AERO-S Manual

Version 1.1

Edition 0.1/19 October 2017

Farhat Research Group (FRG)
Stanford University

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

Short Contents

Table of Contents

- 107 USER DEFINED FORCES *S*
- 108 USER DEFINED PRESCRIBED DISPLACEMENTS *S*
- 109 WEIGHTS
- 110 YOUNGS MODULUS-TEMPERATURE TABLE
- 111 YIELD STRESS-EFFECTIVE PLASTIC STRAIN TABLE
- 112 YIELD STRESS SCALING FACTOR-EFFECTIVE PLASTIC STRAIN RATE TABLE

Up: (dir)

AERO-S

- ATDARB
- ATDDIR
- ATDDNB
- ATDNEU
- ATDROB
- ACTUATORS
- AERO
- AEROH
- ATTRIBUTES
- BOFFSET
- EFRAMES
- BINARY
- CONVECTION
- DISPLACEMENTS
- FLUX
- FORCES
- TEMPERATURES
- BUCKLE
- COMPOSITE
- CFRAMES
- CONDITION
- CONSTRAINTS
- CONTACTSURFACES
- CONTROL
- CONWEP
- DECOMPOSE
- DIMASS
- DYNAMICS
- EIGEN
- FSINTERFACE
- FWEIGHTS
- MFTT
- GRBM
- GEPS
- GRAVITY
- GROUPS
- HZEM
- HZEMFILTER
- HEFSB
- HEFRS
- HELMHOLTZ
- HARB
- HDIR
- HDNB
- HNEU
- HSCB
- HWIB
- HLMPC
- KIRLOC
- IMPEDANCE
- IACCELERATIONS
- IDISPLACEMENTS
- IDISP6
- IDISP6PITA
- IVEL6PITA
- ITEMPERATURES
- IVELOCITIES
- INPC
- MATERIAL
- LMPC
- LOAD
- LOADCASE
- LOCROB

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.

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

[`-d <decomposition_pathandfilename>`]	(Specifies a decomposition file).	[`-v <verbose_frequency>`]	(Turns on verbose and specifies frequency of printing on screen the FETI iteration count and subspace iteration count).
[`-c`]	(Outputs contact status on screen (FETI solver)).		
[`-t`]	(Converts input file to **XPost** format).		
[`-T`]	(Converts input file to **XPost** format after removing all numbering gaps).		
[`-m`]	(Converts input file to **XPost** format after gathering each material in a separate element set).		
[`-M`]	(Converts input file to **XPost** format after removing all numbering gaps and gathering each material in a separate element set).		
[`-P`]	(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).		
`<filename.aeros.aicdf>`	(**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>`]	(Specifies a decomposition file).
[`-v <verbose_frequency>`]	(Turns on verbose and specifies frequency of printing on screen the FETI iteration count and subspace iteration count).
[`-c`]	(Outputs contact status on screen (FETI solver)).
[`-t`]	(Converts input file to **XPost** format).
[`-T`]	(Converts input file to **XPost** after removing all numbering gaps).
[`-m`]	(Converts input file to **XPost** format after gathering each material in a separate element set).
[`-M`]	(Converts input file to **XPost** format after removing all numbering gaps and gathering each material in a separate element set).
[`-P`]	(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).
`<filename.aeros.aicdf>`	(**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>`]	(Specifies a decomposition file).
[`-v <verbose_frequency>`]	(Turns on verbose and specifies frequency of printing on screen the FETI iteration count and subspace iteration count).
[`-c`]	(Outputs contact status on screen (FETI solver)).
[`-t`]	(Converts input file to **XPost** format).
[`-T`]	(Converts input file to **XPost** after removing all numbering gaps).
[`-m`]	(Converts input file to **XPost** format after gathering each material in a separate element set).
[`-M`]	(Converts input file to **XPost** format after removing all numbering gaps and gathering each material in a separate element set).
[`-P`]	(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).
`<filename.aeros.aicdf>`	(**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:

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

and following libraries:

`BLAS` library	`BLAS` is a set of Basic Linear Algebra Subprograms required by various operations performed in **AERO-S**.
`LAPACK` library	`LAPACK` is a high-performance Linear Algebra PACKage with advanced solvers.
`MPI` library `openmpi`	Version 1.2.6 or higher. `Open MPI` 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
Spooles:                  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	Non-zero constant premultiplying the normal derivative of the Robin boundary condition (real).
β	Constant premultiplying the unknown of the Robin boundary condition (float).
γ	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 nonlinear structural models, sensor information on the velocity and acceleration of a nodal degree of freedom is not currently available for use in the "control.C" file. These are currently passed as zero.

Note 3: 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} ($t^{n+\frac{1}{2}}$). 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 fraework 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 $\overline{P}^{n+1} = \frac{1}{\Delta t_S}\int_{t^n}^{t^{n+1}} P(t)dt$. 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 α0 α1
PRESSURE
MATCHER   pathandfilename
EMBEDDED  embeddedsurfaceid
```

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. 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. The READMODE command must be used to input the initial displacements of interest — for example, a set of eigenmodes. 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 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 $\overline{P}^n$ rather than the corresponding pressure fields at t^{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 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.

α_0 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.

α_1 Unlike α_0, this parameter cannot be used with the MPP command (real). It is part of the construction of a predictor for the position of the structure and therefore should be used only as such (see above). Its default value is 0.375 for algorithm C0, and 0 for all other algorithms.

PRESSURE This option, which can take either value COLLOCATED or NONCOLLOCAED, specifies how to compute the corrected pressure field. It is relevant only when ALGORITHM is set to A0, A4, A5, or A6.

COLLOCATED In this case, P_c^{n+1} is set to the received pressure field P^{n+1} (or $\overline{P}^{n+1}$), 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 $P_c^{n+1} = \overline{P}^{n+1}$, 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 $P_c^{n+1} = \frac{1}{\gamma}\overline{P}^{n+1} - \frac{(1-\gamma)}{\gamma}P_c^n$), 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 α method). This specific choice for P_c^{n+1} is consistent with the quadrature rule of the Generalized α 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

pathandfilename 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]

embeddedsurfaceid 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)

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

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

Note 2: The staggered solution algorithm A0 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 A0 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

A0 This is the basic sequential aerothermal partitioned solution algorithm. It is similar to the A0 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 *S*

Command Statement:	**ATTRIBUTES**

The ATTRIBUTES command statement is used to label an element with an attribute identification number for linear 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 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 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: When "empty" (or "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 2: 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 3: Only phantom shell elements and phantom plane stress/plane strain elements can transfer aeroelastic loads to a structure.

Note 4: 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.
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#    S1x S1y S1z    S2x S2y S2z    S3x S3y S3z
```

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 "*S*" 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 distribtued input files.

The syntax for this command is as follows.

```
BINARY
```

```
BINARYINPUT  flagBIN  [<pathandfileprefix>]
BINARYOUTPUT flagBOUT
```

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.

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 of 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# DOF#  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 Ta 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: COMMENTS, 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: COMPOSITE, Previous: BUCKLE

21 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

22 COMPOSITE (OR ORTHOTROPIC SHELL OR ANISOTROPIC SOLID)

Command Statement:	**COMPOSITE**

The command COMPOSITE is used to specify the material properties of a composite shell, orthotropic shell, anisotropic solid, anisotropic shell, or anisotropic membrane element. All information concerning the constitutive coefficients or the composite layering is inputted using this command.

Currently, this command 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 nonlinear membrane elements type 128 and 129.

For the above solid elements, the constitutive matrix is inputted using the sub-command COEF. This matrix is defined in the local frame of the element specified in the command CFRAMES. 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 (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 sub-command COEF should be used. In the second case, either of the sub-commands LAYC or LAYN 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 orthotropic material, the sub-commands LAYD and LAYMAT can be used instead of the sub-command LAYC to simplify the data entry process. Likewise, in similar circumstances, the sub-command LAYO and LAYMAT can be used instead of the sub-command LAYN.

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 sub-command COEF can be currently used only to define an anisotropic hyperelastic or hyper-viscoelastic maerial. Hence, it can be specified together with the following material laws: Linear, HenckyElastic, StVenantKirchhoff, ViscoLinearElastic, ViscoStVenantKirchhoff, PlaneStressLinear, and PlaneStressViscoStVenantKirchhoff (see MATLAW). In the case of the StVenantKirchhoff family of material laws, C relates the Green-Lagrange (engineering) strain and its conjugate stress, the second Piola-Kirchhoff stress. In the case of the HenckyElastic material law, it relates the Lagrangian Hencky (engineering) strain and its conjugate stress, the rotated Kirchhoff stress.

The input format for the COMPOSITE command 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 γxy . It can be omitted, in which case its default value is 0.
w_5	Thermal expansion coefficient associated with the strain γyz . It can be omitted, in which case its default value is 0.
w_6	Thermal expansion coefficient associated with the strain γxz . 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 (seeMATLAW), the constitutive matrix *C* relates the stresses to the engineering strains in the element's local frame system {x ; y ; z} as follows:

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

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:

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

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 sub-command keyword COEF 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 command MATERIAL 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 sub-command keywords LAYC or LAYN (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 sub-commands LAYC and LAYN.

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 sub-commands LAYD and LAYO.

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	$E_1^{(k)}$	$E_2^{(k)}$	$\nu_{12}^{(k)}$	$G_{12}^{(k)}$	$\mu_{1,12}^{(k)}$	$\mu_{2,12}^{(k)}$	$\rho^{(k)}$	$w_1^{(k)}$	$w_2^{(k)}$	$w_{12}^{(k)}$	
or											
LAYER_MATERIAL_ID	$E_1^{(k)}$	$E_2^{(k)}$	$\nu_{12}^{(k)}$	$G_{12}^{(k)}$	$\rho^{(k)}$						

LAYC, LAYD, LAYN, or LAYO	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 $\sigma_1^{(k)}$ = 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 $\tau_{12}{}^{(k)}$ = 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 $\tau_{12}{}^{(k)}$ = 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 command ATTRIBUTES, 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 command ATTRIBUTES, the reference vector is the vector obtained by rotating the directional edge connecting local nodes 1 and 2 of this element around its normal by an angle equal to θ_{Ref} . In both cases, this angle must be inputed in degrees.
$w_1^{(k)}$	Coefficient of thermal expansion in the direction $1^{(k)}$. It can be omitted, in which case its default value is 0.
$w_2^{(k)}$	Coefficient of thermal expansion in the direction $2^{(k)}$. It can be omitted, in which case its default value is 0.
$w_{12}^{(k)}$	Coefficient of thermal expansion associated with the in-plane shear strain. It can be omitted, in which case its default value is 0.
LAYER_MATERIAL_ID	Identifier of a set of material properties (integer).
LAYMAT	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 cois M. Hemez, provides a detailed description of the theory and implementation for the type-20 composite (orthotropic) shell element.

Next: CONDITION, Previous: COMPOSITE

23 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

24 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: CONTACTSURFACES, Previous: CONDITION

25 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 $\times\epsilon\times$ |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 $\underline{10}$. • 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 $\underline{10}$.

- `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.

`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 10^8 (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 10^8 (no default value is provided).

Next: CONTROL, Previous: CONSTRAINTS

26 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 `flagTDENFORCE` of the DYNAMICS object. For implicit computations, the discrete kinematic constraint equations are defined using **AERO-S**'s mortar method and enforced using the method specified in CONSTRAINTS or in `CONSTRAINT_METHOD` below.

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 `flagTDENFORCE` 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 `flagTDENFORCE` 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 `flagTDENFORCE` 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 `flagTDENFORCE` 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 `flagTDENFORCE` 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     CONSTRAINT_METHOD
```

For and explicit dynamic computation with the flag `flagTDENFORCE` 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     CONSTRAINT_METHOD
```

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	The Lagrange multiplier method.
	penalty beta	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 $$f_1 = \rho_1 c_1/(\rho_1 c_1 + \rho_2 c_2), \qquad f_2 = \rho_2 c_2/(\rho_1 c_1 + \rho_2 c_2)$$ 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).
DYANAMIC_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

27 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 *any* thermal (heat conduction) analysis involving *radiation*, 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 `FNAME.top` 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 `FNAME.top` outputted in the **XPost** format, when **AERO-S** is executed with the "-t" (or "-T") option. The default name is "elems".

Next: DECOMPOSE, Previous: CONTROL

28 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

29 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, two mesh partitioning strategies are available:

- A trivial, one-step strategy which 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 with the lowest ID number, element 2 to that with the second lowest ID number, ...
  - On the other hand, if NELES%NSUBS > 0, the first NELES%NSUBS subdomains get each one more element.

  It should be noted that :
  - This mesh partitioning strategy is not suitable for FETI solvers (see STATICS).
  - Instead, it is intended for parallel element sampling (see RMSHC).
  - For applications where the finite element model contains multiple disconnected components, each subdomain is allowed 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.
- A non-trivial, two-step strategy where during the first step, an initial mesh partition is generated using the Greedy algorithm, and during the second step, the subdomain aspect ratios of this partition are optimized for the benefit of the FETI (see STATICS) iterative solution algorithms. Unlike the trivial mesh partitioning strategy described above, this strategy:
  - Is most suitable for all FETI solvers (see STATICS).
  - For applications where the finite element model contains multiple disconnected components, each subdomain is allowed to contain elements from the same component only.
  - It 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.

The syntax of this input file 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] <filename.aeros.aicdf>`

where `--dec` requests a mesh decomposition, `--nsub` specifies the number of subdomains, `--trivial` specifies the trivial mesh partitioning strategy (otherwise, the non-trivial strategy is automatically chosen), and the meaning of the other arguments can be easily determined from the explanations of the syntax of the DECOMPOSE input file 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** input file, the value of any argument specified on the command line overrides that specified in the input 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 subsystem.

DECOMPOSE

mediumskip

```
TRIVIAL
NSUBS   nsubs
DETER
FSGL
OUTFILE decomposition_pathandfilename
OUTLOAD
OUTMEM
EXIT
SKIP
```

TRIVIAL	Sub-command keyword to request the trivial mesh partitioning strategy. In the absence of this request, the non-trivial mesh partitioning strategy is automatically chosen (characters).
NSUBS	Sub-command keyword for specifying the desired number of subdomains (characters).
nsubs	Number of subdomains (integer). The generated mesh partition is guaranteed to contain nsubs subdomains only if the TRIVIAL mesh partitioning strategy is chosen (see above). Otherwise, the generated mesh partition may contain a slightly different number of subdomains because of some desired optimizations that are performed by the non-trivial mesh partitioning strategy. The default value is 1.
DETER	This sub-command keyword is relevant only for the non-trivial mesh partitioning strategy whose second step is performed by default using a simulated annealing procedure. It requests using instead a deterministic approach in order to reduce the CPU time (characters).
FSGL	This sub-command keyword is relevant only for the non-trivial mesh partitioning strategy and vibro-acoustic (aka elastoacoustic and fluid-structure) analysis (for example, see IMPEDANCE). 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	Sub-command keyword to request outputting the decomposition in <decomposition_pathandfilename> (characters).
<decomposition_pathandfilename>	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 FNAME.optDec where FNAME is the prefix specified under the command CONTROL or its own default value.
OUTLOAD	This sub-command keyword is relevant only for the non-trivial mesh partitioning strategy. It requests outputting the load distribution in the file <decomposition_pathandfilename>.load (characters). The load distribution is based on the weights of the elements in the subdomains (see WEIGHTS and FWEIGHTS). The default value is off.
OUTMEM	This sub-command keyword is relevant only for the non-trivial mesh partitioning strategy, and when a FETI iterative algorithm is chosen as an equation solver (see STATICS). It requests outputting in the file <decomposition_pathandfilename>.mem an estimate of the subdomain-based memory consumption (characters). 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. Default value is off.
SKIP	Sub-command keyword to request skipping this command. Default value is off.

Next: DYNAMICS, Previous: DECOMPOSE

30 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 DOF2# (integer). Note that only the contribution to the lower triangular part of the mass matrix should be specified, and therefore DOF1# should be greater than DOF2#. 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 DOF1# (integer). Note that only the contribution to the lower triangular part of the mass matrix should be specified, and therefore DOF2# should be smaller than DOF1#. 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

31 DYNAMIC ANALYSIS

Command Statement:	**DYNAMICS**

The DYNAMICS command is used to specify a time-integrator for a desired dynamic (transient, time-domain) structural, acoustic, or heat transfer (thermal) analysis and other relevant parameters. Currently, the following time-integrators are available in **AERO-S**:

1. The implicit generalized Ω method and the explicit central difference method for linear and nonlinear (in this case, the NONLINEAR command must also be present in the input file) structural dynamic analyses.
2. The implicit generalized Ω 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 heat transfer analysis.
4. The implicit midpoint rule for nonlinear heat transfer analysis.

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

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

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

where

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

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

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

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

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

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

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

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

$$\ddot{u}^0 = M^{-1}(F^0_{external} - F_{internal}(u^0) - D\dot{u}^0)$$
$$u^{n+1} = u^n + \Delta t\dot{u}^n + \Delta t^2\left(\left(\tfrac{1}{2}-\beta\right)\ddot{u}^n + \beta\ddot{u}^{n+1}\right)$$
$$\dot{u}^{n+1} = \dot{u}^n + \Delta t\left((1-\gamma)\ddot{u}^n + \gamma\ddot{u}^{n+1}\right)$$

where $\beta \neq 0$. The Newmark method is obtained by setting $\alpha_m = \alpha_f = 0$. The midpoint rule is obtained by setting $\alpha_m = \alpha_f = \frac{1}{2}$, $\beta = \frac{1}{4}$ and $\gamma = \frac{1}{2}$. The *classical* central difference method is obtained by setting $\alpha_m = \alpha_f = 0$, $\beta = 0$ and $\gamma = \frac{1}{2}$.

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

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

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

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

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

where

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

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

$$\ddot{u}^0 = M^{-1}(F^0_{external} - F_{internal}(u^0) - D\dot{u}^0)$$
$$u^{n+1} = u^n + \Delta t\dot{u}^n + \tfrac{\Delta t^2}{2}\ddot{u}^n$$
$$\dot{u}^{n+1} = \dot{u}^n + \tfrac{\Delta t}{2}(\ddot{u}^n + \ddot{u}^{n+1})$$

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

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

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

$$M\ddot{u}^{n+1} + D\dot{u}^{n+1} + (K - \tfrac{\Delta t^2}{q}KM^{-1}K)u^{n+1} = F^{n+1}$$

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

$$\ddot{u}^0 = M^{-1}\left(F^0 - D\dot{u}^0 - K(u^0 - \tfrac{\Delta t^2}{q}M^{-1}Ku^0)\right)$$
$$\dot{u}^{\frac{1}{2}} = \dot{u}^0 + \tfrac{\Delta t}{2}\ddot{u}^0$$
$$u^{n+1} = u^n + \Delta t\dot{u}^{n+\frac{1}{2}}$$
$$G^{n+1} = K\left(I - \tfrac{\Delta t^2}{q}(M + \tfrac{\Delta t}{2}D)^{-1}K\right)u^{n+1}$$

$$\dot{u}^{n+\frac{3}{2}} = (M + \tfrac{\Delta t}{2}D)^{-1}\left(M\dot{u}^{n+\frac{1}{2}} + \tfrac{\Delta t}{2}(2F^{n+1} - 2G^{n+1} - D\dot{u}^{n+\frac{1}{2}})\right)$$

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 heat transfer analysis enforces the equation of equilibrium
$$Q\dot{u}^{n+\frac{1}{2}} + Hu^{n+\frac{1}{2}} = F^{n+\frac{1}{2}}$$
where
$$u^{n+\frac{1}{2}} = \tfrac{1}{2}(u^{n+1} + u^n)$$
$$\dot{u}^{n+\frac{1}{2}} = \tfrac{1}{2}(\dot{u}^{n+1} + \dot{u}^n)$$
and solves it as follows
$$\dot{u}^0 = Q^{-1}(F^0 - Hu^0)$$
$$\dot{u}^{n+\alpha} = \alpha\dot{u}^{n+1} + (1-\alpha)\dot{u}^n$$
$$u^{n+1} = u^n + \Delta t\,(\alpha\dot{u}^{n+1} + (1-\alpha)\dot{u}^n)$$
The generalized midpoint family of methods outlined above is unconditionally unstable for $\alpha > \frac{1}{2}$. Hence, only the choices satisfying $\alpha \leq \frac{1}{2}$ 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 $\alpha = \frac{1}{2}$.

Nonlinear Heat Transfer Analysis

- The implicit midpoint rule for nonlinear heat transfer analysis enforces the following equation of equilibrium
$$Q\dot{u}^{n+\frac{1}{2}} + F_{internal}(u^{n+\frac{1}{2}}) = F^{n+\frac{1}{2}}_{external}$$
where
$$u^{n+\frac{1}{2}} = \tfrac{1}{2}(u^{n+1} + u^n)$$
$$\dot{u}^{n+\frac{1}{2}} = \tfrac{1}{2}(\dot{u}^{n+1} + \dot{u}^n)$$
Given $F_{external}(t)$ and u^0, it solves the above equation as follows
$$\dot{u}^0 = Q^{-1}\left(F^0_{external} - F_{internal}(u^0)\right)$$
$$u^{n+1} = u^n + \tfrac{\Delta t}{2}(\dot{u}^{n+1} + \dot{u}^n)$$

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 (or moment-of-interia-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 heat transfer 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
MODAL					
SROM	rob_id_1	[rob_id_2 ...		rob_id_L]	[CONTACT rob_id_L+1]
MECH	β	γ			(midpoint implementation of Newmark — that is, $\alpha_m = \alpha_f = \frac{1}{2}$)
MECH	β	γ	α_f	α_m	(generalized α)
MECH	ρ_∞				(generalized α)
ACOU	β	γ			(midpoint implementation of Newmark — that is, $\alpha_m = \alpha_f = \frac{1}{2}$)
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				

IACC — For linear and nonlinear explicit dynamic computations, and for nonlinear implicit dynamic computations, the initial acceleration is always computed to satisfy equilibrium at $t = 0$. For linear implicit dynamic computations, the initial acceleration is always set to zero if a FETI solver is specified under the STATICS command; otherwise, the user can request to have it either set to zero or computed to satisfy equilibrium at $t = 0$. IACC is the sub-command keyword that enables the user to make this choice in the latter case (characters).

flagIACC — On/Off flag (characters). The default value is On.

on — In this case, for linear implicit dynamic computations, as long as a FETI method is not chosen as the equation solver under the STATICS command, the acceleration is initialized to satisfy equilibrium at $t = 0$. This initialization requires the factorization of the mass matrix. The sparse direct solver is always used for this purpose, no matter which direct equation solver is specified under the STATICS command. If for some reason the finite element model has one or several massless or moment-of-inertia-less degrees of freedom, the sparse solver will typically encounter zero pivots and set the initial acceleration to zero at the corresponding degrees of freedom.

off — In this case, for linear implicit dynamic computations, the initial acceleration is set to zero.

STABLE — Sub-command keyword for managing the time-step of a linear or nonlinear, structural dynamic or acoustic time-integration using the 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 (float). 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 .
`MODAL`	Sub-command keyword for requesting a *linear* structural dynamic analysis using modal superposition (based on mass-orthonormalized eigen vectors) instead of direct time-integration (characters). **In this case, the Reduced-Order Basis (ROB) must be inputted using the first format of the READMODE** command, and its type must be `eigen`. 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`	Sub-command keyword (characters) for:
	• Requesting the construction offline of a structural dynamic ROM based on:
	  ◦ A primal ROB that 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 that must also be inputted using the *second format* of the READMODE command and identified by `rob_id_L+1` (see below).
	• Performing the requested structural dynamic analysis online using this structural dynamic ROM.
	• Reconstructing the high-dimensional solution offline using RODC, the Reduced-Order Basis (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 a mass-orthonormalized primal structural ROB can be used for both explicit and implicit dynamic computations using the corresponding SROM, but 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 a mass-orthonormalized primal structural ROB can be used for both explicit and implicit dynamic computations using the corresponding SROM, but 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 contact problems (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.
`MECH` or `ACOU` or `HEAT`	The sub-command keyword `MECH` signals a structural dynamic analysis (characters). The sub-command keyword `ACOU` signals a time-domain acoustic analysis (characters). The sub-command keyword `HEAT` signals a thermal analysis (characters).
β	Newmark coefficient β for a second-order system (real). Set $\beta = 0$ ($\alpha_m = \alpha_f = 0$) and $\gamma = \frac{1}{2}$ to specify the explicit central difference method.
γ	Newmark coefficient γ for a second-order system (real).
α_f	Generalized α coefficient α_f for a second-order system (real). To obtain the standard Newmark algorithm, set $\alpha_f = 0$. The default is $\alpha_f = \frac{1}{2}$.
α_m	Generalized α coefficient α_m for a second-order system (real). To obtain the standard Newmark algorithm, set $\alpha_m = 0$. The default is $\alpha_m = \frac{1}{2}$.
ρ_∞	Generalized α infinite frequency spectral radius for a second-order system (real). When this option is used and this coefficient is specified, 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.

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 that preceds 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 $\alpha \leq \frac{1}{2}$.
TIME	Sub-command keyword for specifying time-interval parameters (characters).
TH	Integration time-step for heat transfer 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 when the MECH option is used. It specifies *default* values of the Rayleigh proportional damping coefficients for the entire structure (characters). These default coefficients can be overwritten at the element level by counterpart coefficients specified using the MATERIAL command.
a	Rayleigh damping stiffness coefficient a (real). In the nonlinear case, this coefficient is assigned at each iteration to the *initial* linearized stiffness matrix and the product of these two quantities, which therefore remains constant throughout the nonlinear iterations, constitutes the contribution to the damping term.
b	Rayleigh damping mass coefficient b (real).
MODDAMP	Sub-command keywords for signaling that the following lines specify modal damping ratios (characters). This option is active only when the MECH and MODAL sub-commands are also specified — that is, for a dynamic analysis of a mechanical system using mode superposition. If a mode is repeated, the modal damping ratio values are added for that mode. If both of the 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 value (ξ) for mode MODE# (real).
TDENFORCE	For explicit dynamic computations with TIEDSURFACES and/or CONTACTSURFACES constraints, 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).
flagTDENFORCE	On/Off flag (characters). The default value is On.
On	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.
Off	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.

Next: END, Previous: DYNAMICS

32 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 Ω^2 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 — or in the shifted case $A - \sigma B$ — is singular, and the equation solver specified in STATICS is sparse, skyline, mumps pivot, or FETI DP, the null space computed by this equation solver is exploited by the eigen solver described herein in the solution of the above generalized eigenvalue problem.

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 — Value of a specified shift for the A matrix (real).

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 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 10× nsbspv.

Next: FSINTERFACE, Previous: EIGEN

33 END

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

Next: FWEIGHTS, Previous: END

34 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: MFTT, Previous: FSINTERFACE

35 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

$$\frac{w_e \times w_{field\_of\_e}}{\sum_{involved fields} w_{field}}$$

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 acou_weight
FLUI flui_weight
MECH mech_weight
HEAT heat_weight
```

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: GRBM, Previous: FWEIGHTS

36 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 — A specified time point (float).

AMP_1 — 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

37 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 use 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.

```
GRBM     [ X Y Z ]
```

```
VALUE_1          VALUE_2          VALUE_3
```

or

```
GRBM     [ X Y Z ]
```

```
VALUE_2
```

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.

`VALUE_1` 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.

`VALUE_2` 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.

`VALUE_3` Optional integer parameter for the case where the problem contains LMPCs. It can take either of the following two values:

- `VALUE_3` = 0. 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:
  - The `GRBM` 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 `sparse` and `skyline` (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 `VALUE_2`. On the other hand, if `GRBM` 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 `VALUE_3` should be used with care — that is, when the user has sufficient insight in the problem to be solved to conclude that `GRBM` 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: GRAVITY, Previous: GRBM

38 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: GROUPS, Previous: GEPS

39 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: HZEM, Previous: GRAVITY

40 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: HZEMFILTER, Previous: GROUPS

41 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.

```
HZEM
```

Next: HEFSB, Previous: HZEM

42 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.

HZEMFILTER

Next: HEFRS, Previous: HZEMFILTER

43 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: HELMHOLTZ, Previous: HEFSB

44 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: HARB, Previous: HEFRS

45 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 $P = pe^{-i\omega t}$, 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 souce 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          i1   r1   r2
POINTSOURCE   ns
              xs   ys   zs
PLANEWAVE     nv
              vx   vy   vz
              .    .    .
              .    .    .
```

. . .
v_x v_y v_z

BGTL

i_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).

r_1 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 Keyword indicating that the Dirichlet and/or Neumann boundary conditions (see HDIR and HDNB) of the acoustic (Helmholtz) or elastoacoustic problem defined by this command are associated with a time-harmonic spherical wave of the form $\frac{1}{4\pi}\frac{e^{ikr}}{r}$, where $r=|X-X_s|$, and propagated by a point 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 HDNB), this option is available only for the faces of type 10 and 11 (see HDNB).

n_s Number of point sources propagating spherical waves (integer).

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

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

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

PLANEWAVE Keyword indicating that the Dirichlet and/or Neumann boundary conditions (see HDIR and HDNB) of the acoustic (Helmholtz) or elastoacoustic problem defined by this command are associated with an incident time-harmonic plane wave of the form $e^{ik\vec{x}\cdot\vec{v}}$ (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_v, and n_v lines specifying for each wave its direction of propagation v.

n_v Number of incident planar waves (integer).

v_x Normalized x component of the direction of a planar wave (float).

v_y Normalized y component of the direction of a planar wave (float).

v_z Normalized z component of the direction of a planar wave (float).

Next: HDIR, Previous: HELMHOLTZ

46 HELMHOLTZ ARTIFICIAL BOUNDARY *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# 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 **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 quadrilaterial 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: HDNB, Previous: HARB

47 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: HNEU, Previous: HDIR

48 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 quadrilaterial 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: HSCB, Previous: HDNB

49 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 keyword 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: HWIB, Previous: HNEU

50 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 quadrilaterial 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: HLMPC, Previous: HSCB

51 HELMHOLTZ WET INTERFACE BOUNDARY *S*

Command Statement: **HWIB**

The HWIB command satement can be used to describe the wet interface boundary of the structure in a coupled frequency response elastoacoutic (fluid-structure) problem where a single mesh is used to discretize both of the computional 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 quadrilaterial 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: KIRLOC, Previous: HWIB

52 HELMHOLTZ LINEAR MULTIPOINT CONSTRAINTS

Command Statement: **HLMPC**

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

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

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).

HLMPC

```
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* (integer).

RHS This is the right-hand side r_i of the *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.
- 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]

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).

REAL_PART_VALUE	This is the real part of the coefficient c_{ij} of the i-th constraint equation (float).
IMAGINARY_PART_VALUE	This is the imaginary part of the coefficient c_{ij} of the i-th constraint equation (float).

Next: IMPEDANCE, Previous: HLMPC

53 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: IACCELERATIONS, Previous: KIRLOC

54 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,

$$\begin{bmatrix} K_s - i(2\pi f_j)D_s - (2\pi f_j)^2 M_s & C^T \\ (2\pi f_j)^2 C & \frac{1}{\rho_f}K_f - \frac{(2\pi f_j)^2}{\rho_f c_f^2}M_f - \frac{1}{\rho_f}S_a(2\pi f_j) \end{bmatrix} \begin{bmatrix} u_s \\ p_f \end{bmatrix} = \begin{bmatrix} g_s((2\pi f_j), \varphi_k) \\ \frac{1}{\rho_f} g_f((2\pi f_j), \varphi_k) \end{bmatrix}, \qquad j = 1, 2, \cdots, \qquad k = 1, 2, \cdots$$

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 thererefore 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 η_E or 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_{Δ_f}

FREQSWEEP f_l f_u n_s^* n_s

FREQSWEEP f_l

f_j^* n_j

$\vdots$ $\vdots$

FREQSWEEPA f_l f_u n_s IMOR $[\epsilon$ n_s^{*max} n_v^{min} n_v^{max} $\Delta n_v]$

FREQSWEEPAW f_l f_u n_s IMOR $[\epsilon$ localflag n_v tol tole1f]

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).

FREQSWEEP1

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 $[f_0-n_{\Delta_f}\Delta_f,\ f_0+n_{\Delta_f}\Delta_f]$

$$f_j = f_0 + j\Delta_f, \quad -n_{\Delta_f} \le j \le n_{\Delta_f}$$

This sub-command does not support frequency sweep problems with multiple right hand-sides.

f_0

Frequency defining the center of the frequency band $[f_0-n_{\Delta_f}\Delta_f,\ f_0+n_{\Delta_f}\Delta_f]$ (real).

Δ_f

Frequency sweep increment (real).

n_{Δ_f}

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).

FREQSWEEP

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 $n_s^* \ge 2$ equally-spaced

interpolation frequencies f_j^* in $[f_l,\ f_u]$, including f_l and f_u . Hence, these frequencies are given by

$$f_j^* = f_l + (j-1)\frac{(f_u - f_l)}{(n_s^* - 1)}, \quad j = 1, \ldots n_s^*$$

Then, the regular scheme samples each interval $[f_{j-1}^\star,\ f_j^\star]$ into $n_s \geq 1$ equal frequency increments, and therefore into (n_s+1) frequency points including $f_{j-1}^\star$ and $f_j^\star$, where it rapidly reconstructs the frequency response function using the algorithm `alg`. Hence, the first scheme computes the frequency response at $((n_s^\star-1)n_s+1)$ 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 `FREQSWEEP` the lower end of the frequency spectrum, f_l, and then every *other* desired interpolation frequency $f_j^\star$ (in ascending order) on a separate line together with an integer number n_j specifying the sampling of the interval $[f_{j-1}^\star,\ f_j^\star]$ into n_j equal frequency increments. Hence, it defines (n_j+1) sampling frequencies in $[f_{j-1}^\star,\ f_j^\star]$ including $f_{j-1}^\star$ and $f_j^\star$. The upper end of the frequency spectrum, f_u, is in this case the last inputted interpolation frequency. It follows that if, for example, $n_s^\star$ 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 $((n_s^\star-1)n_s+1)$ 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 `alg` below).

f_l Lower end of the frequency band of interest $[f_l,\ f_u]$ (real).

f_u Upper end of the frequency band of interest $[f_l,\ f_u]$ (real).

$n_s^\star$ Number of equally-spaced interpolation frequencies $f_j^\star$ 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

$$f_j^\star = f_l + (j-1)\frac{(f_u - f_l)}{(n_s^\star - 1)}, \quad j = 1, \ldots n_s^\star$$

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" $[f_{j-1}^\star,\ f_j^\star]$ — where $f_{j-1}^\star$ and $f_j^\star$ are interpolating frequencies (see above) — is to be sampled (integer). In this case, $[f_{j-1}^\star,\ f_j^\star]$ is sampled into (n_s+1) frequencies, including $f_{j-1}^\star$ and $f_j^\star$. When part of the definition of `FREQSWEEPA` or `FREQSWEEPAW`, 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 `FREQSWEEPA` or `FREQSWEEPAW`, it must satisfy $n_s \geq 2$. In both cases, the frequency response function is computed at all sampled frequency points.

$f_j^\star$ Interpolation frequency (real).

n_j Number of equal size frequency increments in which a frequency interval of the form $[f_{j-1}^\star,\ f_j^\star]$ is to be sampled (integer). In this case, $[f_{j-1}^\star,\ f_j^\star]$ is sampled into (n_j+1) frequency points.

`FREQSWEEPA` 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, specifiy 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.

`FREQSWEEPAW` 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.

IMOR

Speficies an interpolatory model order reduction algorithm for the adaptive frequency sweep procedures associated with the sub-command keywords FREQSWEEPA and FREQSWEEPAW (characters). Currently, three options are available: (1) KrylovGalProjection, (2) GalProjection, and (3) WCAWEGalProjection. 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.

ϵ

Specifies the tolerance level for assessing the convergence of the adaptive frequency sweep procedures associated with the sub-command keyword FREQSWEEPA and FREQSWEEPAW 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 $\epsilon = 10^{-2}$.

n_s^{*max}

Specifies the maximum number of interpolation frequencies for the adaptive frequency sweep procedure associated with the sub-command keyword FREQSWEEPA. 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 FREQSWEEPA (integer). The recommended value is the default value 8.

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 FREQSWEEPA (integer). The recommended values are 48 for the KrylovGalProjection algorithm, and 16 - 24 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 FREQSWEEPA (integer). The recommended value is the default value 4.

localflag

This integer parameter can be be set either to 0 or to 1. It controls the functioning of the adaptive frequency sweep with a windowing strategy command FREQSWEEPAW 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 FREQSWEEPAW (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 FREQSWEEPAW. A suitable value is such that tol*epsilon $\leq 10^{-1}$ (real).

tol1f

Tolerance for controling the global-within-one-frequency algorithm (localflag = 0) that attempts to ensure that residual at the interpolation frequency is less than tol*tol1f (real). A suitable value is 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 *undamped* 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.
WCAWEGalProjection	Interpolatory model order reduction algorithm based on the combination of the Well-Conditioned Asymptotic Waveform Evaluation (WCAWE) scheme, and the multipoint Galerkin projection interpolatory scheme with orthogonalized vectors. This algorithm is also known as the WCAWE+DGP algorithm. It is 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 $n_s^\star$ 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 = $n_s^\star$ are currently the only options).
para_2	This parameter is relevant only for the Pade, GalProjection, and KrylovGalProjection reconstruction algorithms (integer). For Pade, it specifies the order of the numerator of the rational function (L of [L/M]) (integer). For GalProjection, it specifies the number of frequency derivatives of the response to compute at each interpolation frequency. For KrylovGalProjection, it specifies the number of Krylov vectors to compute at each interpolation frequency.
para_3	This parameter is relevant only for the Pade and PadeLanczos reconstruction algorithms. It specifies the order of the denominator of the rational function (M of [L/M]) (integer).
PADEPOLES	This sub-command keyword is active only with alg = PadeLanczos and needs input from the sub-command FREQSWEEP (see above) (characters). It also requires the presence in the input file of the MODE 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 MODE 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 FREQSWEEP 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 para_1, para_2 and para_3 (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

55 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

56 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 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.

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.

For nodal displacement dofs, the format of this command is as follows.

```
IDISPLACEMENTS
```

or

```
IDISPLACEMENTS ZERO
```

```
NODE#          DOF#          VALUE
```

NODE#	Node number where the initial displacement or acoustic field is specified (integer).
DOF#	Local identification number of the degree of freedom where the initial displacement or acoustic field is specified (integer).
VALUE	Value of the specified initial displacement or acoustic field (real).
ZERO	This initializes the entire intital displacement or acoustic field to zero. No other IDISPLACEMENTS 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 Reduced-Order Basis (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. 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 *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 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

57 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 IDISP 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 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.

IDISP6

NODE#	VAL_DOF1	VAL_DOF2	VAL_DOF3	VAL_DOF4	VAL_DOF5	VAL_DOF6

	Amplification factor that multiplies each VAL_DOFi value for each node (real).
NODE#	Node number where the initial displacement is specified (integer).

VAL_DOFi#	Value of the specified initial displacement for the i-th degree of freedom (real).

Next: IVEL6PITA, Previous: IDISP6

58 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

59 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

60 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

61 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.

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

62 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

63 LINEAR MATERIAL PROPERTIES *S*

Command Statement:	**MATERIAL**

The MATERIAL command statement is used to signal that the following data lines correspond **and/or point to**: (1) definitions of materials and/or geometric properties for rigid elements and linear elastic or acoustic elements, and/or geometric properties for materially nonlinear elements, and/or (2) requests for specific methods to enforce the constraints associated with joint and rigid elements (see CONSTRAINTS). Several input formats are possible for this command and are described below.

Note 1: 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 2: 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 3: 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 4: The Perfectly Matched Layer (PML) method is supported only by the elements type 95, 96, 98, and 99.

Note 5: 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	ν	ρ	h	k	t	P	Ta	c_p	w	Ixx	Iyy	Izz	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).

ν — 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 LAYD 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).

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).

w — Coefficient of thermal expansion in either a thermal analysis, or a structural analysis if the specified reference temperature Ta 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 *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. In this case, damping is introduced at the element level in the form of a finite element damping matrix C^e that is constructed as

$$D^e = aK^e + bM^e$$

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 specfied below.

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

$$E^\star = E(1 - i\eta_E)$$

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 *loss factor*. 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,

$$\eta_E(f) = \eta_{E_1} + \eta_{E_2} f$$

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

$$E^\star = E(1 - i\eta_E)$$
$$\mu^\star = \mu(1 - i\eta_\mu)$$

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, $b = \eta_\mu$.
- 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 *stiffness* 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 speficied 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 speficied 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 speficied 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 speficied 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 speficied 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 speficied 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     αY     αZ     C1     P     Ta     cp     w     Ixx     Iyy     Izz     ymin     ymax     zmin     zmax
DAMPING_TYPE     a     b
```

MID	The material id number from element attribute table.
A	Cross sectional area (float).
E	Young's Modulus (float).
ν	Poisson's ratio (float).
ρ	Mass density per unit volume (float).
α_Y	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. 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/wiki/Timoshenko_beam_theory#Shear_coefficient and used by other finite element codes such as NASTRAN.
α_Z	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/wiki/Timoshenko_beam_theory#Shear_coefficient and used by other finite element codes such as NASTRAN.
C1	Non-uniform torsion constant (float).
P	Not Applicable
Ta	Not Applicable
c_p	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  dum  dum  dum  dum  dum  dum  dum  dum  dum  dum  dum  dum  dum  dum
```

MID	The material id number from element attribute table.
x	Any nonzero value implies a rigid motion in this direction (float).
y	Any nonzero value implies a rigid motion in this direction (float).
z	Any nonzero value implies a rigid motion in this direction (float).
dum	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 contraint 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.
	penalty	The penalty method. The real-valued parameter beta should be a large positive number, typically of the order
beta		of 10^8 (no default value is provided).
	augmented	The augmented Lagrangian method. The real-valued parameter beta should be a large positive number,
beta		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: $$k^{eff} = 0 \quad \text{for} \quad -\infty \leq \delta \leq freeplay\_ul$$ $$k^{eff} = k \quad \text{for} \quad freeplay\_ul \leq \delta$$ 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
 factor_uz3
```

	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.
	penalty	The penalty method. The real-valued parameter beta should be a large positive number, typically of the order
beta		of 10^8 (no default value is provided).
	augmented	The augmented Lagrangian method. The real-valued parameter beta should be a large positive number,
beta		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 k1 is the stiffness of the first embedded spring, k2 that of the second embedded spring when applicable, and k3 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, or 3), freeplay is modeled using 5 parameters:

- A pair of lower and upper deflection limits freeplay_lli and freeplay_uli 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

$$k_i^{eff} = factor\_lz_i \times k_i \quad \text{for} \quad \delta_i \leq freeplay\_ll_i$$

$$k_i^{eff} = factor\_dz_i \times k_i \quad \text{for} \quad freeplay\_ll_i \leq \delta_i \leq freeplay\_ul_i$$

$$k_i^{eff} = factor\_uz_i \times k_i \quad \text{for} \quad freeplay\_ul_i \leq \delta_i$$

where k_i is the stiffness coefficient ki 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 k1.
- 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 k1 (k2, or k3) for $\delta_1 \leq freeplay\_ll_1$ ($\delta_2 \leq freeplay\_ll_2$, or $\delta_3 \leq freeplay\_ll_3$), 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 k1 (k2, or k3) for $freeplay\_ll_1 \leq \delta_1 \leq freeplay\_ul_1$ ($freeplay\_ll_2 \leq \delta_2 \leq freeplay\_ul_2$, or $freeplay\_ll_3 \leq \delta_3 \leq freeplay\_ul_3$), 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 k1 (k2, or k3) for $freeplay\_ul_1 \leq \delta_1$ ($freeplay\_ul_2 \leq \delta_2$, or $freeplay\_ul_3 \leq \delta_3$), 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 $g(t)$ = SCALE_FACTOR* $f(t)$ + 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 belowassociated with the revolute or primsatic 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.
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).
FUNCTION_TYPE	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:
SINE	Sets $\theta(t)$ / $\mathcal{M}(t)$ / $u(t)$ / $\mathcal{F}(t)$ 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.
RAMP	Sets $\theta(t)$ / $\mathcal{M}(t)$ / $u(t)$ / $\mathcal{F}(t)$ to a time-dependent relative rotation/applied moment/relative translation/applied force partially described by a bounded ramp function of the form $f(t) = 0$ for $t \le a$, $f(t) = \frac{t-a}{b-a}$ for $a \le t \le b$, and $f(t) = 1$ for $t \ge b$ where a and b are two parameters specified below.
TRIA	Sets $\theta(t)$ / $\mathcal{M}(t)$ / $u(t)$ / $\mathcal{F}(t)$ to a time-dependent relative rotation/applied moment/relative translation/applied force partially described by a triangular (hat) function of the form $f(t) = 0$ for $t \le a$, $f(t) = \frac{t-a}{b-a}$ for $a \le t \le b$, $f(t) = \frac{c-t}{c-b}$ for $b \le t \le c$, and $f(t) = 0$ for $t \ge c$ where a , b and c are three parameters specified below.
TRAP	Sets $\theta(t)/\mathcal{M}(t)$ / $u(t)$ / $\mathcal{F}(t)$ 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 $t \le a$, $f(t) = \frac{t-a}{b-a}$ for $a \le t \le b$, $f(t) = 1$ for $b \le t \le c$, $f(t) = \frac{d-t}{d-c}$ for $c \le t \le d$, and $f(t) = 0$ for $t \ge d$ where a , b , c and d are four parameters specified below.
SSHA	Sets $\theta(t)$ / $\mathcal{M}(t)$ / $u(t)$ / $\mathcal{F}(t)$ 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 $t \le a$, $f(t) = \frac{(t-a)^2}{2(b-a)^2}$ for $a \le t \le b$, $f(t) = 1 - \frac{(c-t)^2}{2(c-b)^2}$ for $b \le t \le c$, and $f(t) = 1$ for $t \ge c$ 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.
σ	Stefan's constant, also known as the Stefan-Boltzmann constant (in SI units, $\sigma = 5.670400 \times 10^{-8}$ Js $^{-1}$ m $^{-2}$ 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	c	ρ	pml_type	γ	pmlx1	pmlx2	pmly1	pmly2	pmlz1	pmlz2

or

MID	AMAT	c_R	c_I	ρ	pml_type	γ	pmlx1	pmlx2	pmly1	pmly2	pmlz1	pmlz2

MID	The material id number from element attribute table.
AMAT	Keyword that specifies that the following data entries are associated with acoustic fluid elements.
c	Speed of sound in the material identified by MID (float).
c_R	Real part of the speed of sound in the material identified by MID (float).
c_I	Imaginary part of the speed of sound in the material identified by MID (float).
ρ	Density (mass per unit volume) of the material identified by MID (float).
pml_type	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*x+y*y+z*z). 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*x+y*y), pmlz1 < z < pmlz2, and -pmlz2 < z < -pmlz1.
γ	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 $C(\gamma) = (1 - \frac{1}{\gamma})\frac{e^\gamma}{\gamma} + \frac{1}{\gamma^2} - \frac{1}{2}$. 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	PML geometrical parameter (see above) (real).
pmlx2	PML geometrical parameter (see above) (real).
pmly1	PML geometrical parameter (see above) (real).
pmly2	PML geometrical parameter (see above) (real).
pmlz1	PML geometrical parameter (see above) (real).
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	E	ρ	A	U_c	U_f	λ	h	d	$\delta\lambda$	n_p	N_f

MID	The material id number from element attribute table (integer).
FABMAT	Keyword specifying that the following data entries are associated with fabric elements (characters).
type	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.
E	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.
ρ	Mass density per unit volume of the fibril/yarn (float).
A	Cross sectional area of the entire yarn element (float).
U_c	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.
U_f	Stretch level at which the entire yarn has failed (i.e. the stress response is approximately zero) (float).
λ	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.
h	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.
d	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

64 LINEAR MULTIPOINT 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

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

or

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

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
```

```
CONSTRAINT# 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 contraint 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.
penalty [beta]	The penalty method. The parameter beta 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

65 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: LOCROB, Previous: LOAD

66 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.
LOADSET_ID_1	Integer identifying a "load" set.
COEFF_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].

Next: LUMPED, Previous: LOADCASE

67 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
                  <pathandfilename1>
AUXI
                  <pathandfilename2>
```

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.
<pathandfilename1>	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 *online* selection of the appropriate local ROB during a nonlinear structural dynamic computation performed using a hyper reduced computational model (characters).
<pathandfilename2>	Path and name of a file containing the auxiliary quantities required for enabling a fast *online* selection of the appropriate local ROB during a computation performed using a hyper reduced nonlinear computational model (characters).

Next: MASS, Previous: LOCROB

68 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

69 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

70 MATERIAL LAW

Command Statement:	**MATLAW**

The MATLAW command can be used for two different purposes:

- Specify one or several predefined material laws, whether they are linear or nonlinear.
- Define and label a material law (constitutive model) supported by a user-defined subroutine to be linked with **AERO-S** via packaging in a library with a ".so" extension. For this purpse, 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.

Note that because the MATERIAL command can be used for inputting the material properties of linear elastic elements, the MATLAW command and related MATUSAGE command should be used primarily for defining and/or specifying nonlinear material laws.

Note 1: In the absence of the NONLINEAR command in the ASCII Input Command Data file, a linear elastic material is attributed to all elements even when otherwise specified under this command, and its properties are set to those specified under the MATERIAL command.

Note 2: In the presence of the NONLINEAR command in the ASCII Input Command Data file, the material laws and properties specified under MATLAW and assigned under MATUSAGE take precedence over those inputted under MATERIAL if a conflict arises.

Note 3: In the presence of the NONLINEAR command but absence of the MATLAW command in the ASCII Input Command Data file, linear elasticity with infinitesimal strains is assigned as the default material law to beam and shell elements as the nonlinear geometric effects of these elements are accounted for using the corotational method, and the Saint Venant-Kirchhoff hyperelastic law is assigned as the default material law to 3D solid elements as their nonlinear geometric effects are accounted for using the total Lagrangian method.

Note 4: The various predefined 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 predefined material laws `Linear`, `HenckyElastic`, `MooneyRivlin`, `NeoHookean`, `Ogden`, `StVenantKirchhoff`, `BilinearPlastic`, `FiniteStrainPlastic`, `LogStrainPlastic`, `SimoPlastic`, `ViscoLinearElastic`, `ViscoNeoHookean`, `ViscoMooneyRivlin`, and `ViscoStVenantKirchhoff`.
- The shell elements `15` and `1515` support the predefined material laws `J2Plasticity`, `PlaneStressLinear`, `PlaneStressBilinearPlastic`, and `PlaneStressViscoLinearElastic`.
- The shell element `16` supports the predefined material laws `HypoElastic`, `J2Plasticity`, `KK1`, and `KK2`.
- The membrane elements `128` and `129` support the predefined material laws `LinearPlaneStress`, `HyperElasticPlaneStress`, `PlaneStressLinear`, `PlaneStressMooneyRivlin`, `PlaneStressNeoHookean`, `PlaneStressStVenantKirchhoff`, `PlaneStressBilinearPlastic`, `PlaneStressFiniteStrainPlastic`, `PlaneStressViscoLinearElastic`, `PlaneStressViscoMooneyRivlin`, `PlaneStressViscoNeoHookean`, and `PlaneStressViscoStVenantKirchhoff`.

Note 5: 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.

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 predefined material laws has been described below on separate lines; when preparing a AERO-S input file however, this data should be specified on a single line.

`MATLAW`

READ MATERIAL_NAME	<pathandfilename>					
MATERIAL_ID MATERIAL_NAME	PARA#1	...	...	...	...	PARA#N
MATERIAL_ID Linear	ρ	E	ν	[T_a	w]	[TulerButcher
	σ_0	β_{TB}	K_f]			
MATERIAL_ID HenckyElastic	ρ	E	ν	[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	$[\mu_3 \cdots \mu_9]$	α_1	α_2
	$[\alpha_3 \cdots \alpha_9]$	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	ϵ	YSSFSRT-ID	
MATERIAL_ID FiniteStrainPlastic	ρ	E	ν	E_T	σ_{yield}	β
	[T_a	w]	MAXEPS	ϵ	YSSFSRT-ID	
MATERIAL_ID J2Plasticity	E	ν	ρ	σ_{yield}	K	H
	ϵ	MAXEPS	YSSFSRT-ID			
MATERIAL_ID KK1	E	ν	ρ	σ_{yield}	K	H
	ϵ	MAXEPS	YSSFSRT-ID			
MATERIAL_ID KK2	E	ν	ρ	ϵ	ϵ	
MATERIAL_ID LogStrainPlastic	ρ	E	ν	E_T	σ_{yield}	β
	[T_a	w]	MAXEPS	ϵ	YSSFSRT-ID	

MATERIAL_ID SimoPlastic ρ E ν E_T σ_{yield} $[\beta]$

MATERIAL_ID ViscoLinearElastic ρ E ν [T_a $\underline{w}$] g_1 τ_1 g_2 τ_2 g_3 τ_3 [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID ViscoMooneyRivlin ρ c_1 c_2 c g_1 τ_1 g_2 τ_2 g_3 τ_3 [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID ViscoNeoHookean ρ E ν g_1 τ_1 g_2 τ_2 g_3 τ_3 [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID ViscoStVenantKirchhoff ρ E ν [T_a $\underline{w}$] g_1 τ_1 g_2 τ_2 g_3 τ_3 [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID LinearPlaneStress ρ E ν t

MATERIAL_ID HyperElasticPlaneStress ρ E ν t

MATERIAL_ID PlaneStressLinear ρ E ν [T_a $\underline{w}$] t [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID PlaneStressMooneyRivlin ρ c_1 c_2 c t [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID PlaneStressNeoHookean ρ E ν t [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID PlaneStressStVenantKirchhoff ρ E ν [T_a $\underline{w}$] t [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID PlaneStressBilinearPlastic ρ E ν E_T σ_{yield} β [T_a $\underline{w}$] t MAXEPS ϵ YSSFSRT-ID

MATERIAL_ID PlaneStressFiniteStrainPlastic ρ E ν E_T σ_{yield} β [T_a $\underline{w}$] t MAXEPS ϵ YSSFSRT-ID

MATERIAL_ID PlaneStressViscoLinearElastic ρ E ν [T_a $\underline{w}$] g_1 τ_1 g_2 τ_2 g_3 τ_3 t [TulerButcher σ_0 β_{TB} K_f]

MATERIAL_ID PlaneStressViscoMooneyRivlin ρ c_1 c_2 c g_1 τ_1 g_2 τ_2 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 PlaneStressViscoStVenantKirchhoff ρ E ν [T_a $\underline{w}$] g_1

[TulerButcher τ_1 σ_0 g_2 β_{TB} τ_2 K_f g_3 τ_3 t]

mediumspace 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 **AERO-S**.

mediumspace MATERIAL_ID — ID number of the material law (integer).

mediumspace MATERIAL_NAME — Name of the material law (characters).

mediumspace <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 predefined standard linear elastic material law (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.

HenckyElastic — Name of another predefined 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 sub-command COEF of COMPOSITE, in which case the parameters ρ, ν, and E inputted under this command are ignored.

HypoElastic — Name of the predefined plane stress version of the hypoelastic material law (characters).

MooneyRivlin — Name of the predefined compressible version of the finite strain Mooney-Rivlin hyperelastic material law (characters).

NeoHookean — Name of the predefined compressible version of the finite strain hyperelastic material law (characters).

Ogden (with K_1 only) — Name of the predefined 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 distorsional 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 distorsional response of the material, and K_1 is its bulk modulus.

Ogden (with K_1 and K_2) — Name of the predefined 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 distorsional 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 distorsional response of the material, and K_1 and K_2 are material properties that characterize its volumetric response.

StVenantKirchhoff — Name of a predefined finite strain hyperelastic material law (appropriate for metals and moderate strains) (characters). Its conjugate strain and stress are the Green-Lagrange (engineering) strain and the second Piola-Kirchhoff stress. This material law has an anisotropic variant which can be specified using the sub-command COEF of COMPOSITE, in which case the parameters ρ, ν, and E inputted under this command are ignored.

BilinearPlastic — Name of a predefined 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 predefined finite strain plastic material law featuring kinematic hardening (characters). Its conjugate strain and stress are the Green-Lagrange (engineering) strain and the second Piola-Kirchhoff stress.

J2Plasticity — Name of the predefined 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 predefined 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 predefined 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 predefined 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 predefined finite strain elasto-plastic material law formulated in the logarithmic principal stretches and based on a multiplicative decomposition of the deformation gradient (see *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)*)
ViscoLinearElastic	Name of a predefined linear elastic material law equipped with a 3-term Prony series viscoelastic model (see, for example, *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)*).
ViscoMooneyRivlin	Name of a predefined Mooney-Rivlin material law equipped with a 3-term Prony series viscoelastic model (see, for example, *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)*).
ViscoNeoHookean	Name of a predefined NeoHookean material law equipped with a 3-term Prony series viscoelastic model (see, for example, *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)*).
ViscoStVenantKirchhoff	Name of a predefined finite strain hyperelastic material law equipped with a 3-term Prony series viscoelastic model (see, for example, *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)*).
LinearPlaneStress	Name of the predefined material law that is the plane stress counterpart of the linear elastic material law `Linear` described above (characters). This option is equivalent to the option `PlaneStressLinear` described below.
HyperElasticPlaneStress	Name of the predefined finite strain hyperelastic material law that is the plane stress counterpart of the material law `StVenantKirchhoff` described above (characters). This option is equivalent to the option `PlaneStressStVenantKirchhoff` described below.
PlaneStressLinear	Name of the predefined material law that is the plane stress counterpart of the material law `Linear` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element. This option is equivalent to the option `LinearPlaneStress` described above.
PlaneStressMooneyRivlin	Name of the predefined material law that is the plane stress counterpart of the material law `MooneyRivlin` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressNeoHookean	Name of the predefined material law that is the plane stress counterpart of the material law `NeoHookean` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressStVenantKirchhoff	Name of the predefined material law that is the plane stress counterpart of the material law `StVenantKirchhoff` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element. This option is equivalent to the option `HyperElasticPlaneStress` described above.
PlaneStressBilinearPlastic	Name of the predefined material law that is the plane stress counterpart of the material law `BilinearPlastic` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressFiniteStrainPlastic	Name of the predefined material law that is the plane stress counterpart of the material law `FiniteStrainPlastic` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element.
PlaneStressViscoLinearElastic	Name of the predefined material law that is the plane stress counterpart of the material law `ViscoLinearElastic` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear

equation at each Gauss point of an element.

`PlaneStressViscoMooneyRivlin` Name of the predefined material law that is the plane stress counterpart of the material law `ViscoMooneyRivlin` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element.

`PlaneStressViscoNeoHookean` Name of the predefined material law that is the plane stress counterpart of the material law `ViscoNeoHookean` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element.

`PlaneStressViscoStVenantKirchhoff` Name of the predefined material law that is the plane stress counterpart of the material law `ViscoStVenantKirchhoff` described above (characters). It is implemented by numerically enforcing the plane-stress condition as described in *S. Klinkel and S. Govindjee, "Using Finite Strain 3D-Material Models in Beam and Shell Elements," Engineering Computations, Vol. 19, pp. 254-271 (2002)*, which involves solving a nonlinear equation at each Gauss point of an element.

ρ Mass density per unit volume (real).

E Material Young modulus (real).

ν Poisson ratio (real).

T_a 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.**

w Coefficient of thermal expansion (real). The default value is 0.

`TulerButcher` 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

$$\Psi(J, I_1, I_2) = c(J-1)^2 - 2(c_1 + 2c_2)lnJ + c_1(I_1 - 3) + c_2(I_2 - 3)$$

and J denotes the determinant of the deformation gradient and I_1 , I_2 and I_3 are its three invariants.

μ_1 - μ_9 Material properties of the predefined hyperelastic material law named `Ogden` that characterize the distorsional response of the material (real).

α_1 - α_9 Material properties of the predefined hyperelastic material law named `Ogden` that characterize the distorsional response of the material (real).

K_1 Material property (or Bulk modulus) of the predefined hyperelastic material law named `Ogden` that characterizes the volumetric response of the material (real).

K_2 Material property of the predefined hyperelastic material law named `Ogden` 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 *-INT* (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

curve_id = INT.

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.
MAXEPS	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 MAXEPS 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} .
YSSFSRT-ID	Integer ID (integer) of the yield stress scaling factor vs effective plastic strain rate curve (see YSSFSRT). Ommitting this parameter sets the default mode of computations where the yield stress scaling factor is set to 1 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 predefined material laws ViscoLinearElastic, ViscoNeoHookean, and ViscoMooneyRivlin (real).
τ_1 - τ_3	Relaxation times for the 3-term Prony series viscoelastic model used in the predefined material laws ViscoLinearElastic, ViscoNeoHookean, and ViscoMooneyRivlin (real).
t	Element material thickness (real).

Next: MODE, Previous: MATLAW

71 MATERIAL LAW USAGE

Command Statement:	**MATUSAGE**

The MATUSAGE command statement can be used to assign a material law specified in the MATLAW command to one or several elements.

Note 1: Because the MATERIAL command can be used for inputting the standard material properties of linear elastic elements, the MATLAW and MATUSAGE commands are expected to be used primarily for specifying nonlinear material laws.

Note 2: If an element is assigned two different material laws using the MATUSAGE/MATLAW commands and ATTRIBUTES/MATERIAL commands, the material law defined in MATLAW and assigned in this command takes precedence.

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

72 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: NFRAMES, Previous: MODE

73 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

74 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

75 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

76 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

$$\bar{v} = \int_{-\infty}^{+\infty} v p(v) dv = v_0 = \frac{\sum_{i=1}^{N} v^{(i)}}{N}$$

$$\sigma_v = \sqrt{\int_{-\infty}^{+\infty} (v - \bar{v})^2 p(v) dv} = \sqrt{\sum_{i=1}^{i=P-1} \left(\int_{-\infty}^{+\infty} \psi_i^2 p(\psi_i(\{\xi_j\})) d\psi_i \right) v_i^2}$$

$$= \sqrt{\sum_{i=1}^{i=P-1} \left(\int_{-\infty}^{+\infty} \psi_i^2 p(\psi_i(\{\xi_j\})) d\psi_i \right) \left(\frac{\sum_{j=1}^{N} v^{(j)} \psi_i^{(j)}}{N \int_{-\infty}^{+\infty} \psi_i^2 p(\psi_i) d\psi_i} \right)^2}$$

where v_0 denotes the constant of the PC expansion of the non deterministic v field ("the output"), N denotes the number of realizations, $v^{(i)}$ denotes the deterministic solution of the field v 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: OUTPUT, Previous: NONINPC

77 NONLINEAR ANALYSIS

Command Statement: **NONLINEAR**

The command NONLINEAR specifies that the static, quasistatic, or dynamic analysis to be performed is a nonlinear one. It also specifies the nonlinear solution strategy and sets its parameters.

In the presence of this command in the input file:

- Material nonlinearities are automatically accounted for, in those elements of the computational model to which a nonlinear material law defined in MATLAW is assigned using MATUSAGE.
- Geometric nonlinearities:
  - Are automatically accounted for in a bar, beam, or shell element using the corotational formulation.
  - Are accounted for in a solid element using the total Lagrangian formulation, if the nonlinear material law defined in MATLAW and assigned to this element in MATUSAGE is based on nonlinear kinematics.

For structural problems, if the command NONLINEAR is present in the input file but no nonlinear material law is specified for some element of the model using MATLAW and MATUSAGE, a default material law is assigned to that element and a geometrically nonlinear formulation of the internal force vector and tangent stiffness matrix is activated. The default material law is element-dependent. For bar, beam and shell elements, it is simply linear elasticity, and the elastic properties are those defined in MATERIAL for the material ID of the aforementioned element. For 3D solid elements, the default material law is the Saint Venant-Kirchhoff hyperelastic law, and the properties of this material are those defined in MATERIAL for the material ID of the aforementioned element.

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 can be used for geometrically nonlinear explicit dynamic analyses.

Note 3: Currently, all 3D solid, beam, shell and rigid elements except the BT shell element (type 16) can be used for geometrically nonlinear static and implicit dynamic analyses.

Note 4: Currently, all 3D solid elements, the andes shell element (type 15) and the BT shell element (type 16) can be used for materially nonlinear explicit dynamic analyses.

Note 5: Currently, all 3D solid elements and the andes shell element (type 15) 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 *static* 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 $\underline{10^{-16}}$ 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 QSTATICS. 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 *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 time-step computation during which the aforementioned events occur using half the value of the time-step, until Newton's method converges. Upon convergence, the value of the time-step is doubled in each subsequent dynamic step computation until the original value of the 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 intertial 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 γ `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

$$\text{solve} \quad R(u) = 0$$

where R and $\underline{u}$ 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 $\underline{u}$. 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 agressive iterates as follows

$$u^{k+1} = u^k + \alpha^k \Delta u^k$$

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` 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:

`backtracking` For a given pair of u^k and Δu^k, the main idea in this case is to minimize the "merit" function

$$f(u^{k+1}) = \tfrac{1}{2}\|R(u^{k+1})\|_2^2 = \tfrac{1}{2}\|R(u^k + \alpha^k \Delta u^k)\|_2^2$$

by searching iteratively for the value of α^k which satisfies the first Wolfe condition — aka the "sufficient decrease condition"

$$f(u^k + \alpha^k \Delta u^k) \leq f(u^k) + \frac{c_{bt}}{2} \alpha^k \nabla f(u^k)^T \Delta u^k$$

where ∇f is the gradient of f with respect to u and satisfies

$$\nabla f(w) = K^T(w) R(u)$$

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

$$\|R(u^k + \alpha^k \Delta u^k)\|_2^2 \leq (1 - c_{bt}\alpha^k)\|R(u^k)\|_2^2$$

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")

$$|\Delta u^{k^T} \nabla f(u^k + \alpha^k \Delta u^k)| \leq c_{bs} |\Delta u^{k^T} \nabla f(u^k)|$$

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 defaut 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: OUTPUT6, Previous: NONLINEAR

78 OUTPUT OF RESULTS (OUTPUT completely spelled out)

Command Statement:	**OUTPUT**	[KEYLETTER]

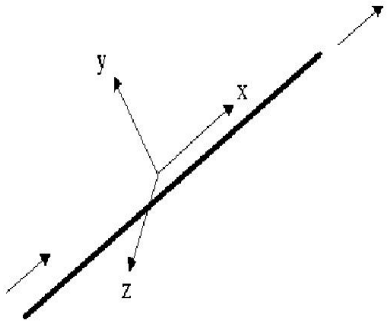
The OUTPUT command statement is used to signal that the following lines of data correspond to specifying which results are to be outputted. For vector results such as displacement fields, this command forces **AERO-S** to output only the three translational components.

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/.

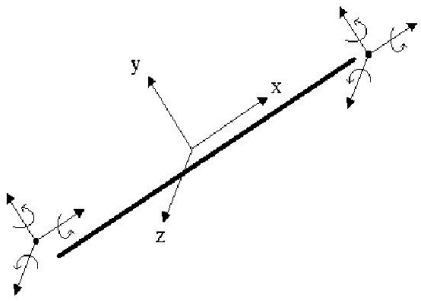
Note 1: In addition to the time-instances implied by the specified value of the parameter INCREMENT, any result specified for output is also outputted at the beginning of a simulation if INCREMENT is non zero, and at the of a simulation if **AERO-S** exits gracefully.

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. SP1DIREC), 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.

```
OUTPUT     [KEYLETTER]
```

```
RESULT     [FORMAT]     PATHANDFILENAME     INCREMENT     [NODE_NUM BER]     [OPTION]     [...]     [OPTION]
```

or

```
RESULT     [FORMAT]     PATHANDFILENAME     INCREMENT     [N NODE_NUMBER]     [OPTION]     [...]     [OPTION]
```

or

```
RESULT     [FORMAT]     PATHANDFILENAME     INCREMENT     [NG GROUP_NUMBER]     [OPTION]     [...]     [OPTION]
```

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` The following results can be outputted (typically in an ASCII file, unless otherwise noted):

`GDISPLAC` 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` Nodal displacements in the x direction.

`DISPLACY` Nodal displacements in the y direction.

`DISPLACZ` Nodal displacements in the z direction.

`ROTATIOX` Nodal rotations in the x direction.

`ROTATIOY` Nodal rotations in the y direction.

`ROTATIOZ` Nodal rotations in the z direction.

`DISPLMOD` Euclidean norms of the nodal displacements.

`ROTATMOD` Euclidean norms of the nodal rotations.

`GDISPMOD` Euclidean norms of the generalized nodal displacements.

`GVELOCIT` 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` Rotation matrix representation of nodal rotations (9 components).

`QUATERNI` Positive unit quaternion representation of nodal rotations (4 components).

`DSVARVCT` Dual state variables (Lagrange multipliers associated with constraints) collected for nonlinear model reduction of contact problems.

`GTEMPERA` Nodal temperatures.

`GTEMPVEL` Nodal first time-derivatives of the temperature.

`GEIGENPA` 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` True fluid nodal displacements associated with the computed sloshing modes.

`SLSHDSPX` x component of the true fluid nodal displacements associated with the computed sloshing modes.

SLSHDSPY	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	$\underline{zx}$ 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	$\underline{xz}$ 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	$\underline{zx}$ 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	$\underline{xz}$ 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.
INYFORCE	Internal forces along the element y -axis.

INZFORCE Internal forces along the element z -axis.

AXMOMENT Applied (internal) moments along the element x -axis.

AYMOMENT Applied (internal) moments along the element y -axis.

AZMOMENT Applied (internal) moments along the element z -axis.

ENERGIES 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):

External : $\sum_{k=0}^{n-1}[(1-\gamma)F^{ext^k}+\gamma F^{ext^{k+1}}]^T(u^{k+1}-u^k)$

Aeroelastic : $\sum_{k=0}^{n-1}[(1-\gamma)F^{aer^k}+\gamma F^{aer^{k+1}}]^T(u^{k+1}-u^k)$

Elastic : $u^{n^T}Ku^n$ (linear analysis)

$\int$ elastic strain energy (nonlinear analysis)

Kinetic : $\dot{u}^{n^T}M\dot{u}^n$

Dissipative : $\sum_{k=0}^{n-1}[(1-\gamma)C\dot{u}^k+\gamma C\dot{u}^{k+1}]^T(u^{k+1}-u^k)$ (linear analysis)

$\sum_{k=0}^{n-1}[(1-\gamma)C\dot{u}^k+\gamma C\dot{u}^{k+1}]^T(u^{k+1}-u^k)$ + material dissipation (nonlinear analysis)

Numerical Production : Kinetic - Previous_Kinetic + Elastic - Previous_Elastic + Increment_Dissipative_Between_Previous_and_Current - Increment_Exterior_Between_Previous_and_Current

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(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).
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 SROMDISP	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 mass-orthonormalized 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 *binary* 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 *nonlinear* structural dynamic analysis. During a linear structural dynamic analysis, displacement snapshots can be collected in the result GDISPLAC using the OUTPUT6 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 *binary* file. They can be used, for example, for constructing one or multiple (hyper) reduced meshes using the NDSCFG sub-command keyword of the RMSHC command.

VELOCVCT
: Snapshots of the velcity 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 RODC 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 RODC command. Currently, this result can be outputted only during a nonlinear structural dynamics analysis.

ROBDATAF
: Prefix of the two binary files containing both instances of the ROB generated by the ROBC command (see in ROBC the MNORMA sub-command keyword).

  - If this ROB is a primal ROB, its orthonormalized instance is outputted in the file <PATHANDFILENAME.orthonormalized>, and its mass-orthonormalized counterpart is outputted in the file <PATHANDFILENAME.massorthonormalized>. 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 <pathandfilename> 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, <pathandfilename_i> must be set to the name of the file containing the mass-orthonormalized version of this ROB unless rob_type_i is set to inorm.
  - If this ROB is a dual ROB, and there can be more than one basis computed (see numrob in ROBC), each ROB is outputted in a separate file labeled <PATHANDFILENAME>.int, where int 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 comes in the form of two files:

  - One file named SAMPLMSH and containing the sampled mesh needed for performing a hyper reduction, in the form of an inclusive list of elements and their corresponding hyper reduction weights or coefficients (see ATTRIBUTES).
  - Another file named SAMPLMSH.elementmesh.inc containing the reduced mesh and that part of the high-dimensional model needed for constructing the corresponding hyper reduced model.

  The format of each of these two files is designed to allow its inclusion in the **AERO-S** input file using the INCLUDE command (see INTRODUCTION).

MODALMAS or SROMMASS
: ASCII file containing the reduced-order mass matrix associated with a linear structural ROM. This file is outputted in the format described below.

```
* Mr: Reduced mass matrix
      n n - n: size of the linear structural dynamics ROM
      Mr11 Mr12 ... Mr1n
      Mr21 Mr22 ... Mr2n
                .
                .
                .
      Mrn1 Mrn2 ... Mrnn
```

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 - n: size of the linear structural dynamics ROM
      Dr11 Dr12 ... Dr1n
      Dr21 Dr22 ... Dr2n
                .
                .
                .
      Drn1 Drn2 ... Drnn
```

  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 or inorm (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 linear structural ROM. This file is outputted in the format described below.

```
* Kr: Reduced stiffness matrix
      n n - n: size of the linear structural dynamics ROM
      Kr11 Kr12 ... Kr1n
      Kr21 Kr22 ... Kr2n
               .
               .
               .
      Krn1 Krn2 ... Krnn
```

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.

FORMAT — 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 — User specified filename associated with result (characters).

INCREMENT — 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 — Sub-command keyword (character) signaling that the next entry is a node identifying number.

NODE_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 — Sub-command keyword (characters) signaling that the next entry is a group identifying number (see GROUPS).

GROUP_NUMBER — 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 DAMAGESC is requested, the user can set this optional parameter to one of the following values: NODALFULL, NODALPARTIAL, 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 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 *linear* 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 $-C\boldsymbol{w}(T - Ta)$, where C is the pertaining constitutive matrix, $\underline{w}$ 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 $C(\epsilon - \boldsymbol{w}(T - Ta))$.

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 ρ of the output values are first printed and then the phases θ are printed. When set to ANIM, the values of $\rho\cos(\theta - \phi_1)$ are printed first, followed by the values of $\rho\cos(\theta - \phi_2)$, etc, concluding with the values of $\rho\cos(\theta - \phi_N)$, 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 (for example, the x, y and z components of a computed displacement field or the xy component of a stress field) in the nodal frames defined at the nodes of the finite element model, or in the global reference frame. In the former case, this parameter should be set to NODFRA, and in the latter case, to GLOFRA. The default value of this parameter is GLOFRA. For this reason, when outputting a directional result to be reused with the command READMODE, this parameter should be set to NODFRA.

ROTVEC_OPTION — This optional parameter can be used to select the specific type of rotation vector output for GDISPLAC (only under OUTPUT6), ROTATIOX, ROTATIOY, ROTATIOZ, ROTATMOD, and GDISPMOD. It can can take values Euler (default value), Complement, Linear, ReducedEulerRodrigues, CayleyGibbsRodrigues, WienerMilenkovic, or BauchauTrainelli. The default Euler rotation vector, also known as the axis-angle representation, is a vectorial parameterization of the rotation group whose magnitude is the (scalar) rotation angle θ and rotation axis is the unit 3-vector $\hat{\underline{e}}$ — that is, $\underline{\Psi} = \theta\hat{\underline{e}}$. 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 *The vectorial parameterization of rotation*, by Bauchau and Trainelli, in *Nonlinear Dynamics*, 32, No 1, pp. 71–92, 2003. Regardless of the value of ROTVEC_OPTION, 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).

RESCALING FLAG — Optional sub-command keyword followed by an On or Off 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 GDISPLAC, GVELOCIT, and GACCELER under OUTPUT6, and to ROTATIOX, ROTATIOY, ROTATIOZ, ROTATMOD, and GDISPMOD. The default value of FLAG is On. The specific interpretation of this option depends on the setting of ROTVEC_OPTION. For example, in the case of an Euler rotation vector, setting FLAG to On 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).

ANGULAR_OPTION — This optional parameter can be used to choose a definition of the angular velocity or acceleration for GVELOCIT or GACCELER under OUTPUT6. It can take the following values: Convected (default), Spatial, and Total. The Convected definition refers to a moving frame that is computed by **AERO-S** and attached to each node. The definition Spatial refers to the fixed reference frame, and the definition Total refers to the time derivatives of the (Euler) rotation vector. In the case of Total, the RESCALING 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 IVEL or USDD should be Convected.

Next: PITA, Previous: OUTPUT

79 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

80 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 imput 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^{-6} .
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

81 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 x direction (float).
PRELOAD_VALUE2	For a membrane element, value of the preload generalized force (force per unit length) in the global y direction (float).
PRELOAD_VALUE3	For a membrane element, value of the preload generalized force (force per unit length) in the global z 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

82 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; for beam elements they can either be of the follower type (default) or non-follower type, depending on a compilation switch that must be set for this purpose.

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 STATICS 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 CONWEP software module.
CONWEPSWITCH	On/Off switch for the pressure load due to an air blast in the free field whose parameters are set in the CONWEP command (characters). If this switch is set to On, the pressure field computed by the software module CONWEP 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: QSTATICS, Previous: PRESSURE

83 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 comand is concerned.

Note 1: This command may only be used in the absence of a mesh decomposition.

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

84 QUASISTATICS ANALYSIS

Command Statement:	**QSTATICS**

The QSTATICS 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

$$A\tilde{x} = b^{n+1}$$
$$\alpha^{n+1} = \alpha^n + \beta R^T b^{n+1}$$
$$\bar{x}^{n+1} = \bar{x}^n + \theta(\tilde{x} - \bar{x}^n)$$
$$x^{n+1} = \bar{x}^{n+1} + R\alpha^{n+1}$$

For problems of the form *r(x,b)=0*, the *iterative* solution algorithm underlying this command can be described as

$$r(\tilde{x}, b^{n+1}) = 0$$
$$x^{n+1} = x^n + \theta(\tilde{x} - x^n)$$

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 aeroelatic 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 `B0` 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.

```
QSTATICS
```

```
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: QSTATICS

85 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 kx, ky, and kz (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: RENUMBERING, Previous: RANDOM

86 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 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_idi below) is automatically set to 0 (characters). Three formats are supported for this file:

- The first 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 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 format.
- The third format is a binary format generated by the command ROBC and readable by the present command.

`numvec` When the input file `<pathandfilename>` is written in the first 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 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 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>`.

- The second format described below should be used in the case of:
  - One or multiple primal, structural, mass- or identity-orthonormalized eigen or other ROBs.
  - One or multiple dual, non-negative ROBs (for example, non-negative Lagrange multipler bases for constrained nonlinear analyses).

`READMODE`

```
rob_id1     rob_type1     <pathandfilename1>     numvec1     [tolerance1]
.
.
.
rob_idi     rob_typei     <pathandfilenamei>     numveci     [tolerancei]
.
.
.
rob_idN     rob_typeN     <pathandfilenameN>     numvecN     [toleranceN]
```

`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 $V^T V = I$
- `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	*Absolute* 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). **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: RESTART, Previous: READMODE

87 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

88 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 of a simulation if **AERO-S** exits gracefully.

The syntax for invoking this command is given below.

```
PATHANDFILENAME1 INCREMENT
PATHANDFILENAME2 EXTENSION
```

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/

Next: RBMFILTER, Previous: RESTART

89 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 (real).

Next: ROBC, Previous: TRBM

90 RIGID BODY MODES FILTER

Command Statement:	**RBMFILTER**	[LEVEL]	[QMATRIX]

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 $P^T \neq P$), according to the user's request (see below). In the latter case, the user must ensure that the density and/or mass information provided in the ASCII Input Command Data file leads to a matrix M that is symmetric positive *definite*.
  - 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, the user must ensure that the density and/or mass information provided in the ASCII Input Command Data file leads to a mass matrix M that is symmetric positive *definite*).
  - 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 aeroelatic 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.

- 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.

`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: RMSHC, Previous: RBMFILTER

91 CLUSTERING SNAPSHOTS OR CONSTRUCTING A REDUCED BASIS

Command Statement:	**ROBC**

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

- Compress one or multiple specified sets of primal displacement and/or velocity solution snapshots, and/or one or multiple specified

sets of previously computed primal Reduced-Order Basis (ROB) vectors using SVD, to construct a global or local primal ROB.
- Compress one or multiple specified sets of dual (Lagrange multiplier) solution snapshots, and/or one or multiple specified sets of previously computed dual ROB vectors, using SVD (Singular Value Decomposition) or NMF (Non-negative Matrix Factorization), to construct a global or local dual ROB. In this case, SVD is appropriate only if the Lagrange multipliers are introduced for enforcing equality constraints.
- 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 command `ROBC`, 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 requested (see `SSCALI`), the inputted solution snapshots are scaled using their associated weights before the SVD or NMF is performed on these snapshots.
- If SVD is chosen as the data compression algorithm and scaling is requested (see `SSCALI`), the inputted ROB vectors are scaled using their associated nonzero singular values before data compression is performed.

Each computed cluster of displacement (and other associated quantities) is outputted by this same command as described in the explanation of the sub-command keyword `DO_CLUSTERING`. On the other hand, any computed ROB can be outputted using the result keyword `ROBDATAF` 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 command DECOMPOSE — 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          flagrs
COMPRESS        flagco
DIMENS          dim
ROMENR          ε
MNORMA          flagmn
SKIP            freq            [offset]
USE_NMF         maxnoi          cvtolr
USE_NMF         numrob          incsiz          numini          maxnoi      cvtolr
USE_GREEDY
```

USE_PQN	maxnoi	cvtolr	maxini	stepln
NSUBS	nblcks			
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 and/or velocity) 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 sub-command keyword ROBFI) is either:

- Compressed to construct the desired ROB: in this case, all inputted solution snapshots can be scaled by their associated weights before they are compressed if requested using SSCALI.
- 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 which case the 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.

<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 solution snapshots is clustered when the sub-command keyword DO_CLUSTERING 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>.cluster*j*, 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 sub-command keyword DO_CLUSTERING 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.

<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. When requested using SSCALI, all inputted ROB vectors are scaled by their associated weights. The union of these potentially scaled ROB vectors (and potentially scaled solution snapshots inputted after the sub-command keyword SNAPFI) is compressed by SVD or NMF, as specified, to construct the desired ROB.

SSCALI

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

flagss

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

flagrs

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

COMPRESS

Sub-command keyword relevant only when the command ROBC 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 command ROBC 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 k of the ROB to be constructed (characters). Specifically, 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, r is its rank, 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.

ϵ

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 defaut 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 sub-command keyword SNAPFI 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 *dual* 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 *dual* 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 *dual* 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).
nblcks	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 sub-command keyword SNAPFI. If the optional sub-command keyword VELSNAPFI 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 sub-command keyword ACCSNAPFI 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: RODC, Previous: ROBC

92 CONSTRUCTING A REDUCED MESH

Command Statement:	**RMSHC**

The RMSHC command can be used to:

- Construct one or multiple (hyper) reduced meshes 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:
  - One or more specified Reduced-Order Bases (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.
- Update the coordinates of previously constructed (hyper) 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 hyper reduced mesh file, and therefore many aspects of this command become irrelevant.

The computed or updated reduced meshes can be outputted in two different ASCII files that have two different formats, using the same SAMPLMSH result keyword of the OUTPUT command.

Note 1: 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 2: If the mesh sampling is to be trained also with the inertial forces, the velocity and acceleration snapshots must be provided using the TRNSOL sub-command keyword described below.

Note 3: Repeating within this command the *pair* of sub-command keywords (PODROB, 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 PODROB — for example, for equipping a suite of n local ROBs with n associated reduced meshes.

Note 4: 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 5: 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 6: 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`

```
PODROB     <pathandfilename1>     dimlrb
DUALBASIS     <pathandfilename2>     dimldb
TRNSOL     <pathandfilename3>
TRNSOL     <pathandfilename3>     <pathandfilename4>
TRNSOL     <pathandfilename3>     <pathandfilename4>     <pathandfilename5>
DIMENS     dimens
SAMFRQ     samfrq
EXTFOL     flag
SOLVER     solver_type
TOLERA     tolera
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>]
```

`PODROB`	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).
`dimlrb`	Specifies the first `dimlrb` columns of the basis stored in `<pathandfilename1>` as the ROB for training (integer). If the basis is not a *local* one (see ROBC), this parameter should be ignored and the dimension of the training ROB should be specified instead in `dimens` (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.
`dimldb`	Specifies the first `dimldb` columns of the dual basis stored in `<pathandfilename2>` 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 explict 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 <pathandfilename1>, when this basis is not a local one (see ROBC) (characters).

dimens — Specifies the first dimens columns of the non-local basis stored in <pathandfilename1> 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 samfrq-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

- 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 ScaLAPACK and MPI 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 MPI library. This method is also known in the literature as the Non-Negative Conjugate Gradient Pursuit method.
- pfpqr — 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 ScaLAPACK and MPI 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 MPI 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 MPI 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, tolera = 0.01 means that 99 % of the energy associated with the training force vectors and the ROB is conserved on the reduced mesh (real).

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 x, y, and z 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 nclusters = 1, in order for **AERO-S** to build the association between a snapshot and the mesh on which it was computed.

x_scale_factori — x-coordinate scale factor for the i-th mesh and the case where the collected training snapshots were computed on different meshes (real).

y_scale_factorj — y-coordinate scale factor for the j-th mesh and the case where the collected training snapshots were computed on different meshes (real).

z_scale_factork — z-coordinate scale factor for the k-th mesh and the case where the collected training snapshots were computed on different meshes (real).

<pathandfilenamei_j> — Name of the *optional* binary file (including path, if needed) containing (characters):

- The j-th mass-orthonormalized local ROB computed on the i-th mesh, when model reduction is to be based on local ROBs.
- The global mass-orthonormalized ROB computed on the 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 nclusters = 1, in order for **AERO-S** to build the association between a snapshot and the mesh on which it was computed.
nodalcoordinatesi	Name of the binary file (including path, if needed) containing the nodal coordinates of the i-th mesh.

Next: SENSITIVITY, Previous: RMSHC

93 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:

- Reconstructing the high-dimensional solution of the nonlinear, dynamic problem of interest using the computed low-dimensional solution (generalized coordinates) of its reduced-order version, and the Reduced-Order Basis (ROB) provided through the READMODE command.
- Reconstructing 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 sub-command keyword SROM 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 CONFIL sub-command keyword 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 command DECOMPOSE — 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
        <pathandfilenameS>     freq
CONFIL
        <pathandfilenameV>     freq
CONFIL
        <pathandfilenameA>     freq
```

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: SENSORS, Previous: RODC

94 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

95 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.

Note 2: For nonlinear analyses, sensor information on the velocity and acceleration of a nodal degree of freedom is not currently available for use in the "control.C" file. These are currently passed as zero.

```
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

96 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

97 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: STATICS, Previous: SZEM

98 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	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.
TIME_1	A specified time point (float).
AMP_1	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: SDETAFT, Previous: HFTT

99 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. Furthermore among these solvers, only sparse, skyline, mumps pivot, and FETI DP compute and return for further usage the null space of the singular matrix. 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):

`spooles`	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.
`skyline`	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.
`blockskyline`	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):

`mumps`	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 `mumps` is chosen as the coarse problem solver, the coarse problem is duplicated on each MPI process: for this reason, if `mumps` 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 `mumps` 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 `mumps` 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 *effective* parallel solution of the coarse problem. On the other hand, if `mumps` 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 ICNTL(18) = 3 (see `mumps_icntl` 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)

`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.
`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.
`spooles` (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.
`spooles 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).
`mumps pivot` (completely spelled out)	For the MUMPS sparse general symmetric direct solver with the pivoting option turned on (characters).

mumps unsymmetric (completely spelled out)	For the MUMPS unsymmetric sparse direct solver (characters).
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 Eigen3 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 LAPACK 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).
bcg	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 $\geq$ 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 $\geq$ 0, default value is 24).
spooles_maxzeros	n *spooles_maxzeros is the maximum number of zeros allowed in a supernode/front (real > 0 and $\leq$ 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 or keywords and values) for the mumps solver, in any order
mumps_icntl	
index integer_value	Pair of integers which can be used to set the integer-valued mumps parameters stored in its ICNTL array, as follows: 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, whereas the default value ICNTL(18) = 0 has it stored entirely on a single MPI process.
mumps_cntl	
index real_value	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: 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.
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	
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	
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).
psparse	Selects the parallel 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	Specifies the local preconditioner (integer or characters).
0 or noprec	Unpreconditioned (integer). This is the default value.
1 or lumped	Lumped preconditioner (integer).
2 or dirichlet	Dirichlet preconditioner (integer). This is also the default preconditioner.
projector	
1	Identity based projector (integer). This is also the default projector.
2	Preconditioner based projector (integer). Uses the selected preconditioner for building the so-called Q matrix.
4	Superlumped projector (integer).
scaling	
1 or stiffness	Stiffness based scaling (integer).
2 or topology	Topology (subdomain connectivity) based scaling (integer). This is the default scaling procedure.
version	
1	The FETI (also known as FETI-1) method (integer).
2	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 and 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 and 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 interace and the fluid/structure interface is chosen as an additional corner node.

`augment` This keyword is useful only for the FETI-DP and 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.

`Gs trans/all` 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.

`WeightedEdgeGs trans/all` 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 `scaling` option is set to `stiffness`, or on topological considerations if the `scaling` option is set to `topology`.

`EdgeWs [type] numdir` This option is exclusive to the FETI-DPH solver and can be combined with the option `EdgeGs`. 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 `type` to `solid`, `shell`, `fluid`, or `any`, 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 `any` is recommended (default value). The parameter `numdir` 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 $2\times$ `numdir` for two- and three-dimensional frequency-domain acoustic problems, $4\times$ `numdir` for two-dimensional elastodynamic (or modelled dynamic shell) problems, and $6\times$ `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*`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 $n^2 + 4(n-1)(n/2-1) + 2(n-1)mod(n,2)$ directions. Hence, *n* is chosen so that

$n^2 + 4(n-1)(n/2-1) + 2(n-1)mod(n,2)$ is as close as possible to `numdir`, with $n^2 + 4(n-1)(n/2-1) + 2(n-1)mod(n,2) \geq$ `numdir`.

`interf_solver` 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).

- `CG` Turns on the CG algorithm as an interface problem solver. This is the default value of `interf_solver`.
- `CGAL` Turns on Dostal's Augmented Lagrangian CG algorithm with adaptive precision control as an interface problem solver.
- `GMRES` Turns on the GMRES algorithm as an interface problem solver.
- `GCR` Turns on the GCR algorithm as an interface problem solver. For frequency sweep problems (see IMPEDANCE), this option is more computationally efficient than GMRES because it can be optimized for systems with multiple right sides (see `maxorth` below).

`orthotol` 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.

`mpc_type` 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 `primal` when the number of MPCs is less or equal to 1000, and `dual` otherwise.

- `dual` In this case, the MPCs are enforced by Lagrange multipliers and are satisfied only at convergence (characters).
- `primal` In this case, the MPCs are put in the coarse problem and are satisfied at every iteration (characters).

`mpc_precn` This keyword is useful only for the FETI-DP and FETI-DPH methods, in the presence of MPCs, and when `mpc\_type` is set to `dual` (characters). The default is `tblock` when running a single thread process, and `full` otherwise.

- `full` In this case, the CC^T matrix, where C denotes the constraint matrix, is treated as a single block matrix (characters).
- `tblock` In this case, the potential algebraic block-structure of the CC^T matrix, is exploited (characters).
- `sblock` In this case, the subdomain-structure of the CC^T matrix, is exploited (characters).
- `mblock` In this case, the mortar-interface-structure of the CC^T matrix, is exploited (characters).
- `diag` In this case, the CC^T matrix is approximated by its diagonal (characters).

`mpc_scaling`

- `1 or stiffness` Stiffness based scaling (integer).
- `2 or topology` Topology (subdomain connectivity) based scaling (integer). This is the default scaling procedure.

`cct_solver`

- `skyline` Selects the skyline direct method as the coarse solver. This is the default choice (characters).
- `sparse` Selects Esmond's (sequential) sparse direct method as the coarse solver (characters).

`cct_tol` 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.

`kryprec` 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 `maxorth` parameter (see below) the scope of the entire nonlinear analysis.
- 2 Turns on the Krylov preconditioner and gives to the `maxorth` parameter (see below) the scope of a load-step in a nonlinear analysis.

`global_cor_rbm_tol` 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.

`global_rbm_tol` 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.

`maxorth` 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 `interf-solver` (see below) is set to `CG` or `GCR`. For nonlinear problems, the acceleration is performed using a multiple left side technique based on the enrichment of the preconditioner, but only if: (a) `interf_solv` (see below) is set to `CG`, and (b) kryprec (see above) is activated. If the kryprec option is not activated, `maxorth` 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 `kryprec` option is used, priority in the accumulation of vectors is given to

enriching the preconditioner. The default value is `maxitr` (see below). To turn-off this reorthogonalization option, set `maxorth` to 1.

`tolfeti` Error tolerance for the convergence of the feti solver (real). The default value is 1.0e-06.

`maxitr` Maximum number of iterations to be performed (integer). The default value is 1000.

`PIECEWISE real_1 real_2` 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: RUBDAFT, Previous: STATICS

100 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`	
f_1	η_{E_1}
.	.
.	.
.	.
f_n	η_{E_n}

`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: SURFACETOPO, Previous: SDETAFT

101 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: TETT, Previous: RUBDAFT

102 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 id number (integer).
SURFACE_THICKNESS	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#	Face id number whose type and connectivity are to be specified (integer).
FACETYPE	
1	4-node quadrilateral.
2	8-node quadrilateral.
3	3-node triangle.
4	6-node triangle.
5	9-node quadrilateral.
6	12-node quadrilateral.
7	10-node triangle.

Next: THERMOE, Previous: SURFACETOPO

103 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).
T_1	A specified temperature value (float).
TE_1	A specified coefficient of thermal expansion value at temperature T_1 (float).

Next: TIEDSURFACES, Previous: TETT

104 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: TOPOLOGY, Previous: THERMOE

105 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 defined using **AERO-S**'s mortar method and enforced using the method specified in CONSTRAINTS or in `CONSTRAINT_METHOD` below.

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
```

`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` — The Lagrange multiplier method.
- `elimination` — The elimination method (see CONSTRAINTS for changing the default values of its parameters).
- `penalty beta` — 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).

`MORTAR_TYPE` — Mortar type: 0 = standard, 1 = dual, default value is 0 (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).

`KPART_TYPE` — 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: USDF, Previous: TIEDSURFACES

106 TOPOLOGY *S*

Command Statement:	**TOPOLOGY**

The TOPOLOGY command statement is used to signal that the following data lines correspond to the connectivity and type of each element. The input format is given below. There should be as many lines as there are elements in the system.

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 predefined material laws Linear, HenckyElastic, MooneyRivlin, NeoHookean, Ogden, StVenantKirchhoff, BilinearPlastic, FiniteStrainPlastic, LogStrainPlastic, SimoPlastic, ViscoLinearElastic, ViscoNeoHookean, ViscoMooneyRivlin, and ViscoStVenantKirchhoff. The shell elements 15 and 1515 support the predefined material laws J2Plasticity, PlaneStressLinear, PlaneStressBilinearPlastic, and PlaneStressViscoLinearElastic (see MATLAW). The shell element 16 supports the predefined material laws HypoElastic, J2Plasticity, KK1, and KK2 (see MATLAW). The membrane elements 128 and 129 support the predefined material laws LinearPlaneStress, HyperElasticPlaneStress, PlaneStressLinear, PlaneStressMooneyRivlin, PlaneStressNeoHookean, PlaneStressStVenantKirchhoff, PlaneStressBilinearPlastic, PlaneStressFiniteStrainPlastic, PlaneStressViscoLinearElastic, PlaneStressViscoMooneyRivlin, PlaneStressViscoNeoHookean, and PlaneStressViscoStVenantKirchhoff (see MATLAW).

Note 2: AERO-S supports a run-time generation of frames for *flexible* beams (see EFRAMES) that is activated when a third node in the local *x*-*z* plane is found in the definition of a flexible beam element within this TOPOLOGY command. The only requirement for the third node is that it does not be colinear with the other two beam nodes. Using this third node option releaves the user from specifying the EFRAMES command. An example illustrating the third node option can be found in FEM.d/fem_examples/Third_Node.d.

Note 3: Thermal loads (see TEMPERATURES) are currently implemented only for the Euler beam, 4-noded plane stress/plane strain, 4-noded as well as 10-noded tetrahedron, 8-noded brick, 6-noded penta (prism with triangular cross section), and 3-noded and 4-noded shell elements.

Note 4: For thermal analysis, **AERO-S** constructs the contribution of convection effects to the right-hand side vector of the resulting system of equations using the CONVECTION command. However, it constructs the contribution of convection effects to the stiffness matrix using the “boundary convection” elements (type 47, 48 and 49) which must be superposed to the edges or faces of existing thermal elements. The data for these boundary convection elements (depth, area, convection coefficients and reference temperature) must be specified in the MATERIAL command.

Note 5: Currently, only the SPOOLES and MUMPS sparse direct solvers and the FETI-DP(H) solvers can handle rigid elements.

Note 6: 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 of 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 7: 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.

Note 8: The heat radiation elements 56, 57, and 58 are primarily *nonlinear* 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 reference temperature of the enclosure receiving the radiation which is specified in the parameter Tr of the MATERIAL command. However, the linearized thermal 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 9: 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).

```
ELEMENT#          ELEMENT_TYPE          CONNECTIVITY_NODES
```

ELEMENT#	Element identification number whose type and connectivity are to be specified (integer). A (*) indicates the element is not yet fully implemented and/or validated.
ELEMENT_TYPE:	M=Mechanic - A=Acoustic (Helmholtz or Time-Domain) - H=Heat - C=Coupled Thermoelastic - F=Fluid
1	M: 3d truss (bar) element with 3 dof/node (Figure 1).
2	M: 2d 4-node plane stress/plane strain quad element with 2 dof/node (Figure 2). **Note that only a lumped mass matrix is available for this element**.
3	H: 3d 4-node quad element with 1 dof/node (Figure 2).
4	M: 2d 3-node triangular element with 2 dof/node (Figure 3). **Note that only a lumped mass matrix is available for this element**.
6	M: 3d 2-node Bernoulli beam element with 6 dof/node (Figure 1).
7	M: 3d 2-node Timoshenko beam element with 6 dof/node (Figure 1). **Note that only a lumped mass matrix is available for this element**.
8	M: 3d 3-node triangular AQR shell element with 6 dof/node (Figure 3). **Note that only a lumped mass matrix is available for this element**.
88	M: 3d 4-node shell element element with 6 dof/node that splits into 2 triangular shell elements of type 8 (Figure 2). **Note that only a lumped mass matrix is available for this element**.
9	H: 3d 2-node lineal element with 1 dof/node (Figure 1).
10	H: 2d 4-node quad element with 1 dof/node (Figure 2).
11	M: 3d 1-node (lumped) torsional spring element with 3 dof/node for linear analysis (Figure 4).
12	M: 3d 1-node (lumped) translational spring element with 3 dof/node for linear analysis.
15	M: 3d 3-node triangular AQR shell element 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 element 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 element 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 element with 3 dof/node (Figure 5) and nonlinear material capability.
18	M: 3d 4-node shear panel element with 3 dof/node (Figure 2). **Note that only a lumped mass matrix is available for this element**.
19	M: 3d 3-node triangular membrane element with 6 dof/node (Figure 3). (Only in-plane and drilling stiffnesses). **Note that only a lumped mass matrix is available for this element**.
20	M: 3d 3-node triangular composite or orthotropic shell element with 6 dof/node (Figure 3). **Note that only a lumped mass matrix is available for this element**.
2020	M: 3d 4-node composite or orthotropic shell element with 6 dof/node (Figure 2). **Note that only a lumped mass matrix is available for this element**.
21	M: 3d translational spring-link element with 3 dof/node for linear analysis (Figure 1).
22	M: 3d torsion spring-link element with 3 dof/node for linear analysis (Figure 1).
23	M: 3d 4-node tetrahedral element with 3 dof/node (Figure 6) and nonlinear material capability.
24	M: 3d 6-node pentahedral element (prism with triangular cross section) with 3 dof/node (Figure 7) and nonlinear material capability.
25	M: 3d 10-node tetrahedral element with 3 dof/node (Figure 8) and nonlinear material capability.
30	A: 2d 4-node quad element with 1 dof/node (Figure 2). 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.
31	A: 2d 4-node quadrilateral GLS element with 1 dof/node (Figure 2).
32*	A: 2d 8-node quadrilateral element 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 triangular element 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 triangular element with 1 dof/node (Figure 10).
40	A: 3d 4-node tetrahedron element with 1 dof/node (Figure 6).
41	A: 3d 4-node tetrahedron GLS element with 1 dof/node (Figure 6).
42	A: 3d 10-node tetrahedron element 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 element 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 element with 1 dof/node for boundary convection (Figure 1).
48	H: 3d 4-node quadrilateral element with 1 dof/node for boundary convection (Figure 2).
49	H: 3d 3-node triangular element with 1 dof/node for boundary convection (Figure 3).
50	H: 3d 4-node, 10-node, 20-node, or 35-node tetrahedron element with 1 dof/node. **These elements use a special local node-numbering for connectivity which goes line-by-line** (Figure 11).
51	H: 3d 8-node brick element with 1 dof/node (Figure 5).
52*	M: 3d 6-node triangular shell element (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).
65	M: 3d 2-node rigid truss (bar) element (Figure 1); enforces constant length of the element.
66	M: 3d 2-node rigid beam element (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 (displacement and rotation, Figure 1); for each dof at one node, 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 specified rotational dof at one node, it enforces equality to the corresponding dof at the other node.
70	M: 3d 8-node rigid brick element 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 element with 3 dof/node (Figure 12) and nonlinear material capability.
73	M: 3d 3-node rigid shell element (Figure 3); equivalent to two rigid beam elements, each defined by an edge of the element (special case of element 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 element (Figure 2); equivalent to three rigid beam elements, each defined by an edge of the element (special case of element 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 element with 6 dof/node (Figure 2). (Only in-plane and drilling stiffnesses). **Note that only a lumped mass matrix is available for this element**.
90	A: 3d 6-node wedge element with 1 dof/node (Figure 7).
91	M: 3d 32-node serendipity brick element with 3 dof/node (Figure 13) and nonlinear material capability.
92	M: 3d 26-node serendipity wedge element with 3 dof/node (Figure 17) and nonlinear material capability.
93	A: 3d 32-node serendipity brick element with 1 dof/node (Figure 13).

94	A: 3d 26-node serendipity wedge element with 1 dof/node (Figure 17).
95	A: 3d 8-node, 27-node, 64-node, or 125-node hexahedron element with 1 dof/node. **These 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.
96	A: 3d 4-node, 10-node, 20-node, or 35-node tetrahedron element with 1 dof/node (Figure 11). **These elements use a special local node-numbering for connectivity which goes line-by-line** They also support the PML (Perfectly Matching Layer) computational technology.
97	M: 3d 15-node wedge element with 3 dof/node and nonlinear material capability (Figure 31).
98	A: 2d 4-node, 9-node, 16-node, or 25-node quadrilateral element with 1 dof/node. **These 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.
99	A: 2d 3-node, 6-node, or 10-node triangular element with 1 dof/node. **These 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.
100	M: 2d 4-node, 9-node, 16-node, or 25-node quadrilateral element with 3 dof/node. **These elements use a special local node-numbering for connectivity which goes line-by-line** (Figure 19).
101	M: 2d 3-node, 6-node, or 10-node triangular element with 3 dof/node. **These elements use a special local node-numbering for connectivity which goes line-by-line** (Figure 20).
102	M: 3d 8-node, 27-node, 64-node, or 125-node hexahedral element with 3 dof/node. **These elements use a special local node-numbering for connectivity which goes line-by-line** (Figure 18).
103	M: 3d 4-node, 10-node, 20-node, or 35-node tetrahedral element with 3 dof/node. **These elements use a special local node-numbering for connectivity which goes line-by-line** (Figure 11).
105	A: 3d 8-node, 27-node, 64-node, or 125-node spectral hexahedral element with 1 dof/node. **These spectral elements use a special local node-numbering for connectivity which goes line-by-line** (Figure 18).
108	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).
109	H: 3d 8-node, 27-node, 64-node, or 125-node hexahedral element with 1 dof/node. **These elements use a special local node-numbering for connectivity which goes line-by-line** (Figure 18).
111	M: 3d fabric truss element with 3 dof/node (Figure 1). **Note that only a lumped mass matrix is available for this element**.
118	M: 3d 2-node planar joint element 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 element. Its x and y axes define the orientation in the undeformed configuration of the x and y axes of the co-rotating reference frame at node 1. The position of node 2 is constrained to lie on the plane defined by the x and y axes of the co-rotating reference frame at node 1.
119	M: 3d 2-node welded joint element with 6 dof/node (Figure 30). This element constrains the relative translations and rotations between two nodes.
120	M: 3d 2-node spherical joint element with 3 dof/node. This element constrains the relative translations between two nodes (Figure 21).
121	M: 3d 2-node translational joint element with 3 rotational dof/node (Figure 28). This element constrains the relative rotations between two nodes.
122	M: 3d 2-node universal joint element 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 element. Its y axis defines the orientation in the undeformed configuration of the y axis of the co-rotating reference frame at node 2. Its z axis defines the orientation in the undeformed configuration of the z axis of the co-rotating reference frame at node 1. These two axes remain orthogonal during the deformations.
123	M: 3d 2-node revolute joint element 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 element. Its x axis defines the orientation in the undeformed configuration of the axis of free relative rotation.
124	M: 3d 2-node cylindrical joint element 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 element. Its x axis defines the orientation in the undeformed configuration of the axis of free relative displacement and rotation.
125	M: 3d 2-node prismatic joint element 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 element. Its x axis defines the orientation in the undeformed configuration of the axis of free relative translation.

126 M: 3d 2-node revolute joint-with-driver element with 6 dof/node (Figure 23) and a relative rotation (between node 2 and 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 element. Its x axis defines the orientation in the undeformed configuration of the axis of forced relative rotation.

127 M: 3d 2-node pin-in-slot joint element 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 element. Its x -axis defines the orientation in the undeformed configuration of the axis of free relative translation, and its y -axis defines the orientation in the undeformed configuration of the axis of free relative rotation.

128 M: 3d 4-node plane stress quadrilateral membrane element with 3 dof/node (Figure 2).

129 M: 3d 3-node plane stress triangular membrane element 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.

134 M: 3d 2-node prismatic joint-with-driver element with 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 element. Its x axis defines the orientation in the undeformed configuration of the axis of prescribed relative translation.

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.

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 element. This translational spring element 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 element.

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 element. 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 element. 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. 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 element (type 120) and three embedded torsional spring elements (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 element (type 121) and three embedded translational spring elements (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. This element is equivalent to one universal joint element (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. This element is equivalent to one revolute joint element (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. This element is equivalent to one cylindrical joint element (type 124), one embedded translational spring element (type 201), and one embedded torsional spring element (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 is equivalent to one prismatic joint element (type 125) and one embedded translational spring element (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 element 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 element. 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 is equivalent to one pin-in-slot joint element (type 127), one embedded translational spring element (type 201), and one embedded torsional spring element (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.

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 element. 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.

301 F: 2d 4-node sloshing (fluid) quadrilateral element with 1 dof/node (Figure 2).

302 F: 2d 2-node free-surface (fluid) element with 1 dof/node for two-dimensional sloshing computations using element type = 301 (Figure 1).

311 F: 3d 4-node sloshing (fluid) tetrahedron element with 1 dof/node (Figure 6).

312 F: 3d 3-node free-surface (fluid) triangular element with 1 dof/node for three-dimensional sloshing computations using element type = 311 (Figure 3).

321 F: 2d 4-node hydroelastic vibration (fluid) quadrilateral element with 1 dof/node (Figure 2).

323 M: This element is a variant of element type 223 that is equipped with a freeplay 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 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 element.

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 element.

331 F: 3d 4-node hydroelastic vibration (fluid) tetrahedral element with 1 dof/node (Figure 6).

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 T-4-1 (Figure 20 with p = 2, or Figure 3).
1111	A: 2d 3-node Helmholtz DGM element T-8-2 (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 T-4-1 (Figure 20 with p = 2, or Figure 3).
1131	A: 2d 3-node Helmholtz DEM element T-8-2 (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.

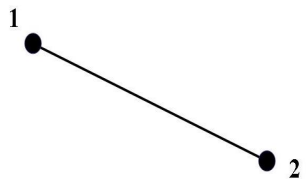


Figure 1: local node numbering for element types 1, 6, 7, 9, 21, 22, 26, 47, 56, 65--69, 111, 302

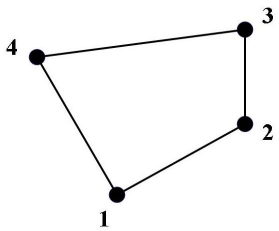


Figure 2: local node numbering for element types 2, 3, 88, 10, 1515, 16, 18, 2020, 30, 31, 33, 34, 4646, 48, 58, 76, 81, 87, 128, 301, 321

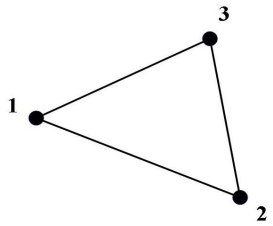


Figure 3: local node numbering for element types 4, 8, 15, 16, 19, 20, 35, 36, 46, 49, 53, 57, 73, 129, 312, 1110,1111,1130,1131

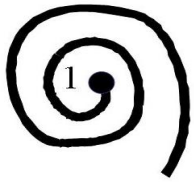


Figure 4: local node numbering for element type 11

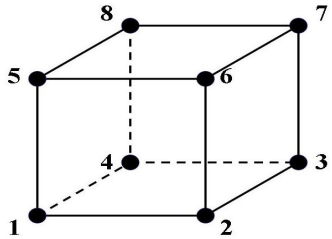


Figure 5: local node numbering for element types 17, 44, 45, 51, 70, 82, 201, 202

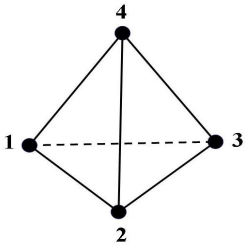


Figure 6: local node numbering for element types 23, 40, 41, 311, 331, 1160, 1161, 1162

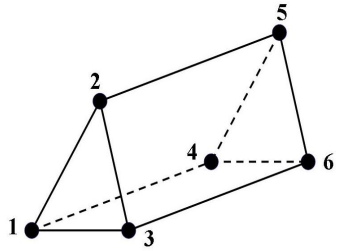


Figure 7: local node numbering for element types 24, 83, 90

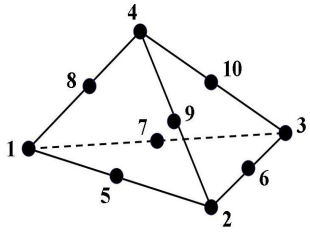


Figure 8: local node numbering for element types 25, 42

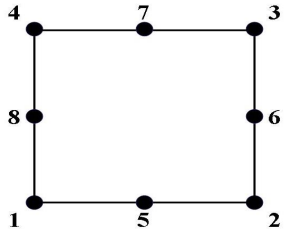


Figure 9: local node numbering for element type 32

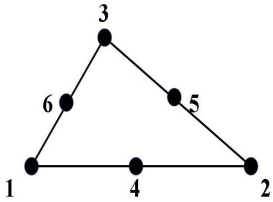


Figure 10: local node numbering for element types 38, 52

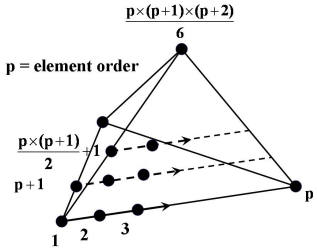


Figure 11: local node numbering for element types 50, 96, 103, 1160, 1161, 1162

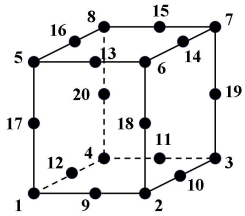


Figure 12: local node numbering for element types 71, 72

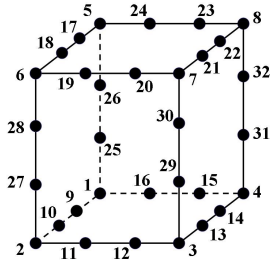


Figure 13: local node numbering for element types 74, 91, 93

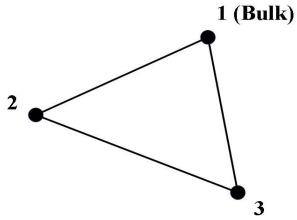


Figure 14: local node numbering for element type 84

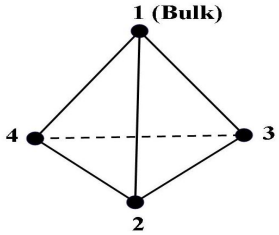


Figure 15: local node numbering for element type 85

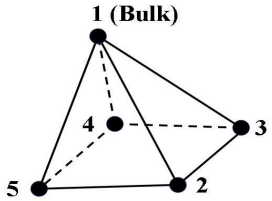


Figure 16: local node numbering for element type 86

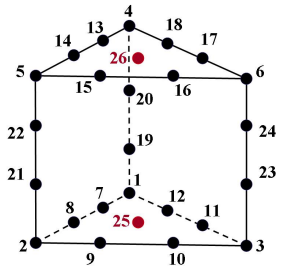


Figure 17: local node numbering for element types 92, 94

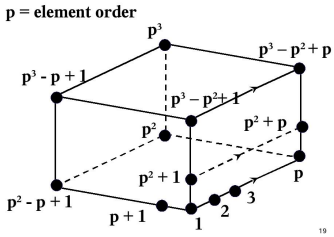


Figure 18: local node numbering for element types 95, 102, 105, 109, 1150, 1151, 1152, 1153, 1170, 1171, 1172, 1173, 1250, 1251, 1252

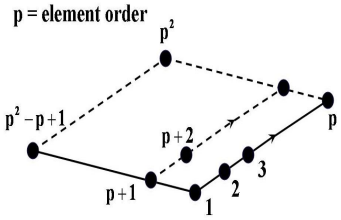


Figure 19: local node numbering for element types 98, 100, 108, 1100, 1101, 1102, 1103, 1120, 1121, 1122, 1123, 1200, 1201, 1220

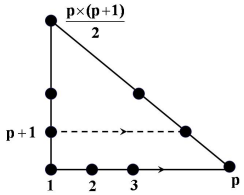


Figure 20: local node numbering for element types 99, 101, 1110, 1111, 1130, 1131

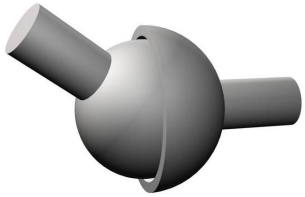


Figure 21: element types 120, 220

Figure 22: element type 122

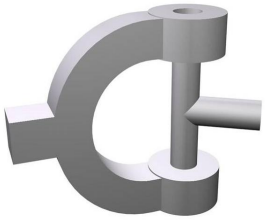


Figure 23: element types 123, 126, 223, 226, 323

Figure 24: element type 124

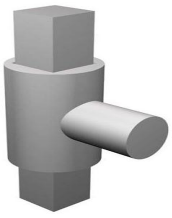


Figure 25: element types 125, 134, 225, 234, 325

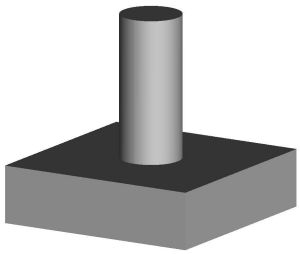


Figure 27: element type 118

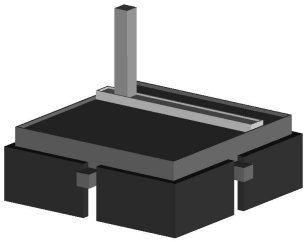


Figure 28: element types 121, 221

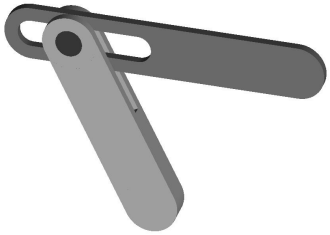


Figure 29: element types 127, 227

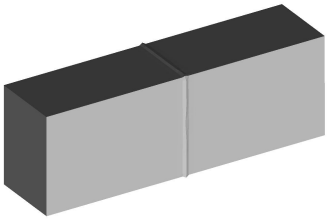


Figure 30: element type 119

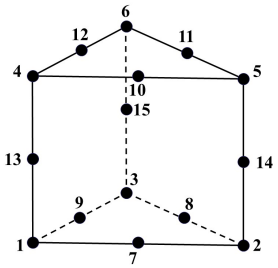


Figure 31: element type 97

Next: USDD, Previous: TOPOLOGY

107 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

108 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#	Node number where the displacement is specified (integer).
DOF#	Degree of freedom local number where the displacement is specified (integer).

Next: YMTT, Previous: USDD

109 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) element with 3 dof/node (**W = 1**).
2	M: 2d 4-node quad element with 2 dof/node (**W = 2**).
3	H: 3d 4-node quad element with 1 dof/node (**W = 2**).
4	M: 2d 3-node triangular element with 2 dof/node (**W = 2**).
6	M: 3d Bernoulli beam element with 6 dof/node (**W = 1**).
7	M: 3d Timoshenko beam element with 6 dof/node (**W = 1**).
8	M: 3d 3-AQR shell element with 6 dof/node (**W = 3**).
88	M: 3d 4-node shell element element with 6 dof/node (splits into 2 elements of type 8) (**W = 4**).
9	H: 3d 2-node lineal element with 1 dof/node (**W = 1**).
10	H: 2d 4-node quad element with 1 dof/node (**W = 2**).
11	M: 3d 1-node torsional spring element with 3 dof/node (**W = 1**).
12	M: 3d 1-node translational spring element with 3 dof/node (**W = 1**).

15	M: 3d 3-node triangular AQR shell element with 6 dof/node and composite as well as nonlinear material capability (**W = 3**).
1515	M: 3d 4-node quadrilateral AQR shell element with 6 dof/node and composite as well as nonlinear material capability (**W = 4**).
16	M: 3d Belytschko-Tsay shell element with 6 dof/node (**W = 4**).
17	M: 3d 8-node brick element with 3 dof/node (**W = 3**).
18	M: 3d 4-node shear panel element with 3 dof/node (**W = 1**).
19	M: 3d 3-node membrane element with 6 dof/node (**W = 3**). (only in-plane and drilling stiffnesses) (**W = 1**).
20	M: 3d 3-node composite or orthotropic shell element with 6 dof/node (**W = 3**).
2020	M: 3d 4-node composite or orthotropic shell element with 6 dof/node (splits into 2 elements of type 20) (**W = 4**).
21	M: 3d translational spring-link element with 3 dof/node (**W = 1**).
22	M: 3d torsion spring-link element with 3 dof/node (**W = 1**).
23	M: 3d 4-node tetrahedral element with 3 dof/node (**W = 3**).
24	M: 3d 6-node pentahedral element with 3 dof/node (**W = 3**).
25	M: 3d 10-node tetrahedral element with 3 dof/node (**W = 4**).
30	A: 2d 4-node quad element with 1 dof/node (**W = 1**).
31	A: 2d 4-node quad GLS element with 1 dof/node (**W = 1**).
32*	A: 2d 8-node quad element 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 triangular element 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 triangular element with 1 dof/node (**W = 3**).
40	A: 3d 4-node tetrahedral element 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 tetrahedral element 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 element 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 lineal element with 1 dof/node for boundary convection (**W = 1**).
48	H: 3d 4-node quad element with 1 dof/node for boundary convection (**W = 2**).
49	H: 3d 3-node triangular element with 1 dof/node for boundary convection (**W = 2**).
50	H: 3d 4-node tetrahedral element with 1 dof/node (**W = 3**).
51	H: 3d 8-node brick element with 1 dof/node (**W = 4**).
52*	M: 3d 6-node triangular shell element (**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**).
65	M: 3d 2-node rigid truss (bar) element (**W = 1**).
66	M: 3d 2-node rigid beam element (**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 element (**W = 4**).
73	M: 3d 3-node rigid shell element (**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 element (**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 tetrahedral 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 (**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 (**W = 4**).
87	M: 3d 4-node membrane element with 6 dof/node (**W = 4**).
90	A: 3d 6-node wedge element with 1 dof/node (**W = 3**).
91	M: 3d 32-node serendipity brick element with 3 dof/node (**W = 6**).
92	M: 3d 26-node serendipity wedge element with 3 dof/node (**W = 5**).
93	A: 3d 32-node serendipity brick element with 1 dof/node (**W = 5**).
94	A: 3d 26-node serendipity wedge element with 1 dof/node (**W = 4**).
95	A: 3d 8-node, or 27-node, or 64-node, or 125-node hexahedral element with 1 dof/node (**W = 3, 4, 5, 6**).
96	A: 3d 4-node, or 10-node, or 20-node, or 35-node tetrahedral element with 1 dof/node (**W = 1, 2, 3, 4**).
97	M: 3d 15-node serendipity wedge element with 3 dof/node (**W = 4**).
98	A: 2d 4-node, or 9-node, or 16-node, or 25-node quadrilateral element with 1 dof/node (**W = 2, 3, 4, 5**).
99	A: 2d 3-node, or 6-node, or 10-node triangular element with 1 dof/node (**W = 2, 3, 4**).
100	M: 2d 4-node, or 9-node, or 16-node, or 25-node quadrilateral element with 3 dof/node (**W = 2, 3, 4, 5**).
101	M: 2d 3-node, or 6-node, or 10-node triangular element with 3 dof/node (**W = 2, 3, 4**).
102	M: 3d 8-node, or 27-node, or 64-node, or 125-node hexahedral element with 3 dof/node (**W = 2, 3, 4, 5**).
103	M: 3d 4-node, or 10-node, or 20-node, or 35-node tetrahedral element with 3 dof/node (**W = 1, 2, 3, 4**).
105	A: 3d 8-node, or 27-node, or 64-node, or 125-node spectral hexahedral element with 1 dof/node (**W = 2, 3, 4, 5**).
108	A: 2d 4-node, or 9-node, or 16-node, or 25-node spectral quadrilateral element with 1 dof/node (**W = 2, 3, 4, 5**).
109	H: 3d 8-node, or 27-node, or 64-node, or 125-node hexahedral element with 1 dof/node (**W = 2, 3, 4, 5**).
111	M: 3d fabric truss element with 3 dof/node (**W = 1**).
118	M: 3d 2-node planar joint element with 6 dof/node (**W = 1**).
120	M: 3d 2-node spherical joint element with 3 dof/node (**W = 1**).
121	M: 3d 2-node translational joint element with 3 rotational dof/node (**W = 1**).
122	M: 3d 2-node universal joint element with 6 dof/node (**W = 1**).
123	M: 3d 2-node revolute joint element with 6 dof/node (**W = 1**).
124	M: 3d 2-node cylindrical joint element with 6 dof/node (**W = 1**).
125	M: 3d 2-node prismatic joint element with 6 dof/node (**W = 1**).
126	M: 3d 2-node joint-with-driver element with 6 dof/node (**W = 1**).
127	M: 3d 2-node pin-in-slot joint element with 6 dof/node (**W = 1**).
128	M: 3d 4-node plane stress/plane strain quadrilateral element with 3 dof/node (**W = 1**).
129	M: 3d 3-node plane stress/plane strain triangular element with 3 dof/node (**W = 1**).
131	M: 3d 1-node discrete mass and inertia element with 6 dof/node (**W = 1**).
134	M: 3d 2-node prismatic joint-with-driver element and 6 dof/node (**W = 1**).
177	M: 3d 2-node point-to-moving-point constraint element with 3 dof/node (**W = 1**).
178	M: 3d 3-node point-to-moving-line constraint element with 3 dof/node (**W = 1**).
179	M: 3d 4-node point-to-moving-plane constraint element with 3 dof/node (**W = 1**).
179	M: 3d 4-node point-to-moving-plane constraint element with 3 dof/node (**W = 1**).
200	M: 3d 2-node uniaxial translational spring with 3 dof/node for linear or nonlinear analyses (**W = 1**).
201	M: 3d 2-node uniaxial translational spring with 6 dof/node for linear or nonlinear analyses (**W = 3**).
202	M: 3d 2-node uniaxial torsional spring with 3 rotational dof/node for linear or nonlinear analyses (**W = 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 (**W = 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 (**W = 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 (**W = 1**).
220	M: 3d 2-node spherical joint spring combination with 6 dof/node (**W = 1**).
221	M: 3d 2-node translational joint spring combination element with 6 dof/node (**W = 1**).
222	M: 3d 2-node universal joint spring combination element with 6 dof/node (**W = 1**).
223	M: 3d 2-node revolute joint spring combination element with 6 dof/node (**W = 1**).
224	M: 3d 2-node cylindrical joint spring combination element with 6 dof/node (**W = 1**).
225	M: 3d 2-node prismatic joint spring combination element with 6 dof/node (**W = 1**).
226	M: 3d 2-node revolute joint-with-actuator element and 6 dof/node (**W = 1**).
227	M: 3d 2-node pin-in-slot joint spring combination element with 6 dof/node (**W = 1**).
234	M: 3d 2-node prismatic joint-with-actuator element and 6 dof/node (**W = 1**).
301	F: 2d 4-node sloshing (fluid) quadrilateral element 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 element with 1 dof/node (**W = 3**).
312	F: 3d 3-node free-surface (fluid) triangular element with 1 dof/node for three-dimensional sloshing computations using element type = 311 (**W = 2**).
321	F: 2d 4-node hydroelastic vibration (fluid) quadrilateral element 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) tetrahedral element 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: YSST, Previous: WEIGHTS

110 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

111 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

112 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	"Id number" for the following curve (or table) (integer).
EPSR_1	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.
YSSF_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.