights of the elements in the subdomains (see WEIGHTS and FWEIGHTS). The default value is <code>off</code> .

OUTMEM	This sub-command keyword is relevant only for the two-step mesh partitioning strategy based on the greedy algorithm (STRAT2), and when an iterative FETI method is chosen as an equation solver (see STATICS) (characters). It requests outputting in the file <decomposition_pathandfilename>.mem an estimate of the subdomain-based memory consumption. This estimate is based on the requirements for factoring the subdomain matrices by a direct skyline method. By default, mid-side nodes are ignored by the overall mesh partitioning algorithm. However, when this option is specified, mid-side nodes are taken into account. Default value is off.
EXIT	Sub-command keyword to request exiting from AERO-S after the mesh partition is generated (characters). Default value is off.
SKIP	Sub-command keyword to request skipping this command (characters). Default value is off.

Next: [NFRAMES](#), Previous: [DECOMPOSE](#)

91 NODAL CONTACT

Command Statement: **NODALCONTACT**

The **NODALCONTACT** command can be used to specify node-to-node contact of the form

$$(u_2 - u_1) \cdot n_{12} = 0$$

or

$$(u_2 - u_1) \cdot n_{12} \geq gap$$

where u denotes the displacement field, n_{12} denotes the normal to the contact surface oriented from node 1 to node 2, and gap denotes the initial gap. The input format of this command can be as follows.

Note 1: For `mode_v = 1` and `mode_v = 3`, the enforcement of the nodal contact constraints by the Lagrange multiplier method is supported only by the FETI-DP family of solvers, and for static and implicit dynamic analyses only.

Note 2: For `mode_v = 1` and `mode_v = 3`, the enforcement of the nodal contact constraints by the elimination method is not supported.

Note 3: For `mode_v = 0` and `mode_v = 2`, the enforcement of the nodal tied contact constraints by the Lagrange multiplier method is supported only by the FETI-DP family of solvers, the GMRES solver, and the SPOOLES and MUMPS direct sparse solvers with pivoting enabled, and in all but explicit dynamic analyses.

NODALCONTACT

or

NODALCONTACT MODE `default_mode_v`

followed by one or more lines of the form

NODE_1 **NODE_2** **N_X** **N_Y** **N_Z**

or

NODE_1 **NODE_2** **N_X** **N_Y** **N_Z** **GAP** `gap_v`

NODE_1 **NODE_2** **N_X** **N_Y** **N_Z** **MODE** `mode_v`

NODE_1 **NODE_2** **N_X** **N_Y** **N_Z** **MODE** `mode_v` **GAP** `gap_v`

NODE_1	NODE_2	N_X	N_Y	N_Z	MODE	mode_v	GAP	gap_v	CONSTRAINT_METHOD
NODE_1									Id number of the first node (integer).
NODE_2									Id number of the second node (integer).
N_X									x-component of the normal to the contact surface oriented from NODE_1 to NODE_2 (float).
N_Y									y-component of the normal to the contact surface oriented from NODE_1 to NODE_2 (float).
N_Z									z-component of the normal to the contact surface oriented from NODE_1 to NODE_2 (float).
GAP									Keyword to be spelled out (characters).
gap_v									Initial gap between NODE_1 and NODE_2. A negative gap means an initial penetration (float).
MODE									Keyword to be spelled out (characters).
mode_v									Specifying mode_v = 0 designates a tie-of-the-normal-component-of-the-displacement constraint. Specifying mode_v = 1 designates a normal contact constraint. Specifying mode_v = 2 designates a tie-of-the-normal-and-tangential-components-of-displacement constraint. Specifying mode_v = 3 designates both normal contact and tie-of-the-tangential-components-of-displacement constraints (integer).
default_mode_v									Default mode_v for all constraints. If not specified, the default mode is set to 1, i.e. normal contact (integer).
CONSTRAINT_METHOD									This is the method for enforcing the associated constraints (characters). The default method is set in CONSTRAINTS and used whenever this entry is omitted.
multipliers									The Lagrange multiplier method.
elimination									The elimination method.
penalty									The penalty method. The parameter beta should be a large positive number, typically of the order of 10^8 (no default value is provided).
[beta]									

Next: [NODES](#), Previous: [NODALCONTACT](#)

92 NODAL FRAMES

Command Statement: **NFRAMES**

The NFRAMES command statement can be used to specify nodal reference frames to which any node defined in [NODES](#) can refer to either for the purpose of specifying in which frame the nodal coordinates are inputted, or in which frame the nodal degrees of freedom are defined, or both.

Note 1: All nodal degrees of freedom referred to in the [ACTUATORS](#), [DISPLACEMENTS](#), [DIMASS](#), [FORCES](#), [IDISPLACEMENTS](#), [IDISP6](#), [IDISP6PITA](#), [IVELOCITIES](#), [IVEL6PITA](#), [LMPC](#), [NODALCONTACT](#), [SENSORS](#), [USDD](#), and [USDF](#) commands are defined in the corresponding nodal degree of freedom frames (see [NODES](#)) which by default are the same as the global reference frame. However, all computed directional results (for example, the x, y and z components of a displacement field or the xy component of a stress field) can be outputted in either the nodal reference frames, or the global reference frame (see [OUTPUT](#)).

Two input formats are available for this command: one for the case of a position frame, and one for the case of a degree of freedom frame. The total number of lines below this command statement should be equal to the total number of different nodal frames referenced in [NODES](#), whether they are position frames or degree of freedom frames.

NFRAMES

NFRAME#	x_O	y_O	z_O	$S_{1x} \ S_{1y} \ S_{1z}$	$S_{2x} \ S_{2y} \ S_{2z}$	$S_{3x} \ S_{3y} \ S_{3z}$
---------	-------	-------	-------	----------------------------	----------------------------	----------------------------

or

NFRAMES

NFRAME#	$S_{1x} \ S_{1y} \ S_{1z}$	$S_{2x} \ S_{2y} \ S_{2z}$	$S_{3x} \ S_{3y} \ S_{3z}$
---------	----------------------------	----------------------------	----------------------------

NFRAME#

ID number of a nodal position (if 13 entries are inputted on the same line) or degree of freedom (if 10 entries are inputted on the same line) reference frame (integer).

x_O

x coordinate of the origin of the nodal position frame identified by the ID number specified above (float).

y_O

y coordinate of the origin of the nodal position frame identified by the ID number specified above (float).

z_O

z coordinate of the origin of the nodal position frame identified by the ID number specified above (float).

 $S_{1x} \ S_{1y} \ S_{1z}$

First axis of the nodal reference frame identified by the ID number specified above, **the nodal x-axis**, expressed in the global reference frame (floats).

 $S_{2x} \ S_{2y} \ S_{2z}$

Second axis of the nodal reference frame identified by the ID number specified above, **the nodal y-axis**, expressed in the global reference frame (floats).

 $S_{3x} \ S_{3y} \ S_{3z}$

Third axis of the nodal reference frame identified above, **the nodal z-axis**, expressed in the global reference frame (floats).

Next: [NONINPC](#), Previous: [NFRAMES](#)

93 NODES *S*

Command Statement: **NODES**

The **NODES** command statement is used to signal that the following data lines correspond to the coordinates, position frame, and degree of freedom frame of each node. The input data format is given below. There should be as many lines as the number of nodes in the mesh.

Note 1: All nodal degrees of freedom referred to in the [ACTUATORS](#), [DISPLACEMENTS](#), [DIMASS](#), [FORCES](#), [IDISPLACEMENTS](#), [IDISP6](#), [IDISP6PITA](#), [IVELOCITIES](#), [IVEL6PITA](#), [LMPC](#), [READMODE](#), [SENSORS](#), [USDD](#), and [USDF](#) commands are defined in the corresponding nodal degree of freedom frames which by default are the same as the global reference frame. However, all computed directional results (for example, the x, y and z components of a displacement field or the xy component of a stress field) can be outputted in either the nodal reference frames, or the global reference frame (see [OUTPUT](#)).

NODES

NODE#	X-ORDINATE	Y-ORDINATE	Z-ORDINATE	POS_FRM	DOF_FRM
-------	------------	------------	------------	---------	---------

NODE#	ID number of a node (integer).
X-ORDINATE	x-ordinate of the node identified by the specified node ID number (float).
Y-ORDINATE	y-ordinate of the node identified by the specified node ID number (float).
Z-ORDINATE	z-ordinate of the node identified by the specified node ID number (float).
POS_FRM	ID number of the nodal position frame (see NFRAMES) in which the coordinates of the node identified by the node ID number specified above are defined (integer). The value 0, which is the default value, specifies that the nodal position frame is the global frame. (See NFRAMES).
DOF_FRM	ID number of the nodal degree of freedom frame (see NFRAMES) in which the degrees of freedom of the node identified by the node ID number specified above are defined (integer). The value 0, which is the default value, specifies that the nodal degree of freedom frame is the global frame. (See NFRAMES).

Next: [NONLINEAR](#), Previous: [NODES](#)

94 NON INTRUSIVE POLYNOMIAL CHAOS

Command Statement: **NONINPC**

The **NONINPC** command signals to the **AERO-S** code to perform a non intrusive uncertainty quantification analysis using the Polynomial Chaos (PC) representation. It requires that the input file also contains the [GROUPS](#), [RANDOM](#), and [STATICS](#) commands as well as their respective data. Essentially, **AERO-S** computes in this case the mean and standard deviation of a computed primal field v as follows

$$\bar{v} = \int_{-\infty}^{+\infty} vp(v)dv = v_0 = \frac{\sum_{i=1}^N v^{(i)}}{N}$$

$$\sigma_v = \sqrt{\int_{-\infty}^{+\infty} (v - \bar{v})^2 p(v)dv} = \sqrt{\sum_{i=1}^{i=P-1} \left(\int_{-\infty}^{+\infty} \psi_i^2 p(\psi_i(\{\xi_j\})) d\psi_i \right) v_i^2}$$

$$= \sqrt{\sum_{i=1}^{i=P-1} \left(\int_{-\infty}^{+\infty} \psi_i^2 p(\psi_i(\{\xi_j\})) d\psi_i \right) \left(\frac{\sum_{j=1}^N v^{(j)} \psi_i^{(j)}}{N \int_{-\infty}^{+\infty} \psi_i^2 p(\psi_i) d\psi_i} \right)^2}$$

where v_0 denotes the constant of the PC expansion of the non deterministic v field ("the output"), N denotes the number of realizations,

$\psi^{(i)}$ denotes the deterministic solution of the field ψ associated with the i-th realization of the random system, p denotes the probability density function, ξ_j denotes the j-th “basic” random variable which is assumed to be Gaussian, v_i denotes the deterministic coefficient of the i-th PC shape function ψ_i , P denotes the total number of PC shape functions ψ_i representing the non deterministic output, and $\psi_i^{(j)}$ denotes the evaluation of ψ_i at the j-th realization of the random system.

Note 1: Currently, this command is limited to the post-processing of the results of a linear, static, structural analysis.

The input format of this command is given below.

NONINPC

DEG_OUT	NUM_RLZ
---------	---------

DEG_OUT	Degree of the PC representation of the output which, together with the data of the RANDOM command, determines P .
NUM_RLZ	Number of realizations N of the random system (which implies the number of deterministic solutions) (integer).

Next: [FLOATINGFRAME](#), Previous: [NONINPC](#)

95 NONLINEAR ANALYSIS

Command Statement:	NONLINEAR
--------------------	------------------

When the formulated static, quasistatic, or implicit dynamic finite element (FE) model is nonlinear, this command must be used to define a nonlinear solution strategy and specify its parameters. For structural problems, this is the case if:

- Some or all elements of the FE model are not assigned a material law using [MATUSAGE](#) – in which case the default material law/s (see below) with the parameters described in [MATERIAL](#) and assigned to the elements in [ATTRIBUTES](#) apply – in which case this command automatically accounts for the geometric nonlinearities in the formulation of the contributions of these elements to the global equations of equilibrium using:
 - The corotational method for all bar, beam, and shell elements.
 - The total Lagrangian method for all 3D solid elements and the membrane elements type 128 and 129.
- A nonlinear material law is specified in [MATLAW](#) and attributed to some or all elements of the FE model using [MATUSAGE](#), in which case this command accounts for the following nonlinearities in the computation of the contributions of these elements to the global equations of equilibrium:
 - The material nonlinearities.
 - The geometric nonlinearities too, using:
 - The corotational method for all bar, beam, and shell elements, if the material law specified in [MATLAW](#) involves infinitesimal strains. (Note that material laws based on finite strains are not permitted for these elements).
 - The total Lagrangian method for all 3D solid elements and the membrane elements type 128 and 129, if the material law specified in [MATLAW](#) involves finite strains. (Note that material laws based on infinitesimal strains are permitted for these elements, but in that case, geometric nonlinearities are not accounted for).

In the presence of this command in the ASCII Input Command Data file, if an element of the FE model is not assigned a material law – either linear or nonlinear – (specified in [MATLAW](#)) using [MATUSAGE](#), it is assigned by default the following material law:

- For a bar, beam, or shell element, the linear elastic material law whose parameters should be specified in [MATERIAL](#).
- For a 3D solid element or a membrane element type 128 or 129, the Saint Venant-Kirchhoff hyperelastic material law, which is an extension of the linear elastic material model to the nonlinear regime. This material law has the form

$$\underline{S} = \lambda \operatorname{tr}(\underline{\epsilon}) I + 2\mu \underline{\epsilon}$$

where \underline{S} is the second Piola-Kirchhoff stress tensor,

$$\underline{\epsilon} = \frac{1}{2} [(\nabla u)^T + \nabla u + (\nabla u)^T \cdot \nabla u]$$

is the Green-Lagrange strain tensor, λ and μ are the Lame constants, and I is the second order unit tensor. The Lame properties of this material law are deduced from the Young modulus E and Poisson ratio ν specified in [MATERIAL](#) for the material ID of the aforementioned element as follows

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \quad \text{and} \quad \mu = \frac{E}{2(1+\nu)}$$

The strain-energy density function for the Saint Venant-Kirchhoff model is defined as

$$W(\epsilon) = \frac{\lambda}{2} [tr(\epsilon)]^2 + \mu tr(\epsilon^2)$$

and the second Piola-Kirchhoff stress is given by

$$S = \frac{\partial W}{\partial \epsilon}$$

In the absence of this command in the ASCII Input Command Data file however, all elements are assigned linear elasticity as the material law and the kinematics are assumed to be linear, even when otherwise implied or specified.

Note 1: In the absence of this command in the input file, all elements are assigned linear elasticity as the material law and the kinematics are assumed to be linear, even when otherwise implied or specified.

Note 2: Currently, all 3D solid, beam, shell and rigid elements, and both membrane elements type 128 and 129 can be used for geometrically nonlinear explicit dynamic analyses.

Note 3: Currently, all 3D solid, beam, shell (except the BT shell element type 16) and rigid elements, and both membrane elements type 128 and 129 can be used for geometrically nonlinear static and implicit dynamic analyses.

Note 4: Currently, all 3D solid elements, the membrane elements (type 128 and 129), the andes shell elements (type 15 and 1515), and the BT shell element (type 16) can be used for materially nonlinear explicit dynamic analyses.

Note 5: Currently, all 3D solid elements, the membrane elements (type 128 and 129), and the andes shell elements (type 15 and 1515) can be used for materially nonlinear static and implicit dynamic analyses.

The input format of this command is given below.

NONLINEAR
METHOD (keyword)
PARAMETERS (keyword or keyword and values)

METHOD	
Newton	This sub-command keyword specifies Newton's method for solving the governing nonlinear equations (characters). This method is also the default nonlinear solution method.
arclength	This sub-command keyword specifies an arclength method for solving the nonlinear equations associated with a nonlinear <i>static</i> computation (characters). Such a method differs from the standard Newton method in the path through which the captured solution converges, which in this case follows at each step a direction orthogonal to the tangent of the solution curve. Specifically, an arc length method causes Newton's method to converge along an arc, thereby often preventing divergence, even when the slope of the "load" vs. solution becomes zero or negative. For this reason, such a method is typically used to enable the nonlinear solution algorithm to pass limit points, such as maximum and minimum loads, and snap-through and snap-back responses. At these limit points, the stability of the numerical system is dependent on whether the analysis is performed under load or solution control. In structural analysis, these limit points are characteristic of buckling or failure.
PARAMETERS	
MAXITR maxitr_v	For a nonlinear static, quasistatic, or implicit dynamic computation, this sub-command keyword (characters) followed by its integer value specifies the maximum number of iterations per solve to be performed by the chosen solution METHOD. The default value is 100.
NLTOL nt1_v nt2_v nt3_v nt4_v	For a nonlinear static, quasistatic, or implicit dynamic computation, this sub-command keyword (characters) and the four real values following it specify the various tolerances needed for assessing the convergence of the chosen solution METHOD. More specifically, the first two parameters nt1_v and nt2_v specify the tolerances for the convergence of the relative residual and solution increment, respectively (real, real). Their default values are 10^{-6} and infinity, respectively. The third and fourth parameters nt3_v and nt4_v specify the tolerances for the convergence of the absolute residual and solution increment, respectively (real, real). Their default values are zero and infinity, respectively. The convergence of the chosen solution METHOD is declared when either of the following occurs (a) both of the relative residual and relative increment are below their specified respective tolerance levels, or (b) both of the absolute residual and absolute increment are below their specified respective tolerance levels. Note that a tolerance can be set to infinity using the characters "INF", and to machine epsilon 10^{-16} using

the characters "EPS".

DLAMBDA real_1 real_2

This sub-command keyword (characters) followed by two real numbers specifies the external "loading" parameters for a nonlinear static or quasistatic analysis only. However, for a quasistatic analysis where the material nonlinearities do not involve any inelasticity, the effect of this sub-command applies only to the first coupling cycle (see [OSTATICS](#)). The first real number, `real_1`, specifies the load fraction increment. The second real number, `real_2`, specifies the total load factor. For example, "DLAMBDA 0.25 1.0" specifies applying the external load in 4 steps, using a load increment equal to 0.25*(Original Inputted Source Terms). On the other hand, "DLAMBDA 0.25 2.0" specifies amplifying first the external loads by the factor 2, then applying the resulting external load in 8 steps using a load increment equal to 0.25*(Original Inputted Source Terms). This incremental loading strategy can help the convergence of the chosen solution `METHOD`.

FITALG fitalg_v

For structural problems, this sub-command keyword (characters) followed by an integer value specifies the fitting algorithm for the corotational formulation. For `fitalg = 1` the tangent stiffness matrix is not consistent but the corotational framework is supposed to be more robust. For `fitalg = 2` (default value), the tangent stiffness matrix is consistent (integer).

FAILSAFE real_1

For a nonlinear static or *implicit dynamic* computation, this keyword activates a fail-safe computational strategy when the chosen solution `METHOD` fails to converge *and* the absolute residual is less than the tolerance specified in `real_1` (real). The fail-safe strategy consists in repeating a load-step or time-step computation during which the aforementioned events occur using half the value of the load-step or time-step, until Newton's method converges. Upon convergence, the value of the load-step or time-step is doubled in each subsequent static or dynamic step computation until the original value of the load-step or time-step is restored. The default value of `real_1` is machine precision.

UNSYMMETRIC

The contributions to the tangent stiffness of various types of configuration-dependent external forces (for example, pressure, follower forces, and both follower and axial moments) lead to an unsymmetric geometric tangent stiffness matrix. For dynamic analysis, the contribution to the dynamic tangent stiffness matrix of some rotary inertial forces (for example, discrete inertial moments and rotary inertial forces emanating from beam and shell elements) also lead to an unsymmetric dynamic tangent stiffness matrix. By default, these matrices are symmetrized in order to reduce storage costs and enable the usage of a symmetric equation solver (see [STATICS](#)), which also reduces CPU costs. Hence, this sub-command keyword can be used to prevent this default symmetrization (characters). In this case, an unsymmetric equation solver must be specified in [STATICS](#) in case the tangent or dynamic tangent stiffness matrix becomes unsymmetric.

REBUILD int_1 [int_2]

This sub-command keyword (characters) followed by two integers manages two different intervals for deciding when to rebuild the tangent "stiffness" matrix during the solution of a nonlinear problem by the chosen `METHOD`. The integer `int_1` specifies the iteration interval within a time-step of a dynamic analysis, or a load-step of a static or quasistatic analysis where `DLAMBDA` is used to apply the external load incrementally. For example, `int_1 = 1` specifies rebuilding the tangent stiffness matrix at every iteration, and `int_1 = 2` specifies rebuilding it at every second iteration. The default value for `int_1` is 1. Usage of the integer `int_2` is optional, and its default value is 1. Otherwise, `int_2` specifies the time-step or load-step interval. For example, `int_2 = 1` specifies every time-step or load-step, and `int_1 = 2` specifies every other time-step or load-step. The default value of `int_2` is also 1. Hence, setting `int_1 = 1` and `int_2 = 1` specifies rebuilding the tangent stiffness matrix at every nonlinear iteration of every time-step or load-step. Setting `int_1 = 2` and `int_2 = 3` specifies rebuilding the tangent stiffness matrix at every second iteration of every third time-step or load-step. The characters "MAX" can be used to set either or both of `int_1` and `int_2` to a very large interval to limit or effectively deactivate rebuilding the tangent stiffness matrix, respectively. An interesting intermediate case is given by the combination `int_1 = "MAX"` and `int_2 = 1` which specifies rebuilding the tangent stiffness matrix only at the start of every time-step or load-step.

LINESEARCH ALPROC [maxitr c  flag]

This sub-command keyword (characters) is applicable only if `Newton` is specified as the nonlinear solution `METHOD`. It requests equipping Newton's method with a line search strategy and specifies the parameters of this strategy. In this case, it is recommended to rebuild the tangent stiffness matrix at each Newton iteration (see `REBUILD` above).

Given a nonlinear problem of the form

$$\text{solve } R(u) = 0$$

where `R` and `u` denote the nonlinear residual and solution of interest, respectively, Newton's method solves this problem by computing the iterates

$$u^{k+1} = u^k + \Delta u^k$$

where the increment Δu^k is the solution of the linearized problem

$$K(u^k)\Delta u^k = -R(u^k)$$

and `K` denotes the Jacobian (or tangent stiffness) of `R` with respect to `u`. For highly nonlinear problems, combining Newton's method with a line search strategy can prove to be a more appropriate solution method as this combination searches for the solution of the above nonlinear

problem in the form of less aggressive iterates as follows

$$\underline{u}^{k+1} = \underline{u}^k + \alpha^k \Delta \underline{u}^k$$

where $\alpha^k \leq 1$ is a step-length computed by **AERO-S** using either a `backtracking` or a `bisection` procedure (see `ALPROC` below).

Note that equipping Newton's method with a line search strategy typically increases the number of iterations required for convergence. However, for many problems, Newton's method may simply fail to converge without a line search strategy.

`ALPROC`

`backtracking`

Sub-command keyword to specify a procedure for computing the step-length parameter α^k of the line search strategy (characters). Two options are currently available:

For a given pair of \underline{u}^k and $\Delta \underline{u}^k$, the main idea in this case is to minimize the "merit" function

$$f(\underline{u}^{k+1}) = \frac{1}{2} \|R(\underline{u}^{k+1})\|_2^2 = \frac{1}{2} \|R(\underline{u}^k + \alpha^k \Delta \underline{u}^k)\|_2^2$$

by searching iteratively for the value of α^k which satisfies the first Wolfe condition — aka the "sufficient decrease condition"

$$f(\underline{u}^k + \alpha^k \Delta \underline{u}^k) \leq f(\underline{u}^k) + \frac{c_{bt}}{2} \alpha^k \nabla f(\underline{u}^k)^T \Delta \underline{u}^k$$

where ∇f is the gradient of f with respect to \underline{u} and satisfies

$$\nabla f(\underline{u}) = K^T(\underline{u}) R(\underline{u})$$

and c_{bt} is a user-specified "sufficient decrease factor" chosen as $0 \leq c_{bt} < 1$ (see `c` below).

If the tangent stiffness K is rebuilt at every iteration k (see `REBUILD` above), the above sufficient decrease condition can also be written as

$$\|R(\underline{u}^k + \alpha^k \Delta \underline{u}^k)\|_2^2 \leq (1 - c_{bt} \alpha^k) \|R(\underline{u}^k)\|_2^2$$

Hence, the value of α^k that satisfies this condition is computed iteratively as follows. The first trial value of α^k is set to the maximum value of 1.0. Then, while the above sufficient decrease condition is not satisfied, the current value of α^k is reduced by the user-specified "contraction factor" $0 < \gamma < 1$ (see γ below) — that is, $\alpha^k \leftarrow \gamma \alpha^k$ — until the sufficient decrease condition

is satisfied.

`bisection`

In this case, the merit function employed, when applicable, is the scalar-valued potential function whose gradient is the nonlinear residual R defined above (see `backtracking`) — that is, $R = \nabla f$. This function is minimized by applying the well-known bisection root-finding

algorithm to search iteratively for the value of α^k which satisfies the second *strong* Wolfe condition — aka the "curvature condition" (or "sufficient slope decrease condition")

$$|\Delta \underline{u}^{k^T} \nabla f(\underline{u}^k + \alpha^k \Delta \underline{u}^k)| \leq c_{bs} |\Delta \underline{u}^k \nabla f(\underline{u}^k)|$$

where c_{bs} is a user-specified "sufficient slope decrease factor" that is usually chosen to be much larger than c_{bt} (see `c` below).

`maxitr`

Specifies the maximum number of line search or bisection iterations — that is, the maximum number of iterations for finding the optimal step-length. The default value of this parameter is 10.

`c`

If `ALPROC` is set to `backtracking`, this parameter specifies the value of the sufficient decrease factor c_{bt} (real). In this case, the chosen value must satisfy $0 \leq c < 1$ and the default value is 1.0e-4.

On the other hand, if `ALPROC` is set to `bisection`, this parameter specifies the value of the sufficient slope decrease factor c_{bs} (real). In this case, the default value of this parameter is 0.1.

γ

This parameter is relevant only when `ALPROC` is set to `backtracking`. It specifies the value of the contraction factor γ (real). The chosen value must satisfy $0 < \gamma < 1$. The default value of this parameter is 0.8.

`flag`

Setting this output flag to `yes` requests outputting to the screen some relevant information at each line search iteration. Setting it to `no` (default setting) turns off this output.

```
PENALTY int_1 real_1 real_2
```

This sub-command keyword (characters) followed by one integer and two real numbers specifies the parameters of the penalty or augmented Lagrangian method for enforcing **CONSTRAINTS** in a nonlinear analysis. The integer **int_1** specifies the maximum number of outer iterations on the penalty parameter in the case of the penalty method, or on the Lagrange multipliers (and optionally the penalty parameter) in the case of the augmented Lagrangian method; its default value is 1. The first real number, **real_1**, specifies the absolute tolerance for assessing the convergence of the outer iteration loop based on the infinite norm of the constraint violations; its default value is 10^{-8} . The second real number, **real_2**, specifies the factor by which the penalty parameter in either method is increased at every outer iteration; its default value is 10.

Next: [OUTPUT](#), Previous: [NONLINEAR](#)

96 NONLINEAR SUPERPOSITION OF RIGID AND DEFORMATION MODES

Command Statement: **FLOATINGFRAME**

The **FLOATINGFRAME** command can be invoked to:

- Superpose the rigid body modes and structural deformation modes of an unrestrained structural system, in order to construct a nonlinear structural dynamic Reduced-Order Model (ROM) that accounts for the coupling between rigid and flexible structural modes and apply this ROM to a nonlinear modal analysis of this structural system. **AERO-S** formulates the dynamic ROM in a floating reference frame attached to the center of gravity of the structural system. The resulting governing equations of dynamic equilibrium are *in general* nonlinear, due to the potentially large rigid rotations as well as the coupling between the rigid body rotation modes and the structural deformation modes. These equations can be written as

$$\underbrace{\begin{pmatrix} M_{uu} & M_{ur} & M_{us} \\ M_{ur}^T & M_{rr} & M_{rs} \\ M_{us}^T & M_{rs}^T & M_{ss} \end{pmatrix}}_{M^*} \underbrace{\begin{pmatrix} \ddot{q}_u \\ \ddot{q}_r \\ \ddot{q}_s \end{pmatrix}}_{\dot{K}^*} + \underbrace{\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & K_{ss} \end{pmatrix}}_{K^*} \begin{pmatrix} q_u \\ q_r \\ q_s \end{pmatrix} = \begin{pmatrix} 0 \\ Q_r(\dot{q}, \ddot{q}) \\ Q_s(\dot{q}, \ddot{q}) \end{pmatrix}$$

where:

- A dot designates a time-derivative.
- The subscript
- The matrices M_{uu} , M_{ur} , M_{us} , M_{rr} , M_{rs} , M_{ss} , and K_{ss} are reduced-order matrices defined by

$$\begin{aligned} M_{uu} &= V_u^T M V_u & M_{ur} &= V_u^T M V_r & M_{us} &= V_u^T M V_s \\ M_{rr} &= V_r^T M V_r & M_{rs} &= V_r^T M V_s & M_{ss} &= V_s^T M V_s \\ K_{ss} &= V_s^T K V_s \end{aligned}$$

where:

- M and K are the symmetric mass and stiffness matrices of the base finite element model of the unrestrained structural system, respectively.
- V_u is the matrix of rigid body translation modes, V_r is the matrix of rigid body rotation modes, and V_s is the matrix of structural deformation modes associated with the same base finite element model mentioned above.
- Collectively, V_u , V_r , and V_s form a Reduced-Order Basis (ROB) and therefore the above governing equations define a projection-based ROM. For this reason, each of V_u , V_r , and V_s can be referred to as a partial ROB and each of q_u , q_r , and q_s can be referred to as a vector of generalized coordinates of some type.
- $Q_r(\dot{q}, \ddot{q})$ and $Q_s(\dot{q}, \ddot{q})$ are two vectors of velocity/acceleration-dependent generalized gyroscopic damping forces due to, among other terms in the above governing equations, the centrifugal, Euler, and Coriolis terms. In *general*, these two vectors of generalized forces are *nonlinear* functions of \dot{q} and \ddot{q} .
- Specify the construction of the aforementioned structural dynamic ROM and its application to the simulation of the dynamic response of a structural system whose center of gravity undergoes a prescribed, planar, circular trajectory at a constant angular velocity (see below the description of the **RMPY** sub-command keyword). In this case, $Q_r(\dot{q}, \ddot{q})$ and $Q_s(\dot{q}, \ddot{q})$ become linear functions of \dot{q} and \ddot{q} and therefore the above governing equations become linear.

Note that the global, reduced-order mass and stiffness matrices M^* and K^* , as well as some quantities involved in the evaluation of the

two vectors of generalized forces Q_r and Q_s , are invariant in a given simulation. Hence, **AERO-S** always pre-computes these quantities *offline* before applying *online* the structural dynamic ROM outlined above to the modal analysis of a structural system. This offline/online staging of the computations accelerates a modal analysis based on the computational technology underlying the **FLOATINGFRAME** command.

It follows that a structural dynamic analysis based on the structural dynamic ROM described above requires including in the same **AERO-S** ASCII Input Command Data file:

- The command [READMODE](#), in order to input the matrix V_s . The matrices V_u and V_r are automatically constructed by **AERO-S** (see the first item under **Note 1** below). There are no orthogonality or orthonormality requirements for or between any of these three matrices (except currently, for those implied by the second item under **Note 1** below).
- The [NONLINEAR](#) command, unless the **RMPt** sub-command keyword (see below) is also specified in the same ASCII Input Command Data file, in which case the governing equations of dynamic equilibrium are linear (see above).
- The [DYNAMICS](#) command, together with the **SRM** sub-command keyword, and if desired, the [OUTPUT](#) command with the result type **MODALFLF** specified, in order to:
 - Construct *offline* the structural dynamic ROM outlined above, output it in a file for subsequent exploitation by a third-party software, and then exit. **AERO-S** will output the ROM in a result file of the type **MODALFLF**, if the user specifies such a file in the same **AERO-S** ASCII Input Command Data file under the [OUTPUT](#) command. Then, **AERO-S** exits if the user has set the value of **tt** in [DYNAMICS](#) to zero.
 - Construct *offline* the structural dynamic ROM, output it if requested, then perform *online* the structural dynamic analysis described in the remainder of the same **AERO-S** ASCII Input Command file.

Note 1: Currently, this command is supported only for:

- Fully unrestrained structural systems with 6 rigid body modes (3 rigid body translation and 3 rigid body rotation modes). In this case (and in the context of the **FLOATINGFRAME** command), **AERO-S** automatically computes V_u and V_r using the hybrid geometric-algebraic method underlying the [GRBM](#) command, which does not need to be specified in the same **AERO-S** ASCII Input Command Data file for this purpose, unless the user wishes to modify its default values.
- Partial ROBs V_s whose columns are eigenvectors of the base finite element model of the undamped structural system (undamped natural modes).
- Implicit dynamic computations using the midpoint rule (which for linear structural dynamic systems, is also known as the trapezoidal rule) obtained by setting $\alpha_m = \alpha_f = 1/2$, $\beta = 1/4$, and $\gamma = 1/2$ in the implicit generalized **LQ** method described in [DYNAMICS](#).

Note 2: After reading the partial ROB V_s inputted via the [READMODE](#) command, **AERO-S** tests whether any of its columns is a rigid body mode and ignores any such modes.

The syntax for invoking this command is given below.

FLOATINGFRAME						
RMIV	u_x	u_y	u_z	ω_x	ω_y	ω_z
RMPT	x_o	y_o	z_o	n_x	n_y	n_z
SCAL	flag	SCAL				

RMIV	Sub-command keyword (characters) for specifying the initial values of the generalized velocity dofs associated with the rigid body modes. Note that the initial values of the generalized velocity dofs associated with the structural deformation modes should be assigned in VELOCITIES using the MODAL sub-command keyword.
u_x	Initial value of the generalized velocity dof associated with the rigid body x -translation mode (real). The default value is <u>0</u> .
u_y	Initial value of the generalized velocity dof associated with the rigid body y -translation mode (real). The default value is <u>0</u> .
u_z	Initial value of the generalized velocity dof associated with the rigid body z -translation mode (real). The default value is <u>0</u> .
ω_x	Initial value of the generalized velocity dof associated with the rigid body x -rotation mode (real). The default value is <u>0</u> .
ω_y	Initial value of the generalized velocity dof associated with the rigid body y -rotation mode (real). The default value is <u>0</u> .
ω_z	Initial value of the generalized velocity dof associated with the rigid body z -rotation mode (real). The default value is <u>0</u> .
RMPT	Sub-command keyword (characters) for specifying a prescribed, planar, circular trajectory at a constant angular velocity, for the center of gravity of the structural system.
x_o	x -coordinate of the fixed center of the prescribed planar, circular trajectory (real).

y_o	y -coordinate of the fixed center of the prescribed planar, circular trajectory (real).
z_o	z -coordinate of the fixed center of the prescribed planar, circular trajectory (real).
n_x	x -component of the unit axis of rotation of the prescribed, planar, circular trajectory (real).
n_y	y -component of the unit axis of rotation of the prescribed, planar, circular trajectory (real).
n_z	z -component of the unit axis of rotation of the prescribed, planar, circular trajectory (real).
ω	Constant angular velocity of the prescribed planar, circular trajectory (real).
SCAL	Sub-command keyword (characters) for specifying whether or not to scale the structural deformation modes stored in V_s by the square root of the total structural mass.
flagSCAL	on/off flag (characters). The default value is off. Note that if the command FLOWINGFRAME is used to construct and output the structural dynamic ROM described above, mass scaling must be disabled for compatibility reasons.
On	In this case, the structural deformation modes are scaled by the square root of the total mass of the structural system.
Off	In this case, the structural deformation modes are not scaled by the square root of the total mass of the structural system.

Next: [OUTPUT6](#), Previous: [FLOWINGFRAME](#)

97 OUTPUT OF RESULTS (OUTPUT completely spelled out)

Command Statement: **OUTPUT** [KEYLETTER]

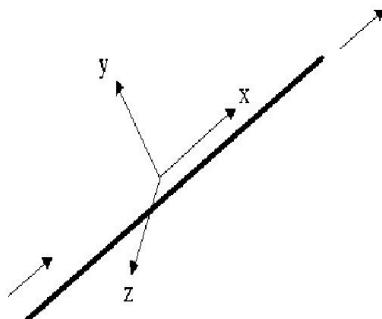
The **OUTPUT** command statement is used to signal that the following lines of data correspond to specifying which results are to be outputted. For vector results such as displacement fields, this command forces **AERO-S** to output only the three translational components.

The shared memory version of **AERO-S** generates ASCII output files. Most of these are in a format that is suitable for postprocessing by the **XPost** software, but some are in a format that is suitable for postprocessing by **gnuplot**. The distributed memory version of **AERO-S** generates binary output files except for the case of selective nodal output which it generates in ASCII format. The **SOWER** software can be used to convert the binary output files into ASCII output files, in a format suitable for postprocessing by **XPost**. All "nodal" output files are generated in the **gnuplot** format. Examples using the various **OUTPUT** commands can be found in **FEM.d/fem_examples/Output.d/**.

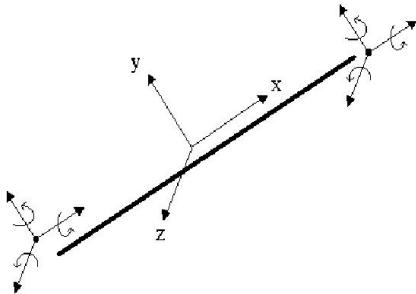
Note 1: In addition to the time-instances implied by the specified value of the parameter **INCREMENT**, any result specified for output is also outputted at the beginning of a simulation if **INCREMENT** is non zero, and at the end of a simulation if **AERO-S** exits gracefully.

Note 2: The strain and stress outputs are not available for the bar and beam elements, except for **STRAINXX** and **STRESSXX**

Note 3: The internal forces and moments and **STRAINXX** and **STRESSXX** for the bar and beam elements are outputted in the element frames using the sign conventions graphically depicted below.



Sign conventions for the output of the internal forces in a bar element



Sign conventions for the output of the internal forces in a beam element

Note 3: For nonlinear structural analysis, the stresses and strains for the corotational elements are outputted on the deformed configuration and in the corotated frames.

Note 4: For nonlinear structural analysis, the stresses and strains for the total Lagrangian elements are outputted on the undeformed configuration.

Note 5: For nonlinear structural analysis, the internal forces and moments and `STRAINXX` and `STRESSXX` for the bar and beam corotational elements are outputted in the element frames associated with the deformed configuration.

Note 6: When multiple outputs are expected, they are printed one set after the other in the same output file.

Note 7: To visualize the directions of principal strains and stresses, the auxiliary code PDIREC should be used to generate the node set and element set used by **XPost** for this purpose. When running PDIREC, the user is asked to enter the name of the mesh **XPost** file (for example, the one obtained from executing **AERO-S** with the option `-t`), a **AERO-S** result file, the type of that result for labeling purposes (eg. `SPDIREC`), and a characteristic length to be used for scaling the length of the elements representing the directions to be visualized. PDIREC can also be used for visualizing any other vector output (eg. `GDISPLAC` or `GVELOCIT`).

Note 8: The vocabulary used for the request of stresses and strains that are associated with the elements listed below differs from the standard continuum mechanics notation. See below for specific notations.

Note 9: When both `OUTPUT` and `OUTPUT6` commands are used in the same input file, the `KEYLETTER` option specified for one of these two commands applies to all output files requested under both of them; if this option is specified twice, once for each of these two commands, and the two specified settings differ, the one which is specified last in the input file prevails.

The input format of this command can be as follows.

<code>OUTPUT</code>	<code>[KEYLETTER]</code>
---------------------	--------------------------

<code>RESULT</code>	<code>[FORMAT]</code>	<code>PATHANDFILENAME</code>	<code>INCREMENT</code>	<code>[NODE_NUM_BER]</code>	<code>[OPTION]</code>	<code>[...]</code>	<code>[OPTION]</code>
---------------------	-----------------------	------------------------------	------------------------	-----------------------------	-----------------------	--------------------	-----------------------

or

<code>RESULT</code>	<code>[FORMAT]</code>	<code>PATHANDFILENAME</code>	<code>INCREMENT</code>	<code>[N NODE_NUMBER]</code>	<code>[OPTION]</code>	<code>[...]</code>	<code>[OPTION]</code>
---------------------	-----------------------	------------------------------	------------------------	------------------------------	-----------------------	--------------------	-----------------------

or

<code>RESULT</code>	<code>[FORMAT]</code>	<code>PATHANDFILENAME</code>	<code>INCREMENT</code>	<code>[NG GROUP_NUMBER]</code>	<code>[OPTION]</code>	<code>[...]</code>	<code>[OPTION]</code>
---------------------	-----------------------	------------------------------	------------------------	--------------------------------	-----------------------	--------------------	-----------------------

`KEYLETTER` Optional keyletter specifying how to handle gaps in the numbering of nodes *as far as output is concerned*.

`e` In this case, the output is expanded — that is, padded with zero(es) at each line corresponding to a gap in the numbering of the nodes. This is the default case. If **XPost** is to be used for visualizing the results, the geometry of the **AERO-S** computational model should be converted in this case to the **XPost** format using the option `-t` or `-m` to perform the same node padding (see [INTRODUCTION](#)).

`c` In this case, no padding is performed, and the output file is ordered according to the ascending order of the freely numbered nodes. If **XPost** is to be used for visualizing the results, the geometry of the **AERO-S** computational model should be converted in this case to the **XPost** format using the option `-t` or `-M` to avoid any node padding (see [INTRODUCTION](#)).

In summary, the user should note that **AERO-S** accepts gaps in the number of elements and nodes (free

numbering) and performs all computations by accounting for such gaps efficiently. The visualizer **XPost** can also handle freely numbered nodes and elements; however, for speed purposes, **XPost** interprets a result as a node-gap-free result. For this reason, if a **AERO-S** computational model contains gaps in the numbering of the nodes, using **XPost** for visualizing the computed results requires performing one of the following:

- Setting KEYLETTER to e and converting the geometry of the **AERO-S** computational model to the **XPost** format using the option -t or -m (see [INTRODUCTION](#)) — in which case the output files will be larger than needed.
- Setting KEYLETTER to c and converting the geometry of the **AERO-S** computational model to the **XPost** format using the option -T or -M (see [INTRODUCTION](#)) — in which case the output files will not be larger than needed but a different node numbering (more specifically, a compressed version) will be used in the **XPost** model.

RESULT
GDISPLAC

The following results can be outputted (typically in an ASCII file, unless otherwise noted):
Nodal displacements. Note that:

- In the context of the [OUTPUT6](#) command, this result can be used to collect displacement snapshots during a structural dynamic analysis, for example, for the purpose of constructing a Reduced-Order Basis (ROB) for (linear or nonlinear) model reduction (for collecting displacement snapshots for the purpose of constructing a ROB for nonlinear model reduction, see also below the description of the output result STATEVCT). Because this result is stored by default in an ASCII input file, it is recommended to output it in this case using a higher precision by setting the optional field FORMAT (see below) appropriately, for example, to 21 15.
- In the context of an online Reduced-Order Model (ROM) simulation and as long as the option NODE_NUMBER is not used, this result corresponds to the generalized coordinates of the displacement field. In this case, these generalized coordinates can be transformed into nodal displacements using the [RODC](#) command and the keyword SROM of the [DYNAMICS](#) command. If however in the same context the option NODE_NUMBER is used, this result corresponds to the reconstructed high-dimensional displacement at the node specified using NODE_NUMBER.

DISPLACX
DISPLACY

Nodal displacements in the *x* direction.
Nodal displacements in the *y* direction.

DISPLACZ
ROTATIOX
ROTATIOY

Nodal displacements in the *z* direction.
Nodal rotations in the *x* direction.
Nodal rotations in the *y* direction.

ROTATIOZ
ROVECTOR

Nodal rotations in the *z* direction.
x, *y*, and *z* components of the rotation vector.

ANGVELOC
ANGACCEL

x, *y*, and *z* components of the angular velocity vector.
x, *y*, and *z* components of the angular acceleration vector.

DISPLMOD
ROTATMOD
GDISPMOD
GVELOCIT

Euclidean norms of the nodal displacements.
Euclidean norms of the nodal rotations.
Euclidean norms of the generalized nodal displacements.
Nodal velocities. In the context of an online Reduced-Order Model (ROM) simulation and as long as the option NODE_NUMBER is not used, this result corresponds to the generalized coordinates of the velocity field. In this case, these generalized coordinates can be transformed into nodal velocities using the [RODC](#) command and the keyword SROM of the [DYNAMICS](#) command. If however in the same context the option NODE_NUMBER is used, this result corresponds to the reconstructed high-dimensional velocity at the node specified using NODE_NUMBER.

GACCELER

Nodal accelerations. In the context of an online Reduced-Order Model (ROM) simulation and as long as the option NODE_NUMBER is not used, this result corresponds to the generalized coordinates of the acceleration field. In this case, these generalized coordinates can be transformed into nodal accelerations using the [RODC](#) command and the keyword SROM of the [DYNAMICS](#) command. If however in the same context the option NODE_NUMBER is used, this result corresponds to the reconstructed high-dimensional acceleration at the node specified using NODE_NUMBER.

ROMATRIX
QUATERNI
DSVARVCT

Rotation matrix representation of nodal rotations (9 components).
Positive unit quaternion representation of nodal rotations (4 components).
Dual state variables (Lagrange multipliers associated with constraints) collected for nonlinear model reduction of contact problems.

GTEMPERA
GTEMPVEL

Nodal temperatures.
Nodal first time-derivatives of the temperature.

Eigenvalues and Eigenvectors. If singular modes are found during an eigen analysis, they are saved in the file FNAME associated with GEIGENPA *before* the non-singular eigen results.

GEIGSLSH

Sloshing Eigenvalues and Eigenvectors (displacement potential). If a singular mode is found during the analysis, it is saved in the file FNAME associated with GEIGSLSH *before* the non-singular eigen results.

SLSHDISP
SLSH DSPX
SLSH DSPY

True fluid nodal displacements associated with the computed sloshing modes.
x component of the true fluid nodal displacements associated with the computed sloshing modes.
y component of the true fluid nodal displacements associated with the computed sloshing modes.

SLSHDSPZ

z component of the true fluid nodal displacements associated with the computed sloshing modes.

STRAINXX	xx component of the (total) strain tensor.
STRAINYY	yy component of the (total) strain tensor.
STRAINZZ	zz component of the (total) strain tensor.
STRAINXY	xy component of the (total) strain tensor.
STRAINYZ	yz component of the (total) strain tensor.
STRAINZX	zx component of the (total) strain tensor.
STRAINVM	von Mises strain.
EFFPSTRN	Effective plastic strain.
STRAINP1	First principal strain.
STRAINP2	Second principal strain.
STRAINP3	Third principal strain.
PLSTRNXX	xx component of the plastic strain tensor.
PLSTRNYY	yy component of the plastic strain tensor.
PLSTRNZZ	zz component of the plastic strain tensor.
PLSTRNXY	xy component of the plastic strain tensor.
PLSTRNYZ	yz component of the plastic strain tensor.
PLSTRNZX	zx component of the plastic strain tensor.
EP1DIREC	Direction of first principal strain.
EP2DIREC	Direction of second principal strain.
EP3DIREC	Direction of third principal strain.
STRESSXX	xx component of the second-Piola stress tensor.
STRESSYY	yy component of the second-Piola stress tensor.
STRESSZZ	zz component of the second Piola stress tensor.
STRESSXY	xy component of the second Piola stress tensor.
STRESSYZ	yz component of the second Piola stress tensor.
STRESSZX	zx component of the second Piola stress tensor.
STRESSVM	(Second Piola) von Mises stress.
STRESSP1	(Second Piola) first principal stress.
STRESSP2	(Second Piola) second principal stress.
STRESSP3	(Second Piola) third principal stress.
SP1DIREC	Direction of first principal stress.
SP2DIREC	Direction of second principal stress.
SP3DIREC	Direction of third principal stress.
BKSTRSXX	xx component of the back stress tensor.
BKSTRSYY	yy component of the back stress tensor.
BKSTRSZZ	zz component of the back stress tensor.
BKSTRSXY	xy component of the back stress tensor.
BKSTRSYZ	yz component of the back stress tensor.
BKSTRSZX	zx component of the back stress tensor.
ELEMDELE	List of elements deleted from the finite element model by the element deletion method, and time at which they were deleted. This output file can be used for post-processing using xp2exo/ParaView .
DAMAGESC	Scalar damage parameter which is equal to 1 when an element is completely damaged and therefore deleted from the finite element model by the element deletion method, and to 0 when the element is not damaged.
HEATFLXX	Temperature flux along the x direction.
HEATFLXY	Temperature flux along the y direction.
HEATFLXZ	Temperature flux along the z direction.
GRDTEMPX	Temperature gradient along the x direction.
GRDTEMPY	Temperature gradient along the y direction.
GRDTEMPZ	Temperature gradient along the z direction.
INXFORCE	Internal forces along the element x -axis. This output result is supported only for bar and beam elements.
INYFORCE	Internal forces along the element y -axis. This output result is supported only for bar and beam elements.
INZFORCE	Internal forces along the element z -axis. This output result is supported only for bar and beam elements.
AXMOMENT	Internal moments along the element x -axis. This output result is supported only for beam elements.

AYMOMENT	Internal moments along the element y -axis. This output result is supported only for beam elements.														
AZMOMENT ENERGIES	Internal moments along the element z -axis. This output result is supported only for beam elements. External, Aeroelastic, Elastic, Kinetic, Dissipative, and Numerical Production energies. The Numerical Production energy corresponds to the energy conservation error for a conservative system. When positive, it suggests a numerically unstable scheme. When negative, it suggests the presence of numerical damping in the scheme. Using the standard notation and that of the DYNAMICS section, these energies are computed as follows (note that in the formulae given below, summation is performed over the outputted time-steps):														
	<table border="0"> <tr> <td><u>External :</u></td> <td>$\sum_{k=0}^{n-1} [(1 - \gamma)F^{\epsilon x t^k} + \gamma F^{\epsilon x t^{k+1}}]^T (u^{k+1} - u^k)$</td> </tr> <tr> <td><u>Aeroelastic :</u></td> <td>$\sum_{k=0}^{n-1} [(1 - \gamma)F^{\alpha e r t^k} + \gamma F^{\alpha e r t^{k+1}}]^T (u^{k+1} - u^k)$</td> </tr> <tr> <td><u>Elastic :</u></td> <td>$u^n^T K u^n$ (linear analysis)</td> </tr> <tr> <td></td> <td>$\int \text{elastic strain energy}$ (nonlinear analysis)</td> </tr> <tr> <td><u>Kinetic :</u></td> <td>$\dot{u}^n^T M \dot{u}^n$</td> </tr> <tr> <td><u>Dissipative :</u></td> <td>$\sum_{k=0}^{n-1} [(1 - \gamma)C\dot{u}^k + \gamma C\dot{u}^{k+1}]^T (u^{k+1} - u^k)$ (linear analysis) $\sum_{k=0}^{n-1} [(1 - \gamma)C\dot{u}^k + \gamma C\dot{u}^{k+1}]^T (u^{k+1} - u^k) + \text{material dissipation}$ (nonlinear analysis)</td> </tr> <tr> <td><u>Numerical Production :</u></td> <td>Kinetic - Previous_Kinetic + Elastic - Previous_Elastic + Increment_Dissipative_Between_Previous_and_Current - Increment_Exterior_Between_Previous_and_Current</td> </tr> </table>	<u>External :</u>	$\sum_{k=0}^{n-1} [(1 - \gamma)F^{\epsilon x t^k} + \gamma F^{\epsilon x t^{k+1}}]^T (u^{k+1} - u^k)$	<u>Aeroelastic :</u>	$\sum_{k=0}^{n-1} [(1 - \gamma)F^{\alpha e r t^k} + \gamma F^{\alpha e r t^{k+1}}]^T (u^{k+1} - u^k)$	<u>Elastic :</u>	$u^n^T K u^n$ (linear analysis)		$\int \text{elastic strain energy}$ (nonlinear analysis)	<u>Kinetic :</u>	$\dot{u}^n^T M \dot{u}^n$	<u>Dissipative :</u>	$\sum_{k=0}^{n-1} [(1 - \gamma)C\dot{u}^k + \gamma C\dot{u}^{k+1}]^T (u^{k+1} - u^k)$ (linear analysis) $\sum_{k=0}^{n-1} [(1 - \gamma)C\dot{u}^k + \gamma C\dot{u}^{k+1}]^T (u^{k+1} - u^k) + \text{material dissipation}$ (nonlinear analysis)	<u>Numerical Production :</u>	Kinetic - Previous_Kinetic + Elastic - Previous_Elastic + Increment_Dissipative_Between_Previous_and_Current - Increment_Exterior_Between_Previous_and_Current
<u>External :</u>	$\sum_{k=0}^{n-1} [(1 - \gamma)F^{\epsilon x t^k} + \gamma F^{\epsilon x t^{k+1}}]^T (u^{k+1} - u^k)$														
<u>Aeroelastic :</u>	$\sum_{k=0}^{n-1} [(1 - \gamma)F^{\alpha e r t^k} + \gamma F^{\alpha e r t^{k+1}}]^T (u^{k+1} - u^k)$														
<u>Elastic :</u>	$u^n^T K u^n$ (linear analysis)														
	$\int \text{elastic strain energy}$ (nonlinear analysis)														
<u>Kinetic :</u>	$\dot{u}^n^T M \dot{u}^n$														
<u>Dissipative :</u>	$\sum_{k=0}^{n-1} [(1 - \gamma)C\dot{u}^k + \gamma C\dot{u}^{k+1}]^T (u^{k+1} - u^k)$ (linear analysis) $\sum_{k=0}^{n-1} [(1 - \gamma)C\dot{u}^k + \gamma C\dot{u}^{k+1}]^T (u^{k+1} - u^k) + \text{material dissipation}$ (nonlinear analysis)														
<u>Numerical Production :</u>	Kinetic - Previous_Kinetic + Elastic - Previous_Elastic + Increment_Dissipative_Between_Previous_and_Current - Increment_Exterior_Between_Previous_and_Current														
DISSIPAT	Dissipative energy corresponding to material dissipation in the output of ENERGIES (see above). When the computational model contains both material dissipation and structural damping (e.g. Rayleigh damping), the output ENERGIES includes a report on their combination. This output result reports on material dissipation separately.														
AEROFORX	Aeroelastic nodal forces along the x direction at $[t^{n+1}]$, after application of the boundary conditions (see DYNAMICS).														
AEROFORY	Aeroelastic nodal forces along the y direction at $[t^{n+1}]$, after application of the boundary conditions (see DYNAMICS).														
AEROFORZ	Aeroelastic nodal forces along the z direction at $[t^{n+1}]$, after application of the boundary conditions (see DYNAMICS).														
AEROMOMX	Aeroelastic nodal moments along the x direction at $[t^{n+1}]$, after application of the boundary conditions (see DYNAMICS).														
AEROMOMY	Aeroelastic nodal moments along the y direction at $[t^{n+1}]$, after application of the boundary conditions (see DYNAMICS).														
AEROMOMZ	Aeroelastic nodal moments along the z direction at $[t^{n+1}]$, after application of the boundary conditions (see DYNAMICS).														
RAEROFOR	Aeroelastic resultant forces along the x , y , and z directions (integrated over the wet surface of the structure) at $[t^{n+1}]$, before application of the boundary conditions (see DYNAMICS).														
RAEROTFL	Fluid heat fluxes transferred to the structure at $[t^{n+1}]$.														
GHELMHOL	Real or complex acoustic pressure scattered field computed by an acoustic analysis in the frequency domain. The real part is always output first, then, if it is non uniformly zero, the imaginary part is output as a separate frame but in the same output file.														
GACOUPRE	Nodal values of the acoustic pressure computed by an acoustic analysis in the time domain.														
GACOUVEL	Nodal values of the first time-derivative of the acoustic pressure computed by an acoustic analysis in the time domain.														
GACOUACC	Nodal values of the second time-derivative of the acoustic pressure computed by an acoustic analysis in the time domain.														
KIRCHHOFF	Acoustic pressure field at locations specified using the KIRLOC command and evaluated using the Kirchhoff integral. This result can be outputted only if the surface of the scatterer is defined using the command HSCB. The corresponding output file contains on each line five words of the form: X-ORDINATE_I Y-ORDINATE_I Z-ORDINATE_I REAL(U_I) IMAG(U_I). X, Y, and Z-ORDINATE_I are the X, Y, and Z coordinates of the I-th point specified using the command KIRLOC, REAL(U_I) and IMAG(U_I) are the real and imaginary parts of the acoustic pressure field computed at the I-th specified point using the Kirchhoff integral method.														

FARFIELD	Farfield pattern of the acoustic pressure field. This field can be output only if the HSCB command is also specified. In this case, the output file contains on each line four words of the form " α , β , real-part(pp), imaginary-part(pp)". The first two quantities are the spherical angles (expressed in radians) that determine the direction $d = [\cos \beta * \cos \alpha, \cos \beta * \sin \alpha, \sin \beta]$. The number of output lines depends on the parameter INCREMENT (see below).
GENCOORD	Projections of a transient solution onto a modal or other basis specified via the MODE command. This field can be outputted only during a linear dynamic analysis without mesh decomposition, and if the MODE command is also specified in the ASCII Input Command Data file.
MODERROR	Relative truncation error using the two-norm of the representation of a transient solution by its projection onto a modal or other basis specified via the MODE command. This field can be outputted only during a linear dynamic analysis without a mesh decomposition, and if the MODE command is also specified.
MODALDSP or SRONDISP	Modal displacement coefficients or generalized displacements, when dynamics computations are done in a modal or other Reduced-Order Basis (ROB). This result can be outputted only during a linear dynamic analysis.
MODALEXF or SROMEXTF	Modal external forces coefficients or generalized external forces, when structural dynamics computations are done in a modal basis. This result can be outputted only during a linear dynamic analysis.
CONPRESS	Contact pressure forces (Lagrange multipliers in the normal directions).
CONFACE	Denotes the status of interactions at a node. A value of 0.5 indicates the node is not in contact. A value of 1, 2, or 3 denotes the number of interactions at that node.
NORMAL_FORCE_MAG	Normal force magnitude at a node. If multiple constraints exist at a node, the value is for the last constraint.
NORMAL_TRACTION_MAG	Normal traction magnitude at a node. If multiple constraints exist at a node, the value is for the last constraint.
TANGENTIAL_FORCE_MAG	Tangential force magnitude at a node. If multiple constraints exist at a node, the value is for the last constraint.
TANGENTIAL_TRACTION_MAG	Tangential traction magnitude at a node. If multiple constraints exist at a node, the value is for the last constraint.
CDIRNORX	x component of the normal direction for the constraint at a node. If multiple constraints exist at a node, the value is for the last constraint.
CDIRNORY	y component of the normal direction for the constraint at a node. If multiple constraints exist at a node, the value is for the last constraint.
CDIRNORZ	z component of the normal direction for the constraint at a node. If multiple constraints exist at a node, the value is for the last constraint.
CDIRTANX	x component of the tangential direction for the constraint at a node. If multiple constraints exist at a node, the value is for the last constraint.
CDIRTANY	y component of the tangential direction for the constraint at a node. If multiple constraints exist at a node, the value is for the last constraint.
CDIRTANZ	z component of the tangential direction for the constraint at a node. If multiple constraints exist at a node, the value is for the last constraint.
SLIP_MAG	Incremental slip at a node. If multiple constraints exist at a node, the value is for the last constraint.
NODAL_DISSIPATION	Frictional energy dissipated at a node.
CONTACT_AREA	Contact area for a node.
GAP_CUR	Normal gap for the current time. If multiple constraints exist at a node, the value is for the last constraint.
GAP_OLD	Normal gap from the previous time. If multiple constraints exist at a node, the value is for the last constraint.
REACTION	Reaction forces and moments along all three orthogonal directions.
HEATREAC	Thermal reaction sources.
VMSTTHIC	Nodal sensitivities of the von Mises stress with respect to all thickness variables specified in SENSITIVITY (column by column).
WEIGTHIC	Sensitivities of the total weight of the structural model with respect to all thickness variables specified in SENSITIVITY (row by row).
VMSTSHAP	Nodal sensitivities of the von Mises stress with respect to all shape (design) variables (column by column).
VMSTMACH	For an aeroelastic computation, nodal sensitivities of the von Mises stress with respect to the Mach number.
VMSTALPH	For an aeroelastic computation, nodal sensitivities of the von Mises stress with respect to the sideslip angle of attack.
WEIGSHAP	Sensitivities of the total weight of the structural model with respect to all shape variables (row by row).
STATEVCT	Snapshots of the state vector (displacements) stored in a <i>binary</i> file. They can be used, for example, for constructing a ROB for the purpose of (linear or nonlinear) model reduction. Currently, this result can be outputted only during a <i>nonlinear</i> structural dynamic analysis. During a linear structural dynamic analysis, displacement snapshots can be collected in the result GDISPLAC using the OUTPUT command. Because in this case the displacement snapshots are stored in an ASCII file, it is recommended to output them using a higher precision by setting the optional field FORMAT appropriately, for example, to 21 15.
NODALCOO	Nodal coordinates of the finite element mesh stored in a <i>binary</i> file. They can be used, for example, for constructing one or multiple (hyper) reduced meshes using the NDSFCG sub-command keyword of the RMSHC command.

VELOCVCT	Snapshots of the velocity vector (velocities). Currently, this result can be outputted only during a nonlinear structural dynamics analysis.
ACCELVCT	Snapshots of the vector of accelerations. Currently, this result can be outputted only during a nonlinear structural dynamics analysis.
FRCNGVCT	Snapshots of the vector of external forces. Currently, this result can be outputted only during a nonlinear structural dynamics analysis.
ROMRESID	Residual associated with a ROM-based approximate solution defined by a Reduced-Order Basis (ROB) inputted using the READMODE command and (time-dependent) generalized coordinates inputted using the RQDC command. Currently, this result can be outputted only during a nonlinear structural dynamics analysis.
ROMEXTFO	Contribution of the (possibly configuration-dependent) external forces to the residual associated with a ROM-based approximate solution defined by a Reduced-Order Basis (ROB) inputted using the READMODE command and (time-dependent) generalized coordinates inputted using the RQDC command. Currently, this result can be outputted only during a nonlinear structural dynamics analysis.
ROBDDATAF	Prefix of the two binary files containing both instances of the Reduced-Order Basis (ROB) generated by the ROBC command (see in ROBC the MNORMA sub-command keyword).
	<ul style="list-style-type: none"> • If this ROB is a primal ROB, its orthonormalized instance is outputted in the file <code><PATHANDFILENAME>.orthonormalized</code>, and its mass-orthonormalized counterpart is outputted in the file <code><PATHANDFILENAME>.massorthonormalized</code>. To this effect, note that when inputting this ROB in READMODE using the first format of this command, the extension ".massnormalized" can be omitted from <code><pathandfilename></code> as it will be internally added by AERO-S. On the other hand, when inputting this ROB in READMODE using the second format of this command, <code><pathandfilename_i></code> must be set to the name of the file containing the mass-orthonormalized version of this ROB unless <code>rob_type_i</code> is set to <code>inorm</code>. • If this ROB is a dual ROB, and there can be more than one basis computed (see <code>numrob</code> in ROBC), each ROB is outputted in a separate file labeled <code><PATHANDFILENAME>.int</code>, where <code>int</code> is the integer indicating the dimension of the ROB (recall that when multiple ROBs are computed, they have monotonically increasing/decreasing dimensions).
SAMPLMSH	This result consists of three files or sets of files: <ul style="list-style-type: none"> • A file named <code><PATHANDFILENAME></code>, containing the attributes of the original mesh and therefore all sampling weights needed for performing a hyper reduction, if computed using the RMSHC command. Specifically, it is outputted as an inclusive list of element attributes and their corresponding hyper reduction coefficients (or weights) (see ATTRIBUTES). In the case of local ROBs, one such file is outputted for each local basis and the extension <code>.cluster<local_basis_id></code> is appended to the filename <code><PATHANDFILENAME></code>. • Another file named <code><PATHANDFILENAME>.elementmesh.inc</code> containing the reduced mesh as well as that part of the high-dimensional model needed for constructing the corresponding hyper reduced projection-based reduced-order model - that is: <ul style="list-style-type: none"> ◦ The reduced mesh and its weights. ◦ The attributes and material properties of the elements of the reduced mesh. ◦ Initial reduced-order displacement and/or velocity conditions or hyper reduced counterparts, if the corresponding command(s) is/are included in the same input file containing RMSHC and the latter command is properly configured. ◦ The constant reduced-order mass matrix, when the ROB is not mass-orthonormalized (see READMODE); and the flag <code>flag_cmf</code> was set to <code>off</code> in the command RMSHC that generated the content of the file <code><PATHANDFILENAME>.elementmesh.inc</code>. ◦ Reduced-order external, constant (non-follower) forces, if the corresponding command(s) is/are included in the same input file containing RMSHC. ◦ Reduced-order linear multipoint constraints (see LMPC), if the corresponding command is included in the same input file containing RMSHC. • And a third file containing the trace on the reduced mesh of its training ROB: this file is named <code><PATHANDFILENAME>.elementmesh.inc.compressed.basis.massorthonormalized</code>, if the training ROB is mass-orthonormalized; otherwise, it is named <code><PATHANDFILENAME>.elementmesh.inc.compressed.basis</code>. In the case of local ROBs, one such file is outputted for each local basis and the extension <code><local_basis_id></code> is inserted in the filename <code><PATHANDFILENAME>.elementmesh.inc.compressed.basis.massorthonormalized</code> after the word <code>basis</code>, or appended to the filename <code><PATHANDFILENAME>.elementmesh.inc.compressed.basis</code>. <p>In the case of local ROBs, the above information is outputted for each (training) local ROB in the same output file <code><PATHANDFILENAME>.elementmesh.inc</code>. Note: When considering an online computation using a hyperreduced ROM, the user should include in the corresponding ASCII Input Command Data file the aforementioned file <code><PATHANDFILENAME>.elementmesh.inc</code> using an <code>INCLUDE</code> statement (see INTRODUCTION).</p> <ul style="list-style-type: none"> • And a third file containing the trace on the reduced mesh of its training ROB: this file is named <code><PATHANDFILENAME>.elementmesh.inc.compressed.basis.massorthonormalized</code>, if the training ROB is mass-orthonormalized; otherwise, it is named <code><PATHANDFILENAME>.elementmesh.inc.compressed.basis</code>. In the case of local ROBs, one such file is outputted for each local basis and the extension <code><local_basis_id></code> is inserted in the filename <code><PATHANDFILENAME>.elementmesh.inc.compressed.basis.massorthonormalized</code> after the word <code>basis</code>, or appended to the filename <code><PATHANDFILENAME>.elementmesh.inc.compressed.basis</code>. <p>The format of each of the above files is designed to allow their inclusion in AERO-S's input file using <code>INCLUDE</code> (see INTRODUCTION).</p> <p>MODALMAS OR SROMMASS</p> <p>ASCII file containing the reduced-order mass matrix associated with a structural dynamic ROM. In the case of a nonlinear structural dynamic ROM constructed in a floating reference frame attached to the center of gravity of the structural system (see FLOATINGFRAME), this reduced-order mass matrix corresponds to the matrix block M_{ss}.</p> <ul style="list-style-type: none"> • If the Reduced-Order Basis (ROB) underlying the linear structural ROM is of the type <code>inorm</code> or <code>nnorm</code>, this file is outputted in the following format <ul style="list-style-type: none"> * Mr: Full reduced mass matrix inorm (or nnorm) n n Mr11 Mr12 ... Mr1n

```
Mr21 Mr22 ... Mr2n
```

```
.
```

```
Mrn1 Mrn2 ... Mrnn
```

where n denotes the size of the linear or nonlinear structural dynamics ROM.

- If the ROB underlying the linear structural ROM is of the type `mnorm` (or `eigen`), this file is outputted in the following format

```
* Mr: Diagonal reduced mass matrix
  mnorm (or eigen)
  n
  Mr11 Mr22 ... Mrnn
```

where n denotes the size of the linear or nonlinear structural dynamics ROM.

MODALDMP OR SROMDAMP

ASCII file containing the reduced-order damping matrix associated with a linear structural ROM. This file is outputted in the format described below.

```
* Dr: Reduced damping matrix
  n n
  Dr11 Dr12 ... Dr1n
  Dr21 Dr22 ... Dr2n
.
.
.
Drn1 Drn2 ... Drnn
```

where n denotes the size of the linear structural dynamics ROM. This file is outputted if:

- The ROB underlying the linear structural ROM is of the type `eigen` (see [READMODE](#)), and damping in the underlying HDM is modeled in [DYNAMICS](#) as modal damping ([MODDAMP](#)) or Rayleigh damping ([RAYDAMP](#)).
- The ROB underlying the linear structural ROM is of the type `mnorm`, `inorm`, or `nnorm` (see [READMODE](#)), and damping in the underlying HDM is modeled in [DYNAMICS](#) as Rayleigh damping ([RAYDAMP](#)).

MODALSTF OR SROMSTIF

ASCII file containing the reduced-order stiffness matrix associated with a structural ROM. In the case of a nonlinear structural dynamic ROM constructed in a floating reference frame attached to the center of gravity of the structural system (see [FLOATINGFRAME](#)), this reduced-order mass matrix corresponds to the matrix block M_{ss} .

- If the Reduced-Order Basis (ROB) underlying the linear structural ROM is of the type `inorm`, `mnorm`, or `nnorm`, this file is outputted in the following format

```
* Kr: Full reduced stiffness matrix
  inorm (mnorm or nnorm)
  n n
  Kr11 Kr12 ... Kr1n
  Kr21 Kr22 ... Kr2n
.
.
.
Krn1 Krn2 ... Krnn
```

where n denotes the size of the linear structural ROM.

- If the ROB underlying the linear structural ROM is of the type `eigen`, this file is outputted in the following format

```
* Kr: Diagonal reduced stiffness matrix
  eigen
  n
  Kr11 Kr22 ... Krnn
```

where n denotes the size of the linear structural dynamics ROM.

SROMMATR

ASCII file containing the reduced-order mass, damping, and stiffness matrices associated with a (linear) structural dynamics ROM. This file is essentially the union of all 3 files associated with the results [MODALMAS](#) OR [SROMMASS](#), [MODALDMP](#) OR [SROMDAMP](#), and [MODALSTF](#) OR [SROMSTIF](#).

MODALACT OR SROMACTU

ASCII file containing the reduced-order displacement, velocity, and acceleration control matrices associated with a linear structural dynamics ROM and three gain variables (displacement, velocity, and acceleration) of an output feedback control law defined using the user-defined subroutine "control.C", [ACTUATORS](#), and [SENSORS](#). This file is outputted in the format described below.

```
* Dr: Full reduced-order displacement control matrix
  rob_type
  n n - n: size of the linear structural dynamics ROM
  Dr11 Dr12 ... Dr1n
  Dr21 Dr22 ... Dr2n
.
.
.
Drn1 Drn2 ... Drnn
* Vr: Full reduced-order velocity control matrix
  rob_type
```

```

n n - n: size of the linear structural dynamics ROM
Vr11 Vr12 ... Vr1n
Vr21 Vr22 ... Vr2n
.
.
.
Vrn1 Vrn2 ... Vrnn
* Mr: Full reduced-order acceleration control matrix
rob_type
n n - n: size of the linear structural dynamics ROM
Ar11 Ar12 ... Ar1n
Ar21 Ar22 ... Ar2n
.
.
.
Arn1 Arn2 ... Arnn

```

MODALFLF

This output result is relevant only when the simulation is performed using a nonlinear structural dynamic ROM constructed in a floating reference frame attached to the center of gravity of the structural system (see [FLOATINGFRAME](#)). In this case, this is the ASCII file containing the projection-based, reduced-order coupling tensors required to evaluate the generalized gyroscopic damping forces associated with the nonlinear structural dynamic ROM. It is outputted in the following format

```

* B_p[0][0]: Full reduced-order fourth-order tensor
3 3
B_p[0][0]11 B_p[0][0]12 B_p[0][0]13
B_p[0][0]21 B_p[0][0]22 B_p[0][0]23
B_p[0][0]31 B_p[0][0]32 B_p[0][0]33
repeat for B_p[0][1], ..., B_p[0][n-1], B_p[1][0], ..., B_p[1][n-1], ..., B_p[n-1][n-1]
* B_t[0]: Full reduced-order third-order tensor
n 3
B_t[0]11 B_t[0]12 B_t[0]13
B_t[0]21 B_t[0]22 B_t[0]23
...
B_t[0]n1 B_t[0]n2 B_t[0]n3
repeat for B_t[1], ..., B_t[n-1]
* B_h[0]: Full reduced-order third-order tensor
3 3
B_h[0]11 B_h[0]12 B_h[0]13
B_h[0]21 B_h[0]22 B_h[0]23
B_h[0]31 B_h[0]32 B_h[0]33
repeat for B_h[1], ..., B_h[n-1]

```

FORMAT

where n denotes the number of flexible modes of the structural dynamic system.

This optional field can be used to override the default format for outputting a numerical value by one which consists of 2 integer numbers F and P separated by a blank (integers). F specifies the field width — that is, the total number of digits to be printed including the decimal point, the "E" of the exponent and the sign of the exponent, but excluding the sign of the output value. P is the number of digits after the decimal and before the exponent. For example, 8 2 results in printing the output value 1523 as 0.15E+03.

FNAME
INCREMENT

User specified filename associated with result (characters).

Output increment for transient and eigenvalue results (integer). For a FARFIELD output result, this field specifies, in the three-dimensional case, the number of longitudinal directions where the farfield is to be evaluated. In this case, $(INCREMENT/2)+1$ latitudinal directions are also considered and therefore the farfield is evaluated at $((INCREMENT/2)+1)*INCREMENT$ points uniformly distributed in spherical coordinates. In two-dimensions, the farfield is evaluated at INCREMENT points that are evenly distributed on a circle. For transient and quasi-static analyses, the final time-instance is always outputted for the requested fields — except for the aeroelastic forces and moments — independently of the value of INCREMENT.

N
NODE_NUMBER

Sub-command keyword (character) signaling that the next entry is a node identifying number.

When this optional field is used, with or without the sub-command keyword N, only the computed results associated with the node NODE_NUMBER are outputted in the same format as otherwise, except that the header of the **XPost** file is omitted for easier use of a visualization software such as gnuplot (integer).

NG
GROUP_NUMBER

Sub-command keyword (characters) signaling that the next entry is a group identifying number (see [GROUPS](#)).

When this optional field is used together with the sub-command keyword NG, only the computed results associated with the nodes belonging to the group of nodes GROUP_NUMBER (see [GROUPS](#)) are outputted in the following format which is suitable for a visualization software such as gnuplot (integer).

FLOAT-1

NODE_NUMBER-1 X Y Z RESULT-1 RESULT-2 ... RESULT-n

NODE_NUMBER-2 X Y Z RESULT-1 RESULT-2 ... RESULT-n

...

FLOAT-2

NODE_NUMBER-1 X Y Z RESULT-1 RESULT-2 ... RESULT-n

NODE_NUMBER-2 X Y Z RESULT-1 RESULT-2 ... RESULT-n

...

where FLOAT-i is an i -th time-instance in the case of a dynamic analysis or a circular frequency value

	(number of cycles per second) in the case of a frequency sweep analysis (see IMPEDANCE), NODE_NUMBER-j is the j -th node number of group GROUP_NUMBER, X, Y and Z are the x , y , and z coordinates of NODE_NUMBER-j, and RESULT-1 RESULT-2 ... RESULT-n are all the computed results pertaining to NODE_NUMBER-j and the chosen instance of RESULT.
OPTION	Any of the following optional fields can be specified after the previous optional and/or non-optional entries documented above:
NDTYPE [NUM_OUT_RLZ]	NDTYPE is an optional parameter that can take one of the following values: MEAN, STDV, or PDF. It should be used only when an intrusive or non intrusive non deterministic analysis is performed. When MEAN is selected and a non intrusive non deterministic analysis is performed, the mean value of the RESULT is outputted using the number of realizations NUM_RLZ specified in the NONINPC command. When STDV is chosen and a non intrusive non deterministic analysis is performed, the standard deviation of the RESULT is outputted using the number of realizations NUM_RLZ specified in the NONINPC command. When PDF is specified, the probability density function is outputted using a number of realizations equal to the integer set in the suboptional field NUM_OUT_RLZ (integer). First, the shape functions are evaluated at NUM_OUT_RLZ realizations of the random system. Then, NUM_OUT_RLZ realizations of the non deterministic RESULT are reconstructed from these evaluations using $v^{(j)} = \sum_{i=0}^{P-1} v_i^{(j)} \psi_i^{(j)}$ in the case of a displacement or $\sigma^{(j)} = \sum_{i=0}^{P-1} \sigma_i^{(j)} \psi_i^{(j)}$ in the case of a stress field σ and written in the output file (note that the case of a von Mises stress is treated slightly differently because it is a nonlinear function of the stress tensor components). A histogram can be constructed from these results.
AVG_OPTION	If the output of a stress field, strain field, or the damage parameter field DAMAGEC is requested, the user can set this optional parameter to one of the following values: NODALFULL, NODALPARTIAL, NODALPARTIALGROUP, ELEMENTAL, or GAUSS (characters), to specify a type of stress/strain/damage averaging. If NODALFULL is specified, the target field is averaged at each node using all elements attached to this node. In this case, the contributions of the beam and bar elements are first transformed to the global frame. If NODALPARTIAL is specified, the averaging is performed using only the elements that are neither of a bar or a beam type. If NODALPARTIALGROUP is specified and the keyword NG is used to activate output for a group of nodes, the averaging is performed using only those elements whose nodes belong to the specified group. If ELEMENTAL is specified, the field is outputted without any averaging at each node of each element. Note however that currently, the option ELEMENTAL is not supported when a FETI solver is invoked. If GAUSS is specified, the target field is outputted without any averaging at each Gauss point of each element. This latter option is currently supported however only for nonlinear analysis using three-dimensional solid elements (type 17, 23, 24, 25, 72, 91, 92 and 94) for which constitutive laws are specified using MATLAW .
STR_TYPE_OPTION	This optional field is applicable only for <i>linear</i> analyses. It can have any of the following settings: THERMAL, MECHANICAL, or THERMOMECHANICAL (default setting). In the first case, the computed and outputted stresses are the thermal stresses defined as $-Cw(T - Ta)$, where C is the pertaining constitutive matrix, w is the thermal expansion coefficient specified in MATERIAL , T is the temperature (absolute), and T_a is the ambient (reference) temperature (absolute) also specified in MATERIAL . In the second case, the computed and outputted stresses are the mechanical stresses defined as $C\epsilon$, where ϵ is the strain tensor. In the last case, which is the default setting, the computed and outputted stresses are the thermomechanical stresses defined as $C(\epsilon - w(T - Ta))$.
STR_SHELL_OPTION	This optional field is applicable only when the output of a stress/strain field of a shell element is requested. It can take any of the following values UPPER, MEDIAN, LOWER (characters). If UPPER is specified, the stresses are computed on the upper surface of the element. If MEDIAN is specified (default value), they are computed on the median surface of the element. If LOWER is specified, they are computed on the lower surface of the element.
STR_BEAM_OPTION	This optional field is applicable only when the output of STRAINXX or STRESSXX of a beam element is requested. It specifies two real numbers (real). The first (second) real number represents the positive or negative y (z) coordinate in percentage of the maximum positive (top fiber) or negative (bottom fiber) y (z) coordinate of the beam cross section, in the local frame attached to the beam, of the fiber at which ϵ_{xx} or σ_{xx} is requested. The default value for both real numbers is zero.
COMPLEX_OUTPUT_OPTION [NSTEPS]	These optional and suboptional fields are applicable only when the output is complex-valued. COMPLEX_OUTPUT_OPTION can take any of the following values REALIMAG, MODPHASE, ANIM (characters). When set to REALIMAG, the real parts of the complex-valued output are printed first and the imaginary parts are printed next. When set to MODPHASE, the moduli p of the output values are first printed and then the phases θ are printed. When set to ANIM, the values of $p \cos(\theta - \phi_1)$ are printed first, followed by the values of $p \cos(\theta - \phi_2)$, etc, concluding with the values of $p \cos(\theta - \phi_N)$, where ϕ_i is defined as $\frac{2\pi}{N}$ and N is specified in the suboptional field NSTEPS (integer).
FRM_OPTION	This optional parameter can be used to request outputting directional results (such as, for example, the x , y and z components of a computed displacement field, or the xy component of a computed stress field) in the nodal reference frames defined at the nodes of the finite element model, in the global reference frame, or in the material reference frame. In the first case, this parameter should be set to NODFRA. In the second case, its should be set to GLOFRA. In the third case, it should be set to MATFRA. Its default value is GLOFRA. For this reason, when outputting a directional result to be reused with the READMODE command, this parameter should be set to NODFRA.
ORTHONORM_OPTION	This optional field is applicable only when the result to be outputted is GEIGENPA. It specifies whether to output the computed mass-orthonormalized eigenvectors or associated orthonormalized vectors (characters). Hence, it can take one of the two following values:

	<code>mnorm</code>	In this case, which is the default case, the computed mass-orthonormalized eigenvectors are outputted.
	<code>inorm</code>	In this case, (identity-) orthonormalized vectors that span the same subspace as the mass-orthonormalized eigenvectors are outputted.
<code>ROTVEC_OPTION</code>		This optional parameter can be used to select the specific type of rotation vector output for <code>GDISPLAC</code> (only under OUTPUT6), <code>ROTATIOX</code> , <code>ROTATIOY</code> , <code>ROTATIOZ</code> , <code>ROTATMOD</code> , and <code>GDISPMOD</code> . It can take values <code>Euler</code> (default value), <code>Complement</code> , <code>Linear</code> , <code>ReducedEulerRodrigues</code> , <code>CayleyGibbsRodrigues</code> , <code>WienerMilenkovic</code> , or <code>BauchauTrainelli</code> . The default <code>Euler</code> rotation vector, also known as the axis-angle representation, is a vectorial parameterization of the rotation group whose magnitude is the (scalar) rotation angle $ \theta $ and rotation axis is the unit 3-vector \hat{e} — that is, $\Psi = \theta\hat{e}$. This is not to be confused with non-vectorial Euler angles for the parameterization of rotation. The complement rotation vector is specifically the complement of the Euler rotation vector and corresponds to a rotation angle of $(2\pi - \theta)$ about the axis $-\hat{e}$. For details about the other options, refer to <i>The vectorial parameterization of rotation</i> , by Bauchau and Trainelli, in <i>Nonlinear Dynamics</i> , 32, No 1, pp. 71–92, 2003. Regardless of the value of <code>ROTVEC_OPTION</code> , the rotation vector is a representation of the “total rotation” — that is, it is measured with respect to the reference orientation (which corresponds in AERO-S to the case where the rotation matrix is the identity matrix).
<code>ORTHONORM_OPTION</code>		This optional field is applicable only when the result to be outputted is <code>GEIGENPA</code> . It specifies whether to output the computed mass-orthonormalized eigenvectors or associated orthonormalized vectors (characters). Hence, it can take one of the two following values:
	<code>mnorm</code>	In this case, which is the default case, the computed mass-orthonormalized eigenvectors are outputted.
	<code>inorm</code>	In this case, (identity-) orthonormalized vectors that span the same subspace as the mass-orthonormalized eigenvectors are outputted.
<code>RESCALING</code>	<code>FLAG</code>	Optional sub-command keyword followed by an <code>on</code> or <code>off</code> flag which can be used to define whether or not to rescale the rotation vector — which is a one-to-many mapping from the rotation matrix. It applies to <code>GDISPLAC</code> , <code>GVELOCIT</code> , and <code>GACCELER</code> under OUTPUT6 , and to <code>ROTATIOX</code> , <code>ROTATIOY</code> , <code>ROTATIOZ</code> , <code>ROTATMOD</code> , and <code>GDISPMOD</code> . The default value of <code>FLAG</code> is <code>on</code> . The specific interpretation of this option depends on the setting of <code>ROTVEC_OPTION</code> . For example, in the case of an Euler rotation vector, setting <code>FLAG</code> to <code>on</code> means that the outputted rotation vector is that with a rotation angle $\theta \leq \pi$. When the rotation vector is rescaled, its components become discontinuous at the point of rescaling. Otherwise, its components may be discontinuous at certain singularity points ($\theta = 2\pi$ in the case of the Euler rotation vector).
<code>ANGULAR_OPTION</code>		This optional parameter can be used to choose a definition of the angular velocity or acceleration for <code>GVELOCIT</code> or <code>GACCELER</code> under OUTPUT6 . It can take the following values: <code>Convected</code> (default), <code>Spatial</code> , and <code>Total</code> . The <code>Convected</code> definition refers to a moving frame that is computed by AERO-S and attached to each node. The definition <code>Spatial</code> refers to the fixed reference frame, and the definition <code>Total</code> refers to the time derivatives of the (Euler) rotation vector. In the case of <code>Total</code> , the <code>RESCALING</code> option may also be used to specify whether the angular velocity (acceleration) is the first (second) time derivative of the rescaled or non-rescaled Euler rotation vector. Note that angular velocities and accelerations inputted using <code>IVEL</code> or <code>USDD</code> should be <code>Convected</code> .

Next: [PITA](#), Previous: [OUTPUT](#)

98 OUTPUT OF RESULTS 6 COLUMNS (OUTPUT6 completely spelled out)

Command Statement: **OUTPUT6** [KEYLETTER]

The `OUTPUT6` command has the same options and format of the `OUTPUT` command. Its outcome differs from that of `OUTPUT` only when vector results such as generalized displacements fields are to be outputted. In that case, the `OUTPUT6` command forces **AERO-S** to output all six components of the generalized displacement field in a format slightly different from that of the file created by `OUTPUT`, in that the node number is included at the beginning of each line.

Note 1: When multiple outputs are expected, they are printed one set after the other in the same output file.

Note 2: When both `OUTPUT` and `OUTPUT6` commands are used in the same input file, the `KEYLETTER` option specified for one of these two commands applies to all output files requested under both of them; if this option is specified twice, once for each of these two commands, and the two specified settings differ, the one which is specified last in the input file prevails.

Next: [PRELOAD](#), Previous: [OUTPUT6](#)

99 PARALLEL-IN-TIME ALGORITHM (PITA)

Command Statement: **PITA**

The `PITA` command statement is used to request the parallelization in time of the time-integration algorithm specified under the `DYNAMIC` command. This option is supported only when executing the distributed version of the **AERO-S** executable. The simultaneous presence of the command statement `NONLINEAR` in the input file automatically triggers the nonlinear version of the `PITA` algorithm. By default, the initialization of the time-slices is performed by running the chosen time-integrator on the coarse time-grid. Alternatively, the `IDISP6PITA` (see [IDISP6PITA](#)) and `IVEL6PITA` (see [IVEL6PITA](#)) commands can be used to specify the seed displacements and velocities. The input format is given below.

Note 1: The current implementation allows only geometric nonlinearities.

Note 2: The foundations of the PITA methodology are described in: J. Cortial, *Time-Parallel Methods for Accelerating the Solution of Structural Dynamics Problem*, PhD Thesis, Stanford University, 2011.

[PITA]

J_RATIO	MAX_ITER
PITA_OPTIONS	

or

J_RATIO	MAX_ITER	MAX_SLICES_ON_CPU
PITA_OPTIONS		

J_RATIO	Ratio between the time-steps on the fine and coarse time-grids. Also, number of time-steps in each time-slice (integer).	
MAX_ITER	Maximum number of outer PITA iterations (integer).	
MAX_SLICES_ON_CPU	Maximum number of time-slices assigned to a processor (integer). If the number of time-slices is larger than the number of available CPUs, the supernumerary time-slices are ignored. For optimal performance, there should be as many available CPUs as time-slices. The default value is 1.	
PITA_OPTIONS	None, one or several of the following optional statements	
TIMEREVERSIBLE	Instructs AERO-S to exploit the time-reversibility of the problem. In this case, each time-slice is divided into 2 halves, one treated by usual forward-in-time integration, and the other by backward-in-time integration. For optimal performance, 2 CPUs should be assigned to each time-slice.	
ORTHOPROJTOL	orthoprojtol	Sets the relative tolerance used to construct the orthogonal projectors for the correction step. The default value is 10^{-5} .
READINITSEED	Instructs AERO-S to use the initial seed information provided via the commands IDISP6PITA and IVEL6PITA . If additional seed values are required, they are obtained in this case via time-integration on the coarse time-grid.	
REMOTECOARSE	Instructs AERO-S to perform the coarse time-grid integration on a dedicated CPU, which will not be available during the parallel integration on the fine-time grid (linear case only). For optimal performance in the linear case, it is recommended to use this option and provide one additional CPU. This command is ignored when only one CPU is available.	
NOFORCE	Informs AERO-S about the absence of any external force so that a special version of the PITA can be executed for optimal performance (linear case only).	
JUMPCVG	jumpcvg	Instructs AERO-S to enable the jump-reduction-based convergence criterion with the specified tolerance jumpcvg (linear case only). The default value is J_RATIO^2 .
GLOBALBASES	integer	Determines the scheme to use for updating the global bases using the following convention: 1 = all seed vectors, 2 = all seed and propagated vectors (nonlinear case only). The default value is 1.
LOCALBASES	Instructs AERO-S to use the local basis updating scheme instead of the global one (nonlinear case only). This option may improve performance but at the expense of time-accuracy. It is not implemented (and therefore ignored) for time-reversible simulations.	
JUMPOUTPUT	Instructs AERO-S to output the magnitudes of the jumps.	

Next: [PRESSURE](#), Previous: [PITA](#)

100 PRELOAD

Command Statement:	PRELOAD
--------------------	----------------

The PRELOAD command statement can be used to specify a preload for a truss or a membrane element. For a truss element, the preload is an axial force that affects its stiffness matrix. For a membrane element, the preload is a three-dimensional generalized force (or force per unit directional length) that also affects its element stiffness matrix. In both cases, the default value(s) of the the preload is (are) zero. It is assumed that the nodal coordinates of the model specified under [NODES](#) include the deformations due to preload.

Preload affects all of linear, nonlinear, static, transient, and eigen analyses. For a truss element, the outputted internal axial force includes the amount of preload. For a membrane element, the outputted stresses also include the amount of preload.

The following two formats are supported and can be mixed.

[PRELOAD]

ELEMENT#	PRELOAD_VALUE1	PRELOAD_VALUE2	PRELOAD_VALUE3
----------	----------------	----------------	----------------

STARTING_ELEMENT#	THRU	ENDING_ELEMENT#	PRELOAD_VALUE1	PRELOAD_VALUE2	PRELOAD_VALUE3
-------------------	------	-----------------	----------------	----------------	----------------

ELEMENT#	Element number where a preload is to be specified (integer).
PRELOAD_VALUE1	For a truss element, value of the axial preloading force. For a membrane element, value of the preload generalized force (force per unit length) in the global <i>x</i> direction (float).
PRELOAD_VALUE2	For a membrane element, value of the preload generalized force (force per unit length) in the global <i>y</i> direction (float).
PRELOAD_VALUE3	For a membrane element, value of the preload generalized force (force per unit length) in the global <i>z</i> direction (float).
STARTING_ELEMENT#	First element of a sequence of elements where a specified preload value or set of values apply (integer).
THRU	Keyword indicating that the following entry is the last element of a sequence of elements (characters).
ENDING_ELEMENT#	Last element of a sequence of elements where a preload value or set of values apply (integer).

Next: [PRINTMAT](#), Previous: [PRELOAD](#)

101 PRESSURE

Command Statement:	PRESSURE [LOADSET_ID]
--------------------	------------------------------

The PRESSURE command can be used to specify a piece-wise uniform pressure field, and/or a pressure field resulting from an air blast in the free field and whose parameters are specified under the [CONWEP](#) command. The specified pressure field can be applied to beam elements, membrane elements, shell elements, and surface elements defined in [SURFACETOPO](#). For a beam element (type 6 or 7), the pressure is actually a *load per unit length* applied in the Y-direction of the **element frame** specified in the [EFRAMES](#) command or implied by the third node technique for beam elements (see [TOPOLOGY](#)).

For a membrane element (type 128, or 129), a shell element (type 8, 88, 15, 1515, 16, 20, 2020, 73, or 76), or a surface element defined in [SURFACETOPO](#) for the sole purpose of defining a surface computation that may be independent from any specific element type, the pressure field is applied normal to the surface of the element using the usual convention of a positive outward normal — that is, if the three nodes of a shell element are denoted by *a*, *b*, and *c* and numbered in this order, the normal is computed as $\vec{ab} \wedge \vec{ac}$. Furthermore, a positive value of PRESSURE_VALUE implies a force in the direction of the normal.

In a nonlinear analysis, the generated pressure forces are of the **follower** type for shell elements and surface elements defined in [SURFACETOPO](#).

Note 1: AERO-S always generates pressure loads using a consistent method.

Note 2: The piece-wise uniform pressure field represented by the pressure value PRESSURE_VALUE can be varied in time using the command [MFTT](#).

Note 3: If the surface of interest is already discretized with beam or shell elements, using the ELEMENT# method of pressure input is more consistent with the remainder of the finite element analysis than using the SURFACE counterpart, because in this case the pressure loads are computed using the shape functions of the corresponding beam or shell elements.

Note 4: The blast pressure generated by CONWEP is currently not applicable to beam or membrane elements.

The following three formats are available for this command and can be mixed.

PRESSURE [LOADSET_ID]

ELEMENT#	PRESSURE_VALUE	[CONWEPSWITCH]
----------	----------------	----------------

STARTING_ELEMENT#	ENDING_ELEMENT#	PRESSURE_VALUE	[CONWEPSWITCH]
-------------------	-----------------	----------------	----------------

SURFACE	SURFACE#	PRESSURE_VALUE	[CONWEPSWITCH]
---------	----------	----------------	----------------

LOADSET_ID	Optional non-negative integer which identifies explicitly the "load" set to which the source term generated by this command belongs to (integer). The default value is 0. Hence, the PRESSURE command can be repeated as many times as desired within the same input file using each time a different value for LOADSET_ID and different data. The LOADCASE command can refer to LOADSET_ID to define one or multiple "load" cases for static analysis (see the QSTATIC s command and the explanation of its sub-command keyword CASES), and/or the "load" case for dynamic analysis.
ELEMENT#	Element number where a pressure field is to be specified (integer).
PRESSURE_VALUE#	Value of the piece-wise uniform pressure field (or load per unit length for a beam element) in that element (float). If CONWEPSWITCH is set to on, this pressure value is added to the blast pressure generated by the CONWEPS module.
CONWEPSWITCH	On/Off switch for the pressure load due to an air blast in the free field whose parameters are set in the CONWEPS command (characters). If this switch is set to On, the pressure field computed by the software module CONWEPS is added to that specified in PRESSURE_VALUE. Otherwise, only the pressure value specifies in PRESSURE_VALUE is used to generate the pressure load to be applied on the specified element or set of elements. The default value of this switch is On.
STARTING_ELEMENT#	First element of a sequence of elements where the piece-wise uniform pressure field has the same value (integer).
ENDING_ELEMENT#	Last element of a sequence of elements where the piece-wise uniform pressure field has the same value (integer).
SURFACE	Keyword indicating that a surface defined in SURFACETOPO is to be identified next by its integer identification number (characters).
SURFACE#	Integer identification of the surface defined in SURFACETOPO where the uniform pressure value PRESSURE_VALUE is to be applied (integer).

Next: [QSTATIC](#), Previous: [PRESSURE](#)

102 PRINTMAT

Command Statement: **PRINTMAT**

The PRINTMAT command statement can be used to request outputting to ASCII files the global stiffness and mass matrices in matrix market format. This format is a sparse matrix format that can be read by other programs such as MATLAB.

The numbering of the rows and columns of the printed sparse matrices corresponds to a trivial numbering of all unconstrained degrees of freedom attached to an element of the mesh. It is constructed as follows. First, any constrained and unused degrees of freedom are eliminated. Second, the remaining degrees of freedom are numbered by sorting them in ascending order of their node number. Third, the degrees of freedom with a common node number are further sorted in ascending order according to the degree of freedom index (1: x-displacement, 2: y-displacement, 3: z-displacement; 4: x-rotation, etc). Any other renumbering system specified using the [RENUM](#) command is ignored as far as this command is concerned.

Note 1: This command may only be used in the absence of a mesh decomposition: currently, it is not supported for explicit dynamics.

Note 2: If any hyper reduction coefficients are specified in the [ATTRIBUTES](#) command, the outputted matrices are the stiffness and mass matrices assembled using the inputted mesh or sampled mesh, weighted by these coefficients.

Note 3: In the presence of this command in the ASCII Input Command Data file, **AERO-S** will exit after printing the aforementioned matrices and therefore will not perform any specified analysis.

PRINTMAT pathandfilenameprefix

pathandfilenameprefix Specifies a prefix for the output files to contain the global stiffness and mass matrices in matrix market format (character string). The suffixes ".stiffness" and ".mass" will be automatically appended to this prefix by **AERO-S**.

Next: [RANDOM](#), Previous: [PRINTMAT](#)

103 QUASISTATICS ANALYSIS

Command Statement: **QSTATIC**

The QSTATIC command statement is used to signal that the subsequent data lines correspond to the following quasistatics *iterative* algorithm for solving either $Ax=b$, where A represents a diagonal block of a coupled linear-linear or linear-nonlinear (multidisciplinary) system and can be singular, or $r(x,b)=0$, where r represents a row block of a coupled nonlinear-linear or nonlinear-nonlinear (multidisciplinary) system.

For problems of the form $Ax=b$, the *iterative* solution algorithm underlying this command can be described as follows

$$\begin{aligned} A\tilde{x} &= b^{n+1} \\ \alpha^{n+1} &= \alpha^n + \beta R^T b^{n+1} \\ \bar{x}^{n+1} &= \bar{x}^n + \theta(\tilde{x} - \bar{x}^n) \end{aligned}$$

$$x^{n+1} = \bar{x}^{n+1} + R\alpha^{n+1}$$

For problems of the form $r(x,b)=0$, the *iterative* solution algorithm underlying this command can be described as

$$\begin{aligned} r(\tilde{x}, b^{n+1}) &= 0 \\ x^{n+1} &= x^n + \theta(\tilde{x} - x^n) \end{aligned}$$

For single discipline analysis, the above algorithm is at best ($\theta = 1$) a direct solution method.

This command is currently relevant for aeroelastic and aerothermal static analyses where the structural or thermal system (respectively) is linear, or aeroelastic static analyses where the structural system is nonlinear. In the first case, A can represent a structural stiffness matrix for aeroelastic analysis, or a conductivity matrix for an aerothermal analysis. In this case, the second and fourth steps of the iterative solution algorithm are enabled only for an aerothermal analysis in which the thermal problem is singular, A represents the matrix of the thermal sub-system, and R represents a basis of its null space. In the second case, r represents the residual of the nonlinear equations governing static equilibrium of the structural system.

This command can also be used to compute a flow-induced load and perform the corresponding static structural analysis. In this case, it must be used together with the staggered solution procedure `&0` of the [AERO](#) command.

For all purposes outlined above, this command requires the additional usage of the [AERO](#) command to specify either the structure matcher file, or the discrete surface to be embedded in the fluid grid, as needed.

Note 1: An equation solver must be specified under the [STATICS](#) command.

The input format of this command can be as follows.

[OSTATICS]

MECH	θ	tolqs	maxqs	delta
HEAT	θ	β	tolqs	maxqs

θ Underrelaxation factor $0 < \theta \leq 1$.
 β Underrelaxation factor $0 < \beta \leq 1$.

tolqs Convergence tolerance.

maxqs Maximum number of iterations.

delta Time-step equivalent-value of the increment between any two iterations. This parameter is taken into account only in:
(a) the presence of a user-defined "control.C" file (see [ACTUATORS](#), see [SENSORS](#), see [USDD](#), and see [USDF](#)),
and/or (b) the time-stamp in the output format for quasi-static analysis. It is the equivalent of a time-step and is used to convert the current iteration into current time. The default value is 0. When `delta` is specified to a non-zero value, θ is automatically set to $\theta = 1$ and convergence is not checked because in that case, convergence is reached in one iteration and it is understood that the purpose of the quasistatics analysis is to solve a series of consecutive static problems whose right-hand sides are set by a "control.C" file.

Next: [READMODE](#), Previous: [OSTATICS](#)

104 RANDOM

Command Statement: **RANDOM**

The `RANDOM` command statement is used to signal that some materials have properties with random values. All random material properties are assumed to have a Gaussian probability distribution.

Note 1 : Currently, an input file can specify only one random material property per group.

Note 2 : To ensure the positivity of the material properties at every realization, the following non-Gaussian model is used

$$\text{MATPROP} = \text{MEAN} + \frac{\text{STDV}}{\sqrt{2}}(\xi^2 - 1)$$

with the constraint $\frac{\text{STDV}}{\sqrt{2}} < \text{MEAN}$. Here ξ denotes a standard normal random variable.

The input format of this command is given below.

RANDOM

GROUP#	MATPROP	MEAN	STDV
--------	---------	------	------

GROUP# Id of the element group for which the random properties are to be specified (integer).
 MATPROP# Material property for which the random properties are to be specified (string). Currently, this keyword can take any of the following values: e (Young's modulus), A (cross section of a bar or beam element), and k_x , k_y , and k_z (linear spring coefficients (see [MATERIAL](#))).
 MEAN Mean value of the non deterministic MATPROP (real). This value overwrites the deterministic value of MATPROP specified under the MATERIAL command for the elements in the corresponding GROUP#.
 STDV Standard deviation of the non deterministic MATPROP (real).

Next: [RODC](#), Previous: [RANDOM](#)

105 READING REDUCED BASES

Command Statement: **READMODE**

The command READMODE can be used for:

- Reading one or multiple primal Reduced-Order Bases (ROBs) V .
- Reading one or multiple dual ROBs W constructed for the model reduction of constrained *nonlinear* problems.
- Signaling to **AERO-S** the context of a *projection-based* Reduced-Order Model (ROM).

Each inputted ROB may be identified by an integer ID number and characterized by a type that covers, among others, different forms of an eigen basis (basis of eigenvectors) or a POD (Proper Orthogonal Decomposition) basis (see [ROBC](#)), and a non-negative basis. Such a ROB can be used, for example, to perform reduced-order dynamic computations (see [DYNAMICS](#)), or a "Basis Ping-Pong" analysis (see [AERO](#)).

Initial conditions for the generalized coordinates associated with a chosen primal ROB can be directly specified using, for example, [IDISPLACEMENTS](#), [IVELOCITIES](#), and/or [ITEMPERATURES](#), as appropriate.

A primal ROB V and a dual ROB W can also be used to hyper reduce linear and nonlinear problems with equality or inequality constraints (see below for limitations).

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: Currently, the reduction of inequality constraints is supported only for nonlinear implicit dynamic computations such as nonlinear implicit dynamic contact problems, and only for those cases where the constraints are linear and enforced using the Lagrange multipliers method.

Note 3: In such cases, snapshot collection for the construction of a dual basis can be performed only when the constraint method (see [CONSTRAINTS](#)) is one of the following methods:

- multipliers equipped with the FETI DP solver (see [STATICS](#)).
- augmented equipped with the parallel `mumps` solver (see [STATICS](#)).

Note 4: Currently, the reduction of equality constraints is supported only when such constraints are enforced using the penalty method.

Two formats are available for this command:

- The first command format described below should be used only in the case of:
 - A primal structural ROB that is a mass-orthonormalized eigen basis.
 - A primal thermal ROB that is a capacitance-orthonormalized eigen basis.
 - An arbitrary primal ROB that is destined for a "Basis Ping-Pong" analysis (see [AERO](#)).

READMODE **<pathandfilename>** **numvec**

<pathandfilename> Name of a file (including path, if needed) containing a primal ROB to be read and whose ID (see `rob_id` below) is automatically set to 0; or prefix of two binary files containing the orthonormalized and mass-orthonormalized instances of a primal ROB generated by the [ROBC](#) command (see [ROBDAF](#) under the [OUTPUT](#) command)

(characters). In the first case, all three formats described below are supported for this file, whereas in the second case, only the third format described below is supported:

- The first file format is an ASCII format that can be described as follows. The first line contains the number of ROB vectors included in this file. The second line specifies the number of nodes of the underlying mesh. The ROB vectors follow afterwards. Each ROB vector is specified by a scalar written on a separate line, followed by a number of lines equal to the number of nodes of the high-dimensional model. The scalar is:
 - A frequency (number of cycles per second) in the case of a primal structural ROB.
 - A singular value in the case of a POD-based structural ROB.
 - An eigenvalue of the thermal pencil in the case of a thermal ROB.
- Each line following this scalar contains the 1 (temperature), 3 (displacements), or 6 (displacements and rotations) values of the ROB that node. Alternatively, each ROB vector can be specified by the same scalar described above written on a separate line, followed by a number of lines equal to the number of nodes, each containing the node number followed by the 1 (temperature), 3 (displacements), or 6 (displacements and rotations) values of the ROB at this node.
- The second file format is the **XPost** ASCII format. In this case, the first line of the file contains a header and the second one contains the number of nodes in the underlying mesh. The ROB vectors follow afterwards as in the first file format.
- The third file format is a binary format generated by the **ROBC** command and readable by the present command. In this case: `<pathandfilename>` should be set to the prefix of the two binary files containing the orthonormalized and mass-orthonormalized instances of a primal ROB generated by the **ROBC** command (see **ROBDAF** under **OUTPUT**); and **AERO-S** uses the flag `flag_mnb` set in the **RMSHC** command and/or the type of the time-integration scheme specified in the **DYNAMICS** command to select one of the two inputted ROB instances – as explained in **RMSHC**.

`numvec`

When the input file `<pathandfilename>` is written in the first file format described above, the size of the ROB adopted by **AERO-S** is the number of ROB vectors specified on the first line of this input file. When this file is written in the second or third file format described above, the size of this ROB is equal to the number of ROB vectors contained in the aforementioned file. However, if this parameter is specified and set to a non-zero value, only the first `numvec` ROB vectors are read from the input file `<pathandfilename>`: in this case, whatever file format `<pathandfilename>` is written in, the size of the primal ROB adopted by **AERO-S** becomes equal to `numvec` (integer). On the other hand, if this parameter is omitted or set to 0, the number of vectors constituting the ROB – and therefore the size of this ROB – is obtained from the content of `<pathandfilename>`.

Finally, if this parameter is set to a value greater than the size of the inputted ROB: if the input file `<pathandfilename>` is written in the first or third file format described above, `numvec` is reset to the size of the inputted ROB; but if the input file `<pathandfilename>` is written in the second file format described above, **AERO-S** throws an error message and interrupts the simulation.

- The second command format described below should be used in the case of:

- One or multiple primal, structural, mass- or identity-orthonormalized eigen or other ROBs.
- One of multiple primal, structural, ROBs that are not necessarily orthonormalized.
- One or multiple dual, non-negative ROBs (for example, non-negative Lagrange multiplier bases for constrained nonlinear analyses).

`READMODE`

<code>rob_id1</code>	<code>rob_type1</code>	<code><pathandfilename1></code>	<code>numvec1</code>	<code>[tolerance1]</code>
.	.			
.	.			
<code>rob_idi</code>	<code>rob_typei</code>	<code><pathandfilenamei></code>	<code>numveci</code>	<code>[tolerancei]</code>
.	.			
.	.			
<code>rob_idN</code>	<code>rob_typeN</code>	<code><pathandfilenameN></code>	<code>numvecN</code>	<code>[toleranceN]</code>

`rob_idi`

Integer number identifying the *i*-th inputted (primal or dual) ROB to support its usage in other commands such as **DYNAMICS**, **IDISPLACEMENTS**, **IVELOCITIES**, and **ITEMPERATURES** (integer).

`rob_typei`

Type of the *i*-th inputted ROB (characters). It can take one of the following values:

- `eigen`: in this case, the inputted ROB V must satisfy — within the specified absolute tolerance `tolerance` — both properties $V^T M V = I$ and $V^T K V = \Omega^2$, where M and K are the mass and stiffness matrices associated with the finite element model inputted in this ASCII Input Command Data file, Ω^2 is the diagonal matrix of the squares of its natural circular frequencies, and I is the identity matrix.
- `mnorm`: in this case, the inputted ROB V must satisfy — within the specified absolute tolerance `tolerance` — the property $V^T M V = I$.
- `inorm`: in this case, the inputted ROB V must satisfy — within the specified absolute tolerance `tolerance` — the property $V^T V = I$, or may simply have full column rank and not satisfy any orthonormality property.
- `nnorm`: in this case, the inputted ROB does not have to satisfy any orthogonality property but its columns must be linearly independent.
- `noneg`: in this case, the inputted ROB V must satisfy the property that none of its entries is negative.

`<pathandfilenamei>`

Name of the *i*-th file (including path, if needed) containing a primal or dual ROB to be read (characters). The type (ASCII/binary) and format of this file follow the same rules as those governing the type and format of `<pathandfilename>` (see above). If the ROB to be read is a primal ROB, this name must have the extension ".massorthonormalized", unless `rob_type_i` is set to `inorm`.

`numveci`

Specifies the size of the *i*-th inputted primal or dual ROB (integer). This parameter is governed by the same rules as those governing `numvec`.

tolerancei

Tolerance to be used by **AERO-S** to verify that the i -th inputted primal ROB satisfies the mathematical property associated with its type (real). For example if the ROB V is declared to be mass-orthonormalized, **AERO-S** performs the following verification

$$\frac{\|V^T MV - I\|_F}{\|I\|_F} < \text{tolerancei}$$

If the ROB V is declared to be of the type `nnorm`, **AERO-S** performs the Singular Value Decomposition (SVD) of V and verifies that each of its singular values σ_k satisfies

$$\sigma_k > \text{TOLERANCE}_i$$

AERO-S performs this verification if and only if this parameter is set to a non zero value, and the requested analysis is a linear structural dynamic analysis using a ROM constructed with the sub-command `MODAL` or `SROM` of **DYNAMICS**. If this parameter is skipped or assigned a zero value, **AERO-S** skips this verification. One good reason for skipping this verification is if the computational model does not include a mass or stiffness matrix, as possible in the case of a modal analysis using a ROB of the type `eigen`.

Next: [RENUMBERING](#), Previous: [READMODE](#)

106 RECONSTRUCTING THE SOLUTION FROM THE REDUCED COORDINATES

Command Statement: **RODC**

The `RODC` command can be used to perform either or both of the following tasks:

- Reconstruct the high-dimensional solution of a nonlinear dynamic problem of interest using the computed low-dimensional solution (generalized coordinates) of the reduced-order version of this problem, and the Reduced-Order Basis (ROB) provided through the [READMODE](#) command and specified after the `SROM` sub-command keyword of the [DYNAMICS](#) command. While for nonlinear problems this capability is the only means for reconstructing the high-dimensional solution using computed generalized coordinates of this solution, it is not currently supported when the nonlinear dynamic reduced-order model is constructed in a floating reference frame attached to the center of gravity of the structural system of interest using the [FLOATINGFRAME](#) command.
- Reconstruct also the first and second time-derivatives of this high-dimensional solution using the computed first and second time-derivatives of its low-dimensional counterpart (generalized coordinates), and the ROB provided through the [READMODE](#) command.

The reconstructed high-dimensional solution and its time-derivatives can be outputted in ASCII files using the standard `GDISPLAC`, `GVELOCIT`, and `GACCELER` result keywords of the [OUTPUT](#) and/or [OUTPUT6](#) commands.

Note 1: The proper execution of this command requires the presence in the same ASCII Input Command Data file of the [DYNAMICS](#) command and its `SROM` sub-command keyword because this is where the ROB is specified.

Note 2: **AERO-S** determines the type of data — that is, displacement, velocity, or acceleration — contained in an ASCII input file specified using the sub-command keyword `CONFIL` from the order in which this file is specified. Specifically, the displacement generalized coordinates file should always be inputted first, followed by the associated velocity and acceleration generalized coordinates files, in this order.

Note 3: Inputting the velocity and acceleration generalized coordinates is optional. However, if the corresponding files are specified through additional entries of `CONFIL`, `freq` should be the same for all instances of this sub-command keyword.

Note 4: This command can be executed in parallel by partitioning the mesh associated with the underlying finite element model of interest using the [DECOMPOSE](#) command — or the corresponding command line to request mesh partitioning or provide `aeros` a previously computed mesh partition (see [DECOMPOSE](#)).

The input format of this command is given below.

ROBC

CONFIL	<code><pathandfilenameS></code>	<code>freq</code>
CONFIL	<code><pathandfilenameV></code>	<code>freq</code>
CONFIL	<code><pathandfilenameA></code>	<code>freq</code>

CONFIL

Sub-command keyword for specifying the name of the ASCII file containing the low-dimensional solution (generalized coordinates) of a nonlinear, reduced-order problem constructed by the projection of its high-dimensional counterpart on a ROB. Typically, this file is outputted in a previous **AERO-S** simulation using the result keyword `GDISPLAC` of [OUTPUT](#) or [OUTPUT6](#).

`<pathandfilenameS>`

Path and name of the ASCII file containing the low-dimensional solution (generalized coordinates) of a nonlinear, reduced-order problem constructed by projecting its high-dimensional counterpart of interest on a ROB (characters). Typically, this file is outputted in a previous **AERO-S** simulation using the result keyword `GDISPLAC` of [OUTPUT](#) or [OUTPUT6](#).

`freq`

Frequency at which the reconstruct the dynamic solution (integer). Its default value is 1.

`<pathandfilenameV>`

Path and name of the ASCII file containing the first time-derivative of the low-dimensional solution (generalized coordinates) of a nonlinear, reduced-order problem constructed by projecting its high-dimensional counterpart of interest on a ROB (characters). Typically, this file is outputted in a previous **AERO-S** simulation using the result keyword `GVELOCIT` of [OUTPUT](#) or [OUTPUT6](#).

<pathandfilenameA> Path and name of the ASCII file containing the second time-derivative of the low-dimensional solution (generalized coordinates) of a nonlinear, reduced-order problem constructed by projecting its high-dimensional counterpart of interest on a ROB (characters). Typically, this file is outputted in a previous **AERO-S** simulation using the result keyword `GACCELER` of [OUTPUT](#) or [OUTPUT6](#).

Next: [RESTART](#), Previous: [RODC](#)

107 RENUMBERING

Command Statement: **RENUMBERING**

The **RENUMBERING** command statement is used to specify the type of node renumbering to be performed on the mesh for either or both of the skyline and sparse solvers. The input format is given below.

RENUMBERING

TYPE

TYPE For FEM, the "rcm" and "sloan" schemes are available for the skyline solvers, and the "esmond" (minimal degree ordering) and "metis" (recursive spectral bisection ordering) are available for the (esmond) sparse solver. When a FETI solver is used, two renumbering schemes can be specified if a skyline as well as a sparse solver are to be used by this FETI solver. In this case, the two renumbering schemes can be specified in any order, each on a separate line (string).

Next: [TRBM](#), Previous: [RENUMBERING](#)

108 RESTART

Command Statement: **RESTART**

The **RESTART** command is used to request saving computational data in a **RESTART** file in order to enable later the restart of a dynamics simulation, and/or initializing a dynamics computation using computational results previously saved in a **RESTART** file (restart data). The processes of creating and reading from a **RESTART** file are specified on separate lines using different syntaxes.

Note 1: In addition to the time-instances implied by the specified value of the parameter **INCREMENT**, the restart data is also saved at the end of a simulation if **AERO-S** exits gracefully.

The syntax for invoking this command is given below.

PATHANDFILENAME1 INCREMENT
<**PATHANDFILENAME2>** EXTENSION [FLAG_RST]

INCREMENT	An integer number that specifies at every how many time-integration steps the "rcfem.restart" file will be updated. For an updating restart, this number specifies at every how many iterations the file "restart.upd" will be updated.
PATHANDFILENAME1 INCREMENT	This command, which can be combined with the following one, instructs AERO-S to save/overwrite the restart data into the specified file PATHANDFILENAME1 (string) every INCREMENT (integer) time-steps. An example using RESTART can be found in FEM.d/fem_examples/Restart.d .
PATHANDFILENAME2 EXTENSION	This command, which can be combined with the previous one, instructs AERO-S to read the restart data from the specified file PATHANDFILENAME2 (string) and append the extension EXTENSION (string) (e.g. ".2") to all output filenames specified under the command OUTPUT . An example using RESTART can be found in FEM.d/fem_examples/Restart.d .
FLAG_RST	This optional parameter can be used to specify whether a hot or cold restart is to be performed (characters). A hot restart (which is the default type) is activated by setting FLAG_RST to "old" (case-sensitive and must include the quotation marks): it specifies that the restart is to continue a previously performed simulation. A cold restart is activated by setting FLAG_RST to "new" (case-sensitive and must include the quotation marks): it specifies that the restart is to perform a new simulation using as initial condition the state stored in the restart file created by a previous simulation. Cold restart of AERO-S is required when performing a coupled fluid-structure simulation using the tandem AERO-F-AERO-S and AERO-F is also restarting in cold mode - that is, not continuing a previous simulation - (for example, if AERO-F is starting an unsteady flow simulation in which the initial condition is specified to be the outcome of a previous steady-state flow computation).

Next: [RBMFILTER](#), Previous: [RESTART](#)

109 RIGID BODY (AND OTHER ZERO ENERGY) MODES

Command Statement:	TRBM
--------------------	-------------

The **TRBM** command is used to specify a tolerance for monitoring small pivots during the factorization of a matrix. "Small" pivots are deemed to be zero pivots. Therefore, they are associated with rigid body modes (or zero energy modes in general). When this command is specified in the **AERO-S** input file and the **sparse**, **skyline**, **blockskyline**, or **mumps** pivot solver is specified under the **STATICS** command, the generalized inverse of the stiffness, conductivity, mass (when an initial acceleration is to be computed to satisfy the governing equation (see **DYNAMICS**)), or other relevant matrix is computed in factored form by eliminating the equation associated with a deemed zero pivot and setting the corresponding unknown to zero. This command can be used together with the **GRBM** command (see below).

Note 1: See **GRBM** for an alternative option for analyzing singular systems.

Note 2: This command can be used together with the **GRBM** command. If both of the **GRBM** and **TRBM** commands are specified in the same **AERO-S** input file, then:

- If the equation solver specified in **STATICS** is the **sparse** or **skyline** solver, and the analysis to be performed is a *linear* static, quasistatic, or eigen (structural) analysis, then **GRBM** is used to:
 - Determine the rigid body — or more generally, the zero energy — modes of the prevailing stiffness matrix that are due to a lack of sufficient Dirichlet boundary conditions or **LMPCs** to guarantee the invertibility of this matrix.
 - Assist the direct equation solvers **sparse** and **skyline** (see **STATICS**) in solving the admissible system of linear equations governed by this singular matrix. For an **EIGEN** analysis, this assistance is performed however only when the shift is zero. Therefore the **TRBM** command is ignored in this case.
- Otherwise:
 - **GRBM** is used to determine, *for the purpose of information and only for this purpose*, the rigid body — or more generally, the zero energy — modes of the prevailing stiffness matrix that are due to a lack of sufficient Dirichlet boundary conditions or **LMPCs** to guarantee the invertibility of this matrix.
 - **TRBM** is used to assist the direct equation solver specified in **STATICS** in solving the admissible system of linear equations governed by this singular matrix.

TRBM

VALUE

VALUE Tolerance for monitoring the zero pivots of a matrix during its factorization. The default value is 1.0e-16 except for the MUMPS solver for which the default is $\epsilon \times 10^{-5}$, where ϵ is the machine precision (real).

Next: [RUBDAFT](#) Previous: [TRBM](#)

110 RIGID BODY MODES FILTER

Command Statement:	RBMFILTER	[LEVEL]	[QMATRIX]	[TOLERANCE]
--------------------	------------------	----------------	------------------	--------------------

This command is suitable for structural, aeroelastic, and aerothermoelastic simulations involving an *unrestrained* (or *partially restrained*) structural system, when any of the following reasons applies:

- The context is that of a *linear* static or quasistatic analysis, and the total external load is not self-equilibrated.
- The context is that of a dynamic analysis, and it is desired to artificially eliminate some or all of the rigid body modes of the system.

In either context outlined above, if this command is specified in the ASCII Input Command Data file, **AERO-S** automatically determines the rigid body modes of the modeled system using the **GRBM** method, constructs a projector $P = I - R(R^T Q R)^{-1} R^T Q$, where I is the

identity matrix, Q is a specified symmetric positive definite matrix (see below), and R is the matrix storing the computed rigid body modes, and uses this projector as described below.

- In a linear static or quasistatic analysis, **AERO-S**:
 - Sets $Q = I$ (in this case, P is symmetric and therefore $P^T = P$) or $Q = M$ (in this case $P^T \neq P$), where M is the mass matrix, according to the user's request (see below). In the latter case, if M is singular, $(R^T M R)$ will be singular; in this case,

AERO-S computes a generalized inverse of this matrix and therefore requires that the user specifies an absolute tolerance (**TOLERANCE**) for detecting zero pivots during the factorization of this matrix.

- Applies P^T to the total external load to self-equilibrate it (even if this load happens to be already self-equilibrated) so that the singular system of equations governing the linear structural analysis problem is solvable. For aeroelastic and aerothermoelastic simulations involving an unrestrained (or partially restrained) structural system, this step is applied at each iteration. It corresponds to an artificial *trimming* of the system that is required for ensuring the solvability of each linear system of equations that arises.

In this case, **AERO-S** computes a particular displacement solution that satisfies the self-equilibrated equations and is associated with a minimally restrained configuration of the system of interest that depends on the equation solver specified in **STATICS** (in principle, any equation solver equipped for handling singular systems introduces in this case the minimum number of non-unique artificial restraints that removes all singularities of the system associated with rigid body modes). Furthermore, this particular solution

depends on the choice of the projector P^T — and therefore on the choice of Q . In other words, it depends on the selected artificial trimming method implied by the application of the projector P^T to the total external load.

- In a linear or nonlinear dynamic analysis, **AERO-S**:

- Sets $Q = M$, where M is the mass matrix. In this case, $P^T \neq P$. If due to modeling assumptions M is singular, $(R^T MR)$ will

be singular: in this case, **AERO-S** computes a generalized inverse of this matrix and therefore requires that the user specifies an absolute tolerance (**TOLERANCE**) for detecting zero pivots during the factorization of this matrix.

- Applies P to the initial solution (initial displacement *and* velocity fields) to *filter out* a specified *subset* of the rigid body modes stored in R .

- Applies, at each time-step, P^T to the (time-dependent) total external load to *filter out* the specified subset of the rigid body modes stored in R from the governing equations of dynamic equilibrium — that is, to compute a dynamic solution that is M -orthogonal to the specified subset of columns of R .

In this case, at each time-step, **AERO-S** *filters out* the specified *subset* of the rigid body modes from the solution it computes before it outputs it. In the case of a dynamic aeroelastic or aerothermoelastic analysis, it performs this filtering before it exchanges the solution it computes with the flow solver **AERO-S**. Such a filtering amounts to an artificial self-equilibration (or trimming in the case of an aeroelastic or aerothermoelastic analysis) of the system in the translational and rotational axes represented by the filtered rigid body modes.

An example using this command can be found in [APPENDIX 7](#).

Note 1: This command is not active when a modal dynamic analysis is performed (see [DYNAMICS](#)). The reason is that one can achieve the same objective by simply not including the rigid body modes in the input for the [READMODE](#) command. It is also inactive in an [EIGEN](#) or [IMPEDANCE](#) analysis as there is no motivation for it in both cases.

The input format of this command is given below.

RBMFILTER	[LEVEL]	[QMATRIX]				
LISRBM	id1	id2	id3	id4	id5	id6

LEVEL	Omitting the specification of this integer is equivalent to specifying the value of 1. This is the default value. In this case, AERO-S uses the projector as described above.
2	This value is applicable only for linear quasistatic and dynamic analyses. In this case, AERO-S also applies the projector P at each iteration or time-step to the computed displacement and velocity fields in order to explicitly Q -orthogonalize them against the specified subset of rigid body modes. This is a safety measure that can be useful, for example, when ill-conditioning prevents the default case from functioning properly.
QMATRIX	This integer information is relevant for a linear static or quasistatic analysis only. It allows the user to choose between two different settings for the matrix Q . Omitting this information is equivalent to choosing the value of 0.
0	This is the default value. In this case, Q is set to the mass matrix.
1	In this case, Q is set to the identity matrix.
TOLERANCE	Optional tolerance for the case where QMATRIX is set to 0 (mass matrix M) (real). This tolerance is used by AERO-S to identify zero pivots during the factorization of the matrix $(R^T MR)$, when M is singular due to modeling assumptions. Its default value is 0 (which means that by default, AERO-S assumes that M is invertible).
LISRBM	Sub-command keyword to specify a list of rigid body modes to be filtered out (characters). It is relevant only in the case of a dynamic analysis. The list consists of integers ranging between 1 and the number of rigid body modes N computed by the GRBM method and stored in R , each identifying a rigid body mode by its column number in the matrix R . To this effect, the reader is reminded that R stores first the translational rigid body modes in the x , y and z directions, then the rotational ones in the x , y , and z directions. The integer identifiers can be listed in any order. A specified identifier that is greater than N is simply ignored.
id1 (... id6)	ID number ranging between 1 and the number N of rigid body modes computed by the GRBM method and stored in the matrix of rigid body modes R . This integer identifies the column number of R where a rigid body mode to be filtered out from the computed solution is stored: if it is greater than N , it is ignored.

Next: [MODE](#), Previous: [RBMFILTER](#)

111 RUBBER DAMPING PROPERTIES TABLE

Command Statement:	RUBDAFT
--------------------	----------------

The command **RUBDAFT** should be used to describe the variations of Young's modulus E , its loss factor η_E , the shear modulus μ , and its loss factor η_μ with the frequency f (see the sub-command keyword **RUBDAMP** in [MATERIAL](#)) anytime anyone of these four parameters is frequency dependent. These evolutions can be specified here in one or multiple tables defined for one or multiple materials by quintuplets of f , E , η_E , μ , and η_μ . In each table, linear interpolation is used for the "in between" points, and constant extrapolation (using the parameter values at the closest frequency) is adopted for the "outside" points. When multiple tables are defined, they are inputted one after the other, and each is identified by an id number as described below.

RUBDAFT

TABLE table_id				
f_1	E_1	η_{E_1}	μ_1	η_{μ_1}
.
.
f_n	E_n	η_{E_n}	μ_n	η_{μ_n}

TABLE Sub-command keyword (characters) for defining a table describing the variation of Young's modulus E , its loss factor η_E , the shear modulus μ , and its loss factor η_μ with the frequency f in the form of quintuplets $(f_i, E, \eta_E, \mu, \eta_\mu)$.

table_id Id number for the defined table (integer).

f_i A sampled frequency value (real).

E_i Value of Young's modulus at the frequency f_i (real).

η_{E_i} Value of Young's modulus loss factor at the frequency f_i (real).

μ_i Value of the shear modulus at the frequency f_i (real).

η_{μ_i} Value of the shear modulus loss factor at the frequency f_i (real).

Next: [SENSITIVITY](#), Previous: [RUBDAFT](#)

112 SAVING EIGENMODES OR PROJECTING ONTO A BASIS

Command Statement: **MODE**

The **MODE** command can be used to signal that:

- When the [EIGEN](#) command is used, the eigen solutions are to be saved in a binary file named EIGENMODES (or binary files EIGENMODES# when using **AERO-S** in distributed memory mode with one file per MPI process) and located in the execution path of **AERO-S**.
- When the [DYNAMICS](#) command is used, the computation of the projections of the transient solution onto a modal or other basis can be requested in the [OUTPUT](#) command. In this case, this basis can be inputted either in the binary file named EIGENMODES (in which case it is a modal basis), or in an ASCII or binary file inputted via [READMODE](#). Specifically, **AERO-S** determines which basis to read and use for this purpose as follows:
 - If the **rob_id** parameter of this command is omitted, the chosen basis is the modal basis inputted in the file named EIGENMODES.
 - If **rob_id** is specified to a non-zero value, the command [READMODE](#) must be also specified in the ASCII Input Command Data file using its second format. In this case, the chosen basis is that identified in [READMODE](#) using the same value for **rob_id**.
 - If **rob_id** is specified as 0, the command [READMODE](#) must be also specified in the ASCII Input Command Data file using its first format. In this case, the chosen basis is that inputted via [READMODE](#).
- When [IMPEDANCE](#) is used with **FREQSWEEP**, **PADEPOLES** and **alg = PadeLanczos**, the binary file named EIGENMODES (or binary files EIGENMODES# when using **AERO-S** in distributed memory mode with one file per MPI process) located in the execution path of **AERO-S** is to be read, and the eigen modes stored in this (or these) file(s) are to be exploited to identify within a range of interest the eigenvalues missed by a previous eigen computation and their corresponding eigenvectors.

Note 1: Currently, the use of this command for the second purpose identified above is supported only for linear dynamic analyses, and in the absence of a mesh decomposition.

Note 2: The EIGENMODES file is written using the "internal" numbering of **AERO-S** for the degrees of freedom. This internal numbering depends on the equation solver and renumbering scheme that were specified under the [STATICS](#) and [RENUMBERING](#) commands, respectively, when the eigen solutions were computed. Hence, when reusing the EIGENMODES file for computing the projections of the transient solution of a problem onto the vectors contained in this file, the same equation solver and renumbering scheme must be specified under the [STATICS](#) and [RENUMBERING](#) commands, respectively.

[MODE [rob_id]]

rob_id

Integer number identifying the basis inputted via [READMODE](#) (integer). See above for a detailed description of the use of this parameter.

Next: [SENSORS](#), Previous: [MODE](#)

113 SENSITIVITY ANALYSIS (SENSITIVITY completely spelled out)

Command Statement: **SENSITIVITY**

The **SENSITIVITY** command statement is used to request the formulation and solution of a structural or aeroelastic sensitivity analysis problem. It can also be used to request that **AERO-S** participates in the gradient-based solution of a structural or aeroelastic optimization problem by providing the needed sensitivities.

[SENSITIVITY]

THGRLI	int1 int2 ... intN
READSE	pathandfilename
TOLSEN	tolsen

THGRLI

int1 int2 ... intN Sequence of integers inputted on the same line with blank spaces in between that specifies the list of element [GROUPS](#) whose thicknesses are to be declared as sensitivity parameters.

READSE This subcommand is relevant only when performing a sensitivity analysis with respect to shape variables.

pathandfilename Name of the ASCII file containing shape sensitivities to be read (characters). Specifically, this file contains $\frac{d\mathbf{x}}{ds_j}$

, the derivatives of the structural mesh position \mathbf{x} with respect to a number of shape design variables s_j .

The file starts with an **XPost**-like header (see below), followed by the total number of nodes in the structural model. Then, the information $\frac{d\mathbf{x}}{ds_j}$ is specified in this file for each shape design parameter s_j , one parameter

at a time, in block form. First, the index j of s_j is specified on a separate line starting from $j = 0$ (zero).

Then, all nodes of the finite element model are considered in the same ordering as that adopted in the command [NODES](#). On each line corresponding to node i , the derivatives $\frac{dx_i}{ds_j}$, $\frac{dy_i}{ds_j}$, and $\frac{dz_i}{ds_j}$ (where x_i , y_i ,

and z_i denote the coordinates of the node i) are provided. An example of this ASCII file is given below.

```
Vector <file name> under load for StructureNodes
<vector size = total number of nodes in the finite element structural model>
0^M
.
.
.

```

```
dx_i/ds_0 dy_i/ds_0 dz_i/ds_0
```

```
.
.
.
```

```
1
```

```
.
.
.
```

```
dx_i/ds_1 dy_i/ds_1 dz_i/ds_1
```

```
.
.
.
```

```
2
```

```
.
.
.
```

```
dx_i/ds_2 dy_i/ds_2 dz_i/ds_2
```

```
.
.
.
```

TOLSEN

tolsen

Convergence tolerance used only when the **SENSITIVITY** command is used to request that **AERO-S** participates in the gradient-based solution of an aeroelastic optimization problem. The default value is 1.0e-5.

Next: [SLOSH](#), Previous: [SENSITIVITY](#)

114 SENSORS *S*

Command Statement: **SENSORS**

The **SENSORS** command statement is used to specify to **AERO-S** the degrees of freedom to be observed and whose structural state is to be passed to the user defined control subroutine "control.C" (see [ACTUATORS](#)). An example input file using the **SENSORS** command can be found in FEM.d/fem_examples/Control.d

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

SENSORS

NODE# DOF#

NODE# Node number where the sensor is to be placed (integer).
DOF# Degree of freedom local number where the sensor is to be placed (integer).

Next: [SZEM](#), Previous: [SENSORS](#)

115 SLOSHING PROBLEMS

Command Statement: **SLOSH**

The command **SLOSH** is used to specify the computation of the sloshing modes of an incompressible and inviscid fluid whose free surface is orthogonal to a specified gravity field. The computational approach assumes linear kinematics and is based on a fluid displacement potential. In three dimensions, it does not involve any fluid material property. In two dimensions, it requires inputting the thickness of the considered slice of the problem. In both cases, the **SLOSH** analysis associated with this command requires identifying a free surface using free-surface elements (element type 312 in three dimensions and element type 302 in two dimensions, see [TOPOLOGY](#)) but does not require any other boundary condition.

Hence, this command should be used in conjunction with the commands [EIGEN](#) and [STATICS](#) in order to solve the arising eigenvalue problem.

SLOSH

SLGRAV slgrav

slgrav Magnitude (scalar) of the gravitational acceleration whose direction must be orthogonal to the free surface of the problem. **AERO-S** uses this specified value to convert the sloshing eigenvalues to sloshing frequencies (number of cycles per second) for output (real).

Next: [HFTT](#), Previous: [SLOSH](#)

116 SLOSHING ZERO ENERGY MODE

Command Statement: **SZEM**

This command is effective only for a sloshing eigen computation involving a zero energy mode. It is used to request the computation of the constant potential mode using a physics-based algorithm (rather than the tolerance-based algorithm associated with the **TRBM** command). It should be used in conjunction with the **SLOSH** and related commands.

SZEM

Next: [SCTT](#), Previous: [SZEM](#)

117 SOURCE TIME TABLE-HEAT CONDUCTION

Command Statement: **HFTT[TABLE_ID]**

The **HFTT** command statement must be used to specify the time-dependent amplification of the boundary fluxes. For FEM, pairs of time and

amplification values are input. Linear interpolation is also used for "in between" points.

HFTT [TABLE_ID]

TIME_1	AMP_1
.	.
.	.
.	.
TIME_n	AMP_n

TABLE_ID

TIME_1
AMP_1

Optional non-negative integer which uniquely identifies a source-time table so that it can be associated with a "load" set to define the "load" case for a dynamic analysis using the **LOADCASE** command. The default value is 0. Hence, the **HFTT** command can be repeated as many times as desired within the same input file using each time a different value for TABLE_ID and different data.

A specified time point (float).

A specified amplification value at time point TIME_1 (float). This amplification factor is automatically set to zero for all times prior to the earliest specified time point and all times later than the latest specified time point.

Next: [STATICS](#), Previous: [HFTT](#)

118 SPECIFIC HEAT COEFFICIENT-TEMPERATURE TABLE

Command Statement: SCTT

The **SCTT** command can be used to describe, for a given material, the evolution of the specific heat coefficient with temperature. This evolution can be specified here in a curve (or one-dimensional table) defined by pairs of specific heat coefficient and temperature values. Linear interpolation is used for "in between" points, and the extrema values are adopted for "outside" points. Several curves can be specified, one after the other. Each curve is identified by an ID number as described below.

Note 1: Currently, this command is supported only by the 4-noded tetrahedral thermal element (type 50).

SCTT

CURVE curve_id
T_1 cp_1
.
.
.
T_n cp_n

CURVE Sub-command keyword (characters) that signals the definition of a specific heat coefficient-temperature curve (or one-dimensional table).

curve_id ID number for the following curve (or one-dimensional table) (integer).

T_i A specified temperature value (real).

YM_i A specified specific heat coefficient value at temperature T_i (real).

Next: [SS1DT](#), Previous: [SCTT](#)

119 STATICS

Command Statement: STATICS

The **STATICS** command statement can be used mainly for three independent purposes: (a) select a static analysis, (b) and/or select an equation solver needed for a static analysis or any other type of analysis specified in an additional command such as **DYNAMICS**, **EIGEN**, or other, (c) and/or set parameters that are relevant to many other analyses besides static analysis, such as, for example, a load case among those defined in the command **LOADCASE**.

For linear static analysis, this command can also be used for requesting the interpretation of concentrated follower (see **FORCES**) and pressure-induced (see **PRESSURE** and **CONWEP**) forces and moments as configuration-dependent external forces and moments and applying to them a piecewise constant treatment, thereby leading to a quasi-static analysis.

The input format of this command is given below.

Note 1: Except when otherwise specified, the solvers proposed below are available only for symmetrical systems of equations.

Note 2: Among all solvers proposed below, the following ones are suitable for the solution of singular (but consistent) systems: `skyline`, `blockskyline`, `sparse`, `mumps pivot`, `superlu`, and `FETI DP`. Among these solvers, only `sparse`, `skyline`, `mumps pivot`, and `FETI DP` compute and return for further usage the null space of the singular matrix. Furthermore, only the `sparse` and `skyline` solvers can benefit for this purpose from the assistance of the `GRBM` command: all other aforementioned solvers benefit from the assistance of the `TRBM` command. Note that singular systems arise: (1) if the problem is formulated without sufficient Dirichlet boundary conditions, or (2) the formulated problem contains redundant constraints and the Lagrange multiplier method is chosen for enforcing these constraints (see [CONSTRAINTS](#)).

Note 3: Among all solvers proposed below, the following ones are suitable for the solution of indefinite systems: `spooles` and `mumps` with pivoting turned on, `superlu`, `gmres`, and `FETI DP`. Indefinite systems arise if: (1) the analysis involves the `HELMHOLTZ`, `IMPEDANCE`, or `EIGEN` command with a positive shift (see `SHIFT` in [EIGEN](#)) or for buckling analysis (see `ARPACK` in [EIGEN](#)), or (2) the structural model includes rigid and/or joint elements (see [TOPOLOGY](#)), linear multi-point constraints (see [LMPC](#)), or tied surfaces (see [TIEDSURFACES](#)), and the Lagrange multiplier method is chosen for enforcing the associated *equality* constraints (see [CONSTRAINTS](#)) in a static, eigen, or implicit dynamic computation.

Note 4: Among all solvers proposed below, only `FETI DP` is suitable for the solution of static or implicit dynamic contact problems if the Lagrange multiplier method is chosen for enforcing the associated *inequality* constraints.

Note 5: Among all FETI methods, only `FETI DP` is maintained for nonlinear structural static and dynamic analyses.

Note 6: When a non deterministic analysis is performed using the intrusive version of the Polynomial Chaos method, only the `pcg`, `bcg`, and `cr` solvers can be selected for solving the resulting large system of equations.

Note 7: In all aforementioned cases, the following direct solvers can be executed in parallel on a shared memory system using OpenMP, in the context of a single domain, that of a coarse solver for a FETI-type method, or that of multiple subdomains (see [DECOMPOSE](#)):

<code>spooles</code>	Can be executed in parallel on a shared memory system using OpenMP, in both contexts of a single domain and a coarse solver for a FETI-type method.
<code>skyline</code>	Can be executed in parallel on a shared memory system using OpenMP, in both contexts of a single domain and a coarse solver for a FETI-type method.
<code>blockskyline</code>	Can be executed in parallel on a shared memory system using OpenMP, in both contexts of a single domain and a coarse solver for a FETI-type method.

Note 8: In all aforementioned cases, the following direct solvers can be executed in parallel on a distributed memory system using MPI, in the context of a single domain, that of a coarse solver for a FETI-type method, or that of multiple subdomains (see [DECOMPOSE](#)):

<code>mumps</code>	Can be executed in parallel on a distributed system using MPI, in all contexts of a single domain, a coarse solver for a FETI-type method, and multiple subdomains.
--------------------	---

Note 9: In all aforementioned cases, the following iterative solvers can be executed in parallel on a distributed memory system using MPI or MPI and OpenMP, or on a shared memory system using OpenMP in the context of multiple subdomains (see [DECOMPOSE](#)):

All FETI-type solvers	Can be executed in parallel on a shared memory system using OpenMP, a distributed memory system using MPI, and a hybrid system using OpenMP on its shared memory subsystem and MPI across its subsystems. In the case of a distributed or hybrid memory system, and except when <code>mumps</code> is chosen as the coarse problem solver, the coarse problem is duplicated on each MPI process: for this reason, if <code>mumps</code> is not chosen as the coarse problem solver, a single MPI process should be created in general within each computational node in order to minimize the memory penalty associated with this parallel implementation of a FETI-type solver. In the specific case of a hybrid system, if <code>mumps</code> is not chosen as the coarse problem solver, the coarse problem is solved in parallel using the OpenMP threads forked within an MPI process: for this reason, if <code>mumps</code> is not chosen as the coarse problem solver, the maximum speedup factor that can be expected from the parallel solution of the coarse problem is in this case equal to the number of OpenMP threads forked within each MPI process, and a single MPI process should be created within each computational node in order to be able to assign all computational units within this node to the <i>effective</i> parallel solution of the coarse problem. On the other hand, if <code>mumps</code> is chosen as the coarse problem solver, whether the computations are performed on a parallel distributed or hybrid memory system, the coarse problem is solved using all MPI processes without storage duplication. However, by default, the coarse problem is stored on a single MPI process, which is memory inefficient: to distribute this problem across all MPI processes, the user should set <code>ICNTL(18) = 3</code> (see <code>mumps_icntl</code> below).
-----------------------	--

Note 10: The solvers `eisgal` and `dbsgal` can be used only for reduced-order models (ROMs). **AERO-S** detects such models by the presence of the [READMODE](#) command in its ASCII Input Command Data file.

STATICS

METHOD
PARAMETERS
PIECEWISE real_1 real_2
CASES case_id_1 case_id_2 ... case_id_n

METHOD	Keyword (characters) specifying one of the solvers named below.
solverhandle solver_id	For any of the equation solvers listed below, if configured in the command SOLVERCNTL using for SOLVER_ID the same integer specified here in solver_id — that is, if this solver and its parameters and identification are specified in SOLVERCNTL (characters, integer). Note that while any of the equation solvers listed below can be configured in SOLVERCNTL , a multi-level FETI-DP solver must be configured in SOLVERCNTL and specified here as a solverhandle solver_id .
direct	For a skyline direct solver (characters). This is also the default solver (except when the INPC command is used) and is available for frequency-domain acoustic (Helmholtz) problems.
skyline	For a skyline direct solver (characters). This is also the default solver and is available for frequency-domain acoustic (Helmholtz) problems.a
blockskyline	For a skyline symmetric positive direct solver with loop unrolling for optimal performance (characters).
sparse	For Esmond's sparse direct solver (characters). This solver is also available for frequency-domain acoustic (Helmholtz) problems.
sgisparse (completely spelled out)	For SGI's sparse direct solver (characters). Runs only on SGI machines.
spoole (completely spelled out)	For the SPOOLES sparse symmetric positive definite direct solver (characters). Cannot be used for: (a) a singular system, and (b) a system with less than 24 equations and unknowns.
spoole pivot (completely spelled out)	For the SPOOLES sparse general symmetric direct solver with the pivoting option turned on (characters).
mumps (completely spelled out)	For the MUMPS sparse symmetric positive definite or semi-definite direct solver (characters). This equation solver is not supported by the data organization of BINARY .
mumps pivot (completely spelled out)	For the MUMPS sparse general symmetric direct solver with the pivoting option turned on (characters). This equation solver is not supported by the data organization of BINARY .
mumps unsymmetric (completely spelled out)	For the MUMPS unsymmetric sparse direct solver (characters). This equation solver is not supported by the data organization of BINARY .
superlu (completely spelled out)	For the SUPERLU unsymmetric sparse direct solver (characters).
sgisky (completely spelled out)	For SGI's skyline direct solver (characters). Runs only on SGI machines.
frontal (completely spelled out)	For a frontal direct solver (characters).
eisgal	For EISGAL, the dense, linear equation solver of the <code>Eigen3</code> library. It is applicable to arbitrary dense systems of equations arising from a projection-based Reduced-Order Model (ROM). It is also the default equation solver for such computational models.
dbsgal	For DBSGAL, the dense, symmetric, linear equation solver of the <code>LAPACK</code> library. It is applicable to symmetric, dense systems of equations arising from a projection-based Reduced-Order Model (ROM).
pcg	For a preconditioned conjugate gradient solver (characters).
bcd	For a bi-conjugate gradient solver (characters).
cr	For a conjugate residual solver (characters).
gmres	For a generalized minimum residual solver (characters). Applicable to symmetric and unsymmetric systems.
FETI	For a one-level FETI solver (default) (characters).
FETI 1	For a FETI-1 (one-level FETI method) solver (characters).
FETI 2 OLD	For a FETI-2 solver using a “full” coarse problem implementation (characters).
FETI 2 NEW	For a FETI-2 solver using a “sparse” coarse problem implementation (characters).
FETI DP	For a FETI-DP (FETI dual-primal method) solver (characters).
FETI DPH	For a FETI-DPH (FETI dual-primal method with augmentation basis) solver (characters). This solver is to be used only with the IMPEDANCE command. Depending on whether the system to be solved is damped or not, or contains a complex boundary condition or multipoint constraint or not, this solver performs in the real or complex domain.
PARAMETERS	(keywords or keywords and values) for pcg/bcg/cr/gmres solvers, in any order.
precno	Specifies the preconditioner (characters).
0	Unpreconditioned (integer). This is the default value.
1	Diagonal scaling (integer). This option should not be used when solving an indefinite system arising in conjunction with the usage of the LMPC or TIEDSURFACES command, or whenever a structural model includes rigid and/or joint elements (see TOPOLOGY), because in all these cases, the system matrix will contain zero diagonal entries.
2	Incomplete block diagonal scaling (integer). This option is available only for a system of equations resulting from the intrusive version of the Polynomial Chaos method. In this case, the subsystems of equations associated with the diagonal blocks can be solved either using the sparse solver (default), or the FETI DP method with its acceleration for multiple right sides. If the FETI DP method is chosen as a block diagonal solver, then its keyword and the keywords associated with its options should be specified after all other parameters of the main (outer) solver have been specified.
tolpcg	Error tolerance for the convergence of the pcg solver (real). The default value is 1e-8.
tolbcg	Error tolerance for the convergence of the bcg solver (real). The default value is 1e-8.
tolcr	Error tolerance for the convergence of the cr solver (real). The default value is 1e-8.
tolgmres	Error tolerance for the convergence of the gmres solver (real). The default value is 1e-8.

maxitr	Maximum number of iterations to be performed (integer). The default value is 1000.
PARAMETERS	(keywords or keywords and values) for spooles solver, in any order.
spooles_renum	Specifies the renumbering (integer, default value is 0).
0	Best of nested dissection and multisection.
1	Multiple minimum degree.
2	Multisection.
3	Nested dissection.
spooles_scale	Specifies the scaling (integer, default value is 0).
0	No scaling.
1	Symmetric scaling.
spooles_tau	Upper bound on the magnitude of the largest element in L or U when pivoting enabled (real 1.0). If this number is too small the results will be wrong. The default value of 100 is generally safe if scaling is used (see spooles_scale).
spooles_msglvl	Message output level (integer ≥ 0 , default value is 0).
spooles_maxdomainsize	n /spooles_maxdomainsize is the maximum subgraph size used by SPOOLES orderings, where n is the number of equations in the system to be solved. This parameter is used to control the incomplete nested dissection process. Any subgraph whose weight is less than maxdomainsize is not split further (integer ≥ 0 , default value is 24).
spooles_maxzeros	$n * \text{spooles_maxzeros}$ is the maximum number of zeros allowed in a supernode/front (real > 0 and ≤ 1.0 , default value is 0.04).
spooles_maxsize	Maximum number of internal columns in supernode/front (integer > 0 , default value is 64).
PARAMETERS	Keywords and values for the solver mumps , in any order. Note that:
	<ul style="list-style-type: none"> When specific values are assigned to the mumps parameters discussed below under this command (STATICS) – to override their default values – AERO-S interprets these parameters as those of a global equation solver: hence, AERO-S ignores them if mumps is not specified as the global equation solver. If mumps is to be used as the coarse solver of a FETI-type method and specific values are to be assigned to the parameters discussed below, COARSE_SOLVER (see below) should be defined as a solverhandle identified by solver_id and the solver mumps and its parameters discussed below should be specified using SOLVERCNTL (see above).
MUMPS_ICNTL index integer_value	Sub-command keyword and corresponding pair of integers which can be used to set the integer-valued mumps parameters stored in its ICNTL array, as follows: MUMPS_ICNTL index integer_value (for ICNTL[index] = integer_value , see the MUMPS documentation for further details). This keyword and corresponding pair of integers can be repeated as often as desired to set different parameters of the array ICNTL. For example, setting ICNTL[18] = 3 requests that the sparse matrix governing the system of equations to be solved be distributed across all MPI processes, while the default setting ICNTL[18] = 0 has this matrix stored entirely on a single MPI process.
MUMPS_CNTL index real_value	Sub-command keyword and corresponding pair of integer index and real value which can be used to set the real-valued mumps parameters stored in its CNTL array, as follows: MUMPS_CNTL index integer_value (for CNTL[index] = real_value , see the MUMPS documentation for further details). This keyword and corresponding pair of integer index and real value can be repeated as often as desired to set different parameters of the array CNTL.
MUMPS_MINEQ integer_value	This parameter is relevant only when the solver mumps is run in distributed mode. It specifies the minimum number of equations to be allocated to each mumps MPI process (integer), which itself determines how many of the mumps MPI processes requested within the mpirun command are to be effectively allocated to mumps : this number is equal to the floor of the ratio of the size of the system of equations to be solved by mumps and integer_value . Hence, this parameter is particularly important for parallel speedup when mumps is used within a FETI-DP solver.
PARAMETERS	(keywords or keywords and values) for FETI, FETI-DP, FETI-DPH solvers, in any order.
spacedim	This parameter is needed only for the FETI-DPH method with augmentation. It should be set to 2 for two-dimensional problems, and to 3 for three-dimensional ones (integer).
sparse	Specifies Esmond's sparse direct method as the local (subdomain and Dirichlet preconditioner) solver (characters). The default is a skyline solver which can also be invoked by specifying skyline instead of sparse on this line. Note also that there is another mechanism for specifying this option (see below).
local_solver solverhandle solver_id	This choice can represent any of the equation solvers listed below, if configured in the command SOLVERCNTL using for SOLVER_ID the same integer specified here in solver_id — that is, if this solver and its parameters and identification are specified in SOLVERCNTL (characters, integer).
skyline	Selects the skyline direct method as the local solver (subdomain and Dirichlet preconditioner problems) (characters). This is also the default choice.
sparse	Selects Esmond's sparse direct method as the local solver (characters).
spooles	Selects the SPOOLES sparse direct method as the local solver (characters).
spooles pivot	Selects the SPOOLES sparse direct method as the local solver with pivoting option turned on (characters).
mumps	Selects the MUMPS sparse direct method as the local solver (characters).

mumps pivot	Selects the MUMPS sparse direct method as the local solver with pivoting option turned on (characters).
coarse_solver solverhandle solver_id	This choice can represent any of the equation solvers listed below, if configured in the command SOLVERCNTL using for SOLVER_ID the same integer specified here in solver_id — that is, if this solver and its parameters and identification are specified in SOLVERCNTL (characters, integer). It can also represent any FETI-DP solver, including a multi-level FETI-DP solver.
blockskyline	Selects the block-skyline direct method as the coarse solver. This is also the default choice (characters).
skyline	Selects the skyline direct method as the coarse solver (characters).
sparse	Selects Esmond's (sequential) sparse direct method as the coarse solver (characters).
spooles	Selects the SPOOLES sparse direct method as the coarse solver (characters).
spooles pivot	Selects the SPOOLES sparse direct method as the coarse solver with pivoting option turned on (characters).
mumps	Selects the MUMPS sparse direct method as the coarse solver (characters).
mumps pivot	Selects the MUMPS sparse direct method as the coarse solver with pivoting option turned on (characters).
precno 0 or nprec 1 or lumped 2 or dirichlet	Specifies the local preconditioner (integer or characters). Unpreconditioned (integer). This is the default value. Lumped preconditioner (integer). Dirichlet preconditioner (integer). This is also the default preconditioner.
projector 1 2 4	Identity based projector (integer). This is also the default projector. Preconditioner based projector (integer). Uses the selected preconditioner for building the so-called Q matrix. Superlumped projector (integer).
scaling 1 or stiffness 2 or topology	Stiffness based scaling (integer). Topology (subdomain connectivity) based scaling (integer). This is the default scaling procedure.
version 1 2	The FETI (also known as FETI-1) method (integer). The two-level FETI (also known as FETI-2) method (integer).
nocoarse	This option is only for dynamics. When specified, the FETI algorithm is executed without any "coarse grid".
corners	This keyword is useful only for the FETI-2, FETI-DP, and FETI-DPH methods. It specifies the treatments of the corners and corner dofs for the construction by these iterative methods of their respective coarse problems. For the FETI-2 method, the user can define both the corner and corner dof selection algorithms. For the FETI-DP and FETI-DPH methods, the corners are automatically chosen by AERO-S but the user can still control the number of dofs at these corners.
cp3	In that case, a corner is defined by FETI-2 as a crosspoint, and only the three active translational dof attached at each corner node are included in the construction of the FETI-2 or FETI-DP and FETI-DPH coarse problems. This is the default value for nodes with 3 dofs. Here, a crosspoint is defined as a point that belongs to more than four subdomains.
cp6	In that case, a corner is defined by FETI-2 as a crosspoint, and all six active dof attached at each corner node are included in the construction of the FETI-2 or FETI-DP and FETI-DPH coarse problems. This is the default value for nodes with 6 dofs. Here, a crosspoint is defined as a point that belongs to more than two subdomains.
be3	In that case, a corner is defined by FETI-2 as the beginning or end of an edge, and only the three active translational dof attached at each corner node are included in the construction of the FETI-2 or FETI-DP and FETI-DPH coarse problems. Note that a crosspoint is also the beginning or end of an edge.
be6	In that case, a corner is defined by FETI-2 either as the beginning or end of an edge, and all six active dof attached at each corner node are included in the construction of the FETI-2 or FETI-DP and FETI-DPH coarse problems. Note that a crosspoint is also the beginning or end of an edge.
fsi_corners	This keyword is useful only for the FETI-DPH method when applied to the solution of a fluid-structure interaction problem (see FSINTERFACE and HWIB) in the frequency domain. It manages the selection of additional, non_essential corner nodes except for numerical scalability. Note that if the decomposition is performed using DECOMPOSE and a separate run of AERO-S , the setting of this parameter must be the same when later running AERO-S using the generated mesh partition to solve a fluid-structure interaction problem using FETI-DPH.
0	In that case, no node of the fluid/structure interface is chosen as a corner node.
1	In that case, every fluid node at the intersection of a subdomain boundary interface and the fluid/structure interface is chosen as an additional corner node.
2	In that case, every node — whether it is a structure or fluid node — on the intersection of a subdomain boundary interface and the fluid/structure interface is chosen as an additional corner node.
augment	This keyword is useful only for the FETI-DP and/or FETI-DPH methods. It specifies the augmentation of the "coarse grid" by various methods defined by the following commands.
EdgeGs trans/all	In that case, the "coarse grid" is augmented using extra equations generated by the rigid body modes (rbms) of the subdomain interfaces, and organized edge-by-edge. Here an edge refers to an interface between two subdomains and not to the usual geometric edge. The "trans/all" option denotes the equation type where "trans" refers to the translational rbms, and "all" refers to both the translational and rotational rbms applied per edge.

<code>Gs trans/all</code>	In that case, the "coarse grid" is augmented using extra equations generated by the traces of the subdomain rigid body modes (rbms) on the subdomain interface boundaries, and organized subdomain-by-subdomain. The "trans/all" option denotes the equation type where "trans" refers to the translational rbms, and "all" refers to both the translational and rotational rbms.
<code>WeightedEdgeGs trans/all</code>	In that case, the "coarse grid" is augmented using extra equations generated by the weighted rigid body modes (rbms) of the subdomain interfaces, and organized edge-by-edge. Here an edge refers to an interface between two subdomains and not to the usual geometric edge. The "trans/all" option denotes the equation type where "trans" refers to the translational rbms, and "all" refers to both the translational and rotational rbms applied per edge. The weights are similar to those used for scaling the residuals. Hence, they are based on stiffness considerations if the <code>scaling</code> option is set to <code>stiffness</code> , or on topological considerations if the <code>scaling</code> option is set to <code>topology</code> .
<code>primal</code>	This choice is relevant for the one-level (standard) FETI-DP method, when either <code>Gs trans/all</code> or <code>WeightedEdgeGs trans/all</code> augmentation option is selected. It is also required in the context of an n -level FETI-DP method, for the FETI-DP solvers applied to levels 1 to $n - 1$. In this case, the rigid body modes are used to define primal unknowns on the "coarse grid" problems instead of additional constraints.
<code>EdgeWs [type] numdir</code>	This option is exclusive to the FETI-DPH solver and can be combined with the option <code>EdgeGs</code> . It augments the corner-based coarse problem of the FETI-DPH algorithm with extra equations generated by the free-space solutions of the frequency-domain acoustic or elastodynamic (or modelled dynamic shell) equation — these are real cosine and sine waves of arbitrary directions — and organized edge-by-edge. Here an edge refers to an interface between two subdomains and not to the usual geometric edge. There are as many free-space solutions to be considered as there are wave directions to be considered. Setting the optional parameter <code>type</code> to <code>solid</code> , <code>shell</code> , <code>fluid</code> , or <code>any</code> , depending on the type of the elements used in the mesh delivers the best performance. If the mesh contains several types of elements, the option <code>any</code> is recommended (default value). The parameter <code>numdir</code> specifies the number of desired directions and therefore controls the total number of such wavy augmentation modes (integer). Its default value is 0. The exact number of augmentation modes per interface edge is equal to $2 \times \text{numdir}$ for two- and three-dimensional frequency-domain acoustic problems, $4 \times \text{numdir}$ for two-dimensional elastodynamic (or modelled dynamic shell) problems, and $6 \times \text{numdir}$ for three-dimensional elastodynamic (or modelled dynamic shell) problems. In the latter case, the factor 6 comes from the fact that for each direction, there are 2 shear waves and 1 pressure wave, and each of these three waves has a cosine mode as well as a sine mode. The implemented directions are chosen according to the following scheme. In two dimensions, a sector is discretized into n sectors, with n an even integer. A direction is defined by connecting the center of the circle to a point on the circle delimiting a sector. Since both the cosine and sine modes are included, only one direction for each pair of opposite directions needs be retained, which results in a total of $n/2$ directions. Hence, n is chosen to be $2 * \text{numdir}$. In three dimensions, a cube is discretized into $n \times n \times n$ points. A direction is defined by connecting the center of the cube to a point lying on a face of the cube. Since both the cosine and sine modes are included, only one direction for each pair of opposite directions needs be retained, which results in a total of $n^2 + 4(n-1)(n/2-1) + 2(n-1)\text{mod}(n,2)$ directions. Hence, n is chosen so that $n^2 + 4(n-1)(n/2-1) + 2(n-1)\text{mod}(n,2)$ is as close as possible to <code>numdir</code> , with $n^2 + 4(n-1)(n/2-1) + 2(n-1)\text{mod}(n,2) \geq \text{numdir}$.
<code>interf_solver</code>	This option is currently available only for the FETI-DP and FETI-DPH solvers. It specifies the Krylov method to be used with these algorithms for solving the interface problem (characters).
<code>cg</code>	Turns on the CG algorithm as an interface problem solver. This is the default value of <code>interf_solver</code> .
<code>cgal</code>	Turns on Dostal's Augmented Lagrangian CG algorithm with adaptive precision control as an interface problem solver.
<code>gmres</code>	Turns on the GMRES algorithm as an interface problem solver.
<code>agmres</code>	Turns on the GMRES algorithm with a deflation-based accelerator for systems with multiple left and/or right sides (characters). In this case, a non zero deflation subspace size must also be specified using the sub-command keyword <code>deflation_size</code> : otherwise, <code>agmres</code> will revert to <code>gmres</code> . Note that by default, the acceleration of the solution of systems with multiple left sides (or nearby systems) using this configuration of the GMRES algorithm is disabled because unless the left side matrices have some special structure (that is not currently exploited), it requires a substantial overhead. However, this acceleration can be enabled using the sub-command keyword <code>mlhs</code> .
<code>deflation_size</code>	Specifies the size of the deflation subspace when AGMRES is specified as the Krylov-based interface solver for the FETI-DP or FETI-DPH iterative domain decomposition method (integer). The default value for this size is 0. Hence, activating the deflation-based acceleration component of the GMRES solver for systems with multiple left and/or right sides requires setting the deflation size to a strictly positive integer value.
<code>deflation_type</code>	Specifies how to construct the deflation subspace of the iterative solver <code>agmres</code> (integer). The default value is 0.
0	Specifies that the deflation subspace is to be constructed using eigenvectors associated with the smallest eigenvalues XXX of what?? XXX.
1	Specifies that the deflation subspace is to be constructed using eigenvectors associated with the largest eigenvalues XXX of what?? XXX.

<code>mlhs</code>	Specifies whether to turn on or off the deflation-based accelerator for systems with multiple left sides when <code>agmres</code> is specified as the Krylov-based interface solver for the FETI-DP or FETI-DPH iterative domain decomposition method. The default value is 0.
1	Turns on the deflation-based accelerator.
0	Turns off the deflation-based accelerator.
<code>gcr</code>	Turns on the GCR algorithm as an interface problem solver.
<code>orthotol</code>	This option is currently available only for the FETI-DPH solver. It specifies the tolerance to be used for filtering out "small" vectors during the local Gram-Schmidt-like orthogonalization of the augmentation vectors. The default value is 1.0e-02.
<code>mpc_type</code>	This keyword is useful only for the FETI-DP and FETI-DPH methods. It specifies the algorithm to be used for handling potential multipoint constraints (MPCs) (characters). The default is <code>primal</code> when the number of MPCs is less or equal to 1000, and <code>dual</code> otherwise.
<code>dual</code>	In this case, the MPCs are enforced by Lagrange multipliers and are satisfied only at convergence (characters).
<code>primal</code>	In this case, the MPCs are put in the coarse problem and are satisfied at every iteration (characters).
<code>mpc_prcn</code>	This keyword is useful only for the FETI-DP and FETI-DPH methods, in the presence of MPCs, and when <code>mpc_type</code> is set to <code>dual</code> (characters). The default is <code>tblock</code> when running a single thread process, and <code>full</code> otherwise.
<code>full</code>	In this case, the $[CC^T]$ matrix, where C denotes the constraint matrix, is treated as a single block matrix (characters).
<code>tblock</code>	In this case, the potential algebraic block-structure of the $[CC^T]$ matrix, is exploited (characters).
<code>sblock</code>	In this case, the subdomain-structure of the $[CC^T]$ matrix, is exploited (characters).
<code>mblock</code>	In this case, the mortar-interface-structure of the $[CC^T]$ matrix, is exploited (characters).
<code>diag</code>	In this case, the $[CC^T]$ matrix is approximated by its diagonal (characters).
<code>mpc_scaling</code>	
1 or <code>stiffness</code>	Stiffness based scaling (integer).
2 or <code>topology</code>	Topology (subdomain connectivity) based scaling (integer). This is the default scaling procedure.
<code>cct_solver</code>	This choice can represent any of the equation solvers listed below, if configured in the command SOLVERCNTL using for <code>SOLVER_ID</code> the same integer specified here in <code>solver_id</code> — that is, if this solver and its parameters and identification are specified in SOLVERCNTL (characters, integer).
<code>solverhandle solver_id</code>	Selects the skyline direct method as the coarse solver. This is the default choice (characters).
<code>skyline</code>	Selects Esmond's (sequential) sparse direct method as the coarse solver (characters).
<code>sparse</code>	
<code>cct_tol</code>	Specifies the tolerance to be used for detecting singularities in the solution of the $[CC^T]$ problems of the FETI-DP and FETI-DPH methods in the presence of MPCs (real). The default is 1.0E-12.
<code>kryprec</code>	This option is only for nonlinear problems. It turns on the Krylov preconditioner enrichment for the solution of nearby problems in a Newton method (characters).
1	Turns on the Krylov preconditioner and gives to the <code>maxorth</code> parameter (see below) the scope of the entire nonlinear analysis.
2	Turns on the Krylov preconditioner and gives to the <code>maxorth</code> parameter (see below) the scope of a load-step in a nonlinear analysis.
<code>global_cor_rbm_tol</code>	Specifies the tolerance to be used for detecting singularities in the solution of the corner-based coarse problems of the FETI-DP and FETI-DPH methods (real). The default is 1.0E-6.
<code>global_rbm_tol</code>	Specifies the tolerance to be used for detecting the global rigid body modes when using a FETI method (real). The default is 1.0E-6.
<code>maxorth</code>	Specifies the maximum total number of reorthogonalization vectors used for accelerating a FETI algorithm and/or enriching its chosen preconditioner during a simulation (integer). For linear problems or while the tangent operator of a nonlinear problem is frozen, the acceleration is performed using a multiple right side technique based on reorthogonalization, but only if <code>interf-solver</code> (see below) is set to <code>cg</code> or <code>gcr</code> . For nonlinear problems, the acceleration is performed using a multiple left side technique based on the enrichment of the preconditioner, but only if: (a) <code>interf_solv</code> (see below) is set to <code>cg</code> , and (b) <code>kryprec</code> (see above) is activated. If the <code>kryprec</code> option is not activated, <code>maxorth</code> is to be understood as the maximum total number of reorthogonalization vectors per Newton iteration, as these vectors are flushed when the tangent operator is rebuilt. For nonlinear problems where the tangent operator is periodically frozen, if the <code>kryprec</code> option is used, priority in the accumulation of vectors is given to enriching the preconditioner. The default value is <code>maxitr</code> (see below). To turn-off this reorthogonalization option, set <code>maxorth</code> to 1.
<code>tolfeti</code>	Error tolerance for the convergence of the fetsi solver (real). The default value is 1.0e-06.
<code>maxitr</code>	Maximum number of iterations to be performed (integer). The default value is 1000.
<code>PIECEWISE real_1 real_2</code>	This sub-command keyword (characters) followed by two real numbers is applicable to linear static and dynamic analyses. It requests the interpretation of concentrated follower (see FORCES), pressure-induced (see PRESSURE and CONWEP), temperature-induced (see TEMPERATURES and THERMOE), and freeplay-induced (see MATERIAL) forces and moments, and moments due to gravity (see GRAVITY) and discrete masses (see DIMASS) as configuration-dependent external forces and moments, and enables their piecewise constant treatment. In this treatment, the application of the aforementioned

external loading is divided into steps and is performed at the beginning of each step using the updated deformed configuration. To this effect, and in the case of a static analysis: the first entered real number, `real_1`, specifies the load fraction increment to apply at each step; and the second real number, `real_2`, specifies the total load factor to apply to all inputted external loads. For example, consider the input "PIECEWISE 0.25 1.0". The second entry "1.0" requests keeping all external loads inputted using various commands of **AERO-S** unchanged, and the first entry "0.25" requests their application to the system of interest in 4 steps — that is, the application at each step of 25% only of these external loads. On the other hand, inputting "PIECEWISE 0.25 2.0" requests multiplying first all external loads resulting from various external load input commands by the factor 2, then applying at each step a load increment equal to 0.25 times of the *original* external loads. This implies splitting the application of the magnified external loads in 8 steps and applying at each step an external load increment equal to 12.5% of the magnified external load.

For a dynamic analysis, this sub-command is automatically activated for the freeplay-induced (see [MATERIAL](#)) forces and moments.

CASES

This optional sub-command keyword (characters) can be used to select a "load" case among those defined in the [LOADCASE](#) command. In this capacity, it is relevant to all analyses. However for *linear* static analysis and single-frequency frequency response analysis, it can also be used to select multiple "load" cases among those defined in the [LOADCASE](#) command. Selecting no "load" case is equivalent to selecting the load case 0 which by default contains the "load" set 0 and any "load" generated by a command which does not support the "load" set construct ([LOADSET_ID](#)) (see the [FORCES](#), [PRESSURE](#), [HNEU](#), [FLUX](#), and/or [CONVECTION](#) command).

case_id_j

Non-negative integer identifying uniquely a j -th "load" case that is defined in the [LOADCASE](#)

command. Specifying more than one case identifier on the same line results in a multiple "load" case analysis.

Next: [SS2DT](#), Previous: [STATICS](#)

120 STRESS-STRAIN ONE-DIMENSIONAL TABLE

Command Statement: **SS1DT**

The `ss1dt` command can be used to describe the evolution of stress with strain, for a given material. This evolution can be specified here via a curve (or one-dimensional table) defined by pairs of strain and stress values. Linear interpolation is used for "in between" points, and the extrema values are adopted for "outside" points. Several curves can be specified, one after the other. Each curve is identified by an ID number as described below.

SS1DT

```
CURVE    curve_id
strain_1 stress_1
.
.
.
strain_n stress_n
```

CURVE Sub-command keyword (characters) signaling the definition of a stress-strain curve (or table).
curve_id "Id number" for the following curve (or one-dimensional table) (integer).
strain_i A specified strain value (real).
stress_i A specified stress value at strain `strain_i` (real).

Next: [SDEAFT](#), Previous: [SS1DT](#)

121 STRESS-STRAIN TWO-DIMENSIONAL TABLE

Command Statement: **SS2DT**

The `ss2dt` command can be used to describe the evolution of stress with biaxial strain, for a given material. This evolution can be specified here in a surface (or two-dimensional table) defined by triplets of two strain values and one stress value. Linear interpolation is used for "in between" points, and the extrema values are adopted for "outside" points. Several surfaces can be specified, one after the other. Each surface is identified by an ID number as described below.

SS1DT

```
SURFACE    surface_id [ENGINEERING eng_flag]
           strain22_1 ...      strain22_m
           strain11_1 stress_11 ...      stress_1m
           .
           .
           .
           strain11_n stress_n1 ...      stress_nm
```

SURFACE Sub-command keyword (characters) signaling the definition of a stress-biaxial-strain surface (or table).
surface_id "Id number" for the following surface (or two-dimensional table) (integer).
ENGINEERING Optional sub-command keyword signaling the input of a flag that specifies the interpretation of the inputted stress and strain data (characters).
eng_flag A flag specifying the interpretation of the inputted stress and strain data (characters). The value `on` indicates that the stress and strain data are to be interpreted as engineering quantities. The value `off` indicates that the stress and strain data are to be interpreted as the second-Piola Kirchhoff stress and Green-Lagrange strain, respectively. The default value is `on`.
strain11_i A specified strain11 value (real).
strain22_j A specified strain22 value (real).
stress_ij A specified stress value at strains `strain11_i` and `strain22_j` (real).

Next: [SURFACETOPO](#), Previous: [SS2DT](#)

122 STRUCTURAL DAMPING LOSS FACTOR TABLE

Command Statement: **SDETAFT**

The command `SDETAFT` can be used to describe for a given material, the variation of the Young modulus loss factor η_E (see the sub-command keyword `STRDAMP` in [MATERIAL](#)) with the frequency f . This evolution can be specified here in one or multiple curves (or tables) each defined by pairs of Young modulus loss factor and frequency values. On each curve, linear interpolation is used for the "in between" points, and constant extrapolation (using the loss factor value at the closest frequency) is adopted for the "outside" points. When multiple curves are defined, they are inputted one after the other, and each is identified by an id number as described below.

SDETAFT

```
CURVE curve_id
f1 ηE1
—
.
.
.
fn ηEn
—
```

CURVE Sub-command keyword (characters) for defining a curve (or table) describing the variation of the Young modulus loss factor η_E with the frequency f in the form of pairs (f_i, η_{E_i}) .
curve_id "Id number" for the defined curve (integer).
f_i A sampled frequency value (real).
η_{E_i} Value of the Young modulus loss factor at the frequency f_i (real).

Next: [TETT](#), Previous: [SDETAFT](#)

123 SURFACE TOPOLOGY

Command Statement: **SURFACETOPO**

The `SURFACETOPO` command statement can be used to define a discrete surface on a body by describing its faceted connectivity. Each facet may, but does not necessarily have to, be an element defined in the command [TOPOLOGY](#). Such a surface can be paired with another similar surface using another command statement in order to define a surface-to-surface interaction (see [TIEDSURFACES](#), [CONTACTSURFACES](#), [FSINTERFACE](#), [AERO](#)). It can also be referred to by another command statement to define a surface boundary condition (see [PRESSURE](#), [FORCES](#), [DISPLACEMENTS](#), [FLUX](#) and [TEMPERATURES](#)) or a group of nodes (see [GROUPS](#)).

Each surface is inputted using two sections. The first one identifies the surface to be defined and assigns to it, if needed, a thickness. This

section is followed by a second section which contains the description of all faces constituting the surface. Each face is specified on a separate line. The set of two sections can be repeated as many times as there are surfaces to be defined.

The format of this command is as follows.

SURFACETOPO	ID_NUMBER	SURFACE_THICKNESS	t
-------------	-----------	-------------------	---

ID_NUMBER
SURFACE_THICKNESS

Surface id number (integer).

Optional keyword indicating that the surface identified by ID_NUMBER defines the midplane of a physical surface (string). The importance of this parameter is underlined by the following note and consequences. For contact detection and enforcement, a surface is treated as 2-sided (or "shell" surface) when both of the following conditions are met: (a) the simulation is explicit dynamics with flagTDENFORCE set to on in DYNAMICS (or omitted since this is the default setting), and (b) a non-zero value is specified here for this surface thickness parameter. In all other cases, the contact detection and enforcement is 1-sided. For a 1-sided surface, the directions of the normals to the faces of the surface are important. Contact interactions between two 1-sided surfaces, or self-contact involving one 1-sided surface, can only be detected between two faces with normals in opposite directions. Furthermore, if an interaction is detected, the directions of the normals establish whether the configuration involves penetration or separation. For a 2-sided surface, the directions of the normals of the faces of the surface are not important. Contact interactions between two 2-sided surfaces, or self-contact involving one 2-sided surface, can be detected between two faces regardless of the directions of the normals. For contact between one 2-sided surface and one 1-sided surface, only the normal direction of the 1-sided surface is important. If an interaction is detected, the direction of the normal of the 1-sided surface establishes whether the configuration involves penetration or separation.

t

Real or virtual thickness of the surface used for detecting and enforcing contact conditions on both of its sides (real). The default value is 0.

FACE#	FACETYPE	CONNECTIVITY_NODES
-------	----------	--------------------

FACE#
FACETYPE
1
2
3
4
5
6
7

Face id number whose type and connectivity are to be specified (integer).

4-node quadrilateral.
8-node quadrilateral.
3-node triangle.
6-node triangle.
9-node quadrilateral.
12-node quadrilateral.
10-node triangle.

Next: [THERMOE](#), Previous: [SURFACETOPO](#)

124 THERMAL EXPANSION TEMPERATURE TABLE

Command Statement:	TETT
--------------------	------

The TETT command statement can be used to describe the evolution of the coefficient of thermal expansion with temperature, for a given material. This evolution can be specified here in a curve (or table) defined by pairs of temperature and coefficient of thermal expansion values. Linear interpolation is used for "in between" points, and the extrema values are adopted for "outside" points. Several curves can be specified, one after the other. Each curve is identified by an "id number" as described below.

TETT

CURVE curve_id
T_1 TE_1
.
.
.
T_n TE_n

CURVE
curve_id

"Id number" for the following curve (or table) (integer).

<code>T_1</code>	A specified temperature value (float).
<code>TE_1</code>	A specified coefficient of thermal expansion value at temperature <code>T_1</code> (float).

Next: [TIEDSURFACES](#), Previous: [TETT](#)

125 THERMOELASTICITY

Command Statement:	THERMOE
--------------------	----------------

The **THERMOE** command statement is used to indicate that **AERO-S** is to interact with itself to perform a *transient* thermoelastic (thermostructure-structure vibration) *one-way* coupled simulation.

The syntax for invoking this option is given below.

THERMOE

Next: [USDF](#), Previous: [THERMOE](#)

126 TIED SURFACES

Command Statement:	TIEDSURFACES
--------------------	---------------------

The **TIEDSURFACES** command can be used to tie — that is, enforce perfect contact between — pairs of surfaces defined using the command **SURFACETOPO**. Surface interactions are detected using the search module of the library ACME. For explicit computations, the discrete kinematic constraint equations are defined and enforced as specified in the sub-command keyword `TDENFORCE` and its associated flag `flagTDENFORCE` of the **DYNAMICS** object. For implicit computations, the discrete kinematic constraint equations are formulated using **AERO-S**'s mortar method and enforced using the method specified in **CONSTRAINTS** or in **CONSTRAINT_METHOD** below with the following limitation: in this case, the enforcement method can be only one of the following three methods: the penalty method, the Lagrange multiplier method, or the augmented Lagrange multiplier method.

Note 1: The enforcement of tied surface constraints by the Lagrange multiplier method in all but explicit dynamic analyses is supported only by the FETI-DP family of solvers, the GMRES solver, and the SPOOLES and MUMPS direct sparse solvers with pivoting enabled (see **STATICS**).

TIEDSURFACES

SURF_PAIR_ID#	MASTER	SLAVE
----------------------	--------	-------

or, for static, frequency response, eigenvalue, implicit dynamic, and explicit dynamic computations with the flag `flagTDENFORCE` set to off (see **DYNAMICS**)

SURF_PAIR_ID#	MASTER	SLAVE	MORTAR_TYPE	NORMAL_TOL	TANGENTIAL_TOL	CONSTRAINT_METHOD
----------------------	--------	-------	-------------	------------	----------------	-------------------

or, for explicit dynamic computations with the flag `flagTDENFORCE` set to on (see **DYNAMICS**)

SURF_PAIR_ID#	MASTER	SLAVE	KPART_TYPE	NORMAL_TOL	TANGENTIAL_TOL	NUM_ITER	CONVERG_TOL	CONSTRAINT_METHOD
----------------------	--------	-------	------------	------------	----------------	----------	-------------	-------------------

<code>SURF_PAIR_ID#</code>	Id number of the surface pair to be described (integer).							
<code>MASTER</code>	Identification of the master (mortar method) surface (see SURFACETOPO) (integer).							
<code>SLAVE</code>	Identification of the slave (mortar method) surface (see SURFACETOPO) (integer).							
<code>CONSTRAINT_METHOD</code>	Method for enforcing the associated constraints (characters). The default method is set in CONSTRAINTS and used whenever this entry is omitted.							
<code>multipliers</code>	The Lagrange multiplier method.							
<code>elimination</code>	The elimination method (see CONSTRAINTS for changing the default values of its parameters).							
<code>penalty beta</code>	The penalty method. The parameter <code>beta</code> should be a large positive number, typically of the order of 10^8 (no default value is provided).							
<code>augmented beta</code>	The augmented Lagrangian method. The parameter <code>beta</code> should be a large positive number, typically of the order of 10^8 (no default value is provided).							
<code>MORTAR_TYPE</code>	Mortar type: 0 = standard, 1 = dual, default value is 0 (integer).							
<code>NORMAL_TOL</code>	Normal search tolerance used by ACME to identify interactions, default value is 0.1 (float) (see Figs. 1.2 and 1.3 in Section 1.3 of ACME's User Reference Manual).							
<code>TANGENTIAL_TOL</code>	Tangential search tolerance used by ACME to identify interactions, default value is 0.001 (float) (see Figs. 1.2 and 1.3 in Section 1.3 of ACME's User Reference Manual).							
<code>KPART_TYPE</code>	Kinematic partitioning type: 0 = fixed, 1 = automatic, default value is 0 (integer).							

NUM_ITER	Maximum number of predictor-corrector iterations to be performed at each time step. The default value is 5 (integer).
CONVERG_TOL	Convergence tolerance of the predictor-corrector iteration loop. The default value is 1.0e-10 (float).

Next: [USDD](#), Previous: [TIEDSURFACES](#)

127 USER DEFINED FORCES *S*

Command Statement: **USDF**

The main purpose of the **usdf** command is to specify nodal forces and/or moments of the *follower* or *axial* type for a structural model via a user-defined subroutine. In this case, the user should: (1) write his/her own algorithm for specifying the nodal forces and/or moments within a subroutine named `control.c` (see [APPENDIX 10](#)), (2) compile this subroutine and link it using the makefile provided for this command, and (3) use [LOAD](#) to activate it.

By default, all user-defined forces and moments are interpreted as being of the *axial* type — that is, as being defined in the *fixed* nodal degree of freedom reference frames (see [NODES](#) and [NFRAMES](#)). However, if a node has rotational degrees of freedom, the user can specify that the forces and/or moments prescribed at this node are of the *follower* type — that is, they act in a direction that remains constant in the local frame attached to the node where they are applied. This local frame coincides with the nodal degree of freedom reference frame (see [NODES](#) and [NFRAMES](#)) in the undeformed configuration. In the deformed configuration, the orientation of this local frame is defined by the rotation of the node to which it is attached. In other words, the specified nodal force or moment "follows" in this case the rotation of the node to which it is applied.

User-defined nodal forces and/or moments are not assembled neither at the element level nor at the subdomain level. They can be combined with those specified under the [FORCES](#) and [GRAVITY](#) commands, but will not be amplified by the [MFTT](#) or [HFTT](#) tables. In case of a conflict between this command and the [FORCES](#) and [GRAVITY](#) commands, the specified nodal forces are simply added. A more detailed description of the `control.c` subroutine is given in the form of a template in [APPENDIX 10](#).

Note 1: By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: All forces must be specified in the computational basis.

Note 3: The **usdf** command is redundant with the [ACTUATORS](#) command in the sense that [ACTUATORS](#) can achieve whatever **usdf** can achieve. However, the reverse is not true.

Note 4: Similarly, this command can be used to prescribe time-variant, Neumann boundary conditions (or source terms) for a time-domain acoustic simulation.

Note 5: Specifying a follower force or moment leads to an unsymmetric tangent "load" stiffness matrix during a [NONLINEAR](#) analysis.

The syntax for invoking this option is given below.

USDF

NODE#	DOF#	TYPE
-------	------	------

NODE#	Node number where the force/moment is specified (integer).	
DOF#	Degree of freedom local number where the force/moment is specified (integer).	
TYPE	For structural models, all user-defined nodal sources (forces and/or moments in the case of a structural model) are by default of the axial type. However, if this parameter is set to FOLLOWER and the node NODE# has rotational degrees of freedom, the user-defined source (force or moment in the case of a structural model) at this node and the degree of freedom DOF# is considered to be of the follower type (characters).	

Next: [WEIGHTS](#), Previous: [USDF](#)

128 USER DEFINED PRESCRIBED DISPLACEMENTS *S*

Command Statement: **USDD**

The purpose of the **usdd** command is to specify time-variant prescribed displacements for any node and its associated dofs using a user-defined subroutine. In this case, the user should: 1) write his/her own algorithm for specifying the prescribed displacement field within a subroutine named "control.C", 2) compile this subroutine and link it with **AERO-S** using the makefile that is provided for this purpose, and 3) use the [LOAD](#) command (see [LOAD](#)) to activate it. An example input file using the **usdd** command can be found in `FEM.d/fem_examples/USDD.d`.

This command can be used simultaneously with the time-invariant command [DISPLACEMENTS](#) for prescribing displacements at a node. In

the event of a conflict between the [DISPLACEMENTS](#) and [USDD](#) commands, [USDD](#) prevails. A more detailed description of the "control.C" subroutine is given in its template located in the Control.d directory of **AERO-S**.

The syntax for invoking this option is given below.

Note 1: All degrees of freedom referred to by this command are defined in the nodal degree of freedom reference frames defined at the nodes where these degrees of freedom are attached (see [NODES](#) and [NFRAMES](#)). By default, the nodal degree of freedom reference frames are the same as the global reference frame.

Note 2: Similarly, this command can be used to specify time-variant prescribed Dirichlet boundary conditions for a time-domain acoustic simulation by setting [DOF#](#) to 8 (see below).

[USDD]

[NODE# DOF#]

NODE#
DOF#

Node number where the displacement is specified (integer).
Degree of freedom local number where the displacement is specified (integer).

Next: [YMST](#), Previous: [USDD](#)

129 WEIGHTS

[Command Statement: WEIGHTS]

The [WEIGHTS](#) command statement can be used to modify some or all of the weights attributed by default to an element type (see [TOPOLOGY](#)). These weights are exploited by the mesh partitioning algorithm (see [DECOMPOSE](#)) to achieve load balance when generating the subdomains.

Note 1: Variable weights attributed by default to elements with a variable number of nodes cannot be modified by this command.

The input format of this command is given below.

[WEIGHTS]

[ELEMENT_TYPE# DESIRED_WEIGHT]

ELEMENT_TYPE# This is the element type id number as in the [TOPOLOGY](#) command (integer).
DESIRED_WEIGHT This is the desired weight to be attributed to this element type (integer).

The weight default values are as follows:

M=Mechanic H=Heat C=Coupled Thermoelastic F=Fluid A=Acoustic W = default weight

- | | |
|------|---|
| 1 | M: 3d truss (bar) with 3 dof/node (W = 1). |
| 2 | M: 2d 4-node quadrilateral with 2 dof/node (W = 2). |
| 3 | H: 3d 4-node quadrilateral with 1 dof/node (W = 2). |
| 4 | M: 2d 3-node triangle with 2 dof/node (W = 2). |
| 6 | M: 3d Bernoulli beam with 6 dof/node (W = 1). |
| 7 | M: 3d Timoshenko beam with 6 dof/node (W = 1). |
| 8 | M: 3d 3-AQR shell with 6 dof/node (W = 3). |
| 88 | M: 3d 4-node shell with 6 dof/node (splits into 2 elements of type 8) (W = 4). |
| 9 | H: 3d 2-node line with 1 dof/node (W = 1). |
| 10 | H: 2d 4-node quadrilateral with 1 dof/node (W = 2). |
| 11 | M: 3d 1-node torsional spring with 3 dof/node (W = 1). |
| 12 | M: 3d 1-node translational spring with 3 dof/node (W = 1). |
| 15 | M: 3d 3-node triangular AQR shell with 6 dof/node and composite as well as nonlinear material capability (W = 3). |
| 1515 | M: 3d 4-node quadrilateral AQR shell with 6 dof/node and composite as well as nonlinear material capability (W = 4). |

16 M: 3d Belytschko-Tsay shell with 6 dof/node (**W = 4**).
 17 M: 3d 8-node brick with 3 dof/node (**W = 3**).
 18 M: 3d 4-node shear panel with 3 dof/node (**W = 1**).
 19 M: 3d 3-node membrane with 6 dof/node (**W = 3**).
 (only in-plane and drilling stiffnesses) (**W = 1**).
 20 M: 3d 3-node composite or orthotropic shell with 6 dof/node (**W = 3**).
 2020 M: 3d 4-node composite or orthotropic shell with 6 dof/node (splits into 2 elements of type 20) (**W = 4**).
 21 M: 3d translational spring-link with 3 dof/node (**W = 1**).
 22 M: 3d torsion spring-link with 3 dof/node (**W = 1**).
 23 M: 3d 4-node tetrahedron with 3 dof/node (**W = 3**).
 24 M: 3d 6-node pentahedron with 3 dof/node (**W = 3**).
 25 M: 3d 10-node tetrahedron with 3 dof/node (**W = 4**).
 30 A: 2d 4-node quadrilateral with 1 dof/node (**W = 1**).
 31 A: 2d 4-node quadrilateral GLS element with 1 dof/node (**W = 1**).
 32* A: 2d 8-node quadrilateral with 1 dof/node (**W = 3**).
 33* A: 2d 4-node quad bubble element with 1 dof/node (**W = 1**).
 34* A: 2d 4-node quad two-level bubble element with 1 dof/node (**W = 1**).
 35 A: 2d 3-node triangle with 1 dof/node (**W = 1**).
 36 A: 2d 3-node triangular GLS element with 1 dof/node (**W = 1**).
 38* A: 2d 6-node triangle with 1 dof/node (**W = 3**).
 40 A: 3d 4-node tetrahedron with 1 dof/node (**W = 2**).
 41 A: 3d 4-node tetrahedral GLS element with 1 dof/node (**W = 2**).
 42 A: 3d 10-node tetrahedron with 1 dof/node (**W = 3**).
 44 A: 3d 8-node brick GLS element with 1 dof/node (**W = 3**).
 45* A: 3d 8-node brick with 1 dof/node (**W = 3**).
 46 H: 3d 3-node triangular heat element (**W = 2**).
 4646 H: 3d 4-node triangular heat element (splits into 2 elements of type 46) (**W = 3**).
 47 H: 3d 2-node line with 1 dof/node for boundary convection (**W = 1**).
 48 H: 3d 4-node quadrilateral with 1 dof/node for boundary convection (**W = 2**).
 49 H: 3d 3-node triangle with 1 dof/node for boundary convection (**W = 2**).
 50 H: 3d 4-node tetrahedron with 1 dof/node (**W = 3**).
 51 H: 3d 8-node brick with 1 dof/node (**W = 4**).
 52* M: 3d 6-node triangular shell (**W = 1**).
 53 H: 2d 3-node triangular heat element (**W = 2**).
 56 H: 3d 2-node heat radiation element (**W = 1**).
 57 H: 3d 3-node triangular heat radiation element (**W = 2**).
 58 H: 3d 4-node quadrilateral heat radiation element (**W = 2**).
 59 A: 2d 6-node axisymmetric triangle with 1 dof/node (**W = 3**).
 60 A: 2d 4-node axisymmetric quadrilateral with 1 dof/node (**W = 2**).
 61 A: 2d 3-node axisymmetric triangle with 1 dof/node (**W = 1**).
 62 A: 2d 8-node axisymmetric quadrilateral with 1 dof/node (**W = 3**).
 65 M: 3d 2-node rigid truss (bar) (**W = 1**).
 66 M: 3d 2-node rigid beam (**W = 1**).
 67 M: 3d rigid link (translational and rotational) (**W = 1**).
 68 M: 3d rigid translational link (**W = 1**).
 69 M: 3d rigid rotational link (**W = 1**).
 70 M: 3d rigid plane or solid element with 3 dof/node (3-node to 20-node element) (**W = 3**).
 71 M: 3d rigid plane or solid element with 3 dof/node and anywhere from 3 to 20 nodes per element (**W = 1**).
 72 M: 3d 20-node brick (**W = 4**).
 73 M: 3d 3-node rigid shell (**W = 3**).
 74 M: 3d rigid plane or solid element with 6 dof/node (3-node to 32-node element) (**W = 1**).
 76 M: 3d 4-node rigid shell (**W = 4**).
 78 M: 3d 1-node point-to-line constraint element with 3 dof/node (**W = 1**).
 79 M: 3d 1-node point-to-plane constraint element with 3 dof/node (**W = 1**).
 81 H: 2d 4-node contact resistance thermal element with 1 dof/node that can be inserted between two thermal elements of type = 10 (**W = 2**).
 82 H: 3d 8-node contact resistance thermal element with 1 dof/node that can be inserted between two thermal elements of type = 51 (**W = 4**).
 83 H: 3d 6-node contact resistance thermal element with 1 dof/node that can be inserted between two thermal elements of type = 50 with 4 nodes each (**W = 4**).
 84 H: 2d 3-node triangular bulk fluid (thermal) element with 1 dof/node. The first node appearing in the connectivity list of this element must be inside the bulk fluid and the other two nodes must be located at the interface boundary between the bulk fluid and the surrounding material. The temperature at the first node is by definition the average temperature of the bulk fluid (**W = 2**).
 85 H: 3d 4-node bulk fluid (thermal) tetrahedron with 1 dof/node. The first node appearing in the connectivity list of this element must be inside the bulk fluid and the other three nodes must be located at the interface boundary between the bulk fluid and the surrounding material. The temperature at the first node is by definition the average temperature of the bulk fluid (**W = 3**).

86 H: 3d 5-node pyramidal bulk fluid (thermal) element with 1 dof/node. The first node appearing in the connectivity list of this element must be inside the bulk fluid and the other four nodes must be located at the interface boundary between the bulk fluid and the surrounding material. The temperature at the first node is by definition the average temperature of the bulk fluid ($\mathbf{W} = \mathbf{4}$).
 87 M: 3d 4-node membrane with 6 dof/node ($\mathbf{W} = \mathbf{4}$).
 89 A: 3d 6-node wedge with 1 dof/node ($\mathbf{W} = \mathbf{3}$).
 91 M: 3d 32-node serendipity brick with 3 dof/node ($\mathbf{W} = \mathbf{6}$).
 92 M: 3d 26-node serendipity wedge with 3 dof/node ($\mathbf{W} = \mathbf{5}$).
 93 A: 3d 32-node serendipity brick with 1 dof/node ($\mathbf{W} = \mathbf{5}$).
 94 A: 3d 26-node serendipity wedge with 1 dof/node ($\mathbf{W} = \mathbf{4}$).
 95 A: 3d 8-node, or 27-node, or 64-node, or 125-node hexahedron with 1 dof/node ($\mathbf{W} = \mathbf{3}, \mathbf{4}, \mathbf{5}, \mathbf{6}$).
 96 A: 3d 4-node, or 10-node, or 20-node, or 35-node tetrahedron with 1 dof/node ($\mathbf{W} = \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}$).
 97 M: 3d 15-node serendipity wedge with 3 dof/node ($\mathbf{W} = \mathbf{4}$).
 98 A: 2d 4-node, or 9-node, or 16-node, or 25-node quadrilateral with 1 dof/node ($\mathbf{W} = \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}$).
 99 A: 2d 3-node, or 6-node, or 10-node triangle with 1 dof/node ($\mathbf{W} = \mathbf{2}, \mathbf{3}, \mathbf{4}$).
 100 M: 2d 4-node, or 9-node, or 16-node, or 25-node quadrilateral with 3 dof/node ($\mathbf{W} = \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}$).
 101 M: 2d 3-node, or 6-node, or 10-node triangle with 3 dof/node ($\mathbf{W} = \mathbf{2}, \mathbf{3}, \mathbf{4}$).
 102 M: 3d 8-node, or 27-node, or 64-node, or 125-node hexahedron with 3 dof/node ($\mathbf{W} = \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}$).
 103 M: 3d 4-node, or 10-node, or 20-node, or 35-node hexahedron with 3 dof/node ($\mathbf{W} = \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}$).
 105 A: 3d 8-node, or 27-node, or 64-node, or 125-node spectral hexahedron with 1 dof/node ($\mathbf{W} = \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}$).
 106 M: 3d 2-node rigid beam of arbitrary length, including zero length ($\mathbf{W} = \mathbf{1}$).
 108 A: 2d 4-node, or 9-node, or 16-node, or 25-node spectral quadrilateral with 1 dof/node ($\mathbf{W} = \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}$).
 109 H: 3d 8-node, or 27-node, or 64-node, or 125-node hexahedron with 1 dof/node ($\mathbf{W} = \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}$).
 111 M: 3d fabric truss with 3 dof/node ($\mathbf{W} = \mathbf{1}$).
 118 M: 3d 2-node planar joint with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 120 M: 3d 2-node spherical joint with 3 dof/node ($\mathbf{W} = \mathbf{1}$).
 121 M: 3d 2-node translational joint with 3 rotational dof/node ($\mathbf{W} = \mathbf{1}$).
 122 M: 3d 2-node universal joint with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 123 M: 3d 2-node revolute joint with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 124 M: 3d 2-node cylindrical joint with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 125 M: 3d 2-node prismatic joint with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 126 M: 3d 2-node joint-with-driver with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 127 M: 3d 2-node pin-in-slot joint with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 128 M: 3d 4-node plane stress/plane strain quadrilateral with 3 dof/node ($\mathbf{W} = \mathbf{1}$).
 129 M: 3d 3-node plane stress/plane strain triangle with 3 dof/node ($\mathbf{W} = \mathbf{1}$).
 131 M: 3d 1-node discrete mass and inertia element with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 132 M: 3d 2-node *massless* rigid beam of *non zero* length ($\mathbf{W} = \mathbf{1}$).
 133 M: 3d 2-node *massless* rigid beam of arbitrary length, including zero length ($\mathbf{W} = \mathbf{1}$).
 134 M: 3d 2-node prismatic joint-with-driver and 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 150 M: 3d 4-node rigid tetrahedron with 3 dof/node ($\mathbf{W} = \mathbf{3}$).
 151 M: 3d 6-node rigid pentahedron with 3 dof/node ($\mathbf{W} = \mathbf{4}$).
 177 M: 3d 2-node point-to-moving-point constraint element with 3 dof/node ($\mathbf{W} = \mathbf{1}$).
 178 M: 3d 3-node point-to-moving-line constraint element with 3 dof/node ($\mathbf{W} = \mathbf{1}$).
 179 M: 3d 4-node point-to-moving-plane constraint element with 3 dof/node ($\mathbf{W} = \mathbf{1}$).
 180 M: 3d 8-node Q1-P0 hexahedron with 3 dof/node and nonlinear material capability ($\mathbf{W} = \mathbf{3}$).
 181 M: 3d 20-node Q2-P0 hexahedron with 3 dof/node and nonlinear material capability ($\mathbf{W} = \mathbf{4}$).
 182 M: 3d 20-node Q2-P1 hexahedron with 3 dof/node and nonlinear material capability ($\mathbf{W} = \mathbf{4}$).
 183 M: 3d 4-node P1-P0 tetrahedron with 3 dof/node and nonlinear material capability ($\mathbf{W} = \mathbf{3}$).
 200 M: 3d 2-node uniaxial translational spring with 3 dof/node for linear or nonlinear analyses ($\mathbf{W} = \mathbf{1}$).
 201 M: 3d 2-node uniaxial translational spring with 6 dof/node for linear or nonlinear analyses ($\mathbf{W} = \mathbf{3}$).
 202 M: 3d 2-node uniaxial torsional spring with 3 rotational dof/node for linear or nonlinear analyses ($\mathbf{W} = \mathbf{3}$).
 203 M: 3d 2-node uniaxial tension-only translational spring with 3 dof/node and an *optional* freeplay model, for linear or nonlinear analyses ($\mathbf{W} = \mathbf{1}$).
 204 M: 3d 2-node uniaxial tension-only or compression-only translational spring with 6 dof/node and a freeplay model, for linear or nonlinear analyses ($\mathbf{W} = \mathbf{1}$).
 205 M: 3d 2-node uniaxial tension-only or compression-only torsional spring with 3 rotational dof/node and a freeplay model, for linear or nonlinear analyses ($\mathbf{W} = \mathbf{1}$).
 220 M: 3d 2-node spherical joint spring combination with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 221 M: 3d 2-node translational joint spring combination element with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 222 M: 3d 2-node universal joint spring combination element with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 223 M: 3d 2-node revolute joint spring combination element with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 224 M: 3d 2-node cylindrical joint spring combination element with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 225 M: 3d 2-node prismatic joint spring combination element with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 226 M: 3d 2-node revolute joint-with-actuator and 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 227 M: 3d 2-node pin-in-slot joint spring combination element with 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 234 M: 3d 2-node prismatic joint-with-actuator and 6 dof/node ($\mathbf{W} = \mathbf{1}$).
 280 M: 3d 8-node Q1-P0 incompressible hexahedron with 3 dof/node and nonlinear material capability ($\mathbf{W} = \mathbf{3}$).

```

301      F: 2d 4-node sloshing (fluid) quadrilateral with 1 dof/node (W = 1).
302      F: 2d 2-node free-surface (fluid) element with 1 dof/node for two-dimensional sloshing computations using element type = 301 (W = 1).
311      F: 3d 4-node sloshing (fluid) tetrahedron with 1 dof/node (W = 3).
312      F: 3d 3-node free-surface (fluid) triangle with 1 dof/node for three-dimensional sloshing computations using element type = 311 (W = 2).
321      F: 2d 4-node hydroelastic vibration (fluid) quadrilateral with 1 dof/node (W = 2).
323      M: 3d 2-node revolute joint spring combination element with 6 dof/node and a freeplay model (W = 1).
325      M: 3d 2-node prismatic joint spring combination element with 6 dof/node and a freeplay model (W = 1).
331      F: 3d 4-node hydroelastic vibration (fluid) tetrahedron with 1 dof/node (W = 3).
1100     A: 2d 4-node Helmholtz DGM element Q-4-1 (W = 1).
1101     A: 2d 4-node Helmholtz DGM element Q-8-2 (W = 2).
1102     A: 2d 4-node Helmholtz DGM element Q-16-4 (W = 4).
1103     A: 2d 4-node Helmholtz DGM element Q-32-8 (W = 8).
1110     A: 2d 3-node Helmholtz DGM element T-4-1 (W = 1).
1111     A: 2d 3-node Helmholtz DGM element T-8-2 (W = 2).
1120     A: 2d 4-node Helmholtz DEM element Q-4-1 (W = 1).
1121     A: 2d 4-node Helmholtz DEM element Q-8-2 (W = 2).
1122     A: 2d 4-node Helmholtz DEM element Q-16-4 (W = 4).
1123     A: 2d 4-node Helmholtz DEM element Q-32-8 (W = 8).
1130     A: 2d 3-node Helmholtz DEM element T-4-1 (W = 1).
1131     A: 2d 3-node Helmholtz DEM element T-8-2 (W = 2).
1150     A: 3d 8-node Helmholtz DGM element H-6-1 (W = 1).
1151     A: 3d 8-node Helmholtz DGM element H-26-4 (W = 4).
1152     A: 3d 8-node Helmholtz DGM element H-56-8 (W = 8).
1153     A: 3d 8-node Helmholtz DGM element H-98-12 (W = 12).
1160     A: 2d 3-node Helmholtz DGM element T-6-1 (W = 1).
1161     A: 2d 3-node Helmholtz DGM element T-26-4 (W = 4).
1162     A: 2d 3-node Helmholtz DGM element T-56-8 (W = 8).
1170     A: 3d 8-node Helmholtz DEM element H-6-1 (W = 1).
1171     A: 3d 8-node Helmholtz DEM element H-26-4 (W = 4).
1172     A: 3d 8-node Helmholtz DEM element H-56-8 (W = 8).
1173     A: 3d 8-node Helmholtz DEM element H-98-12 (W = 12).
1200     M: 2d 4-node Elastodynamic DGM element Q-4x2-2 (W = 2).
1201     M: 2d 4-node Elastodynamic DGM element Q-16x2-8 (W = 8).
1220     M: 2d 4-node Elastodynamic DEM element Q-4x2-2 (W = 2).
1250     M: 3d 4-node Elastodynamic DGM element H-6x3-3 (W = 3).
1251     M: 3d 4-node Elastodynamic DGM element H-26x3-15 (W = 15).
1252     M: 3d 4-node Elastodynamic DGM element H-50x3-28 (W = 28).

```

Next: [YMTT](#), Previous: [WEIGHTS](#)

130 YOUNG MODULUS-STRAIN TABLE

Command Statement: **YMST**

The **YMST** command can be used to describe, for a given material, the evolution of the Young modulus with strain. This evolution can be specified here in a curve (or one-dimensional table) defined by pairs of Young modulus and strain values. Linear interpolation is used for "in between" points, and the extrema values are adopted for "outside" points. Several curves can be specified, one after the other. Each curve is identified by an ID number as described below.

YMST
CURVE curve_id
Strain_1 YM_1
.
.
.
Strain_n YM_n

CURVE	Sub-command keyword (characters) that signals the definition of a Young modulus-strain curve (or one-dimensional table).
curve_id	ID number for the following curve (or one-dimensional table) (integer).
T_i	A specified strain value (real).
YM_i	A specified Young modulus value at strain Strain_i (real).

Next: [YSST](#), Previous: [YMST](#)

131 YOUNGS MODULUS-TEMPERATURE TABLE

Command Statement:	YMTT
--------------------	-------------

The **YMTT** command statement can be used to describe the evolution of Young's modulus with temperature, for a given material. This evolution can be specified here in a curve (or table) defined by pairs of temperature and Young's modulus values. Linear interpolation is used for "in between" points, and the extrema values are adopted for "outside" points. Several curves can be specified, one after the other. Each curve is identified by an "id number" as described below.

YMTT

CURVE curve_id T_1 YM_1 T_n YM_n
--

CURVE
curve_id "Id number" for the following curve (or table) (integer).
T_1 A specified temperature value (float).
YM_1 A specified Young's modulus value at temperature T_1 (float).

Next: [YSSFSRT](#), Previous: [YMTT](#)

132 YIELD STRESS-EFFECTIVE PLASTIC STRAIN TABLE

Command Statement:	YSST
--------------------	-------------

The **YSST** command statement can be used to describe the evolution of the yield stress with the effective plastic strain, for a given material. This evolution can be specified here in a curve (or table) defined by pairs of effective plastic strain and yield stress values. Linear interpolation is used for "in between" points, and *linear extrapolation* is used for "outside" points (at the "right"). Several curves can be specified, one after the other. Each curve is identified by an "id number" as described below.

YSST

CURVE curve_id EPS_1 YS_1 EPS_n YS_n
--

CURVE
curve_id "Id number" for the following curve (or table) (integer).
EPS_i A specified effective plastic strain value (float). The first point of the table should be characterized by EPS_1 = 0.
YS_i A specified yield stress value at the effective plastic strain EPS_i (float).

Previous: [YSST](#)

133 YIELD STRESS SCALING FACTOR-EFFECTIVE PLASTIC STRAIN RATE TABLE

Command Statement:	YSSFSRT
--------------------	----------------

The **YSSFSRT** command statement can be used to describe the evolution of the yield stress scaling factor with the effective plastic strain rate, for a given material. This evolution can be specified here in a curve (or table) defined by pairs of effective plastic strain rate and yield stress scaling factor values. Linear interpolation is used for "in between" points, and *linear extrapolation* is used for "outside" points (at the "right"). Several curves can be specified, one after the other. Each curve is identified by an "id number" as described below.

YSST

```
CURVE curve_id
EPSR_1 YSSF_1
.
.
.
EPSR_n YSSF_n
```

CURVE

curve_id

EPSR_1

YSSF_1

"Id number" for the following curve (or table) (integer).

A specified effective plastic strain rate value (float). The first point of the table should be characterized by `EPSR_1 = 0` and `yssf_1 = 1`.

A specified yield stress scaling factor value at the effective plastic strain `EPSR_1` (float). The first point of the table should be characterized by `EPSR_1 = 0` and `yssf_1 = 1`.