Dytran 2021.2

User's Guide

Corporate
5161 California Ave. Suite 200
University Research Park, Irvine, CA 92617
Telephone:(714) 540-8900
Toll Free Number: 1 855 672 7638
Email: americas.contact@mscsoftware.com

Europe, Middle East, Africa
Am Moosfeld 13
81829 Munich, Germany
Telephone: (49) 89 431 98 70
Email: europe@mscsoftware.com

Japan
Shinjuku First West 8F
23-7 Nishi Shinjuku
1-Chome, Shinjuku-Ku
Tokyo 160-0023, JAPAN
Telephone: (81) (3)-6911-1200
Email: MSCJ.Market@mscsoftware.com

Asia-Pacific
100 Beach Road
#16-05 Shaw Tower
Singapore 189702
Telephone: 65-6272-0082
Email: APAC.Contact@mscsoftware.com

Worldwide Web
www.mscsoftware.com

Support
http://www.mscsoftware.com/Contents/Services/Technical-Support/Contact-Technical-Support.aspx

DT:V2021.2:Z:Z:Z:DC-USER-PDF

Documentation Feedback

At MSC Software, we strive to produce the highest quality documentation and welcome your feedback. If you have comments or suggestions about our documentation, write to us at: documentation-feedback@mscsoftware.com.

Please include the following information with your feedback:

- Document name
- Release/Version number
- Chapter/Section name
- Topic title (for Online Help)
- Brief description of the content (for example, incomplete/incorrect information, grammatical errors, information that requires clarification or more details and so on).
- Your suggestions for correcting/improving documentation

You may also provide your feedback about MSC Software documentation by taking a short 5-minute survey at: http://msc-documentation.questionpro.com.

Note:	The above mentioned e-mail address is only for providing documentation specific feedback. If you have any technical problems, issues, or queries, please contact Technical Support.

Contents

Dytran User's Guide

1 Introduction

Overview

Dytran™ is a three-dimensional analysis code for analyzing the dynamic, nonlinear behavior of solid components, structures, and fluids. It uses explicit time integration and incorporates features that simulate a wide range of material and geometric nonlinearity.

It is particularly suitable for analyzing short, transient dynamic events that involve large deformations, a high degree of nonlinearity, and interactions between fluids and structures. Typical applications include:

- Air bag inflation
- Air bag-occupant interaction
- Sheet metal forming analysis
- Weapons design calculations, such as self-forging fragments
- Birdstrike on aerospace structures
- Hydroplaning
- Response of structures to explosive and blast loading
- High-velocity penetration
- Ship collision

Lagrangian and Eulerian solvers are available to enable modeling of both structures and fluids. Meshes within each solver can be coupled together to analyze fluid-structure interactions. Solid, shell, beam, membrane, spring, and rigid elements can be used within the Lagrangian solver to model the structure; three-dimensional Eulerian elements can be used to create Eulerian meshes. Both the Lagrangian and the Eulerian solvers can handle hydrodynamic materials and materials with shear strength.

A general material facility can be used to define a wide range of material models including linear elasticity, yield criteria, equations of state, failure and spall models, and explosive burn models. Specific material properties can also be used for elastoplastic, orthotropic composite materials.

Transient loading can be applied to the Lagrangian elements as concentrated loads and surface pressures or indirectly as enforced motion or initial conditions. Loads can be applied to material in the Eulerian mesh by pressure or flow boundaries, and initial conditions of element variables can be prescribed.

Single-point constraints can be applied to Lagrangian grid points. Rigid walls can also be created that act as barriers to either prevent the motion of Lagrangian grid points or the flow of Eulerian material.

Contact surfaces allow parts of Lagrangian meshes to interact with each other or with rigid geometric structures. This interaction may include contact, sliding with frictional effects, and separation. Single-surface contact can be used to model buckling of structures where material may fold onto itself.

Interaction between Eulerian and Lagrangian meshes is achieved by coupling. This is based on the creation of coupling surfaces on Lagrangian structures. The coupling surface, which must form a closed volume, calculates the forces arising from the interaction and then applies the forces to the material within the Eulerian mesh and the material of the Lagrangian structure.

An alternative of constituting a fluid-structure interaction is by means of Arbitrary Lagrange Euler (ALE). This is based on the interaction at a coupling surface between the structure and the Eulerian region. The Eulerian mesh is capable of following the structure by means of an ALE moving grid algorithm. A typical application where ALE is especially efficient is the birdstrike analysis.

A simple but flexible prestress facility allows structures to be initialized with an Nastran® computed prestate analysis.

A restart facility allows analyses to be run in stages.

Dytran is efficient and extensively vectorized. It provides cost-effective solutions on the latest generation of computers ranging in size from engineering workstations to the largest supercomputers. In addition, some applications can exploit the parallel processing facility for distributed memory systems that is available for simple element processing and rigid body-deformable body contact. For shared memory parallel systems, the Hughes-Liu, BLT, and Keyhoff shell formulations can use parallel processing.

This document is a user's manual for Dytran that describes the facilities available within the code and how they can be used to model the behavior of structures. This manual explains how to run the analysis and includes advice on modeling techniques, checking the data, executing the analysis, a description of the files that are produced, and methods of postprocessing. It also gives a detailed description of all available input commands.

If you need assistance in using the code, understanding the manual, obtaining additional information about a particular feature, or selecting the best way to analyze a particular problem, contact your local MSC representative. MSC offices are located throughout the world.

We welcome your suggestions for improvements to the program and documentation so that we can keep Dytran relevant to your requirements.

Features of Dytran

The main features of Dytran include:

Elements

- Euler solid elements with four, six, and eight grid points
- Lagrange solid elements with four, six, and eight grid points
- Shell and membrane elements with three and four grid points
- Beam, spring, and damper elements with two grid points
- Spotweld elements with failure
- Seat belt elements

Materials

- General material model with the definition of elastic properties, yield criterion, equation of state, spall and failure models, and explosive burn logic
- Constitutive models for elastic, elastoplastic, and orthotropic materials
- Constitutive models for multilayered composite materials
- Constitutive model for sheet metal forming applications
- Strain rate dependent material models for shells and beams

- Constitutive models for foams, honeycombs, and rubbers

Rigid Bodies

- Rigid ellipsoids
- Externally defined rigid ellipsoids
- Multifaceted rigid surfaces
- MATRIG and RBE2-`FULLRIG` rigid body definition

Constraints

- Single-point constraints
- Kinematic joints (shell/solid connections)
- Local coordinate systems
- Rigid body joints
- Drawbead model in contact

Tied Connections

- Connections to rigid ellipsoids
- Two surfaces tied together
- Grid points and surfaces tied together
- Shell edges to shell surface connections

Rigid Walls

- Rigid walls for Lagrangian elements
- Rigid barriers to Eulerian material transport

Contact and Coupling

- Contact between Lagrangian domains
- Efficient single surface contact for shell structures
- Adaptive contact with erosion and failure
- Arbitrary Lagrange-Euler (ALE) coupling
- General Euler-Lagrange coupling for fluid-structure interactions
- Contact with rigid ellipsoids
- Coupling with external programs
- Drawbead model embedded in contact
- Contact algorithm for seat belt elements

Loading

- Concentrated loads and moments
- Pressure loading
- Enforced motion
- Eulerian flow boundaries
- Body forces

Initial Conditions

- Initialize any grid-point and/or element variable
- Initialize by Nastran pre-state
- Initialize contact

Solution

- Structural subcycling
- General coupling subcycling
- Highly efficient, explicit transient solution
- Almost completely vectorized
- Dynamic relaxation for quasi-static solutions
- Simple and flexible restart procedure
- External user subroutines for advanced features
- Application sensitive default setting

Pre- and Postprocessing

- Nastran style input
- Pre- and postprocessing by Patran
- Input compatible with most modeling packages
- Free or fixed format input
- Translator for I-DEAS Version 6
- Readers for The Data Visualizer from WaveFront Technologies
- ATB output in Dytran format

Learning to Use Dytran

The simplest and quickest way to learn to use Dytran is to attend the training courses held regularly throughout the world by MSC.Software Corporation. The courses are designed to enable you to use the code quickly and reliably, to give you an in-depth understanding about how Dytran works, and to show you how to solve problems in the most efficient way with the minimum use of computer resources. For details on when and where the courses are being held, contact your local MSC representative listed at the back of this manual.

If you are unable to attend a course and have to learn to use Dytran by reading this manual, then continue reading this introduction for an overview of Dytran and how it differs from general finite element programs. Then, read those parts of this *Dytran User's Guide* that describe the features you need to use to solve your first problem, concentrating particularly on the Case Control commands and Bulk Data entries that you will use to define the input data. Chapter 9: Running the Analysis in this manual is essential reading since it describes the entire process of running an Dytran analysis from the initial modeling to postprocessing the results. Finally, while you are creating the input file, use the *Dytran Reference Manual* as a reference section to quickly locate the information needed to define the individual entries. If you are familiar with Nastran, read Similarity with Nastran, which describes the main differences between Nastran and Dytran.

Make your first problems as simple as possible and gradually increase their complexity as you build experience in using Dytran. Remember, you can always contact your local MSC representative if you need clarification on any information provided in this manual or encounter problems. Your MSC representative is there to help you!

When to Use Dytran

The time step for implicit solutions can be much larger than is possible for explicit solutions. This makes implicit methods more attractive for transient events that occur over a long time period and are dominated by low frequency structural dynamics. Explicit solutions are better for short, transient events where the effects of stress waves are important. There is, of course, an area where either method is equally advantageous and may be used.

Explicit solutions have a greater advantage over implicit solutions if the time step of the implicit solution has to be small for some reason. This may be necessary for problems that include:

- Material nonlinearity. A high degree of material nonlinearity may require a small time step for accuracy.
- Large geometric nonlinearity. Contact and friction algorithms can introduce potential instabilities, and a small time step may be needed for accuracy and stability.
- Those analyses where the physics of the problem demands a small time step (e.g. stress wave effects).
- Material and geometric nonlinearity in combination with large displacements. Convergence in implicit methods becomes more difficult to achieve as the amount of nonlinearity increases for all types.

Explicit methods have increasing advantages over implicit methods as the model gets bigger. For models containing several thousand elements and including significant nonlinearity, Dytran may provide the cheapest solution even for problems dominated by low-frequency structural dynamics.

Once Dytran is selected to analyze a particular problem, you can use the Lagrangian solver, the Eulerian solver, or Euler-Lagrange coupling.

The benefit of the Lagrangian solver is that the displacements, deformation, and stresses in structures can be monitored with a high degree of precision. However, extreme deformations may lead to drastically reduced time steps and extended run times. The Lagrangian solver should be used for structural components that may undergo large deformation and for which the dimensions, deformed geometry, and residual stress state are of major importance. Try to use the Lagrangian solver whenever possible.

The benefit of the Eulerian solver is that complex material flow can be modeled with no limit to the amount of deformation. With increasing deformation, however, the boundaries between the materials may become less precise. The Eulerian solver should be used for bodies of material, such as fluids or solids, which may experience extremely large deformations, shock wave propagation, and even changes of state.

With the coupling feature, the advantages of both solvers can be used in one analysis. This allows you to model the interaction of precisely defined structural components with fluids and highly deformable materials.

Dytran Memory Requirements

On the Windows platform, Dytran uses a dynamic memory allocation scheme. The size of the memory is preset, but you can change the memory size to be used. You can change the memory size in the input file by using the MEMORY-SIZE Executive Control Statement. (See also the *Dytran Reference Manual*).

When you do not define the memory size, Dytran, by default, allocates the memory size to small (3,000,000 integer and 2,500,000 float words). Although the memory allocation itself is fully dynamic, the size of the available (core) memory is fixed once allocated. Nevertheless, once the analysis is past the first integration cycle, the core memory is fully setup and any further memory allocations are truly dynamic.

Unlike previous versions of Dytran, you do not have to rebuild the executable file when you wish to increase or decrease the memory settings. By using either method as mentioned above, the system allocates the size as defined.

At the end of each analysis, you can find the actual memory usage (in words) at the end of the output file. The data listed represents the exact size of the memory Dytran needed to run the analysis. You can use the data as the memory settings for this analysis if you wish, or need to rerun.

Dytran and Parallel Processing

Dytran calculations can use a substantial amount of CPU time and computing resources. Therefore, optimization is an important part of the development of Dytran. One such optimization effort is parallel processing. Using parallelism allows a speed up of the analysis by harnessing the combined power of a computer with several processors (Shared Memory configuration) or clusters of computers (Distributed Memory configuration).

Dytran has been optimized to run on shared-memory computers with multiple processors. If you define to use more than one CPU for the analysis, the components that run in parallel will do so. The finite elements and the structural contact (contact version 4) can run in parallel.

Shared Memory

To exploit the parallelism available in Dytran, all you need to do is to define the number of processors that you wish to use for the analysis. On UNIX and Linux computers this can be done using a command line option (ncpus="number of CPUs). See below for an example to run a job in parallel on four CPUs.

```
dytran jid = filename ncpus = 4
```

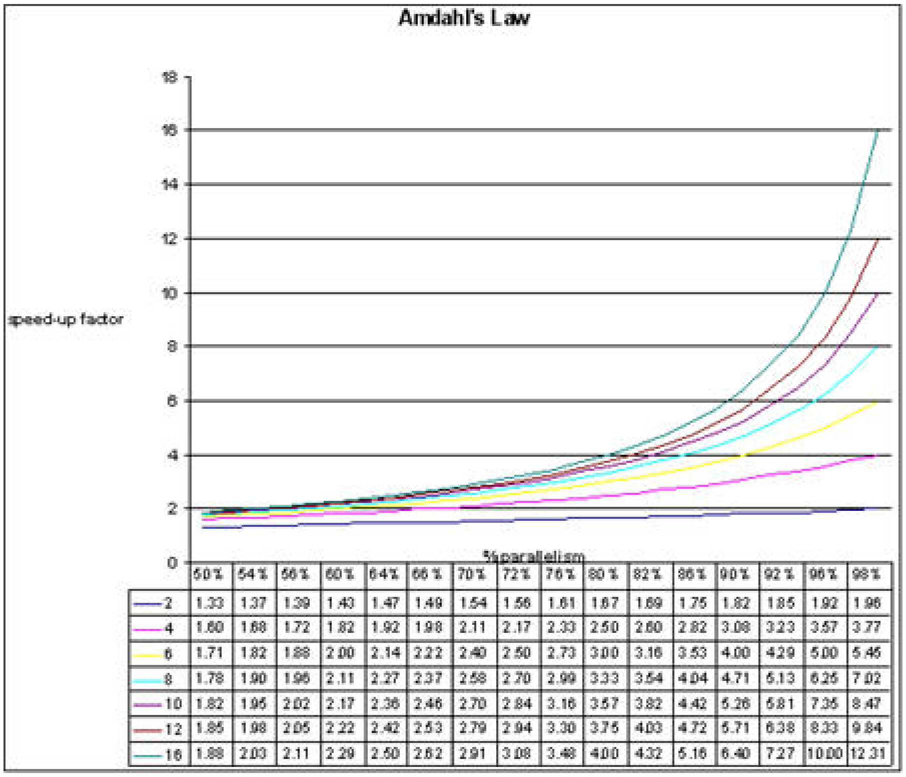
You can also define your personal default in the graphical user interface, Dytran Explorer.

The actual gain in speed up may vary with the type of problem that you wish to analyze. This is called scalability. Scalability depends on the percentage of serial (or parallel) code that the program offers. As certain components in Dytran are not (yet) running in parallel, the actual percentage of serial processing depends on the analysis. Amdahl's law defines the theoretical speed up as a function of the fraction of serial and parallel processing and the number of processors:

$$S = \frac{1}{f_s + \frac{f_p}{N}}$$

where S is the speed-up factor, f_s is the serial fraction, f_p is the parallel fraction and N denotes the number of processors. For example, when the analysis effectively runs in parallel for about 60% on two processors, the theoretical maximum speed-up factor is 1.43. At 70% parallelism on two processors, the speed-up factor increases to 1.54. Figure 1-1 below shows Amdahl's law for different number of processors and percentage parallel work.

The finite-element solvers (beams, springs, dampers, shells, membranes and solid elements) in Dytran run in shared memory parallel. The fraction of parallelism for these solvers approaches 90%. As a result, when you run a problem that only contains elements, the theoretical speed-factor on a dual processor machine is 1.8. When you add components that do not run in parallel, like for example Eulerian elements, the fraction of parallelism decreases and the theoretical speed-up factor also decreases.



	50%	54%	56%	60%	64%	66%	70%	72%	76%	80%	82%	86%	90%	92%	96%	98%
2	1.33	1.37	1.39	1.43	1.47	1.49	1.54	1.56	1.61	1.67	1.69	1.75	1.82	1.85	1.92	1.96
4	1.60	1.68	1.72	1.82	1.92	1.98	2.11	2.17	2.33	2.50	2.60	2.82	3.08	3.23	3.57	3.77
6	1.71	1.82	1.88	2.00	2.14	2.22	2.40	2.50	2.73	3.00	3.16	3.53	4.00	4.29	5.00	5.45
8	1.78	1.90	1.96	2.11	2.27	2.37	2.58	2.70	2.99	3.33	3.54	4.04	4.71	5.13	6.25	7.02
10	1.82	1.95	2.02	2.17	2.36	2.46	2.70	2.84	3.16	3.57	3.82	4.42	5.26	5.81	7.35	8.47
12	1.85	1.98	2.05	2.22	2.42	2.53	2.79	2.94	3.30	3.75	4.03	4.72	5.71	6.38	8.33	9.84
16	1.88	2.03	2.11	2.29	2.50	2.62	2.91	3.08	3.48	4.00	4.32	5.16	6.40	7.27	10.00	12.31

Figure 1-1 Amdahl's Law

The shared-memory parallel implementation in Dytran uses a dynamic processor work-balancing scheme. The aim for this scheme is to evenly spread the work over the requested number of processors. Especially for smaller problems, say less than 5,000 elements, there is still a substantial gain in speed possible. The algorithm also reverts back to serial processing when it detects the amount of work is too little spread over the requested number of processors, thus reducing the inherent parallel processing overhead.

The release notes that come with Dytran contain an overview of several real-life examples that have been benchmarked on different computer hardware and with different processor configurations.

The Parallel Execution Report

After completion of a job, a report may be written about the actual and potential distribution of work among the concurrent processes. This will show information such as the fraction of work executed on a particular number of processors. Note that for different runs this report may show different work distribution, caused by differences in overall load of the system.

To write out the parallel execution report, include a PARAM,PARALLEL,INFPAR,ON entry in your input data file. At this moment, the parallelization information option is available for quad shells solvers only. The outline of the report can be seen as follows:

```
*********************************************************************
*    *** INFORMATION ON PARALLELIZATION
*
*   CPU TIME IN PARALLEL SECTION :   0.23610E+03
*   AVERAGE PARALLEL CPUS        :   0.19929E+01
*   MAXIMUM PARALLEL CPUS        :   0.20000E+01
*   MINIMUM PARALLEL CPUS        :   0.10000E+01
*
*--FURTHER PARALLEL INFORMATION PER PROPERTY
*
*   PROPERTY NAME                : SHELL1
*   CPU TIME IN PARALLEL SECTION :   0.23610E+03
*   NUMBER OF CPUS ALLOCATED     :           2
*   CPU NUMBER                   :           1
*   CALLS INTO PARALLEL SECTION  :      143601
*   PERCENTAGE OF PARALLLEL WORK :   0.50307E+02
*   AVERAGE CALLS (MAXIMUM)      :   0.20000E+01
*   AVERAGE CALLS (MINIMUM)      :   0.10000E+01
*   NUMBER OF ELEMENTS PROCESSED :   0.91905E+07
*
*   CPU NUMBER                   :           2
*   CALLS INTO PARALLEL SECTION  :      141851
*   PERCENTAGE OF PARALLLEL WORK :   0.49693E+02
*   AVERAGE CALLS (MAXIMUM)      :   0.20000E+01
*   AVERAGE CALLS (MINIMUM)      :   0.10000E+01
*   NUMBER OF ELEMENTS PROCESSED :   0.90785E+07
*
*   PROPERTY AVERAGE OF WORK (%) :   0.10000E+03
*   PROPERTY AVERAGE OF TIME (%) :   0.10000E+03
*********************************************************************
```

Note that this CPU time information is only measured in parallel region. In this example the average parallel CPUs is close to the maximum two CPUs available. It indicates that the workload of the two CPUs used are in balance. More detailed information per CPU such as % parallel work, number of calls or elements processed, etc. are also included.

Distributed Memory

Please refer to CHAPTER 10 for details of distributed memory parallel capability.

Solvers

Dytran contains two finite element solvers, Lagrangian (finite element) and Eulerian (finite volume). In the Lagrangian solver, the grid points are fixed to locations on the body under analysis. Connecting the grid points together creates elements of material, and the collection of elements produces a mesh. As the body deforms, the grid points move in space and the elements distort. The Lagrangian solver is therefore calculating the motion of elements of constant mass.

In the Eulerian solver, the grid points are fixed in space and the elements are simply partitions of the space defined by connected grid points. The Eulerian mesh is then a fixed frame of reference. The material of a body under analysis moves through the Eulerian mesh, and the mass, momentum, and energy of the material is transported from element to element. In ALE applications, the Eulerian grid points may move in space, whereby the material flows through a moving and deforming Eulerian mesh. It is important to realize that the Eulerian grid-point motion is decoupled from the material motion.

The input for the two solvers is essentially the same. The only choice you must make is what type of property the element is to have. For example, when a solid element is to be part of a Lagrangian mesh, it is assigned a PSOLID property; however, where it is to be part of an Eulerian mesh it is assigned the PEULER property. The actual definition of the grid points and element connectivity is exactly the same for both types of solvers.

Units

Dytran does not require the model to be defined in any particular set of units. Any set of units may be used as long as it is consistent. It is advisable to use SI units whenever possible. Some examples of consistent sets of units include:

Quantity	SI	Imperial	mm/kg/s/K	mm/tonne/s/K	mm/kg/ms/K
Length	M	in ($2.54\ 10^{-2}$ m)	mm (10^{-3} m)	mm (10^{-3} m)	mm (10^{-3} m)
Time	S	s	s	s	ms (10^{-3} s)
Mass	Kg	lbf-s^2/in ($1.7513\ 10^2$ kg)	kg	tonne (10^3 kg)	kg
Angle( )	Radian	radian	radian	radian	radian
Force	kg-m/s^2	lbf (4.4482 kg-m/s^2)	kg-mm/s^2 (10^{-3} kg-m/s^2)	tonne-mm/s^2 (1 kg-m/s^2)	kg-mm/ms^2 (10^3 kg-m/s^2)
Density	kg/m^3	lbf-s^2/in^4 ($1.068\ 10^7$ kg/m^3)	kg/mm^3 (10^9 kg/m^3)	tonne/mm^3 (10^{12} kg/m^3)	kg/mm^3 (10^9 kg/m^3)
Stress	kg/m/s^2	lbf/in^2 ($6.8948\ 10^3$ kg/m/s^2)	kg/mm/s^2 (10^3 kg/m/s^2)	tonne/mm/s^2 (10^6 kg/m/s^2)	kg/mm/ms^2 (10^9 kg/m/s^2)
Energy	kg-m^2/s^2	lbf-in ($1.1298\ 10^{-1}$ kg-m^2/s^2)	kg-mm^2/s^2 (10^{-6} kg-m^2/s^2)	tonne-mm^2/s^2 (10^{-3} kg-m^2/s^2)	kg-mm^2/ms^2 (1 kg-m^2/s^2)
Temperature	°K	°R (5/9 °K)	°K	°K	°K
Spec. Heat Capacity	m^2/s^2/°K	in^2/s^2/°R ($1.1613\ 10^{-3}$ m^2/s^2/°K)	mm^2/s^2/°K (10^{-6} m^2/s^2/°K)	mm^2/s^2/°K (10^{-6} m^2/s^2/°K)	mm^2/ms^2/°K (1 m^2/s^2/°K)
Heat Convection	kg/s^3/°K	lbf/in/s/°R ($3.1523\ 10^2$ kg/s^3/°K)	kg/s^3/°K	tonne/s^3/°K (10^3 kg/s^3/°K)	kg/ms^3/°K (10^9 kg/s^3/°K)

Quantity	SI	Imperial	mm/kg/s/K	mm/tonne/s/K	mm/kg/ms/K
Thermal Conductivity	kg-m/s^3/°K	lbf/s/°R (8.0068 kg-m/s^3/°K)	kg-mm/s^3/°K (10^{-3} kg-m/s^3/°K)	tonne-mm/s^3/°K (1 kg-m/s^3/°K)	kg-mm/ms^3/°K (10^6 kg-m/s^3/°K)
Thermal Expansion	m/m/°K	in/in/°R (9/5 m/m/°K)	mm/mm/°K	mm/mm/°K	mm/mm/°K

Sometimes the standard units are not convenient to work with. For example, Young's modulus is frequently specified in terms of MegaPascals (MPa or equivalently, N/mm^2) where 1 Pascal is 1 N/m^2 . As shown in the table below, SI units are fundamental units with only conversion factors for stress and temperature

Quantity	Common Units	to	SI Units	Multiplication Factor
Length	meter (m)		meter (m)	1.0
Time	second (s)		second (s)	1.0
Mass	kilogram (kg)		kilogram (kg)	1.0
Angle	degree (°)		radian (rad)	1.745329 10^{-2}
Density	kg/m^3		kg/m^3	1.0
Force	Newton (N)		kg-m/s^2	1.0
Stress	MegaPascal (MPa)		kg/m/s^2	1.0 10^6
Temperature	Celsius (°C)		Kelvin (°K)	°K = °C + 273.15
Spec. Heat Capacity	J/kg/°C		m^2/s^2/°K	1.0
Heat Convection	W/m^2/°C		kg/s^3/°K	1.0
Thermal Conductivity	W/m/°C		kg-m/s^3/°K	1.0
Thermal Expansion	m/m/°C		m/m/°K	1.0

Imperial or American units can cause confusion, however, since the naming conventions are not as clear as in the SI system. Below you can find a conversion table that helps you to derive Imperial Units from common US units:

Quantity	US Common Units	to	Imperial Units	Multiplication Factor
Length	inch (in)		inch (in)	1.0
Time	second (s)		second (s)	1.0
Mass (1)	pound (lb)		lbf-s^2/in	2.590076 10^{-3}
Mass (2)	slug (lbf-s^2/ft)		lbf-s^2/in	8.333333 10^{-2}
Density	lb/in^3		lbf-s^2/in^4	2.590076 10^{-3}
Force	pound force (lbf)		pound force (lbf)	1.0
Stress	lbf/in^2		lbf/in^2	1.0
Temperature	Fahrenheit (°F)		Rankine (°R)	°R = 459.67 + °F
Spec. Heat Capacity	Btu/lb/°F		in^2/s^2/°R	3.605299 10^6
Heat Convection	Btu/in^2/sec/°F		lbf/in/s/°R	9336.0
Thermal Conductivity	Btu/in/s/°F		lbf/s/°R	9336.0
Thermal Expansion	in/in/°F		in/in/°R	1.0

Unit systems cannot be mixed. All the input to Dytran must be defined in the appropriate units for the chosen consistent set.

Input Format

A detailed description of the format of the input file is given in Chapter, but a brief overview is given here. The input data is stored in a text file with up to 80 characters on each line. The input is divided into the following sections: File Management Section, Executive Control Section, Case Control Section, Bulk Data Section, and Parameter Options.

File Management Section (FMS)

This section contains information concerning the file names to be used in the analysis. The section is optional and must be the first section in the input file. Each line of the file in this section is called a File Management statement.

Executive Control Section

The Executive Control Section comes between the FMS and Case Control. This section is little used in Dytran since there is no Executive System. Each line of the file in this section is called an Executive Control statement.

Case Control Section

The Case Control Section precedes the Bulk Data Section and contains information relating to the extent of the analysis and what output is required in printed form and what should be stored in files for subsequent postprocessing. Each line of the file in this section is called a Case Control command.

Bulk Data Section

The Bulk Data Section contains all the information necessary to define the finite element model — its geometry, properties, loading, and constraints. The section consists of a number of Bulk Data entries, each of which defines a particular part of the model. A single entry may occupy several lines of the input file, and it contains several fields, each of which is comprised of a single piece of data. The Bulk Data Section is usually by far the largest section in the input file.

Parameter Options

PARAM entries are defined in the Bulk Data Section. These entries are used to define various options that control aspects of the analysis. Each parameter option has a default value that is used if the option does not appear in the input file.

Grid Points

The grid points define the geometry of the analysis model. A grid point is defined on a GRID Bulk Data entry by specifying the grid-point coordinates in the basic coordinate system or in the coordinate system referred to from theGRID entry.

Coordinate Systems

The basic coordinate system is a rectangular one with its origin at (0.0, 0.0, 0.0) and its axes aligned with the x, y, and z axes of the model. This is implicitly defined within Dytran and is obtained by setting the coordinate system number to blank or zero. Local coordinate systems can be either rectangular (Figure 1-2), cylindrical (Figure 1-3), or spherical (Figure 1-4), and must be related directly or indirectly to the basic system. The CORD1R, CORD1C, and CORD1S entries are used to define rectangular, cylindrical, and spherical coordinate systems in terms of three grid points. The CORD2R, CORD2C, and CORD2S entries define the coordinate system in terms of the coordinates of three points in a previously defined coordinate system. Any number of local coordinate systems can be defined to ease the task of defining the geometry of the model. On input, the geometry of all the grid points is transformed to the basic system, and the sorted output gives the grid points positions in this system.

Degrees of Freedom (DOF)

Each grid point can have up to six displacement components — or degrees of freedom (DOF) — depending on the elements connected to it. The degrees of freedom are three translations and three rotations in a rectangular system at an individual grid point. By default, this system is aligned with the basic coordinate system. The coordinate system used to define the location of the grid point and the coordinate system to define the directions of its degrees of freedom need not be the same. The constraints acting on the grid point are in the direction of the displacement coordinate system. The displacement coordinate system is the basic system.

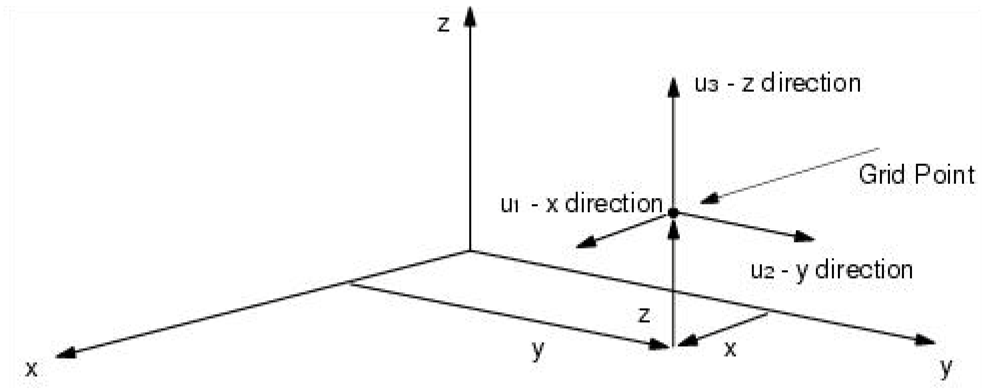


Figure 1-2 Rectangular

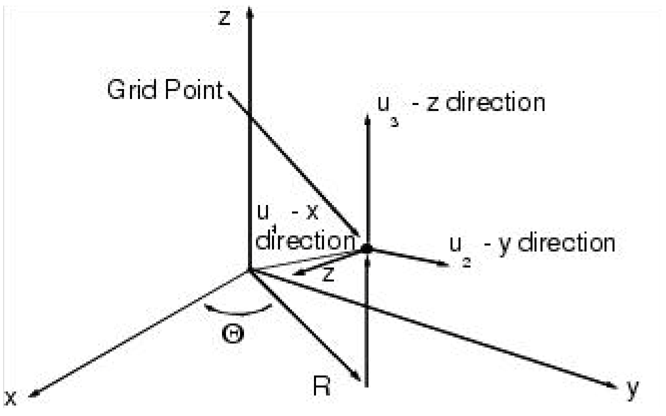


Figure 1-3 Cylindrical

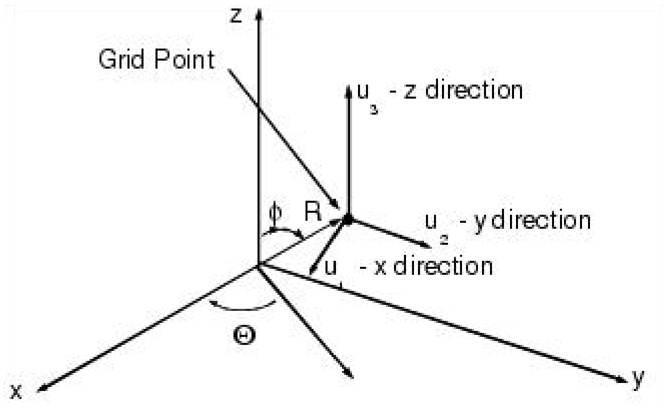


Figure 1-4 Spherical

Constraints

Permanent single-point constraints can be applied on the GRID entry and are used automatically for all solutions. Note that single-point constraints can also be applied using the SPC and SPC1 entries.

The GRDSET entry allows you to specify default values for the definition coordinate system and the single-point constraints. If a zero or blank value is encountered on a GRID entry, the default value from the GRDSET entry is used. This facility saves you entering large amounts of data, for example, in the case of plane structures where all of the out-of-plane motion is prevented.

Grid-Point Properties

Generally, the properties of the model are associated with the structural elements, rather than the grid points. However, there is one exception to this. Mass properties are input at grid points using the CONM2 entry. These masses are in addition to those arising from the density of the structural elements.

Lagrangian Solver

Grid points are the fundamental definition of the geometry of the model. The spatial coordinates of grid points are defined on GRID Bulk Data entries. Each grid point can have up to six displacement components or degrees of freedom, depending on the element to which the grid point is connected. These degrees of freedom are the three translational components and three rotational components in the basic coordinate system. Permanent single-point constraints can be applied to Lagrangian grid points using a field on the GRID entry or by using one of the SPCn entries. The grid points can be constrained in any combination of the three translational components (1,2,3) and the three rotational components (4,5,6).

Solid, plate, and beam elements can be joined together by being attached to common grid points. This connection acts as a hinge where three DOF elements (solids) are connected to six DOF elements (plates/beams). If a connection of the rotational degrees of freedom is desired, you can use the KJOIN entry.

Eulerian Solver

The definition of a grid point is common to both the Eulerian and Lagrangian solver. Grid points are the fundamental definition of the geometry of the model. The spatial coordinates of grid points are defined on GRID Bulk Data entries.

While Lagrangian grid points can have up to six displacement components, grid points used for the definition of Eulerian elements have either zero or three degrees of freedom. These grid points are a geometric device used to define the spatial position of the Eulerian mesh.

Lagrangian and Eulerian elements cannot have common grid points. If you want to connect Lagrangian and Eulerian elements, you must create separate grid points for the two element types and then use the ALE and SURFACE Bulk Data entries.

Grid Point Sequencing

The order of grid point numbering has no effect on the solution; therefore, you are free to choose any numbering system that is convenient for data generation or postprocessing. Gaps in the grid-point numbering are allowed, and you are encouraged to use a numbering system that allows you to easily identify the location of a grid point in the model from its assigned number.

Mesh Generation and Manipulation

A rectangular mesh with an equidistant grid containing CHEXA elements aligned with the basic coordinate system axes can be created using the MESH Bulk Data entry.

If you want to move certain grid points you can apply an offset to the grid-point coordinates with the GROFFS Bulk Data entry.

2 Elements

Lagrangian Elements

There are many types of Lagrangian elements available within Dytran: solid elements (CHEXA, CPENTA, CTETRA), shell elements (CQUAD4 or CTRIA3, membrane elements (CTRIA3, beam elements CBAR, CROD, CBEAM) and spring elements (CSPR, CVISC, CELAS1, CDAMP1). Most of the elements have a large strain formulation and can be used to model nonlinear effects.

Element Definition

The topology of an element is defined in terms of the grid points to which the element is connected. A "C" prefixed to the element name, such as CHEXA or CQUAD4, identifies these connectivity entries. The order of the grid points in this connectivity entry is important since it defines a local coordinate system within the element and therefore the position of the top and bottom surfaces of shell and membrane elements.

The connectivity entry references a property definition entry that may define some other geometric properties of the element, such as thickness. A "P" prefixed to the type of element (for example, PSOLID, PSHELL) identifies these entries. The property entry also references a material entry.

The material entries are used to define the properties of the materials used in the model. The material models are covered in detail in the *Dytran Theory Manual*, Chapter 3: Materials.

The elements can all be used with each other within the limits of good modeling practice. Care is needed when using solid and shell elements in a model since the solid elements only have translational degrees of freedom, while the shells have both translational and rotational degrees of freedom.

All the Lagrangian elements in Dytran are simple in their formulation; the solid and shell elements are based on trilinear and bilinear displacement interpolation, respectively. The elements are integrated at a single point at the centroid of the element.

Parabolic and other higher-order elements are not available to ensure maximum efficiency in the solution. The explicit formulation of Dytran requires many time steps in an analysis, perhaps in excess of 100000. It is vital, therefore, that each step is as efficient as possible. It has been shown that a larger number of simple elements produces a cheaper solution than a smaller number of more complex elements.

Users of Nastran should note that although the Dytran elements have the same names as those in Nastran, they are different in their formulation and behavior.

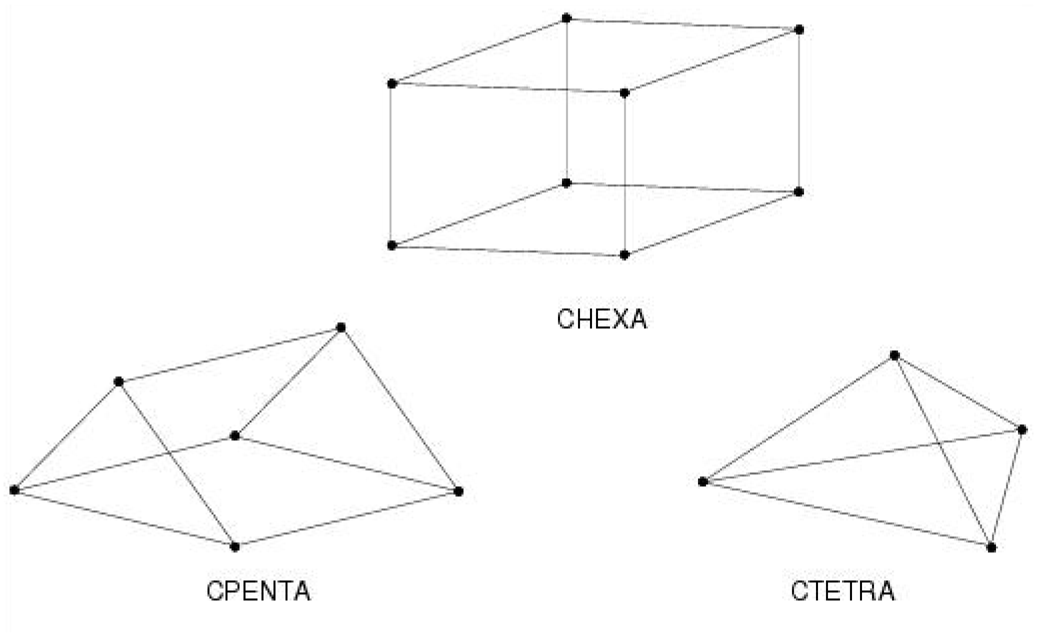
Explicit models tend to have fine meshes in regions of high plasticity or internal contacts since simple, constant force or moment elements are used.

Solid Elements

Dytran has three different forms of solid elements, which are shown below:

CHEXA	Six-sided solid element with eight grid points
CPENTA	Five-sided solid element with six grid points
CTETRA	Four-sided solid elements with four grid points

The PSOLID entry is used to assign material properties to the element.



The elements use one-point Gaussian quadrature to integrate the gradient/divergence operator. The Gauss point is located at the element centroid.

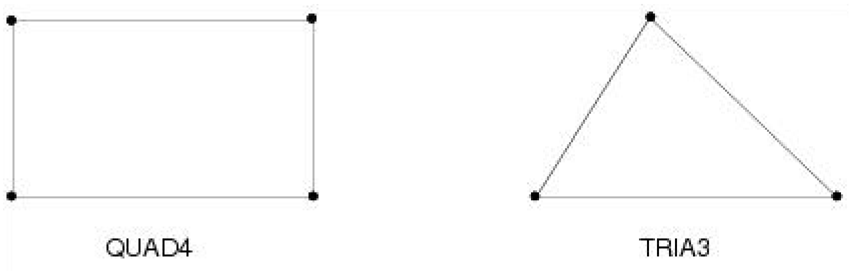
The CPENTA and CTETRA elements are degenerate forms of the CHEXA element where the grid points of the element are coincident. These elements have significantly reduced performance compared to the CHEXA element and should only be used when absolutely necessary and should be placed well away from any areas of interest. The CTETRA element in particular tends to be too stiff and should be avoided if possible. With practice, it is possible to mesh solid regions with very complex geometry using CHEXA elements only.

The elements can be distorted to virtually any shape, although their performance is best when they are close to cuboidal. Elements inevitably become distorted during the analysis, but the code does not perform any checks on element shape, which ensures that the analysis does not abort due to one or two badly distorted elements. Therefore, the burden is on the user to ensure that the elements have sensible shapes both before and during the analysis.

Shell Elements

Two shell elements are available in Dytran: CQUAD4, a quadrilateral shell element with four grid points, and CTRIA3, a triangular shell element with three grid points. The CQUAD4 element uses the Belytschko-Tsay, Hughes-Liu, or Key-Hoff formulation, while the CTRIA3 uses the CO-triangle formulation.

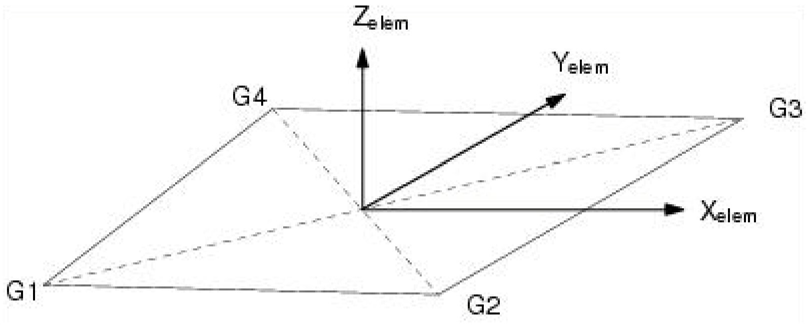
Of the various shell formulations, the Belytschko-Tsay is the most efficient and should be used in most situations. The Key-Hoff is more expensive, but performs better at large strains (over 5%). When a part of the structure suffers very large straining, you should consider using Key-Hoff shells in that area and Belytschko-Tsay shells elsewhere. The Hughes-Liu shell is substantially more expensive than the previous ones and offers an advantage only if the thickness varies within the element.



The PSHELLn or PCOMP entry is used to assign properties to the element.

Element Coordinate System

The connectivity of the Belytschko-Tsay and Hughes-Liu element, as input on the CQUAD4 or CTRIA3 entry, defines the element coordinate system. It is a rectangular coordinate system, and the direction of axes depends on the order of the grid points in the connectivity entry. The z-axis is perpendicular to the two diagonals of the element, which are given by the vectors from grid point 1 to grid point 3 and from grid point 2 to grid point 4. The x-axis is the vector from grid point 1 to grid point 2. The x-axis is always forced to be orthogonal with the z-axis. The y-axis is perpendicular to both the x-axis and the z-axis and is in the direction defined by the right-hand rule.



Each element has its own coordinate system. The top surface of a shell element is defined in the positive z-direction and the bottom surface is in the negative z-direction. The element coordinate system for the Key-Hoff and the shared-memory parallel version of Belytschko-Tsay element defines the x-axis as the line connecting the midpoints of sides G1-G4 and G2-G3.

Membrane Elements

The CTRIA3 element can be specified as a membrane element rather than a normal shell element. This membrane element uses a different formulation that allows the element to carry in-plane loads but no bending stiffness.

Triangular membrane elements are not large strain elements, and therefore the in-plane deformations should be small. Membrane elements can only be elastic.

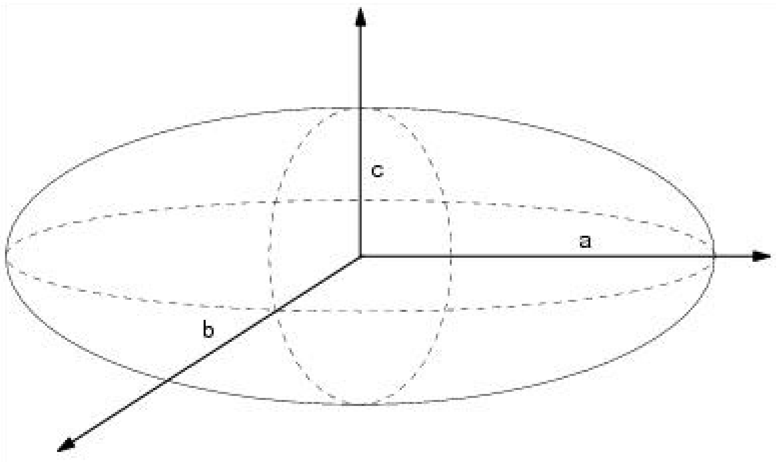
Rigid Structures

Rigid structures are one of the most versatile features in Dytran. Rigids are basically nondeformable structures that can have a user-defined arbitrary shape or have a pre-defined shape such as Rigid Ellipsoids.

Rigid Ellipsoids

A rigid ellipsoid is defined on the RELLIPS Bulk Data entry. The definition consists of the ellipsoid name, mass, orientation in space, and the shape. The ellipsoid orientation is determined by the longest and the shortest axis direction. The shape is defined by three numbers (a , b , and c where $a \geq b \geq c$) which define the length of the axes. In addition, the rotational and/or translational motion of the rigid ellipsoid can be specified. The moments of inertia of the ellipsoid are calculated under the assumption that the mass is evenly distributed over the body.

The initial velocities can be specified in either the basic coordinate system or the body's own coordinate system defined by the vectors of the major and minor axes.



The RELLIPS entry allows the body to be defined in an external program. Only the name of the body is required on the input entry. These are normally used for modeling of anthropomorphic dummies. This can be done by coupling Dytran by using ATB, which is included in Dytran. For ATB, see Chapter 7: Interface to Other Applications, ATB Occupant Modeling Program.

Specific grid points or rigid bodies can be connected to rigid ellipsoids using the RCONREL entry. Contact with rigid ellipsoids can be defined through the use of the CONTREL entry.

Rigid Bodies

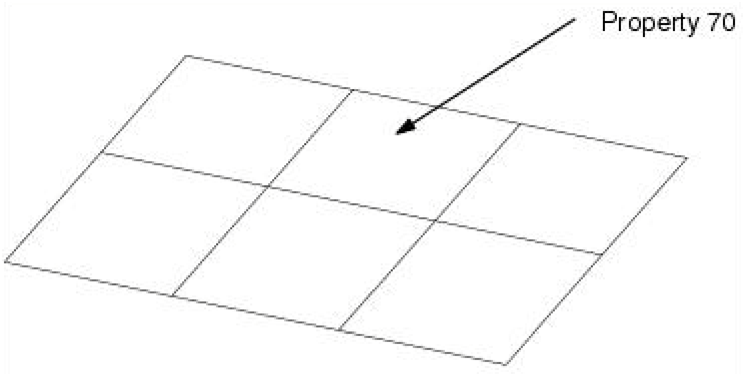
While rigid ellipsoids are geometric entities of a fixed form, rigid bodies are user-defined surfaces that are specified as rigid. A rigid body can have almost any shape as determined by the surface from which it is made.

RIGID

The RIGID entry defines the mass, center of gravity, and inertia tensor of the body and references a surface that describes the body's shape.

The surface is defined on the SURFACE entry.

For example, the following data defines a rigid plate.

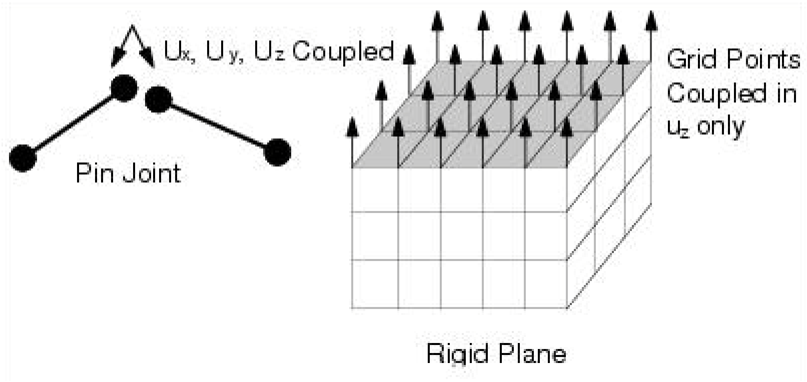


```
RIGID, 1, 100, 359, ,5, 2.5, 0.0, , +
+, , , , , , , , , +
+, 4495., , , 4495., , 4495.
SURFACE, 100, , PROP, 100
SET1, 100, 70
PSHELL1, 70, , DUMMY
CQUAD4, 1, 70, 1, 2, 12, 11
```

When a CONTACT entry references the same surface number as the RIGID entry, the body is also included in the contact surface and may interact with the other defined surfaces. Similarly, when the surface is referenced in a COUPLE or ALE entry, the rigid body is coupled to an Eulerian mesh.

Rigid Element — RBE2

Particular degrees of freedom on grid points can be specified to have the same displacement using the RBE2 entry. The degrees of freedom attached to the RBE2 move the same amount throughout the analysis. This facility can be used, for example, to model pin joints and rigid planes:


U_x, U_y, U_z Coupled
Pin Joint
Grid Points Coupled in u_z only
Rigid Plane

For rigid elements, the motion of all the degrees of freedom that are coupled is obtained by averaging their unconstrained motion. The rigid element constraints act in the basic coordinate system.

In the RBE2 plane shown above, all the grid points in the plane will have the same displacement so the plane itself will not rotate. When rotation is required, you must use RBE2 (FULLRIG), RIGID, or MATRIG.

The location of the grid points is irrelevant, but you must be careful not to over constrain the model. In the rigid plane shown above, all the grid points in the plane must have the same displacement so the plane itself cannot rotate. When rotation is required, you must use rigid elements.

There are a number of restrictions needed when using rigid elements. No grid point connected to an RBE2 can be

- Subjected to enforced motion
- Attached to a rigid body
- Attached to a tied connection
- A secondary contact for a rigid wall

Also, if a degree of freedom on one grid point in an RBE2 is constrained, that degree of freedom on all of the other grid points in the RBE2 should also be constrained. The RBE2 does not automatically constrain the other grid points in that RBE2 since it averages the motion of all grid points.

Translational and rotational degrees of freedom can be coupled.

An RBE2 definition using the FULLRIG option couples all degrees of freedom. All grid points defined on the RBE2 entry together behave as a rigid body.

The PARAM,CFULLRIG entry automatically converts all 123456 constraints on a normal RBE2 to the FULLRIG option.

An RBE2-FULLRIG entry can be merged with other RBE2-FULLRIG entries and with MATRIG entries into one rigid assembly by using PARAM,MATRMERG or PARAM,MATRMRG1. (See the following explanation for MATRIG)

RBE2-FULLRIG basically behaves in the same way as MATRIG. The only difference is that the grid points of an RBE2-FULLRIG are attached to elements that have deformable materials. Therefore, RBE2-FULLRIG is more expensive to use than MATRIG that can skip the whole material solver. In addition, for an element with a deformable material whose grid points belong to one RBE2-FULLRIG, the stresses and strains should vanish. In practice however, there can be spurious noise due to the discretization of the nodal rotations from one cycle to the next. It is advised, therefore, to use MATRIG instead of RBE2-FULLRIG, when possible.

MATRIG

Parts of the mesh can be made rigid by replacing the material definition with a MATRIG entry. All elements referred to by the MATRIG material number behave as a rigid body. This can be convenient in situations where large rigid body motions arise, which are expensive to simulate with deformable elements.

MATRIG definitions can also be merged. In this case, the set of MATRIG entries behaves as one rigid body. In addition, MATRIG entries can also be merged with RBE2 entries which have the FULLRIG option. Merging can be achieved with PARAM,MATRMERG or PARAM,MATRMRG1. The PARAM,MATRMERG merges all MATRIG and RBE2-FULLRIG definitions which are mentioned on the entry in a new rigid assembly. The properties (mass, center of gravity and moments of inertia) are computed from the properties of each of the individual merged definitions. The PARAM,MATRMRG1 entry performs the same merging but there can be pre-defined properties for the new rigid assembly.

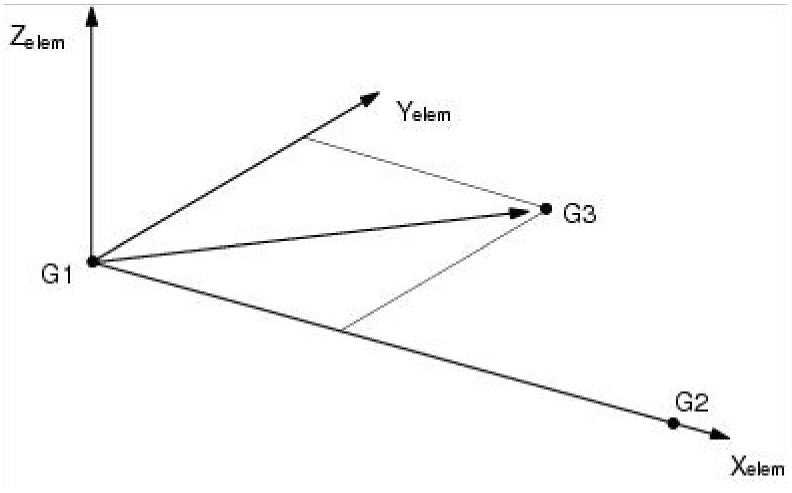
Beam Elements

The beam element is defined using either the CBAR or CBEAM entry. Both have the same effect and define the same element. CBAR is easier to use and is recommended for this reason. The CBEAM entry allows compatibility with modeling packages that do not use the CBAR entry. The properties of the beam can be defined using the PBAR, PBEAM, or PBEAM1 entry. Only the basic data used for the PBAR entry is extracted from PBEAM; the additional features of PBEAM available in Nastran are not used in Dytran.

Element Coordinate System

The beam element connects two grid points, but you must define the orientation of the beam and its element coordinate system. The definition can be done in two ways:

- Use a third grid point in the xy-plane.
- Use a vector in the xy-plane.



The element x-axis is aligned with the direction of `G1` to `G2`. A vector with its origin at `G1` is either defined explicitly or by defining a third grid point, in which case the vector is from `G1` to `G3`. This vector defines the xy- plane with the element y- axis perpendicular to the element x- axis. The element z-axis is perpendicular to both the element's x- and y- axis.

The element coordinate system is defined at the start of the calculation. It is automatically updated depending on the distortion of the beam during the analysis.

Formulations

There are two types of beam formulations:

- Belytschko-Schwer
- Hughes-Liu

The element material can either be defined as elastic by referencing a MAT1 entry, or as elastoplastic by referencing a DMATEP entry.

If an elastoplastic material is specified for Belytschko-Schwer beams, a resultant plasticity model is used, whereby the entire cross-section yields at once. It is not possible to choose a strain rate dependent yield model for elastoplastic Belytschko-Schwer beams.

Rod Elements

A rod element can be defined using a CROD entry. A rod connects two grid points and can carry only axial tension and compression. It cannot carry any torsion or bending; for torsion or bending, the CBARor CBEAM element should be used.

The only required property is the cross-sectional area of the rod that is specified using the PROD entry.

Spring Elements

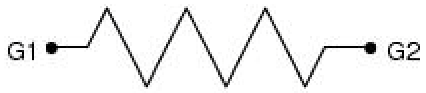
There are two types of spring elements available in Dytran: the CSPR and CELAS spring elements.

CSPR spring elements only connect translational degrees of freedom. The CELAS spring elements can connect both translational and rotational degrees of freedom. For rotational springs, you should define the moment/angle characteristic. In the remainder of this section, force and displacement are described for simplicity. You should substitute these terms by moment and angle for rotational springs.

The spring properties are defined using PSPR or PELAS entries. There are three types of spring elements available: linear, nonlinear, and user-defined spring elements.

CSPR Elements

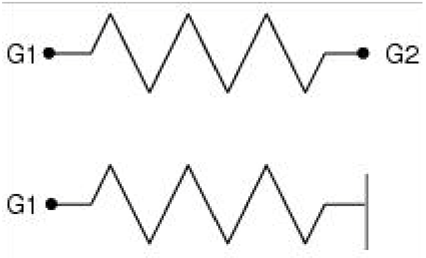
The CSPR element always connects two grid points and defines the force/deflection characteristic between the two points. The force always acts in the direction of the line connecting the grid points. As the position of the grid points changes during the analysis, the line of action of the force changes as well. The CSPR element is similar to the CROD element except that the force/deflection characteristic is defined directly rather than defining the area and material properties.



The spring properties are defined using PSPR entries. There are three types of springs: one linear, one nonlinear, and one that is defined via a user subroutine.

CELAS1 and CELAS2 Elements

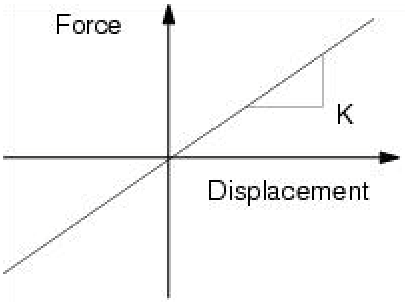
The CELAS elements connect either one or two grid points. If only one grid point is specified, the spring is grounded. In addition, you must specify the direction of the spring. The force in the spring always acts in this direction regardless of the motion of the grid points during the analysis.



The CELAS1 and CELAS2 elements are linear springs. The spring characteristic from a CELAS1 spring element is defined by referring to a PELAS entry. The spring characteristic for a CDAMP2 spring element is defined on the CDAMP2 entry directly.

Linear Elastic Springs (PSPR and PELAS)

The force is proportional to the displacement of the spring.



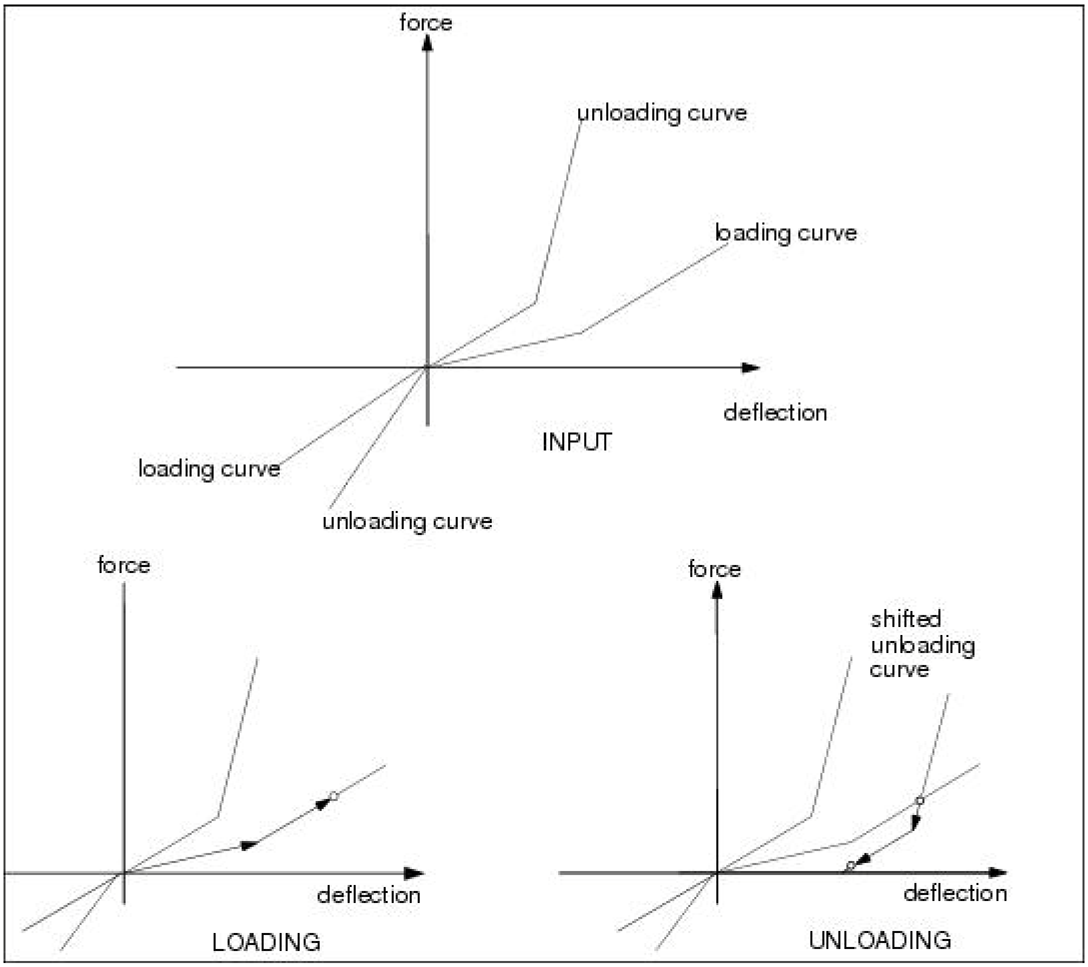
You must define the stiffness K of the spring.

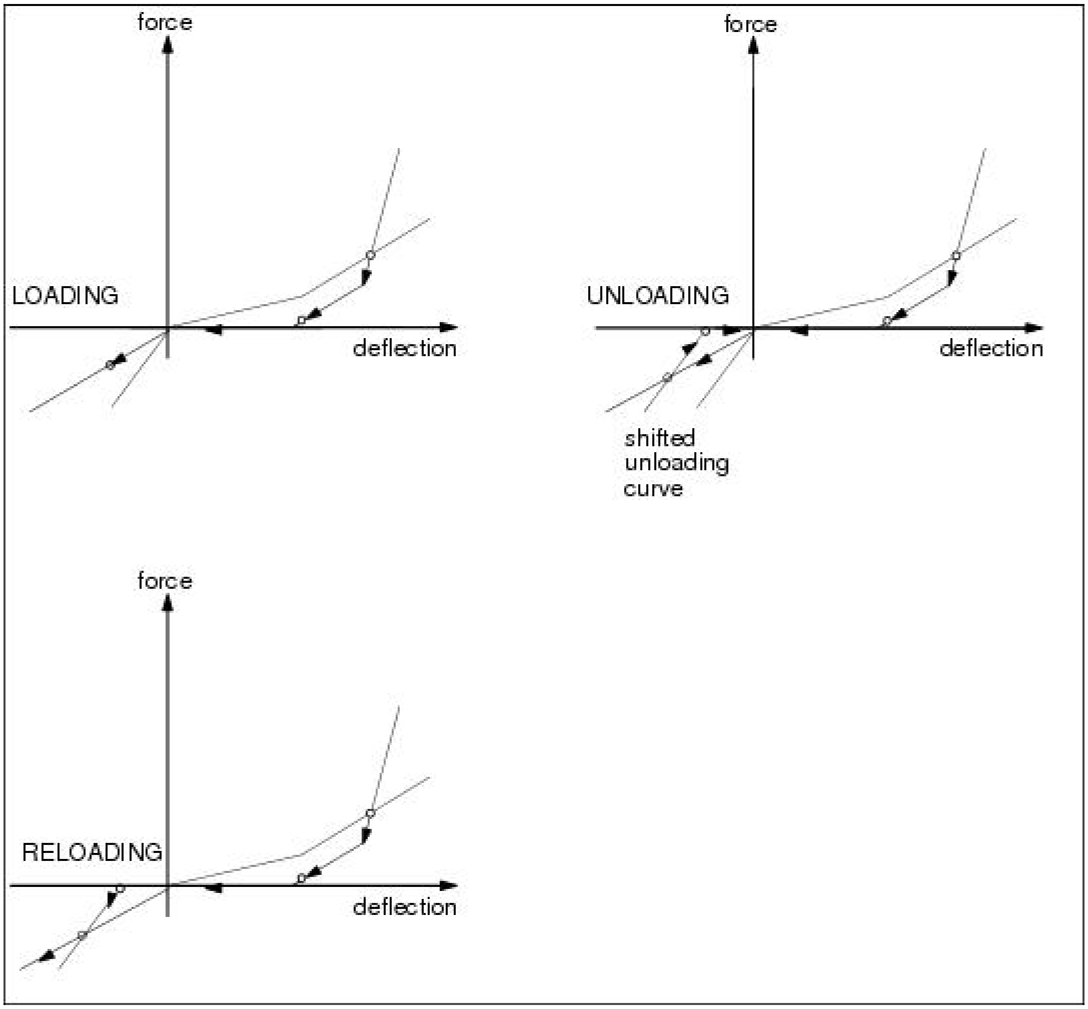
Nonlinear Elastic Springs (PSPR1, PELAS1)

The nonlinear `PSPRI` and `PELAS1` entries can refer to a loading curve and an unloading curve. The forces are not proportional to the displacement.

The force/deflection characteristic can be of any shape and is defined by specifying a table of force/deflection values using a TABLED1 entry. Loading and unloading occurs corresponding to the curves. If the unloading table is not defined, unloading occurs corresponding to the loading curve. Input for loading and unloading must be consistent. Both curves must be either completely defined or have only positive values (start from (0.,0.)). When only positive values are defined, the curves are automatically mirrored. It is suggested to either define the entire curve in both tension and compression. The force associated with a particular displacement is determined by linear interpolation within the table range or by using the end point values outside the table range.

Upon unloading, the unloading curve is shifted long the deflection axis until it intersects the loading curve at the point from which unloading commenced. An example of a typical load, unload, and reload sequence is shown as follows


force
unloading curve
loading curve
deflection
INPUT
loading curve
unloading curve
force
deflection
LOADING
force
shifted unloading curve
deflection
UNLOADING



The unloading table is applied for unloading and reloading until the deflection again exceeds the point of intersection. At further loading, the loading is applied.

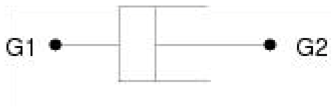
The area enclosed between the loading and unloading curves represents an energy loss this is the hysteresis portion.

Damper Elements

There are two types of damper elements available in Dytran: `CVISC` and the `CDAMP` damper elements.

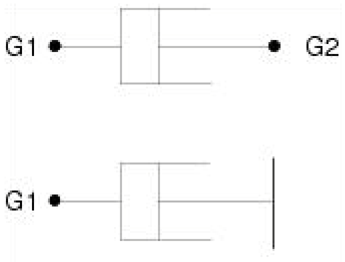
The CVISC damper elements connect translational degrees of freedom only. The `PELAS` damper elements can connect both translational and rotational degrees of freedom. For translational dampers, you should define the force/velocity characteristic. For rotational dampers, you should define the moment/angular velocity characteristic. In the remainder of the section, the force and velocity are described for simplicity. You should substitute these terms with moment and angular velocity for rotational dampers.

The damper properties are defined using PVISC or PDAMP entries. There are three types of dampers available: linear, nonlinear, and user-defined dampers.



CDAMP1 and CDAMP2 Element

The CDAMP elements connect either one or two grid points and are the equivalent of the CELAS spring elements. If only one grid point is specified, the damper is grounded. In addition, you must specify the direction of the damper. The damping force always acts in this direction regardless of the motion of the grid points during the analysis.



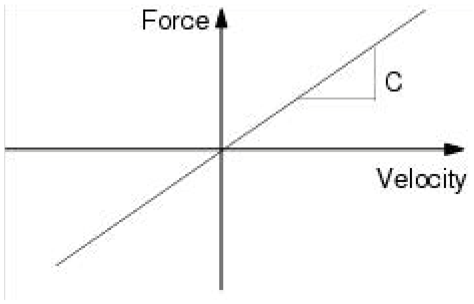
The CDAMP1 and CDAMP2 elements are linear dampers. The damper characteristic for CDAMP1 element is defined by referring to a PDAMP entry. For a CDAMP2 element, the damper characteristic is defined on the CDAMP2 entry directly.

The damper properties are defined using PVISC and PDAMP entries. There are three types of dampers: linear, nonlinear, and one that is defined using a user subroutine.

Linear Dampers (PVISC and PDAMP)

The force is proportional to the relative velocity of the end points.

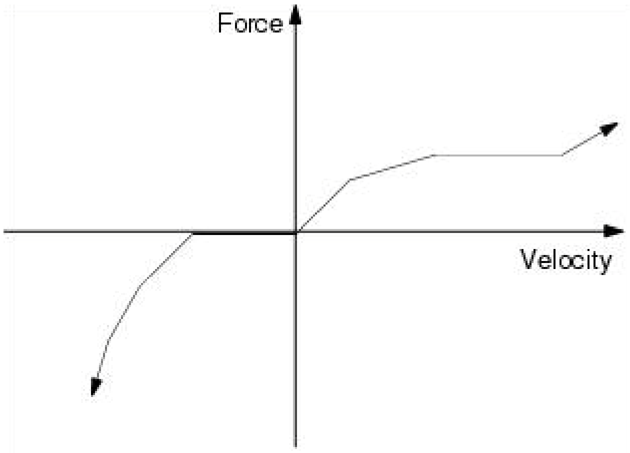
You must define the damping constant C.



Nonlinear Dampers (PVISC1)

The force/velocity characteristic is nonlinear.

The force/velocity characteristic can be of any shape and is defined by specifying a table of force/velocity values using a TABLED1 entry. You must specify the entire curve in both tension and compression. The force associated with a particular velocity is determined by linear interpolation within the table range or by using the end point values outside the table range.



Lumped Masses

Additional mass and inertia can be applied to a grid point using the CONM2 entry.

All grid points in the model have mass, either by the properties of the structural elements attached to the grid points or by using a CONM2 entry. If, for example, a spring is connected at a grid point and there is no other element attached to the grid point, a CONM2 entry is used to define the mass at that grid point.

Eulerian Elements

In the Eulerian solver, grid points and solid elements define the mesh. The elements are specified as being (partially) filled with certain materials or with nothing (VOID), and initial conditions are defined.

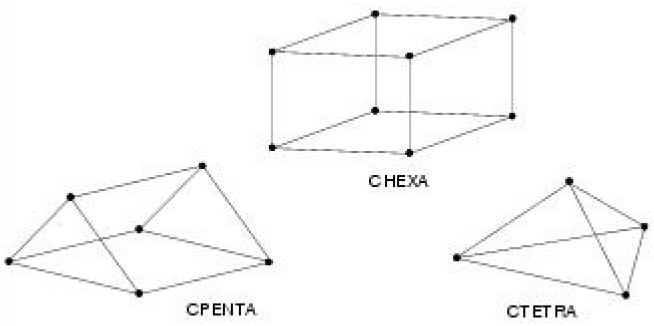
As the calculation proceeds, the material moves relative to the Eulerian mesh. The mass, momentum, and energy of the material is transported from element to element depending on the direction and velocity of the material flow. Dytran then calculates the impulse and work done on each of the faces of every Eulerian element.

Eulerian elements can only be solid but have a general connectivity and therefore are defined in exactly the same way as Lagrangian elements.

Solid Elements

There are three types of Euler elements, a six-sided CHEXA with eight grid points defining the corners, a CPENTA with six grid points, and a CTETRA with four grid points. The connectivity of the element is defined in exactly the same manner as a Lagrangian element, that is, with a CHEXA, CPENTA, or CTETRA entry. However, in order to differentiate between Lagrangian and Eulerian solid elements, the property entry for Euler is PEULER rather than PSOLID.

Unlike Lagrangian solid elements, the CPENTA and CTETRA elements perform just as well as the CHEXA element. They can be used, therefore, wherever meshing demands such use.



The PEULER entry references a DMAT material entry that is used to define the material filling the elements at the start of the calculation. When no material entry is referenced (the field contains a zero), the element is initially void.

Supported Elements in Material Models

The elements in the model must have properties that describe the element's behavior. Many materials can be modeled using the material models in Dytran. Below is a list of materials and the associated required material model entries.

Isotropic Elastic Material	
Shell and membrane elements	DMATEL
Lagrangian solid elements	DMAT + EOSNA + SHREL
Eulerian solid elements	DMAT + EOSNA + SHREL
Isotropic Nonlinear Elastic Material	
Lagrangian solid elements	DMAT + EOSNA + SHRPOL
Eulerian solid elements	DMAT + EOSNA + SHRPOL
Isotropic Fluid Material	
Lagrangian solid elements	DMAT + EOSNA
Eulerian solid elements	DMAT + EOSNA

Orthotropic Elastic Material	
Shell and membrane elements	MAT8
Lagrangian solid elements	DMATOR
Composite Material	
Shell elements	MAT8
Composite Material with Damage	
Shell elements	MAT8 + MAT8A
Anisotropic Elastic Material (Classical Laminate Theory)	
Shell elements	MAT8, MAT2
Isotropic Elastoplastic Material	
Beam elements	DMATEP + YLDVM,DYMAT24
Shell elements	DMATEP + YLDVM,DYMAT24
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDVM, DYMAT24
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDVM
Isotropic Elastoplastic Material with Failure	
Beam elements	DMATEP + YLDVM + FAILEX, DYMAT24
Shell elements	DMATEP + YLDVM + FAILEX, DYMAT24
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDVM + FAILEX, DYMAT24
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDVM + FAILEX
Kinematic/Isotropic Plasticity	
Beam elements	DMATEP + YLDVM,DYMAT24
Shell elements	DMATEP + YLDVM,DYMAT24
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDVM,DYMAT24
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDVM
Resultant Plasticity	
Beam elements	DMATEP
Rate Power Law Plasticity Model	

Shell elements	DMATEP + YLDRPL
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDRPL
Eulerian solid elements	DMATEPYLDRPL
Johnson/Cook Plasticity Model	
Shell elements	DMATEP + YLDJC
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDJC
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDJC
Mohr-Coulomb Plasticity Model	
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDMC
Tanimura/Mimura Plasticity Model	
Shell elements	DMATEP + YLDTM
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDTM
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDTM
Zerilli/Armstrong Plasticity Model	
Shell elements	DMATEP + YLDZA
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDZA
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDZA
User-defined Plasticity	
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDEX
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDEX
Strain-rate Dependent Plasticity	
Beam elements	DYMAT24
Shell elements	DYMAT24
Lagrangian solid elements	DYMAT24
Piece-wise Linear Plasticity	
Beam elements	DMATEP + YLDVM,DYMAT24
Shell elements	DMATEP + YLDVM,DYMAT24
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDVM,DYMAT24
Eulerian solid elements	DMAT + EOSNA + SHREL + YLDVM

Piece-wise Linear Plasticity with Isotropic Hardening	
Beam elements	DMATEP + YLDVM,DYMAT24
Shell elements	DMATEP + YLDVM,DYMAT24
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDVM,DYMAT24
Piece-wise Linear Plasticity with Isotropic Hardening and Failure	
Beam elements	DMATEP + YLDVM + FAILEX,DYMAT24 + FAILEX
Shell elements	DMATEP + YLDVM + FAILEX,DYMAT24 + FAILEX
Lagrangian solid elements	DMAT + EOSNA + SHREL + YLDVM + FAILEX,DYMAT24 + FAILEX
Nonlinear Plasticity	
Lagrangian solid elements	DMAT + EOSNA + SHRPOL + YLDPOL
Eulerian solid elements	DMAT + EOSNA + SHRPOL + YLDPOL
Anisotropic Plasticity Model (Sheet-metal - Krieg)	
Shell elements	SHEETMAT + BARLT89
Anisotropic Plasticity Model with FLD (Sheet-metal - Krieg)	
Shell elements	SHEETMAT + BARLT89
Linear Viscoelastic Material Model	
Lagrangian solid elements	DMAT + EOSNA + SHRLVE
Soil And Concrete (Cap) Material Model	
Lagrangian solid elements	DYMAT25
Soil and Crushable Foam	
Lagrangian solid elements	DYMAT14
Soil and Crushable Foam with Failure	
Lagrangian solid elements	DYMAT14
Combustible material	
Eulerian solid	DMAT + EOSDEF

Crushable Foam (Polypropylene)	
Lagrangian solid elements	FOAM1
Crushable Foam with Hysteresis and Strain Rate Dependency	
Lagrangian solid elements	FOAM2
Orthotropic Crushable Material	
Lagrangian solid elements	DYMAT26
Mooney-Rivlin Rubber	
Lagrangian solid elements	RUBBER1
Explosive Material (JWL)	
Eulerian solid elements	DMAT + EOSJWL
Ignition and Growth Explosive Material	
Lagrangian solid elements	DMAT + EOSIG
Polynomial Equation of State	
Lagrangian solid elements	DMAT + EOSNA
Eulerian solid elements	DMAT + EOSNA
Polynomial Equation of State with Viscosity	
Eulerian solid elements	DMAT + EOSNA
Tait's Equation of State with Cavitation Model	
Lagrangian solid elements	DMAT + EOSTAIT
Eulerian solid elements	DMAT + EOSTAIT
Tait's Equation of State with Cavitation and Viscosity	
Eulerian solid elements	DMAT + EOSTAIT
User-defined Equation of State	
Eulerian solid elements	DMAT + EOSEX
Perfect Gas	
Lagrangian solid elements	DMAT + EOSGAM

Eulerian solid elements	DMAT + EOSGAM
Rigid Material	
Beam elements	MATRIG
Shell elements	MATRIG
Lagrangian solid elements	MATRIG

In addition, for all material definitions under Lagrangian solid and (quad) shell elements, you can combine the associated material model entries with a failure model based on the minimum time step (see *Dytran Theory Manual*,Chapter 4: Models, Material Failure). This failure model can be defined from `PARAM,FAILDT` entry.

Choice of Constitutive Model

As becomes evident from the list above, there are many material models available. It may therefore be difficult to select the most appropriate material model for a specific element type.

The main rule to follow when selecting a material model is to keep it as simple as possible. Simple models are much more efficient since they require fewer calculations, and it is often easier to understand their behavior. You should also consider how accurate your knowledge of material properties is. No matter how sophisticated the material model and the formulation of the elements, the results can only be as accurate as your input data. The large strain properties of materials under dynamic cyclic loading at high strain rates represent area where little information is available. It often requires special testing. Such tests are difficult to carry out and may have a large margin of error associated with them. If you do not have sufficient confidence in your material properties, use a relatively simple material model and consider running several analyses with different models and assumptions to see how sensitive the results are to the input data.

For historical reasons, Dytran is also able to address the materials that originated from Dyna (Dyna) using the material name definitions from Dyna. The Dyna Version 3 materials listed in the Materials section can be addressed in Dytran using the exact same format as used in Dyna. Materials listed in parentheses are implemented in Dytran by the same name and appear in Chapter 5: Bulk Data Entry Descriptions of the *Dytran Reference Manual*. The other material definitions are mapped to equivalent Dytran materials and are described in the Chapter 5: Bulk Data Entry Descriptions of the *Dytran Reference Manual* under the Dytran name.

Dyna Version 3	Material Description	Dytran
MAT1	Isotropic, linear, elastic material.	(MAT1)
DYMAT1	Isotropic, elastic material.	DMATEP
DYMAT2	Orthotropic, elastic material. (Solid Lagrangian elements.)	DMATOR
DYMAT3	Elastoplastic, nonlinear material with isotropic hardening.	DMATEP
DYMAT5	Nonlinear, elastic perfectly plastic soil and crushable foam. Crushing under hydrostatic loading, elastoplastic under deviatoric loading.	DYMAT14

Dyna Version 3	Material Description	Dytran
DYMAT6	Viscoelastic material	DMAT + SHRLVE
DYMAT12	Elastoplastic, nonlinear material with isotropic hardening.	DMATEP
DYMAT12A	Like DYMAT12, but the shear and bulk modulus define the material behavior.	DMATEP
DYMAT13	Nonlinear, isotropic, elastotropic material with failure.	DMATEP
DYMAT13A	Like DYMAT13 but the shear and bulk modulus define the material behavior.	DMATEP
DYMAT14	Nonlinear, elastic perfectly plastic, compressible soil and crushable foam, with failure. Crushing under hydrostatic loading, elastoplastic under deviatoric loading. (Solid Lagrangian elements).	(DYMAT14)
DYMAT24	Elastoplastic, nonlinear, plastic material with isotropic hardening. Stress-strain curve is piecewise linear.	(DYMAT24)
DYMAT26	Orthotropic crushable material. (Solid Lagrangian elements.)	(DYMAT26)

Graded Meshes in Euler

Nonuniform Euler meshes can easily be created by Patran. This is particularly useful when the elements need not be orthogonal. But in orthogonal meshes the nonuniformity propagates to the boundaries of the mesh. Even at the boundary, there will be elements that have a small size in at least one direction. To allow for large element sizes in all three coordinate directions at the boundary, block-structured meshing has to be used. This type of meshing is very effective when modeling the flow over bodies and is often used in CFD. Usually, the Euler elements are fine near the body and become coarser as the distance is increased away from the body. By using the graded mesh capability, a block of fine elements is used in the area of interest and a coarse block is used in other areas. To use this method, the blocks need to be glued together. This is done by adding PARAM,GRADED-MESH.

Graded meshes are supported by all Eulerian solvers except by the single material Euler solver with Strength. Intersection of a coupling surface segment with the interface between a coarse and fine block is not allowed. Graded meshes are not supported by multiple Euler domains. Figure 2-1 shows an example of graded meshes. This mesh is be used in the Using Euler Archive Import in Blast Wave Analyses (EP4_14) example in Chapter 4: Fluid-structure Interaction in the *Dytran Example Problem Manual*.

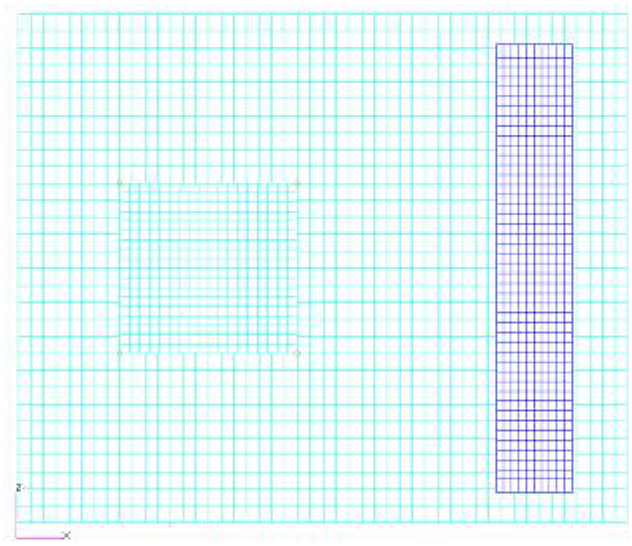


Figure 2-1 Graded Mesh with Structure

Consider a blast wave simulation. Close to the ignition point elements need to be fine; but, at some distance, the blast wave becomes larger in radius and it becomes less steep allowing coarser mesh elements. To reduce the number of elements and to limit the problem size, part of the fine mesh can be replaced by a coarse mesh. For modeling, only part of the fine mesh is constructed and the coarse mesh is created such that it covers the whole problem domain. Next, the fine mesh is glued to the coarse mesh. This gluing is activated by PARAM,GRADED-MESH. The algorithm will identify the elements of the coarse mesh that are covered by the fine mesh. They are deactivated and removed from Euler archive output requests.

Requirements for Gluing Meshes

Connecting meshes with varying mesh sizes was already possible by using multiple Euler domains with porous coupling surfaces. Using this method two connected meshes are surrounded by a coupling surface together with a fully porous subsurface that connects the two domains. The domains are exclusively setup by using the MESH entry. In this approach there are no requirements on the size and location of the meshes. In output requests, only one domain can be examined. Making plots of the whole domain is not possible in one Patran session. This makes the approach cumbersome. With the graded mesh functionality there is no longer any need to create coupling surfaces. An additional advantage is that there is no restriction on how the elements are created for the simulation. Any preprocessor may be used, defining the `CHEXA` elements in an input file, but also blocks of meshes can be created by means of the `MESH` entry. The interface between the fine and coarse mesh is identified in the Dytran solver when PARAM,GRADED-MESH is activated. If the fine mesh is completely contained inside the coarse mesh, no restrictions apply. Then, to avoid any restrictions, the grid points of the fine mesh are slightly displaced. But if parts of the fine mesh are outside the coarse mesh, a restriction applies to graded meshes. In that case, an Euler element of the coarse mesh has to be fully active or fully inactive. This means that the coarse element should not intersect elements of the fine mesh or it should be fully covered by the fine elements. Fine elements are not allowed to cover any part of the coarse elements. In practice, this means that the fine mesh has to fit nicely in the coarse mesh. As shown in the meshes in Figure 2-1, the four marked locations a grid point of the fine mesh coincides with a grid point of the coarse mesh. This matching does not need to be exact, since the Dytran solver uses a tolerance to find the coinciding grid points. Visualization of the results of the complete Euler domain can be done in one session in a postprocessor like Patran.

Gluing Meshes

Gluing of fine and coarse meshes is activated by PARAM,GRADED-MESH. The algorithm will remove coarse elements that are completely covered by fine elements. Here the criterion for removal is based on the element volume. The element with the largest volume will be removed. It is also possible to remove the elements with the lowest element numbers. These two approaches are activated by respectively the `MINVOLUME` and `ELNUM` option of the GRADED-MESH parameter.

The gluing algorithm performs the following steps:

- The Euler elements are sorted such that the connected elements are grouped together.
- For each group of elements, the algorithm determines which elements have to be removed. As mentioned the criterion for removal is based on the fact if these elements are completely covered by elements of another group that are smaller or have a larger element number.
- Next the groups of elements are connected at boundary faces and at locations where elements have been removed. Special faces will be created that connect an element of one element group to an element of another group.

If the fine mesh is fully surrounded by the coarse mesh, the interface between the two meshes consists of the boundary faces of the fine mesh. The geometry formed by these faces will be used to construct special faces that connect an element of the coarse mesh to an element of the fine mesh.

Using Graded Meshes

Blocks of Euler elements can be defined by either Patran or the MESH,BOX entry. It is important that at the interface a face of a coarse element can be matched to several parts of faces of fine elements. This matching does not need to be exact since a tolerance is used.

One way to construct graded meshes is:

- Make the coarse mesh by using MESH,BOX.
- Run the simulation and read in the Euler mesh into Patran. For a part of this mesh the coarse elements have to be replaced by finer elements. Select a part of this mesh by selecting two Eulerian grid points.
- Create the fine mesh by MESH,BOX by using for the reference grid point as the first grid point. The width of the box is given by subtracting the coordinates of the second grid point from the coordinates of the first. To finish the definition of the fine mesh, the number of elements in each direction has to be defined. This ensures that the fine mesh fits nicely in the coarse mesh.
- Add PARAM,GRADED-MESH, which will activate the algorithm that is described above.

If needed a part of fine mesh can be replaced by an even finer mesh, by iterating through the steps above.

To construct graded meshes by Patran, a utility called "the break up of element" can be used.

Visualization with Patran

Elements that are fully covered by the fine mesh will not be included in the Euler archives. This allows for visualization of all Euler elements in one Patran session. The fringe plots can lack smoothness at the interfaces. The reason is that Patran determines colors on the basis of grid point values. They are computed by averaging over the elements that are connected to the grid points. In graded meshes, there can be grid points that belong only to fine elements and not to coarse elements. These grid points are called hanging nodes. An example of a hanging node is shown in Figure 2-2. At the hanging node the value only reflects the fine mesh. This results in loss of smoothness. An example is shown in Figure 2-3. If the results are postprocessed using the element values, this problem does not appear (see Figure 2-4).

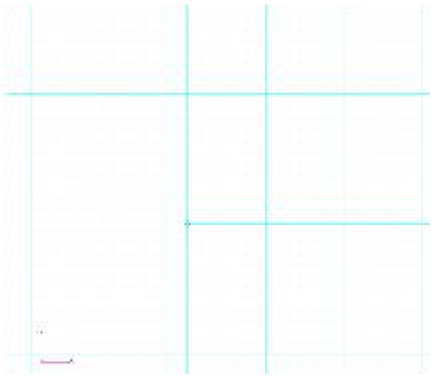


Figure 2-2 A Hanging Node

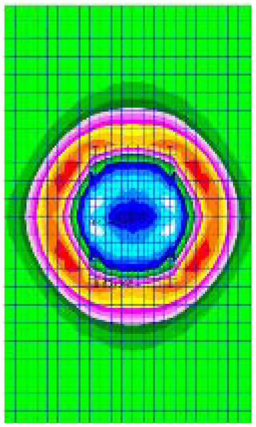


Figure 2-3 Lack of Smoothness

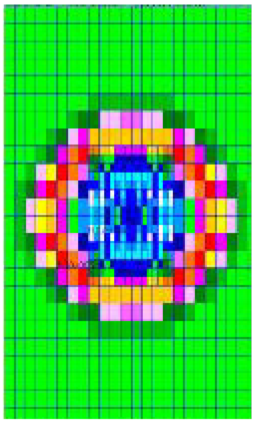


Figure 2-4 Using Element Values

3 Constraints and Loading

Constraint Definition

The motion of part or all of a mesh can be prescribed by application of constraints.

Single Point Constraints

A single point constraint is used to prescribe the motion of a translational or rotational degree of freedom. The constraint is effective throughout the analysis and is used to specify boundary conditions or planes of symmetry.

A single point constraint is defined by an `SPC` entry. The SPC entry defines the constraints on one grid point, while the SPC1 defines the constraints to be applied to a set of grid points. The SPC2 explicitly defines a rotational velocity constraint. SPC3 defines a constraint in a local coordinate system referenced from the SPC2 entry. Several sets of SPC entries can be defined in the Bulk Data Section, but only those selected in the Case Control Section using the SPC = n command are incorporated in the analysis.

Single-point constraints can also be defined using the GRID entry. These constraints are present for the entire analysis and do not need to be selected in Case Control. This is valid only for SPC and SPC1.

Since Dytran is an explicit code, there is no matrix decomposition. Therefore, the problems of singular matrices that occur with some implicit codes do not exist. All, or part of the Lagrangian mesh can be entirely unconstrained and can undergo rigid body motion. Dytran correctly calculates the motion of the mesh. Similarly, the redundant degrees of freedom, such as the in-plane rotation of shell elements, do not need to be constrained since they do not affect the solution. The only constraints that are needed are those representing the boundary conditions of the model and those necessary for any planes of symmetry.

Contact Surfaces

Contact surfaces provide a very simple and flexible way of modeling the interaction among the parts of the finite element model and allowing continuous contact between deforming or rigid bodies. This gives enhanced convergence over a point-to-point gap and allows parts of the model to slide large distances relative to each other.

There are three types of contact surface:

- General Contact and Separation
- Single Surface
- Discrete Grid Points

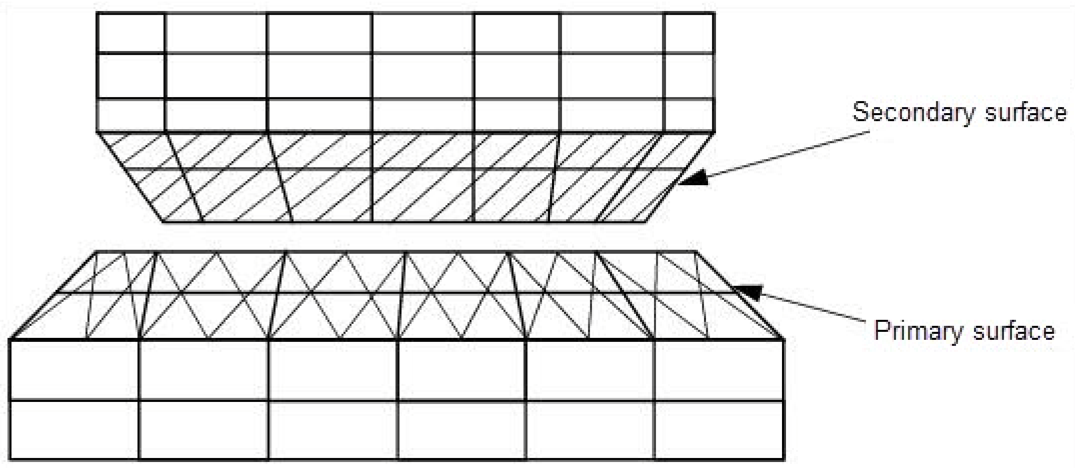
They are defined using the CONTACT entry on which you must specify the type of contact surface, the coefficient of friction, and the entities that might touch the contact surface.

General Contact and Separation

This is the most general of the contact surfaces and the one that is used most frequently. It models the contact, separation, and sliding of two surfaces, which can be frictional if required.

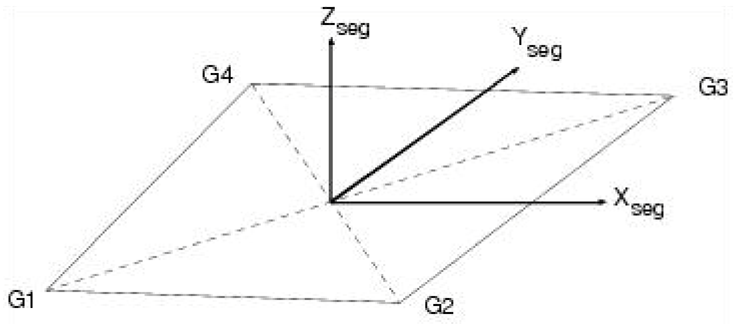
Segments

You must define the two surfaces that may come in contact by specifying the faces of the elements that lie on the surface. Each element face is called a segment of the surface. Segments are specified using the CSEG, CFACE, or CFACE1 entries. They can be attached to either solid or shell elements and can be triangular or quadrilateral. One surface is called the secondary contact surface; the other surface is called the primary contact surface. You must define a set of segments for each.

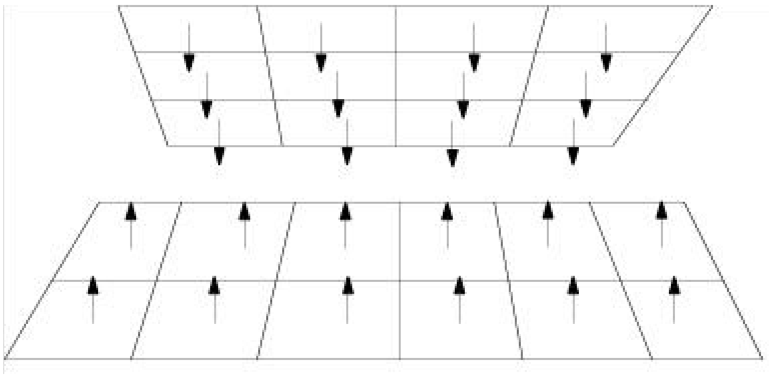


The two surfaces must be distinct and separate. A segment cannot be part of both the primary and secondary contact surfaces. The segments can be defined in a number of ways; they can be defined directly using CSEG, CFACE, or CFACE1 entries, or they can be attached to shell or membrane elements chosen by element number, property, or material. CSEG entries can also be defined using CQUAD4 or CTRIA3 entries, and CFACE1 entries can be defined using PLOAD4 entries, see Chapter 9: Running the Analysis, Using a Modeling Program with Dytran in this manual.

The connectivity of the segments is important since it determines from which side contact occurs. The order of the grid points on the CSEG entry defines a coordinate system just like the element coordinate system for the CQUAD4 and CTRIA3 elements described in Chapter 2: Elements, Lagrangian Elements in this manual.



The SIDE field on the CONTACT entry is used to define the side from which contact occurs. In the example below, the z-axes of the segments point towards each other so that the top surfaces contact. The SIDE field should therefore be set to TOP. It is possible to specify that contact can occur from both sides of the segment. This is dangerous, however, in that initial penetrations of the contact surfaces are not detected.



It requires that the normals of a set of segments all point in the same direction. If this is not the case, the REVERSE option on the CONTACT entry automatically reverses the normals of any segments that do not point in the same general direction as the majority of segments on the surface.

Penetration

There must be no initial penetration of the two surfaces. The surfaces must either be coincident or have a gap between them. If there is initial penetration, Dytran issues a User Warning Message when the INIPEN field on the CONTACT entry is set to ON.

However, the calculation does continue, but the forces are applied to separate the surfaces. If the penetrations are large, these forces are also large and may cause premature termination of the analysis.

The `PENTOL` field on the CONTACT entry sets a tolerance for the penetration checks. Grid points outside of this tolerance are not initialized into the contact surface and so do not take part in the contact.

Method

The detailed theory is outside the scope of this manual, but it is important that you know how the contact surface works if you are to use it effectively. At each time step, each grid point on the secondary contact surface is checked, and the nearest primary contact segment is located. Dytran then checks to see if the grid point has penetrated the primary contact segment. If it has not, the calculation continues. If it has penetrated, forces are applied in a direction normal to the primary contact surface forces to prevent further penetration of the segment. The magnitude of the forces depends on the amount of penetration and the properties of the elements on each side of the contact surface. The magnitude of the forces is calculated internally by Dytran to ensure minimal penetration while retaining a stable solution. The `FACT` field on the CONTACT entry can be used to scale the magnitude of the forces. This can be useful when two components are forced together by large forces. However, instability may occur when the `FACT` value is set to a high value.

A friction force is also applied to each of the surfaces, parallel to the surface. The magnitude of the force during sliding is equal to the magnitude of the normal force multiplied by the coefficient of friction. The direction of the friction force is opposite to the relative motion of the surfaces.

The coefficient of friction μ is calculated as follows

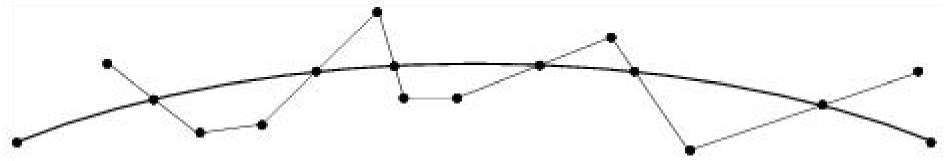
$$\mu = \mu_k + (\mu_s - \mu_k)e^{-\beta v}$$

where

μ	=	static coefficient of friction
μ_k	=	kinetic coefficient of friction
β	=	exponential decay coefficient
v	=	relative sliding velocity of the secondary contact and primary contact surfaces

You must specify μ_s, μ_k, and β.

The algorithm is not symmetrical, since the secondary contact points are checked for penetration of the primary contact segments but not vice versa. This means that the mesh density of the secondary contact surface should be finer than that of the primary contact surface. If not, penetrations can occur as shown in following two dimensions.



This can lead to hourglassing and incorrect results.

Since the closest segment to each point on the surface is constantly updated, the contact surface works correctly regardless of how far the two surfaces slide relative to each other or how much the shape of the surfaces changes as the mesh deforms.

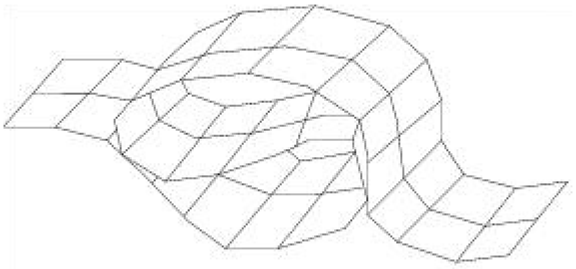
Efficiency

Generally, contact surfaces are very simple to use and very efficient. However, the penetration checks do take time, and therefore, the number of secondary contact and primary contact segments on each interface should be limited to those where contact might occur.

`TSTART` and `TEND` enable you to switch the contact surface on and off at specific times. This means that the contact surfaces are not checked until the contact surface is activated, thus saving computational effort when no contact occurs. By default, the contact surface is active throughout the analysis.

Single Surface

The single-surface contact is similar to the general one described in previous sections, but instead of defining secondary contact and primary contact segments, you define one set of secondary contact segments where the secondary contact segments cannot penetrate themselves. This is particularly useful for modeling buckling problems where the structure folds onto itself as the buckles develop and the points of contact cannot be determined beforehand.



This type of contact surface is defined in the same way as the general type, using the CONTACT entry, except that you only define a set of secondary contact segments — the `PID` field must be left blank. The surface can be frictional by giving nonzero values of the friction coefficients on the CONTACT entry. Friction forces are applied in the same way as for the general contact surface, see General Contact and Separation in this chapter.

Unlike the general contact surface, the connectivity of the segments does not matter. Contact can occur on either side of the surface automatically. However, the normals of all segments on the surface must point in the same direction although it does not matter in which direction. In most of the meshes, this usually is the case. If not, the REVERSE option automatically reverses the normals of segments that do not point in the same direction as the majority of segments on the surface.

The single-surface algorithm works in much the same way as the primary-secondary contact type described in General Contact and Separation in this chapter. The algorithm is particularly efficient, and rather large areas of single surface contact may be defined.

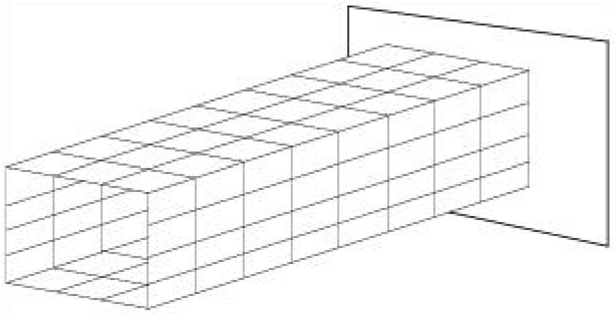
Discrete Grid Points

This type of contact surface allows individual grid points to contact a surface. The SID field on the CONTACT entry must be set to GRID. You must supply a list of the secondary contact grid points—which cannot penetrate the primary contact surface — using the SET1 entry. The primary contact surface must be defined as a set of segments in the same way as general contact surfaces are defined; see *Dytran Theory Manual*, Chapter 4: Models Shear Models.

The secondary contact points can be attached to any type of element. Throughout the analysis the secondary contact points are prevented from penetrating the primary contact surface. When in contact with that surface, the secondary contact points can slide frictionless or with friction along the surface.

Rigid Walls

A rigid wall is a plane through which specified grid points cannot penetrate. The rigid wall provides a convenient way of defining rigid targets in impact analyses.



Any number of rigid walls can be specified using WALL entries. The orientation of each wall is defined by the coordinates of a point on the wall and a vector that is perpendicular to the wall and points towards the model.

At each time step, a check is made to determine whether the grid points have penetrated the wall. These points are defined using a SET1 entry, and there can be any number of them. Since a check is made for every point at each time step, you should specify only those points as points that are expected to contact the wall in order to ensure the most efficient solution.

If a point is found to have penetrated the wall, it is moved back towards the wall so that its momentum is conserved. If the point subsequently moves away from the wall, it is allowed to do so.

Points in a SET1 that are referenced by a WALL cannot have any other constraint. They can, however, be part of other contact and coupling surfaces.

Tied Connections

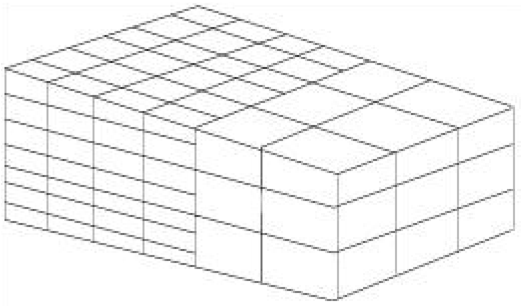
Tied connections are used to join parts of the mesh together. There are three types of connections:

- Two Surfaces Tied Together
- Grid Points Tied to a Surface
- Shell Edge Tied to a Shell Surface

All are defined using the RCONN entry and are described in the following sections.

Two Surfaces Tied Together

With this type of connection, two surfaces are permanently joined together during the analysis. This provides a convenient method of mesh refinement. It is better to use this method of mesh refinement than to use CPENTA or CTETRA elements, which are too stiff. Naturally, tied contact surfaces should not be close to any critical regions or areas that are highly nonlinear. Otherwise, you may use them wherever convenient.



You need to define the primary and secondary contact surfaces that are to be tied together by specifying the faces of elements that lie on the surface. Each element face or segment can be attached to either solid or shell elements and can be either quadrilateral or triangular.

You must define two surfaces that comprise a primary and secondary contact surface by specifying the faces of the elements that lie on the surface. Each element face is called a segment. The segments can be defined using CSEG, CFACE, or CFACE1 entries.

The way the tied surface works is not symmetrical, so your choice of primary and secondary contact and primary contact surface is important. The secondary contact segments must always be attached to the finer mesh, and the primary contact segments are attached to the coarser mesh. To use tied surfaces to connect two meshes that change their mesh density so that in one area one mesh is finest while in another area the other is finest, use more than one tied connection to join them together.

If this rule is not followed, some grid points will penetrate the other mesh, hourglassing will be excited, and spurious results will occur in the region of the tied connection.

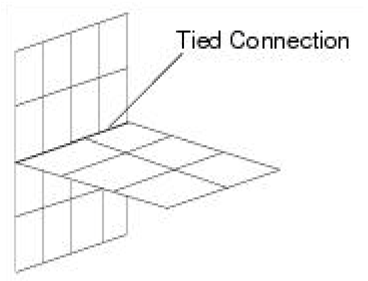
Grid Points Tied to a Surface

Individual grid points can be tied to a surface using this type of tied connection. In this case, the `SID` field of the RCONN entry must be set to `GRID`, and you must give a list of all grid points that are to be tied. The primary contact surface must be defined as a set of segments in the same way as for the two surface connections described in the previous section. In addition, the `OPTION` field must be set to `NORMAL`.

During the analysis, each grid point will be tied to the surface; i.e., its position relative to the surface will not change. Only the translational degrees of freedom are tied. If, for example, a shell element is attached to a tied grid point, then the shell can rotate relative to the surface.

Shell Edge Tied to a Shell Surface

This type of connection is used to connect the edge of one set of shell elements to the surface of another set.



The `SID` field of the RCONN entry must be set to `GRID`, and you must give a list of all the grid points that lie on the edge of the first set of shell elements. The primary contact surface is defined as a set of segments in the same way as the two surface connection described in Two Surface Tied Together, except that the segments can only be attached to shell elements. In addition, the `OPTION` field must be set to `SHELL`.

In addition, the list of grid points can consist of any type of six degrees of freedom grid points (CBEAMs, `CTRIA`s, etc.).

Similar to the previous connection, the secondary contact grid points are tied to the surface during the analysis; i.e., their position relative to the surface will not change. The difference is that, in this case, the rotational degrees of freedom are also coupled so that the angle between the two sets of shells will be maintained.

Contact Penalty Stiffness based on Material Stiffness

Existing contact penalty stiffness based nodal mass is relatively overestimated when element sizes are largely different in contact surface definition. To improve the contact for this case, the contact penalty stiffness based on material stiffness is newly introduced.

Users must be careful to use the contact penalty stiffness based on material stiffness because it can not control numerical stability automatically and it may result in unstable condition.

Since the capability is focused on sheet metal forming simulation, it is applicable limitedly under the conditions below:

- The bulk modulus is identical in the secondary contact bodies.

- Only material stiffness of secondary contact bodies are used.
- Only shell elements (quad and triangular) are considered.
- Only available in serial run. (dmp capability will be added in the future release).

Based on setting, the contact stiffness, k can be calculated as the equation below.

- Quad Elements:

$$k = \sum \frac{1}{4} \cdot \frac{\text{safety\_factor} \times K_i \times A_i}{max(diagonal)} = k = \frac{1}{4} \cdot \frac{\text{safety\_factor} \times K}{max(diagonal)} \sum A_i$$

- Tria Elements:

$$k = \sum \frac{1}{3} \cdot \frac{\text{safety\_factor} \times K_i \times A_i}{max(diagonal)} = k = \frac{1}{3} \cdot \frac{\text{safety\_factor} \times K}{max(diagonal)} \sum A_i$$

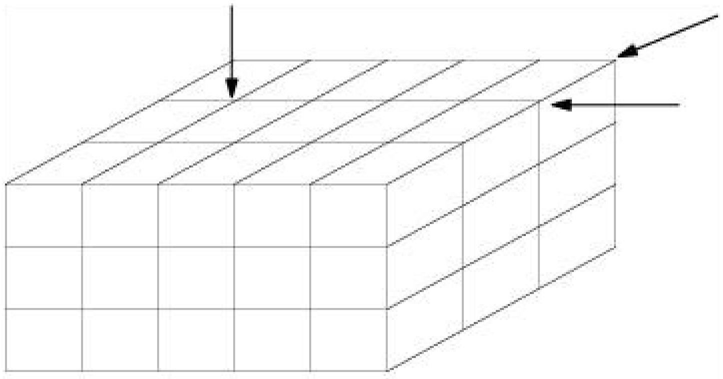
when diagonal is longer diagonal for quad elements and the longest side for triangular elements, K is bulk modulus and the default safety factor is 1.0.

Lagrangian Loading

This section covers the different ways that the analysis model can be loaded. The facilities available are:

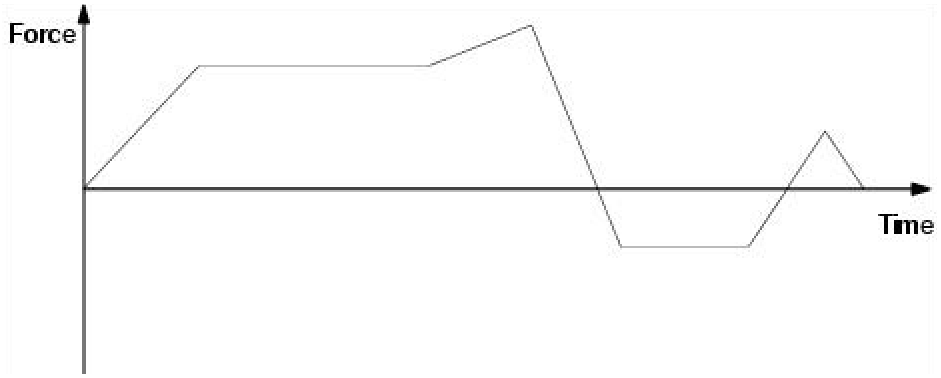
- Concentrated Loads and Moments
- Pressure Loads
- Enforced Motion
- Initial Conditions

Concentrated Loads and Moments



Concentrated loads and moments can be applied to any grid point using the DAREA, FORCE, FORCE1, FORCE2, MOMENT, MOMENT1, or MOMENT2 entries in combination with a TLOAD entry.

The TLOAD1 entry can reference a TABLEXX entry which is used to specify how the force $F(t)$ varies with time t.



The TLOAD2 entry always defines a variation with time by a function of which the coefficients are explicitly defined on the TLOAD2 entry.

The TLOAD entry also references a set of loading entries. These select the type of load, the grid point that is to be loaded, the direction of the load, and a scale factor to be applied to the curve of force versus time. The actual load applied $P(t)$ is given by

$$P(t) = AF(t)$$

where A is the scale factor.

The types of concentrated load that can be applied are discussed in the following section.

FORCE, FORCEn, or DAREA – Fixed-Direction Concentrated Loads

The FORCE, FORCEn, and DAREA entries define fixed direction loads. In other words, the direction of the force is constant throughout the analysis and does not change as the structure moves.

FORCE, FORCEn, and DAREA entries have the same effect but define the loading in different ways. With the DAREA entry, you specify the grid point, the direction in the basic coordinate system in which the load acts, and the scale factor. With the FORCE or FORCEn entry, you define the grid point, the components of a vector giving the loading direction, and the scale factor. In this case, the magnitude of the vector also acts as a scale factor, so the force in direction i is given by

$$P_i = AN_iF_i(t)$$

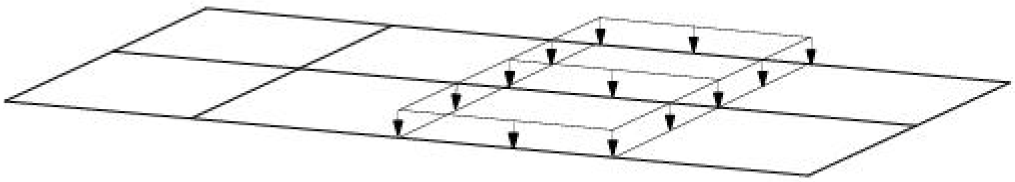
On a rigid body, the concentrated load or enforced motion is specified by defining the load at the rigid body center of gravity. To do so, set the TYPE field of the TLOAD1 or TLOAD2 entries to 13 and 12, respectively. The G field in the FORCE or MOMENT entry references the property number of the rigid body or MR<id> or FR<id>, where id is the number of a MATRIG or RBE2-FULLRIG entry, respectively.

MOMENT, MOMENTn, or DAREA – Fixed-Direction Concentrated Moments

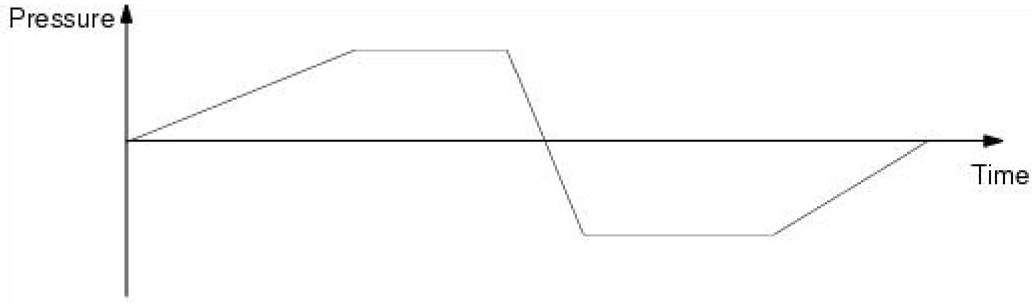
Concentrated moments can be applied using either MOMENT, MOMENTn, or DAREA entries. The difference between the two is the same as that between theFORCE, FORCEn, and DAREA entries described in the previous section.

Pressure Loads

Pressure loads are applied to the faces of solid elements and to shell elements. Pressure loads are defined using the PLOAD or PLOAD4 entry in combination with a TLOADn entry.



The TLOAD1 entry references a TABLEXX entry on which you specify the variation of the pressure $P(t)$ with time t.



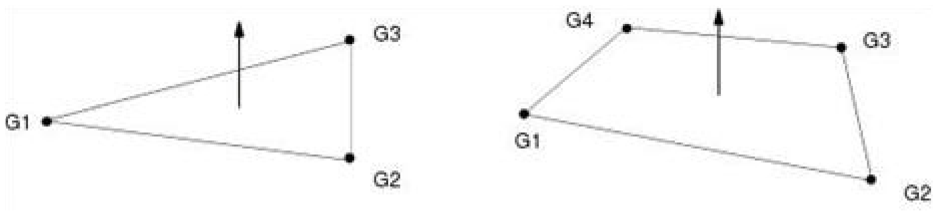
The TLOAD2 entry defines the variation of the pressure $P(t)$ with time t based on an equation, which is defined by the TLOAD2 entry.

TLOAD2 also references a set of PLOAD and/or PLOAD4 entries. Each entry selects the face of the element to be loaded by its grid points and defines the scale factor to be applied to the curve of pressure versus time. The actual pressure acting on the element p_{el} is given as follows:

$$p_{el}(t) = Ap(t)$$

where A is the scale factor.

The direction of positive pressure is calculated according to the right-hand rule using the sequence of grid points on the PLOAD4 entry. For PLOAD4 entries, the pressure is inwards for solid elements and in the direction of the element normal vector for shell elements.



Enforced Motion

This facility specifies the enforced motion of a degree of freedom at grid points by defining the grid point velocity with time. The enforced motion is applied in a way similar to concentrated loads, using a DAREA, FORCE, RFORCE, GRAV, or FORCEEX entry in combination with a TLOAD1 or TLOAD2 entry.

You must specify that the TLOAD1 or TLOAD2 entry defines enforced motion. TLOAD1 references a TABLEXX entry that gives the variation of velocity $V(t)$ with time t. The TLOAD2 entry implicitly defines a function of time. It also references a set of DAREA and/or FORCE entries that define the grid point being excited and the direction of the excitation. FORCE and DAREA entries have the same effect but define the excitation in different ways. With the DAREA entry, you specify the grid point, the direction in which the excitation is applied, and the scale factor S.

For enforced velocity, the velocity of the grid point $V_g(t)$ is given by

$$V_g(t) = SV(t)$$

With the FORCE entry, you define the grid point, the components of a vector N giving the excitation direction, and the scale factor S. In this case, the magnitude of the vector also acts as a scale factor, so the velocity of the grid point $V_g(t)$ is given by

$$V_g(t) = SNV(t)$$

If you want to specify the motion of a grid point in terms of its displacement, you must differentiate the motion to produce a velocity versus time characteristic that can be used by Dytran.

The FORCEEX entry allows the enforced motion of grid points to be defined in an external subroutine. The load number specified on the FORCEEX entry must be referenced in a TLOAD1 entry that specifies enforced motion; i.e., loading type 2. Enforced motion is defined in the user subroutine EXTVEL which is included in Loads User Defined Services(UDS) which will require the definition of a CONNECT FMS statement.The RFORCE entry defines enforced motion due to a centrifugal acceleration field. This motion affects all structural elements present in the problem. The GRAV defines an enforced motion due to a gravitational acceleration field. This motion affects all Lagrangian and Eulerian elements.

Grid points with enforced motion cannot be:

- attached to a rigid body.
- a point for a rigid wall.

- contact or rigid connection with rigid ellipsoids.

To specify the motion of a rigid body, the enforced motion of the rigid-body center of gravity must be defined. To do so, set the `TYPE` field of the TLOAD1 and TLOAD2 entries to 12. The G field on the DAREA, FORCE, or MOMENT entry references the property number of the rigid body, `MR<id>` or `FR<id>`, where id is the property number of the MATRIG entry or the RBE2-`FULLRIG` entry, respectively.

Initial Conditions

The initial velocity of grid points can be defined using TIC, TICGP, TIC1, and TIC2 entries. This allows the initial state of the model to be set prior to running the analysis. It is important to recognize the difference between initial velocities and enforced velocities. Enforced velocities specify the motion of grid points throughout the transient analysis. Initial velocities, on the other hand, specify the velocity of grid points at the beginning of the analysis. Thereafter, the velocities are determined by the calculation.

Where TIC1 and TIC2 set only the initial grid-point velocity, the TICGP entry can be used to set the initial value of any valid grid point variable. It can also refer to a local coordinate system by including the `CID1` and/or `CID2` entry in the list.

Element variables can also be given initial values using the TICEL entry. Any valid element variable can be defined for a set of elements.

Eulerian Loading and Constraints

Loading Definition

The implementation of loading and constraints within Eulerian meshes is somewhat different than that in a Lagrangian mesh. Eulerian constraints apply to element faces within the mesh rather than to the grid points. Dytran allows you to set the initial conditions for material in Eulerian elements, constrain material with fixed barriers, apply gravitational body forces, apply pressure boundaries to element faces, apply flow boundaries where material enters or leaves the mesh, and couple the mesh so that the material interacts with the Lagrangian parts of the model.

If an exterior - or free face - of an Eulerian mesh does not have a specific boundary condition, then, by default, it forms a barrier through which the material cannot flow. The default can be redefined by using a FLOWDEF entry.

Flow Boundary

A flow boundary defines the physical properties of material flowing in or out of Eulerian elements and the location of the flow. The FLOW entry is referenced by a TLOAD1. The `TYPE` field on the TLOAD1 must be set to 4.

The FLOW entry references a set of segments, specified by CFACE, CFACE1, or CSEG entries, through which the material flows. The subsequent fields allow you to specify the x, y, or z velocity, the pressure, and the density or specific internal energy of the flowing material. If only the pressure is defined, this gives a pressure boundary. Any of the variables that are not specified take the value in the element that the material is flowing into or out of.

The FLOWEX entry specifies a similar flow boundary through a set of faces. However, the physical details of the flow are determined from a user subroutine.

The FLOWSQ entry allows specifying a flow condition on basis of a square definition. All Eulerian boundary faces that are covered by the square will get the boundary condition.

The FLOWXSQ entry specifies a similar flow. Details of the flow are determined by a user subroutine.

The entries FLOWT and FLOWTSQ specify time-dependent flow. The time dependence is given by table entries. FLOWTSQ uses a square definition.

The FLOWDIR entry imposes a flow condition on all Eulerian boundary faces pointing in a certain direction.

The FLOWXDR entry imposes a flow condition on all Eulerian boundary faced pointing in a certain direction. Flow conditions are given by a user subroutine.

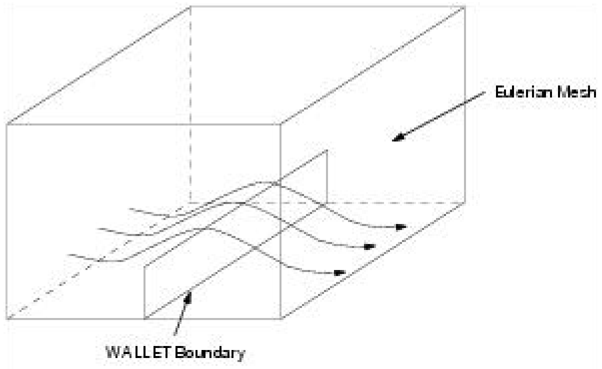
FLOWC, FLOWCDR, and FLOWCSQ enable cyclic flow boundaries, These entries have to be defined in pairs. Outflow from the first flow boundary is used as the inflow of the second flow boundary. And the other way around.

Rigid Wall

The WALLET entry defines a wall that is equivalent to a Lagrangian rigid wall. This is a barrier to material transport in an Eulerian mesh. The barrier is defined by a set of faces generated from a CFACE, CFACE1, or CSEG entry through which no material can flow.

This is the default condition for any exterior faces of the Eulerian mesh that do not have a FLOW boundary specified. However, the WALLET entry can be used to specify rigid walls within an Eulerian mesh.

WALLDIR assigns a WALLET definition to all Eulerian boundary faces pointing in a specific directory and is especially useful when a FLOWDEF has been defined.



Initial Conditions

The initial conditions of Eulerian elements can be defined using the TICEL, TICEUL, or EULINIT entry. This allows the initial state of the model to be set prior to running the analysis. It is important to recognize the difference between initial conditions and enforced conditions. Enforced conditions specify the loading and constraints of material throughout the transient analysis. Initial conditions, on the other hand, specify the state of the material only at the beginning of the analysis. Thereafter, the material state is determined by the calculation.

TICEL

TheTICEL entry defines transient initial conditions for elements. Any valid element variable can be given an initial value.

EULINIT

The EULINIT entry imports an Euler archive into a follow-up run and maps it onto the defined Euler elements. If the follow-up run uses a coupling surface, then, in the first run, this coupling surface can be left out, providing that the physics in the Euler have not reached the coupling surface at the end of the first run. In the follow-up run, a coarser mesh can be used to reduce CPU time. To remap a spherical symmetric or an axial symmetric Euler archive, the `PARAMs` SPREMAP and AXREMAP have to be used. Also, the remapping of a spherical symmetric Euler domain onto a 2-D axial symmetric Euler domain is supported.

TICEUL

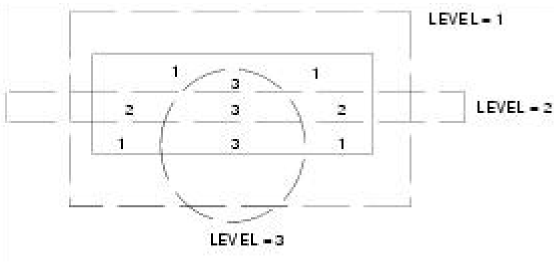
The TICEUL entry defines transient initial conditions for geometrical regions in the Eulerian mesh. The TICEUL entry must be used together with the PEULER1 property definition.

With the TICEUL entry, it is possible to generate initial conditions inside or outside multifaceted surfaces, analytical cylindrical or spherical geometry shapes and in sets of elements.

Each geometrical region (multifaceted surface, cylinder, sphere, box, or set of elements) has a level number. This allows the creation of regions of arbitrary shape by allowing the regions to overlap. An element that lies in two or more geometrical regions is assigned to the region that has the highest level number.

Think of geometrical regions as shapes cut out of opaque paper. Position the region of the lowest level number on the mesh. Then, place the next higher region on top of the first and continue until all the regions are in place. When the last region is placed, you have a map indicating to which region each element in the problem is assigned.

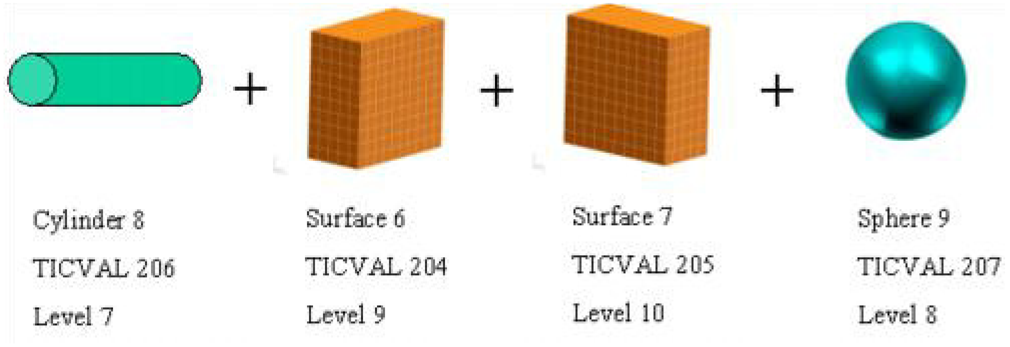
The following figure shows how three different geometrical regions can be used to create regions of arbitrary shape. The solid line represents the boundary of the mesh. Region one (LEVEL = 1) is the large dashed rectangle. Region two (LEVEL = 2) is the long narrow rectangle. Region three (LEVEL = 3) is a circular region. The numbers on the diagram indicate how the elements in different parts of the mesh are assigned to these three regions.



If two or more regions with the same level number but different initial value sets or materials overlap, then the regions are ambiguously defined. This results in an error.

Multifaceted surfaces that are used in initial value generation must be closed and form a positive volume and are defined on the MATINI entry. The MATINI entry is referred to from the TICEL entry and together with a TICVAL entry the initial condition for the initialization surface is defined.

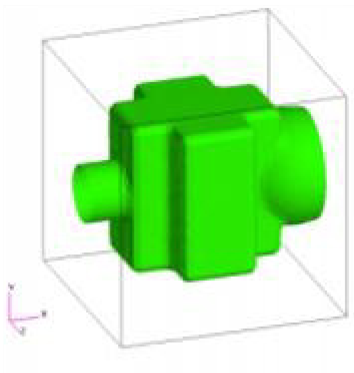
Example 1:



In the example below a combination of two multi-faceted surfaces, a cylindrical and a spherical shape together with a block of elements (all Eulerian elements) are used to define the initial conditions in an Eulerian rectangular mesh.

The different shapes, initial value sets and levels used are shown below. All elements of the Eulerian mesh are defined as an element shape with void and the lowest level 6 (see input file below). This means that the part of an element that doesn't fall inside any of the shapes will be initialized as being void.

Plot of the material inside the Eulerian mesh after initialization:

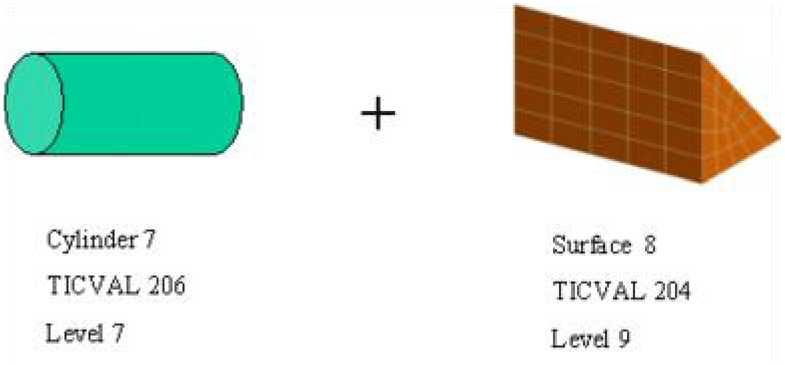


Input:

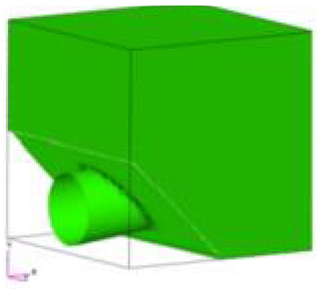
```
TICEUL,101,,,,,,,,+
+,SURF,6,12,204,9.,,,,,+
+,SURF,7,12,205,10.,,,,,+
+,CYLINDER,8,12,206,7.,,,,,+
+,SPHERE,9,12,207,8.,,,,,+
+,ELEM,444,,,6.
SET1,444,50000,THRU,53375
$
TICVAL,204,,DENSITY,1000.,XVEL,50.
TICVAL,205,,DENSITY,1000.,XVEL,50.
TICVAL,206,,DENSITY,1000.,XVEL,50.
TICVAL,207,,DENSITY,1000.,XVEL,50.
$
SURFACE 6               PROP    116
SET1,116,6
PSHELL1,6,,DUMMY
MATINI   6       6      Inside  On      On
$
SURFACE 7               PROP    117
SET1,117,7
PSHELL1,7,,DUMMY
MATINI   7       7      Inside  On      On
$
CYLINDER,8,,1.0,.5,.5,2.0,.5,.5,+
+,.075
SPHERE,9,,1.75,.5,.5,.15
```

Example 2:

In the example below, a multifaceted surface is used with the `OUTSIDE` option. When the `OUTSIDE` option is used on the MATINI entry, the parts of the Eulerian elements that fall outside the initialization surface are initialized.



Plot of the material inside the Eulerian mesh after initialization:



Input:

```
TICEUL,102,,,,,,,,+
+,SURF,8,12,204,9.,,,,+
+,CYLINDER,7,12,206,7.,,,,+
+,ELEM,445,,,5.
SET1,445,60000,THRU,63375
$
SURFACE 8                PROP    118
SET1,118,8
PSHELL1,8,,DUMMY
MATINI   8       8       Outside On      On
$
$SPHERE,10,,2.25,.35,.35,.2
$
CYLINDER,7,,1.0,.20,.50,3.0,.40,.50,+
+,.1
```

Any initial condition that you define acts on the material as it is defined within the confines of the Eulerian mesh. The initial conditions can be given for any Eulerian elemental variables (see Outputting Results for valid elemental variables.

In addition, you can define a radial velocity field for the material in an Eulerian domain. The definition does not apply the standard element variables, but a sequence of four definitions that completely specify the radial velocity field. You need to define the center from where the radial is to emerge (X-CENTER, Y-CENTER, and Z-CENTER), the velocity in the direction of the radial (R-VEL) and the decay coefficient (DECAY).

Assume the element center at location x,y,z and the location of where the radial emerges as x_r,y_r,z_r. With $\dot{R}$ the velocity along the radial and the decay coefficient β, the velocity components for the element (mass) can be computed:

$$\dot{\bar{r}} = \frac{\dot{\bar{x}} - \dot{\bar{x}}_r}{\left\| \dot{\bar{x}} - \dot{\bar{x}}_r \right\|} \text{ and } \dot{\bar{v}} = \dot{\bar{r}} \cdot \dot{R} \cdot (\dot{\bar{x}} - \dot{\bar{x}}_r)^{\beta}$$

The velocity components resulting from the radial field are added to the velocity components otherwise defined for the element.

Note that the dimension of RVEL changes with the value of DECAY.

Example 3:

Assume the initialization is with sphere with origin at (0,0,0) and has a radius of 2.

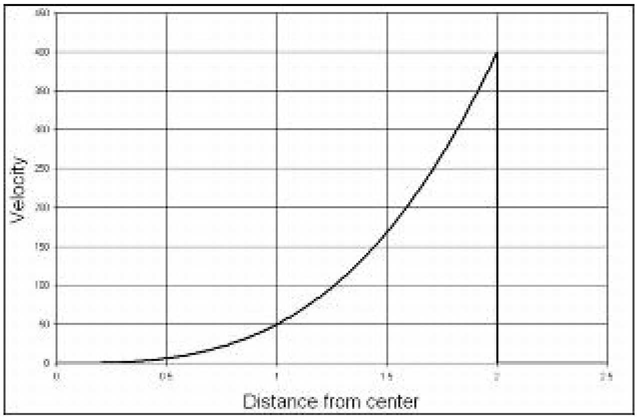
The DECAY coefficient is 3 and the velocity of air at the sphere boundary is 400.

The value of R-VEL is:

Input:

```
PEULER1,100,,2ndOrder,101
$
TICEUL,101,,,,,,,,+
+,SPHERE,1,1,4,2.0,,,,+
+,ELEM,2,1,5,1.0
$
SPHERE,1,,0.,0.,0.,2.0
TICVAL  4               DENSITY 1.1468-7SIE     3.204+8 ZVEL     20000. +CONT
+CONT   X-CENTER0.0     Y-CENTER0.0     Z-CENTER0.0     R-VEL    50.    +CONT
+CONT   DECAY   3.0
$


SET1,2,1,THRU,10000
```



Initializing with a Spherical Symmetric Solution

The TICVAL entry allows specifying initial element variables as function of the radial distance to a preselected center. This enables mapping of spherical symmetric solution results onto a full 3-D mesh.

Deflagration

Euler elements that refer to EOSDEF of state can burn. The region of the Euler mesh that can burn is indicated by the variable DEFMAT. Only regions with DEFMAT= 1 can burn. As part of the Euler Element initialization the variable DEFMAT can be set using TICVAL entries. Burning with EOSDEF is driven by pressure, therefore ignition can take place by prescribing a sufficiently large initial specific internal energy, leading to a rise in more pressure. The burn fraction is given by the Eulerian element variable DEFBURN.

Detonation

Eulerian elements that reference a JWL equation of state (EOSJWL) have to be detonated. A DETSPH entry must be present that defines a spherical detonation wave. You define the location of the detonation point, the time of detonation, and the speed of the detonation wave. Dytran then calculates the time at which each explosive element detonates. Elements that do not have a JWL equation of state are unaffected.

Body Forces

If the GRAV entry is specified, the Eulerian material also has body forces acting on the material mass. The GRAV entry defines an acceleration in any direction. All Eulerian material present in the problem is affected.

The gravity entry works on all material. To allow for body forces inside a specific region and on a specific material, the BODYFR1 can be used. This entry also allows to define time-dependent accelerations.

Hydrostatic Preset

With PARAM, HYDSTAT, the Euler element densities are initialized in accordance to a hydrostatic pressure profile. This PARAM requires the use of the GRAV entry.

To impose matching boundary conditions, FLOW, HYDSTAT, and PORHYDST can be used. These two entries use the following boundary conditions:

- The pressure given by hydrostatic pressure profile. This is defined by the HYDSTAT entry.
- The velocities are set equal to their element values.
- If fluid flows in, its density is derived from the hydrostatic pressure.

Speedup for 2-D Axial Symmetric Models

To simulate a 2-D axial symmetric model with Dytran, a 3-D pie model can be used. To get sufficient accuracy, the angle of the pie should be smaller than five degrees but not too small. The small angle gives a mesh size in circumferential direction that is much smaller than the mesh sizes in the other directions. This results in a small time step. In principle, the mesh size in the circumferential direction can be skipped for the time step computation. But, often there are small errors in the circumferential normals and the circumferential direction has to be taken into account. With PARAM,`AXIALSYM`, these normals are automatically aligned. This allows a time step that is only based on the axial and radial directions, resulting in a significant larger time step. The larger time step is automatically computed when using PARAM,`AXIALSYM`.

Speedup for 1-D Spherical Symmetric Model

For 1-D-spherical symmetric simulations, PARAM, `SPHERSYM` can be used. This transforms a 1-D rectangular mesh into a wedge mesh. In the time step computation, only the mesh-size in radial direction will be taken into account, allowing a significant larger time step.

Viscosity and Skin Friction in Euler

Viscosity is available for the HYDRO, MMHYDRO, and the Roe solver. It is activated by setting the viscosity entry on the EOSNA, EOSTAIT, EOSEX, or EOSGAM options. This entry specifies the dynamic viscosity that relates shear stress to velocity gradients. Besides physical viscosity, the first-order solvers introduce a certain amount of artificial viscosity. This amount is problem dependent and the total viscosity can become significantly larger than the physical one. The Roe solver, using standard settings, has no artificial viscosity. Only in case of a transonic rarefaction waves, is a limited amount of artificial viscosity introduced.

With viscosity active, the Euler equations are replaced by the Navier-Stokes equation. Heat transfer and heat generation by viscous dissipation are not taken into account. In addition, a no-slip condition is enforced at the boundaries.

In most flows, the viscous effects are limited to a small region alongside the boundary. This region is called the boundary layer. Outside this region, the flow is more or less inviscid and its size depends on the Reynolds number. The thickness of the boundary layer scales as $\frac{1}{\sqrt{Re}}$. Flows with large Reynolds number have small boundary layers. To accurately simulate the boundary layer, several elements are needed across it. To see whether Euler elements are sufficiently fine, the mesh can be refined a bit. If element tangential shear stresses at the wall do not change significantly, the boundary layer is sufficiently captured. For flow with a large Reynolds number, this may require very small mesh sizes. To avoid small mesh sizes, the tangential shear stress at the wall can be computed by an empirical law that relates the shear stress to the element velocity:

$$\tau = \frac{1}{2} f \rho v^2$$

Here, f denotes the skin-friction coefficient, ρ the density, and v the relative tangential velocity. The skin-friction coefficient is available either in literature or from experimental studies. Moreover, the skin friction can account, to some degree, for turbulence and roughness of the structural surface.

Considering first-order methods, if the artificial viscosity is larger than the physical viscosity, artificial viscosity enters into the Reynolds number. In this case, artificial viscosity can increase the size of the boundary layer. If the artificial viscosity is larger than the physical viscosity and boundary layers play a dominant role in the flow, first-order methods are not appropriate and the Roe solver has to be used. If the physical viscosity is larger than the artificial viscosity, there are no restriction on the use of viscosity.

Boundary Conditions

At walls, the no-slip condition is enforced. When viscosity is activated, walls cannot be used to simulate symmetry planes. To simulate symmetry planes, define a flow boundary condition with normal velocity set to zero and any tangential velocity left unspecified.

At flow boundaries and porous sub surfaces, a prescribed tangential velocities also give rise to viscous shear stresses when the tangential Euler element velocity does not match the prescribed tangential velocity.

4 Fluid Structure Interaction

General Coupling

The objective of fluid-structure interaction using the coupling algorithm is to enable the material modeled in Eulerian and Lagrangian meshes to interact. Initially, the two solvers are entirely separate. Lagrangian elements that lie within an Eulerian mesh do not affect the flow of the Eulerian material and no forces are transferred from the Eulerian material back to the Lagrangian structure. The coupling algorithm computes the interaction between the two sets of elements. It thus enables complex fluid-structure interaction problems to be analyzed.

The first task in coupling the Eulerian and Lagrangian sections of a model is to create a surface on the Lagrangian structure. This surface is used to transfer the forces between the two solver domains. The surface acts as a boundary to the flow of material in the Eulerian mesh. At the same time, the stresses in the Eulerian elements cause forces to act on the coupling surface, distorting the Lagrangian elements.

By means of a SURFACE entry, you can define a multifaceted surface on the Lagrangian structure. A set of CFACEs,CFACE1s, CSEGs, element numbers, property numbers, material numbers, or any combination of these identifies the element faces in this surface. The method of defining of the surface is, therefore, extremely flexible and can be adapted to individual modeling needs.

The coupling algorithm is activated using the COUPLE entry. It specifies that the surface is used for Euler-Lagrange coupling. You can define whether the inside or the outside domain is covered by the coupling surface by setting the `COVER` field on the entry. This means that the Euler domain cannot contain material where the outside or the inside of the Lagrangian structure covers it. For problems where the Eulerian material is inside a Lagrangian structure (for example, an inflating air bag), `COVER` should be set to `OUTSIDE` since the Eulerian elements outside the coupling surface must be covered. For problems where the Eulerian material is outside the Lagrangian structure (for example a projectile penetrating soft material), the inside of the coupling surface must covered, and `COVER` should be set to `INSIDE`.

The coupling surface must have a positive volume to meet Dytran's internal requirements. This means that the normals of all the segments of the surface must point outwards. By default, Dytran checks the direction of the normal vectors and automatically reverses them when necessary. However, if you wish to switch off the check to save some computational time in the generation of the problem, you can define this using the `REVERSE` field on the COUPLE entry.

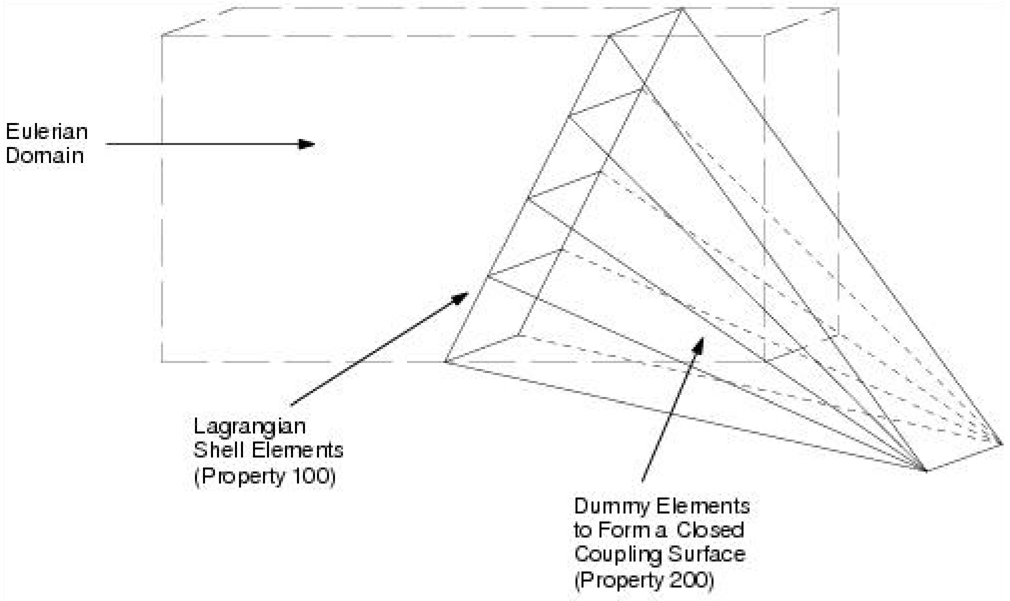
The coupling algorithm activated using the COUPLE entry is the most general interaction algorithm. It can handle any Euler mesh. There is an option however, to switch to a faster algorithm by setting the parameter FASTCOUP This algorithm makes use of knowledge of the geometry of the Euler mesh. As a result, the requirement is that the Euler mesh must be aligned with the basic coordinate system axes.

Closed Volume

The coupling surface must form a closed volume. This requirement is fundamental to the way the coupling works. It means that there can be no holes in the surface and the surface must be closed.

In order to create a closed volume, it may be necessary to artificially extend the coupling surface in some problems. In the example shown below, a plate modeled with shell elements is interacting with an Eulerian mesh. In order to form a closed coupling surface, dummy shell elements are added behind the plate. The shape of these dummy shell elements does not matter. However, it is best to use as few as possible to make the solution more efficient.

The closed volume formed by the coupling surface must intersect at least one Euler element otherwise the coupling surface is not recognized by the Eulerian mesh.


Eulerian Domain
Lagrangian Shell Elements (Property 100)
Dummy Elements to Form a Closed Coupling Surface (Property 200)

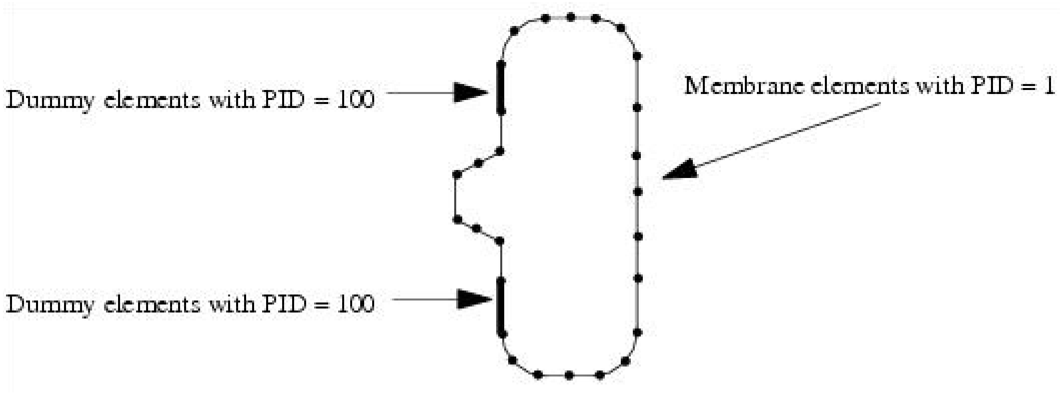
```
COUPLE, 1, 10, INSIDE
SURFACE, 10, , PROP, 10,
SET1, 10, 100, 200
PSHELL, 100, 100, 0.05
PSHELL1, 200, , DUMMY
```

Care must be taken when doing so, however. The additional grid points created for the dummy elements do not move as they are not connected to any structural elements. When the shell elements move so far that they pass beyond these stationary grid points, the coupling surface turns inside out and has a negative volume, causing Dytran to terminate.

Porosity

There is a general-purpose capability to model porosity of a coupling surface. Porosity allows material to flow from the Eulerian region through the coupling surface or vice-versa. This method is addressed using the PORFLOW, PORFCPL or PORFLCPL entry. PORFLOW is used to model the interaction between an Eulerian region to the environment while PORFCPL or PORFLCPL are used for flow from an Eulerian region to another one. For air bags, a newer and better methodology for modeling porosity has been implemented. Please refer to Chapter 6: Air Bags and Occupant Safety, Porosity in Air Bags in this manual.

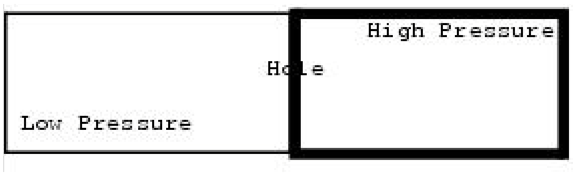
The coupling surface or parts of it can be made porous by referring to a COUPOR entry from the COUPLE entry. This is further explained by means of two examples. The example models an air bag with porous material and two holes using PORFLOW:


Dummy elements with PID = 100
Membrane elements with PID = 1
Dummy elements with PID = 100

The required input is:

```
PSHELL1, 1, , MEMB, , , , , ,+
+, 1.E-3
PSHELL1, 100, , DUMMY
SURFACE, 11, , PROP, 11
SET1, 11, 1, 100
SUBSURF, 22, 11, PROP, 100
SET1, 1, 1
SET1, 100, 100
COUPLE, 1, 11, OUTSIDE, , , 55
COUPOR, 1, 55, , PORFLOW, 42, CONSTANT, 0.009
COUPOR, 2, 55, 22, PORFLOW, 42, CONSTANT, 1.0
PORFLOW, 42, , MATERIAL, 33, PRESSURE, 1.E-5, METHOD, PRESSURE
```

The second example models two chambers divided by a membrane with a hole. Two sets of Euler elements must be defined in which each set belongs to each coupling surface (COUPLE). The interaction between a set of Euler elements and a coupling surface is defined by using IGNORE. The hole is modeled by using COUPOR that refers to either PORFCPL or PORFLCPL.



The required input is:

```
chamber 1 (low pressure)
SURFACE,1,.....
SUBSURF,10,1,....
COUPLE,60,1,,,,70,,,+
+,,,,,,,,,+
+,,21
COUPOR,80,70,10,PORFCPL,50, ,<coeffv>
PORFCPL,50,,,,1020
MESH,21,BOX,,,,,,,+
+,0.,0.,0.,1.,1.,1.,,,+
+,10,10,10,,,,EULER,500
PEULER1,500,,HYDRO,19
TICEUL,19 (initialization to low pressure)

chamber 2  (high pressure)
SURFACE,2,.....
COUPLE,1020,2,,,,,,,+
+,,,,,,,,,+
+,,22
MESH,22,BOX,,,,,,,+
+,0.8,0.8,0.8,2.,2.,2.,,,+
+,10,10,10,,,,EULER,600
PEULER1,600,,HYDRO,20
TICEUL,20 (initialization to high pressure)
```

Note that the porosity characteristics need to be defined for chamber 1 only. The gas automatically flows from chamber 2 into 1 and vice versa.

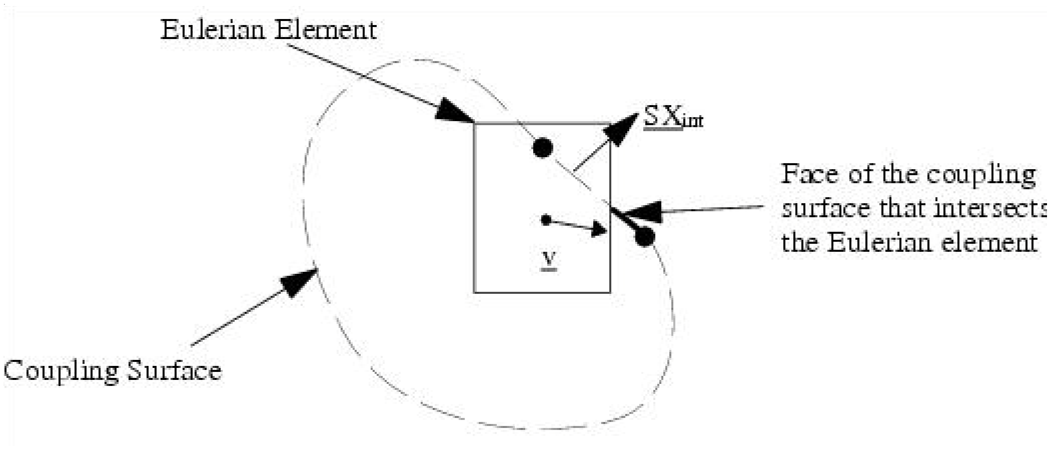
Two different algorithms are available to calculate the mass transport through the coupling surface. The desired method can be activated from the PORFLOW entry or by choosing PORFCPL for velocity method and PORFLCPL for pressure method.

Velocity Method

This algorithm is activated by:

```
PORFLOW, 42, , MATERIAL, 33, PRESSURE, 1.E-5, METHOD, VELOCITY
```

The transport of mass through the porous area is based on the velocity of the gas in the Eulerian elements, relative to the moving coupling surface.


Eulerian Element
SX_int
Face of the coupling surface that intersects the Eulerian element
v
Coupling Surface

The volume of the Eulerian material transported through the faces of the coupling surface that intersect an Eulerian element is equal to

$$V_{trans} = -dt \cdot \alpha \cdot (\vec{v} \cdot \vec{A})$$

where

V_{trans}	=	transported volume during one time step ($V_{trans} > 0$ for outflow; $V_{trans} < 0$ for inflow)
dt	=	time step
α	=	porosity coefficient
$\vec{v}$	=	velocity vector of the gas in the Eulerian mesh
$\vec{A}$	=	area of the face of the coupling surface that intersects the Eulerian element $\|A\|$ is equal to the area of the face that lies inside the Eulerian element.

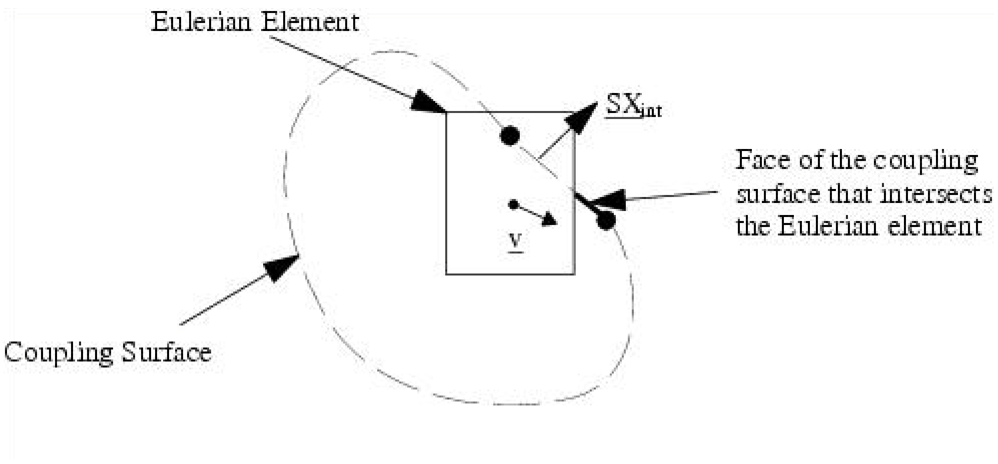
The transported mass through the porous area is equal to the density of the gas times the transported volume.

Pressure Method

This algorithm is activated by:

```
PORFLOW, 42, , MATERIAL, 33, PRESSURE, 1.E5, METHOD, PRESSURE.
```

The transport of mass through the porous area is based on the pressure difference between the gas in the Eulerian element and the outside pressure. The outside pressure is the pressure as specified on the PORFLOW entry.


Eulerian Element
SX_int
Face of the coupling surface that intersects the Eulerian element
v
Coupling Surface

The volume of the Eulerian material transported through the faces of the coupling surface that intersect an Eulerian element is equal to:

$$V_{trans} = dt \cdot \alpha \cdot (\vec{A} \cdot \vec{A}) \cdot \sqrt{\frac{2p\rho\gamma}{\gamma - 1}\left[\left(\frac{p_{exh}}{p}\right)^{\frac{2}{\gamma}} - \left(\frac{p_{exh}}{p}\right)^{\frac{\gamma+1}{\gamma}}\right]}$$

where

V_{trans}	=	transported volume during one time step ($V_{trans} > 0$ for outflow; $V_{trans} < 0$ for inflow)
dt	=	time step
α	=	porosity coefficient
$\vec{v}$	=	velocity vector of the gas in the Eulerian mesh
$\vec{A}$	=	area of the face of the coupling surface that intersects the Eulerian element $\lVert A \rVert$ is equal to the area of the face that lies inside the Eulerian element.
p	=	pressure of the gas in the Eulerian element
ρ	=	density of the gas in the Eulerian element
γ	=	adiabatic exponent $= C_p / C_v$
p_{exh}	=	pressure at the face

The pressure at the face is approximated by the one-dimensional isentropic expansion of the gas to the critical pressure or the environmental pressure according to

$$p = \begin{cases} p_{env} > p_c \\ p_{env} < p_c \end{cases}$$

where p_c is the critical pressure:

$$p_c p \cdot \left(\frac{2}{\gamma + 1}\right)^{\frac{r}{r-1}}$$

In case the outside pressure is greater than the pressure of the gas, inflow through the coupling surface occurs. This porosity model can only be used for ideal gases; i.e. materials modeled with the gamma law equation of state (EOSGAM).

Efficiency

The coupling algorithm requires a very large number of calculations to determine how the coupling surface intersects the Eulerian mesh. The coupling surface should, therefore, be as small as possible to make the solution efficient.

A subcycling technique can be applied to improve the efficiency. When you use the subcycling, the geometry of the coupling surface is not updated every time step, but only when necessary based on the motion of the surface. Dytran automatically controls subcycling. The frequency of geometrical updates may vary during the calculation. It depends on the motion of the coupling surface. The parameters COSUBCYC and COSUBMAX control the subcycling process. Subcycling is switched on using the COSUBCYC parameter.

Multiple Coupling Surfaces

Multiple Coupling Surfaces with Multiple Euler Domains

Multiple coupling surfaces are available for HYDRO, MMHYDRO, and MMSTREN Euler Solvers in combination with the fast coupling algorithm. It is not available for the STRENGTH Euler solver. This fast coupling algorithm is activated by the PARAM,FASTCOUP entry. Each couple surface is associated with an Eulerian domain. An Eulerian domain is a mesh that is aligned with the basic coordinate axes. You can define such a domain using the `MESHID` or `SET1ID` field on the COUPLE entry.

Through a surface that is shared by two coupling surfaces mass can flow from one coupling surface to the other. Such a surface will be called a hole. A hole can be either a porous subsurface of a coupling surface or be part of a coupling surface with interactive failure.

There are two methods to enable flow between Euler domains. These methods are

- The exact method. No approximations are made. All Euler elements need to be created by `MESH` entries. For this, an auxiliary Euler mesh is created at the hole.
- An approximate method. Transport between two Euler domains through a segment is determined by averaging Euler element properties across the segment.

By default the exact method is used. By selecting PARAM,FLOW-METHOD,`FACET` the second method is applied. Support for PARAM , FLOW-METHOD,`FACET` is limited but it allows the use of CHEXA's.

If all Euler domains are defined by MESH,BOX or MESH,ADAPT, then there are no restrictions on the use multiple coupling surfaces. In that case, a simulation may contain both porous holes that connect one Euler domain to another as well as coupling surfaces with interactive failure. If at least one Euler domain is defined by a SET1ID (thus using CHEXA's) field, then PARAM,FLOW-METHOD,FACET has to be added. In that case, only the following types of simulations are supported:

- Coupling surfaces with interactive failure (COUP1INT) with the Roe solver
- Coupling surfaces that are connected through porous holes with the Standard single material solver (using PORFCPL,PORFLCPL)
- Flow between two Euler domains through a fully porous coupling surface with the Roe solver. This is done by using COUP1INT and initiating failure starting at cycle 0 by the use of an exfail routine or by specifying a very low failure value in a failure mode.

With the multi-material solver multiple Euler domains are only supported if all Euler domains are defined by either MESH,BOX or MESH,ADAPT.

Coupling Surface with Failure

A coupling surface is always associated with a Lagrangian structure. When the material model used in the Lagrangian structure supports failure, for example by defining a failure model for the material (see *Dytran Theory Manual*, Chapter 4: Models, Material Failure), the faces in the coupling surface can fail when the underlying material in the structure fails. You can define the failure mode for the coupling surface by specifying PARAM, FASTCOUP, FAIL.

When a Lagrangian element fails and the element is shared by two coupling surfaces, mass from one Eulerian domain flows to the other Eulerian domain through the hole. The interaction between these Eulerian domains is defined through a COUP1INT entry. When you do not define the interaction between the Eulerian domains, but the coupling surface fails, default ambient values for the state variables are used to compute the in- or outflow through the hole in the surface. The ambient values of the variables can be defined on the COUP1INT entry.

Restrictions for using the COUP1FL entry and the COUP1INTentry are described in the manual entry of PARAM, FASTCOUP.

Coupling Surfaces with Porous Holes

The porous hole is a surface that is shared by two coupling surfaces and connects the two coupling surface to each other. By selecting either the porosity model PORFLCPL or PORFCPL, flow is enabled from the Euler domain in one coupling surface to the Euler domain in the other coupling surface. The model PORFLCPL uses the velocity method and is for general use whereas PORFCPL uses the pressure method and is only for small holes. For an overview of the pressure and velocity method, see "Getting Started-General-Coupling". The porous hole can be either partially or fully porous and can be either a subsurface of the whole coupling surface. A major application is the flow inside multi-compartment air bags.

To activate flow between two coupling surfaces through a porous hole, the following steps have to taken:

1. Associate an Euler domain to each of the two coupling surfaces.

2. Make a subsurface of the elements of the coupling surfaces that models the hole. The elements in this subsurface should be shared by both coupling surfaces.
3. Define a COUPOR entry for one of the coupling surfaces. This COUPOR references either a PORFLCPL or PORFCPL entry. It also references the subsurface.
4. Create a PORFLCPL or PORFCPL entry. The other coupling surface has to be referenced by this PORFLCPL entry.

Flow Between Domains

There are two methods available to compute flow from one Euler domain to the other across coupling surfaces:

- The facets in the coupling surfaces that represent an open area are subdivided into smaller facets, that each connect exactly to one Euler element in the first Euler domain and to exactly one Euler element in the second Euler domain. Material flow takes place across these smaller, subdivided facets called POLPACKs. This is the most accurate method. This method is the default and is option POLPACK of PARAM,FLOW-METHOD. For a detailed description of the theory involved see Reference [18.].
- The facets in the coupling surfaces that represent an open area are not subdivided. Material flow takes place across the original facets. If these facets are too large in relation with the Euler elements the method becomes inaccurate. Material flow across one facet can involve several Euler elements on both sides of the hole, and averaging will occur. This method is activated by PARAM,FLOW-METHOD,FACET.

The first method is more accurate but has the following limitations:

- Flow faces and wallets are not supported.

Note:	`flowdef` is supported.

- Viscosity is not supported.
- All Euler domains have to be created by the mesh entry. Euler domains consisting of a set of Euler elements are not supported.

A case where these limitations require the use of FLOW-METHOD=FACET is when the Euler elements are generated in Patran, not using the MESH entry, and one or more of the following options is used:

- FLOW boundaries are defined on some Euler faces.
- WALLET boundaries are defined on some Euler faces.
- Viscosity is defined.

For the default method FLOW-METHOD=POLPACK Euler meshes should have at least one element overlapping at the hole. When the meshes are created dynamically using MESH,ADAPT, this is taken care of automatically. When mesh sizes are comparable for two Euler meshes that are connected by a hole some reduction in costs is achieved by choosing the same mesh size and the same reference point for the two Euler meshes.

In general, holes should not be precisely on Euler element faces.

Deactivation

Deactivation is only supported by the Roe solver. In case you are using the multiple coupling surfaces functionality, it is also possible to deactivate a coupling surface and the associated Eulerian domain at a certain time using the `TDEAC` field on the COUPLE entry. The deactivation stops the calculation of the coupling algorithm and its associated Eulerian domain. The analysis of the Lagrangian structure continues. Activation of the coupling surface is not possible.

Initialization

To initialize Euler elements, several PEULER and TICEUL entries are used as follows:

```
PEULER1,6,,HYDRO,19
TICEUL,19,,,,,,,,+
+,SPHERE,1,3,5,1.0
SPHERE,1,,0.0,0.0,0.0,500.0
TICVAL,5,,SIE,400000.,DENSITY,0.2
MESH,22,BOX,,,,,,,+
+,-0.01001,-0.01001,-0.01001,0.14002,0.12002,0.12002,25,OUTSIDE,+
+,14,12,12,,,,EULER,6
$
PEULER1,7,,HYDRO,20
TICEUL,20,,,,,,,,+
+,SPHERE,2,3,6,2.0
SPHERE,2,,0.0,0.0,0.0,500.0
TICVAL,6,,SIE,400000.,DENSITY,1.9
MESH,23,BOX,,,,,,,+
+,0.11499,-0.00501,-0.00501,0.1302,0.11002,0.11002,50,OUTSIDE,+
+,26,22,22,,,,EULER,7
```

The MESH entry references a unique property number that is also used by the PEULER1 entry. So, the PEULER1 entry provides the link between the MESH entry and a TICEUL entry. In this way, each Euler domain references a unique TICEUL entry. The level indicators that occur on a TICEUL entry only apply to the Euler mesh that is linked to this TICEUL entry.

To initialize all meshes to one initial state only one property set is used in combination with only one PEULER1 and TICEUL entry. Also the PEULER entry may be used in this case, but TICEL is only supported by PARAM,FLOW-METHOD,FACET.

Output

Euler archives can be specified as usual if PARAM,FLOW-METHOD,FACET has been set. Euler archive output is restricted when this option facet has not been set. The restrictions are:

- The entry ELEMENTS on an Euler archive output request is required to be ALLEULHYDRO, ALLMULTIEULHYDRO, or ALLMULTIEULSTREN. Specifying element numbers is not supported.
- For each mesh, a separate Euler archive file is created. Thus, one Euler archive output request gives multiple archive files. Each of these archive files may contain more than one cycle. To distinguish the Euler archive files from each other, the Euler archive files have a tag that specifies to what mesh they belong. This tag has the form *FV_(Mesh-ID). here, FV is an acronym for "Finite Volume."
- When one of the Euler meshes is of TYPE ADAPT, all Euler archives will contain only one cycle.

For two meshes, an Euler archive output request reads (for example):

```
TYPE (ALLEULER) = ARCHIVE
ELEMENTS (ALLEULER) = 2
SET 2 = ALLEULHYDRO
ELOUT (ALLEULER) = PRESSURE,XVEL,YVEL,ZVEL
TIMES(ALLEULER) = 0.0 THRU 0.1 BY 0.01
SAVE (ALLEULER) = 99
```

Suppose a mesh with `ID=22` and a mesh with `ID=23` have been defined. Then this output request would generate the archives `ALLEULER_FV22_0` and `ALLEULER_FV23_0`.

Using the Dytran Preference of Patran

For an example, refer to workshop 11. The preference supports simulations with multiple coupling surfaces, each using a different Euler mesh entry. For Interactive failure, the preference also supports simulations with multiple coupling surfaces, each using a set of Euler elements. Interactive failure is activated by the entry COUP1INT.

Creating multiple coupling surfaces with multiple coupling Euler domains with Patran is done as follows:

1. Create geometry for the coupling surfaces and mesh these geometries. Use Loads-BCs/Coupling or Loads-BCs/Airbag to define each coupling surface. The airbag option allows for porosity definitions.
2. Make an arbitrary solid for each Euler mesh using geometry/Create/Solid. This solid only serves to associate an Euler Mesh to a 3-D property set. The geometry of the solid will not be used and any solid will suffice.
3. Create a material.
4. Create a 3-D Eulerian property for each of the dummy solids of step 2 using Properties/Create/3D/EulerianSolid. In defining the property set, select the material created in step 3.
5. Create the Euler meshes with Loads-BCs/Create/MeshGenerator. Use the coupling or air bag BC's from step 1 and the 3-D properties from step 4.

To define a porous hole between two closed surfaces, the surfaces should not be defined as coupling surfaces but as air bags. The first air bag is defined in one go, using all the elements in the first surface. To define the second air bag, two subsurfaces are created from the second surface. The first subsurface is the hole. The second subsurface consists of all the elements of the second surface that are not part of the hole. The second air bag is the combination of these two subsurfaces.

1. Create, with Loads-BCs/Create/Airbag/Surface, the first air bag using all elements in the first surface. This will be the first air bag.

Define a porous hole with Airbag/Subsurface and choose as porosity mode PORFLCPL or PORFCPL.

2. Create a subsurface consisting of all elements of the second surface that are not part of the subsurface created in step 2.
3. Create the second air bag by selecting the two subsurfaces in steps 2 and 3.

Fluid and Gas Solver for the Euler Equations

For gases and fluids flow, a state-of-the-art Eulerian solver is available that is based on the ideas of Professor Philip Roe. The fluid and gas Euler solver is based on the solution of so-called Riemann problems at the faces of the finite-volume elements. The mathematical procedure amounts to a decomposition of the problem into a discrete wave propagation problem. By including the physics of the local Riemann solution at the faces, a qualitatively better and physically sounder solution is obtained. The fluid and gas solver is also known as an approximate Riemann solver.

The solver can be either first or second order accurate in space in the internal flow field. Second order spatial accuracy is obtained by applying a so-called `MUSCL` scheme in combination with a nonlinear limiter function. The `MUSCL` approach guarantees that no spurious oscillations near strong discontinuities in the flow field will occur. The scheme is total variation diminishing (TVD), meaning it does not produce new minima or maxima in the solution field. The original Roe solver can be activated by using the PARAM,LIMITER,`ROE` entry in the input file.

Improvements have been made to arrive at a full second order scheme in the fluid and gas solver to further gain accuracy in the solution. All boundary conditions –that is, the flow boundary conditions, and the wall boundary conditions, are fully second order accurate in space. You can use the new and improved solver (either first or second order) by entering the keyword `2ndOrder` or `1stOrder` on the PEULER or PEULER1 entry.

Furthermore, a so-called "entropy fix" has been added to avoid the sharp discontinuities in those areas where the eigenvalues of the local Riemann problem vanish. In effect, the entropy fix ensures that an expansion shock (although mathematically sound) is broken down into a correct expansion fan. The expansion shock would yield a physical impossibility of decreasing entropy in the system. That is the reason for the name "entropy fix". The entropy fix brings a very – almost unnoticeable – form of locally necessary dissipation into the solution to improve the differentiability of the equations where the eigenvalues vanish.

The time integration in the fluid and gas solver is performed by a multistage time integrator, also know as a Runge-Kutta type scheme. Higher order temporal accuracy can be achieved by applying multiple stages in the time integration. The required number of stages is automatically selected when you select either a first order or a second order solution. A first order spatial accurate solution uses a one-stage time integration scheme; a second order spatial accurate solution applies a three-stage time integration scheme.

When a coupling surface is required you have to use the COUPLE1 entry. Multiple coupling surfaces with failure can be requested if the fast coupling algorithm is used by setting the PARAM,FASTCOUP, `,FAIL` entry (see Multiple Coupling Surfaces in this chapter). You can also define interaction between multiple coupling surfaces, like in cases where you wish to model "chambers" that after structural failure will show a "connection" between them through which fluid or gas could flow. A typical example is an explosion in the cargo space of an aircraft after which the floor may be ruptured and the high-pressure gas can vent into the passenger cabin.

The fluid solver allows you to introduce viscosity into the solution. You can use Tait's equation of state to model the fluid with additional viscosity terms.

There are some limitations in the current implementation. Eulerian elements must be completely filled with materials, so void or partial void elements are not allowed. For fluid flows where voids sometimes may occur, we recommend that you use Tait's equation of state. This equation of state is fully supported by the improved second order Euler solver, and allows you to define a so-called critical density. When the fluid's density falls below the critical value, the fluid cavitates (i.e. the pressure retains the value associated with the critical density). The density can further drop, but the pressure remains constant. In this fashion, you avoid the creation of voids and still allow the fluid to cavitate. When you only have data available for the fluid that satisfies the simple bulk equation of state, you can still use Tait's model by setting the A_0 parameter to zero, γ to one, and add the critical density value at which cavitation occurs.

The ALE interface to Lagrangian structures is not supported. Air bag applications are not supported as they tend to show void elements during the analysis.

Especially for blast wave types of problems, the full second order solution is recommended because of the accuracy it inherently brings. The JWL equation of state is not supported. A major advantage of using the blast wave approach is the speed at which the analysis can be performed. Especially spherical wave propagation through a Cartesian mesh is much more accurate in a full second order solution than in first order, or second order with (cheaper) first order accurate boundary conditions.

Modeling Fluid Filled Containers

Containers, for example plastic bottles, are often subjected to axial loading. Axial loading occurs when the bottle is crushed, or for example, stacked. It may concern both empty and (partially) filled bottles. Fluid filled containers or bottles can be modeled using a full multi-material Euler description. However, using full multi-material Euler fluid dynamics solver is a quite expensive method to solve the quasi-static behavior of the fluid. An alternative way of modeling is available through the FFCONTR (**F**luid **F**illed **CONT**aine**R**) option. Using this option removes the need for a full fluid dynamics solution.

The FFCONTR option uses the uniform pressure algorithms to calculate the pressure increase due to the compression of the container. The pressure is uniformly distributed but may change in time due to volume changes that occur when the container deforms.

You need to define a surface to indicate the boundary of the container. The volume enclosed by the surface then equals the volume of the container.

The normals of the faces of the surface must point outwards in order to compute the correct (positive) volume. When the normals point inwards, they are automatically reversed such that the resulting volume is positive. The surface must be closed to ensure a correct volume calculation.

You must define the amount of fluid in the container or bottle. The volume of the gas above the fluid then follows immediately from the difference between the volume of the container and the fluid volume in the container. Obviously, the fluid volume cannot exceed the volume enclosed by the surface. The fluid is assumed to be incompressible. Thus, any volume change directly translates to a volume change of the gas above the fluid. The gas above the fluid is assumed to behave as an ideal gas under iso-thermal conditions:

$$p - V = C$$

Where p is the pressure, V is the volume and C represents a constant.

To define the constant C, you have to specify the initial pressure of the gas above the fluid on the FFCONTR entry. The initial pressure is only used for the calculation of the pressure changes and does not have an effect on other boundary conditions that you may have applied. If an over-pressure (for example, a carbonated soft drink) or an under-pressure (for example, a hot filled container) is present, you must model this separately using a PLOAD definition. The values for the pressure on the PLOAD entry and the pressure generated by the FFCONTR are superimposed for the calculation.

Hotfilling

Filling bottles with hot liquid can cause large deformations during cooling. To simulate these deformations, the fluid filled functionality container option can be used. Since Dytran has only a limited cooling functionality, the temperature of the fluid has to be specified by the user. In addition, the volume of the fluid will depend on temperature. A temperature versus time table and water density versus temperature table are input options for the fluid-filled container. If these are set, the gas is no longer iso-thermal but satisfies:

$$PV/T = C$$

Here, V is the volume of the gas. This volume is computed as the difference between the total volume and the volume of fluid. The volume of the fluid is given by:

$$V = \frac{M_{water}}{\rho}$$

Here, the fluid density, ρ, depends on the temperature as specified by table entry.

Arbitrary Lagrange-Euler (ALE) Coupling

As stated for the general coupling, ALE also acts to enable the material modeled by the Eulerian and Lagrangian meshes to interact. The two meshes initially must be coupled to each other by an ALE interface surface. The Lagrangian and Eulerian grid points in the interface surface coincide in physical space but are distinct in logical space. The interface serves as a boundary for the flowing Eulerian material during the analysis. The Eulerian material exerts pressure on the Lagrangian part of the interface that is distributed as forces to the Lagrangian grid points.

The interface moves as the Lagrangian structure deforms. Thus, the Eulerian mesh boundary also moves. In order to preserve the original Eulerian mesh and have it follow the structural motion, the Eulerian grid points can be defined as ALE grid points. In that case, the motion of the ALE interface is propagated through the Eulerian mesh by the ALE motion algorithm.

In an ALE calculation, the Eulerian material flows through the mesh and the mesh can also move at the same time. The material can have a velocity relative to the moving mesh which makes it an Eulerian formulation.

The ALE interface can not be used in combination with Eulerian single material elements with strength and in combination with the Roe Solver.

Efficiency

Since the ALE coupling does not require geometrical calculations during the analysis, it is potentially faster than the general coupling. However, the deformation of the structure at the interface should be smooth but not necessarily small. Bird-strike analyses are typical ALE applications. The deformation is usually large but smooth in time.

Automatic Coupling

Introduction

To simulate fluid-structure interaction two methods are available. The first method called general coupling can be applied to non-orthogonal Euler meshes but is expensive. The second method called fast coupling requires the Euler mesh to be orthogonal and is considerable faster. This make the fast coupling approach the most used method. To enable fluid-structure with fast coupling the following has to be done:

- Define an orthogonal Euler domain using CHEXA's or MESH, BOX
- Define a surface that contains all structural segments that interact with the fluid. This is called the coupling surface.
- Define a COUPLE entry that uses this surface.
- Add PARAM, FASTCOUP

Furthermore, the coupling surface has to closed. It cannot have holes. It also cannot have T-joints. If there are T-joints Dytran terminates with an error.

If there are holes then the user has to mesh the holes with dummy segments. For an example refer to Blastwave Hitting a Bunker (EP4_7) (Ch. 4) in the *Dytran Example Problem Manual*. Here the sides of the bunker are open and have to be meshed with dummy segments. In addition, two coupling surfaces have to be defined. One coupling surface serves to model the fluid inside the bunker and the other one to model the fluid outside the bunker. In addition, two Euler domains have to be defined. In realistic models the treatment of holes can require too much effort from the user.

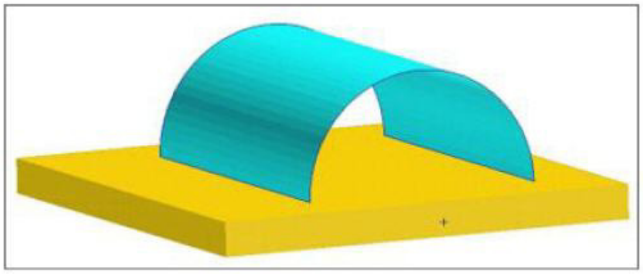


Figure 4-1 Bunker

Similarly, it is possible to handle T-joints by defining multiple coupling surfaces as illustrated in Figure 4-2 and Figure 4-3. Here the "coupling surface 1" consists of the lines in light blue and orange. The porous dummy surface is used by both coupling surfaces and allows flow between the two Euler domains. For realistic sloshing models with baffles, this approach requires much effort from the user and is not feasible.

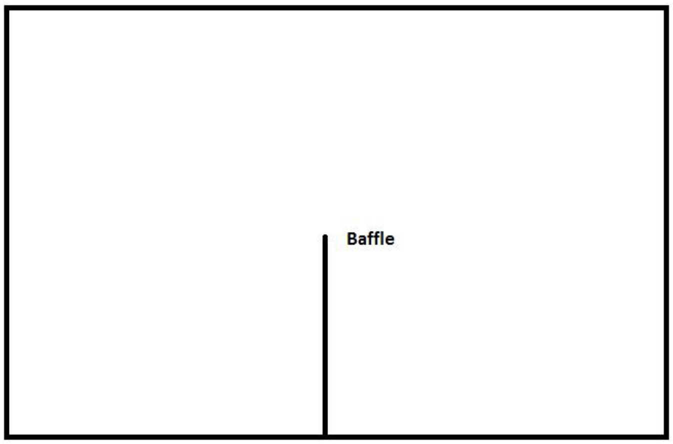


Figure 4-2 Model with baffle

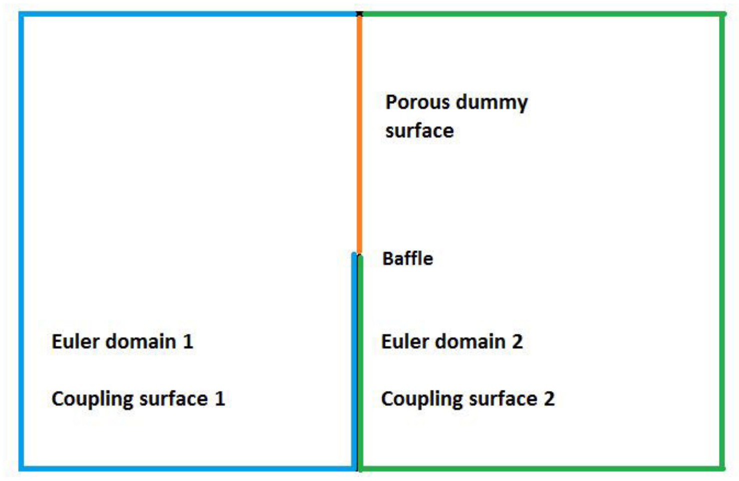


Figure 4-3 Using the fastcoup approach to handle T-joints

To allow for holes and T-joints the requirement of a closed coupling surface has to be removed. A versatile method is to redefine the coupling surface concept. The coupling surface defines what part of the Euler domain is covered by the structure. Therefore, it has to be closed. To avoid this requirement, the meaning of the coupling surface has to be changed. The coupling surface will no longer act as a container of fluid but will only act as a barrier to fluid flow. This new approach is called the auto coupling approach and is activated by using PARAM, AUTOCOUP.

Using the coupling surface as barrier

The requirement of a closed surface means that fluid can only be inside or outside the coupling surface. Consider for example a cylindrical surface. Figure 4-4 shows a cross-section of the cylinder and Euler mesh. The circle in red represents the cylindrical segments. The lines in light blue the Euler elements. Here it assumed that COUPLE uses COVER=OUTSIDE. Therefore, the outside region is covered by the coupling surface and only the inside region can contain fluid. The fluid is marked with dark blue.

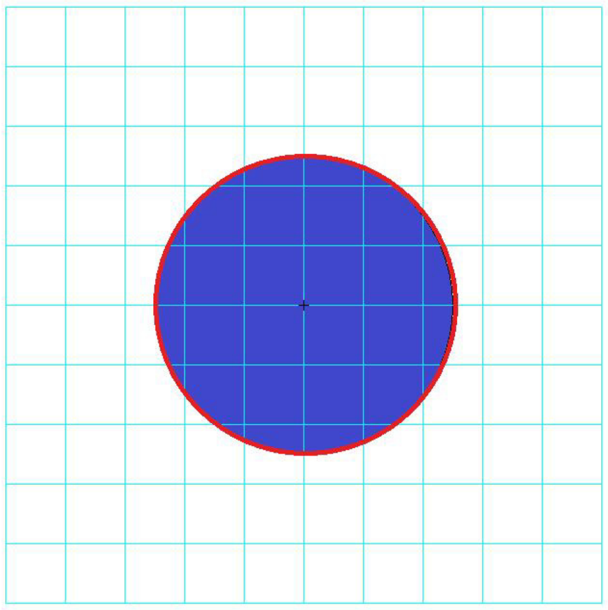


Figure 4-4 Fluid can only be inside the cylinder

There are three types of Euler elements.

- Elements that are outside, they cannot contain fluid
- Element that are completely inside. They can be completely filled with fluid
- Elements that are partially inside. Only the part of the Element that is inside the coupling surface can be filled with fluid. The other part cannot.

When the coupling surface is closed, it is straight forward to determine what parts of the element can contain fluid.

When using the coupling surface as barrier then both sides can contain fluid as shown in Figure 4-5. This is shown by the dark blue color.

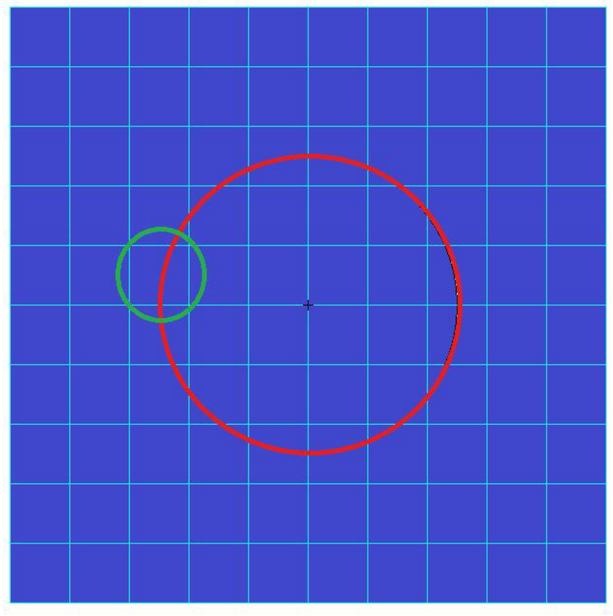


Figure 4-5 Fluid on both sides of the cylinder

All the Euler elements can contain fluid. An Euler element that is intersected by the structure can contain fluid on both sides. Each side can have a distinct mass, pressure and density. This requires that the Euler element is split into parts. These element parts will be called sub elements. Consider for example the element marked with the green circle. Figure 4-6 shows this element. It is split into two parts that are called sub element 1 and 2.

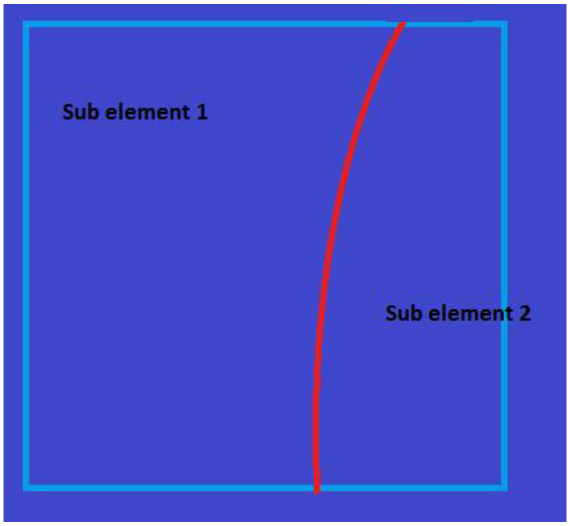


Figure 4-6 Splitting an Euler element into two sub elements

The sub elements are defined by their boundaries. These boundaries are shown in Figure 4-7as oriented line segments. The green lines are parts of Euler element faces. The orange lines are parts of structural segments.

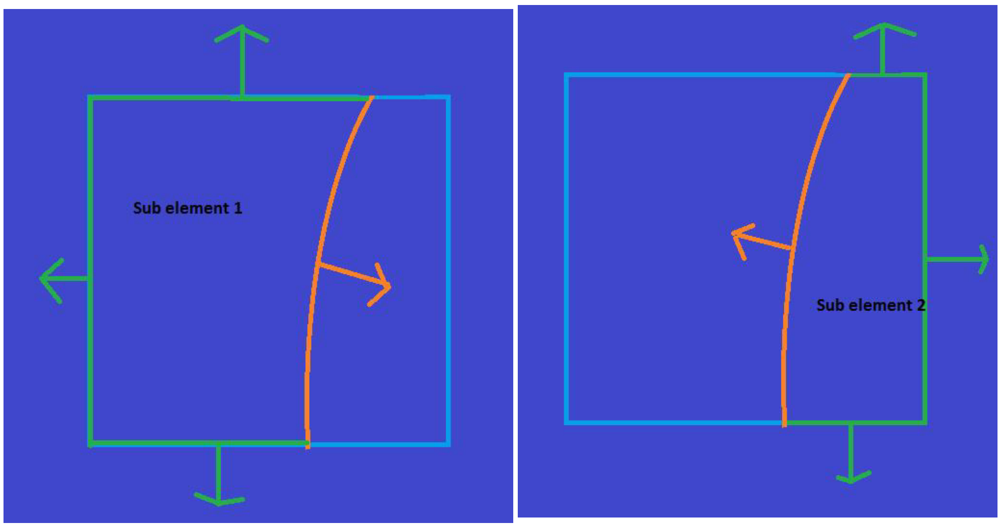


Figure 4-7 Boundaries of sub element 1 and sub element 2

The lines in green and orange form a closed loop, without holes. In general, a sub element is defined by giving a list of oriented structural segments. This list will be called the subsurface of the sub element. To properly define a sub element, the subsurface of the sub element should have no holes within the parent Euler element.

Since a list of oriented segments defines the sub element, they can be used to connect sub elements to each other. In figure 8 two Euler elements are shown. The lines in red are the sub surfaces of sub element 1 and 3. Sub element 1 and 3 share one oriented segment. Therefore, sub element 1 is connected to sub element 3. As a result, fluid can flow between sub element 1 and 3. Also Sub element 1 is not connected to sub element 4, because they do not share a common oriented segment.

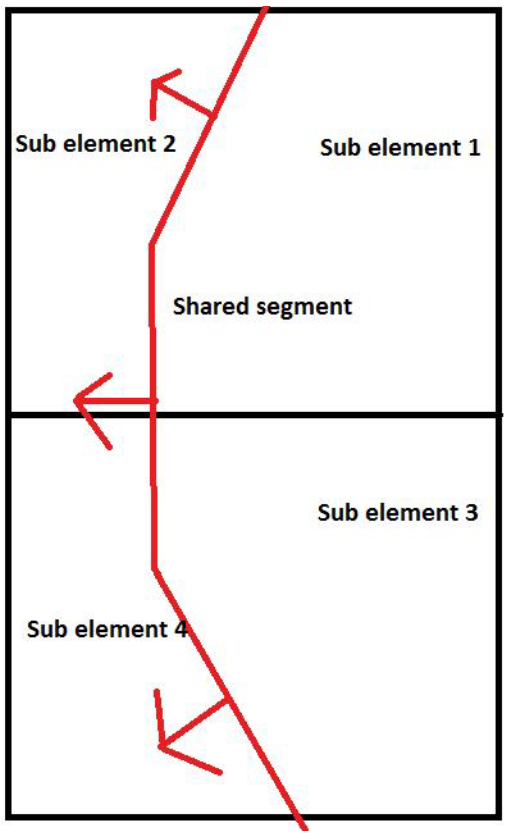


Figure 4-8 Connecting sub elements

Enabling holes

Consider a coupling surface with a hole as shown in Figure 4-9.

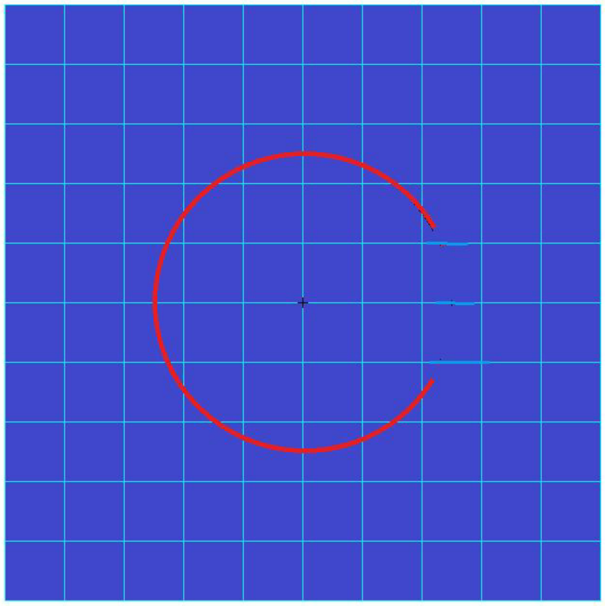


Figure 4-9 Cylinder with hole

Here the structure ends within two Euler elements. With the fast coupling approach dummy segments have to be created for the full hole. This is shown in Figure 4-10. The dummy segments make the coupling surface closed. Here the dummy segments are shown in orange. If there is flow through the hole and if the fast coupling approach is used, two Euler domains have to be used. One for the inside and one for the outside. Figure 4-11 and Figure 4-12 show the two Euler domains. By using two Euler domains fluid can be inside but also outside the coupling surface.

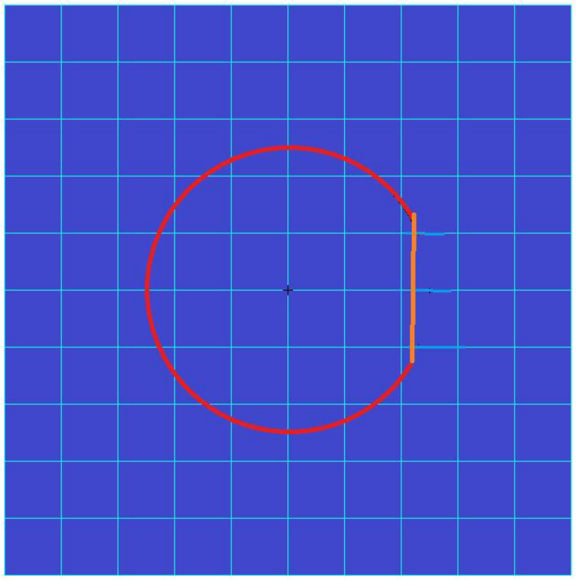


Figure 4-10 Creating dummy segments for the fast coupling approach

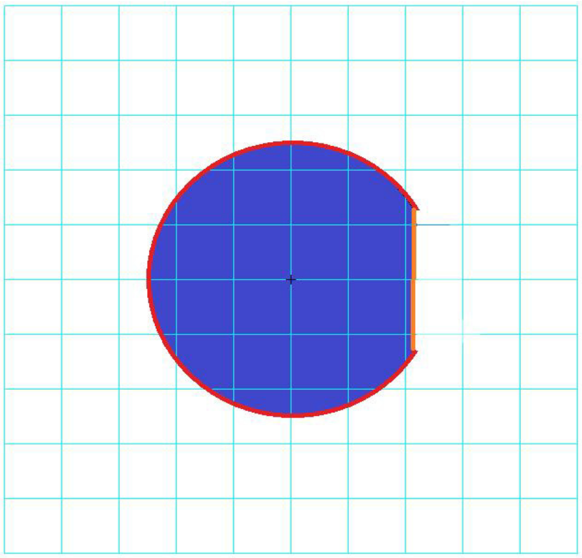


Figure 4-11 Euler domain for the outside region

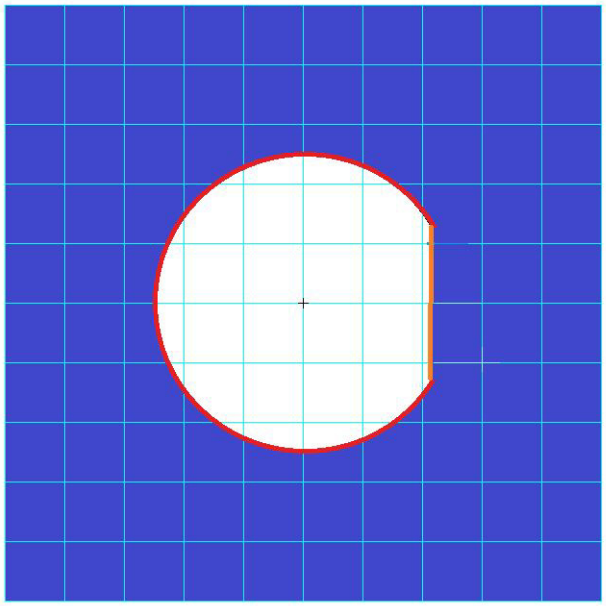


Figure 4-12 Euler domain for the outside region

By viewing the coupling surface as barrier only the elements that contain a hole need a special treatment. In these elements there is a barrier that ends within the element. This barrier splits the Euler element into two sub elements. In Figure 4-9 only two Euler elements have holes. Consider the top element. The sub surfaces used for the two sub elements are shown in Figure 4-13. But they have holes within the Euler element. Therefore, to properly define the sub elements dummy segments are added to fill the holes within the Euler element. This is shown in Figure 4-14 and Figure 4-15. Note, that the hole has only to be closed within the Euler element, that contains the hole. This is a major difference with the fast coupling approach, where the complete hole has to filled with dummy segments.

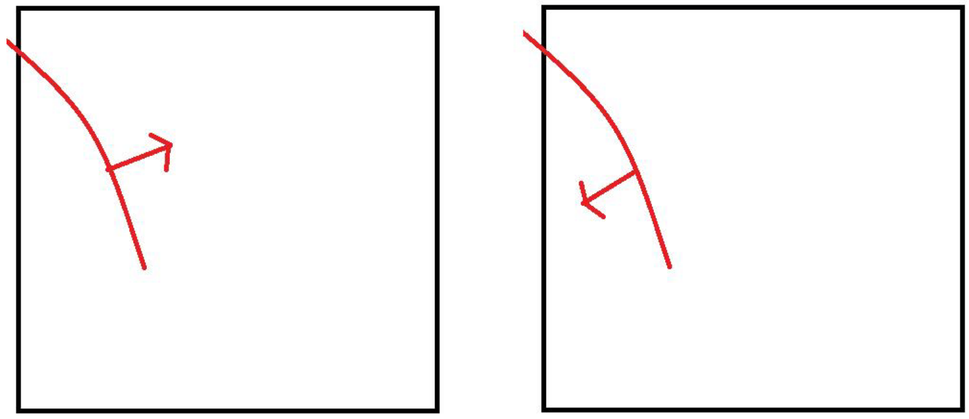


Figure 4-13 Subsurface for sub element 1 and sub-surface for sub element 2

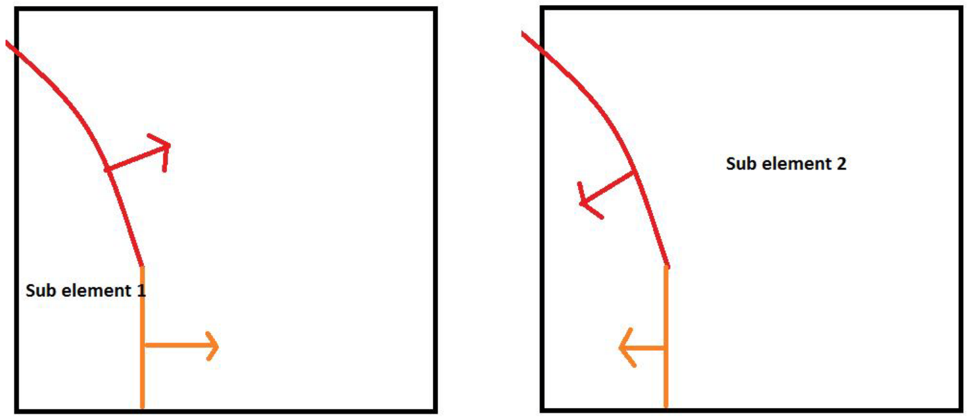


Figure 4-14 Filling holes in the sub surfaces of the sub elements

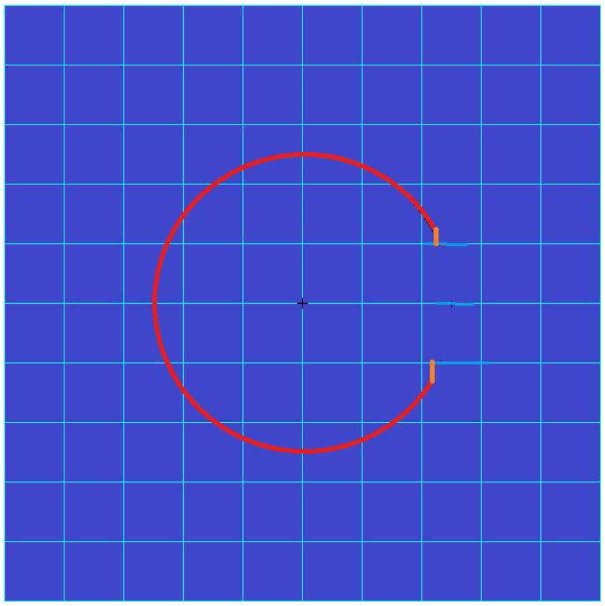


Figure 4-15 Adding dummy segments

To show the creation of dummy segments in more detail, consider a square plate with hole.

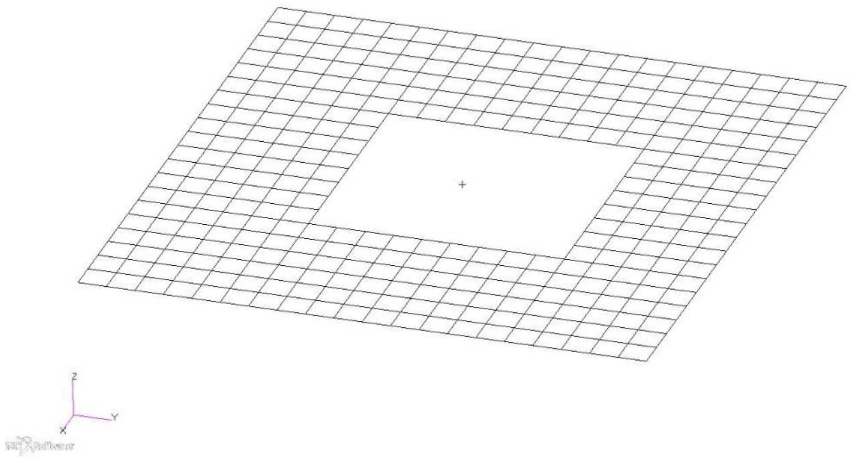


Figure 4-16 Plate with hole

Figure 4-17 shows that some of the structural segments end within Euler elements. Figure 19 shows one of these Euler elements. It is the element in the center of the red circle as shown in Figure 18. Here the lines in green are the free edges of the structure. These free edges are part of the hole. At the hole the structural segments are not attached and they just end within the Euler elements. Therefore, within the Euler element the coupling surface has a hole. These holes within the Euler element will be filled by adding dummy segments.

To mesh the hole shown in Figure 17 the free edges of the hole have to be attached to the Euler faces This is done by orthogonal projecting of points A and B to the opposing Euler faces. Point A is projected to point C at the right Euler face and point B is projected on point D at the to the top Euler face. The two green edges and two orange edges form a rectangle that is meshed by four tria's. They will be called dummy segments.

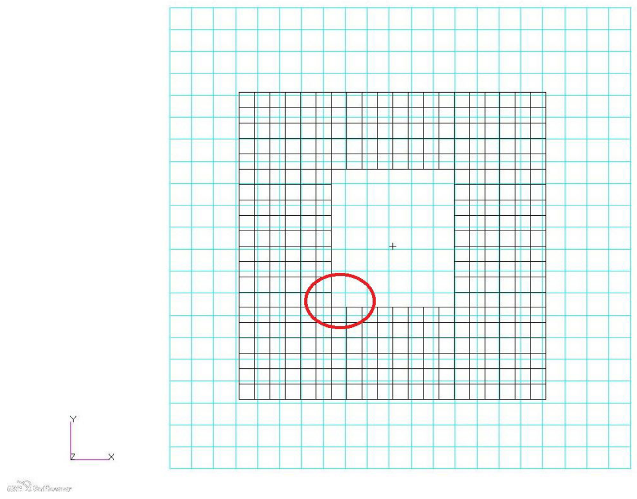


Figure 4-17 Plate with Euler mesh

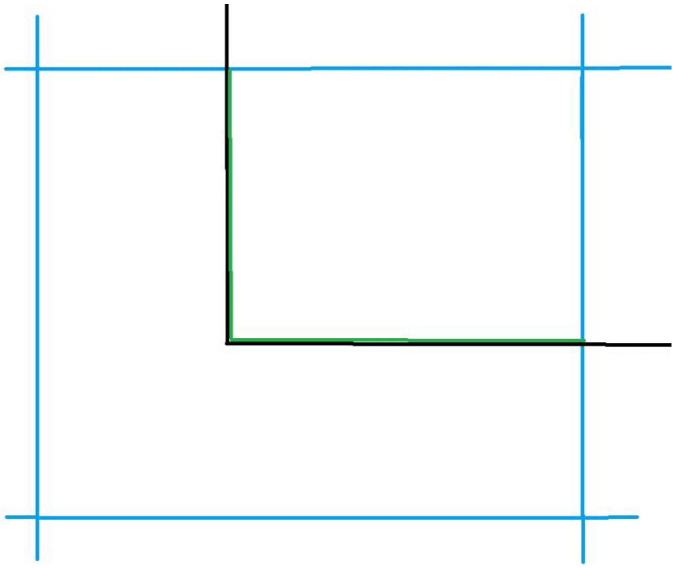


Figure 4-18 Free edges of the hole in green

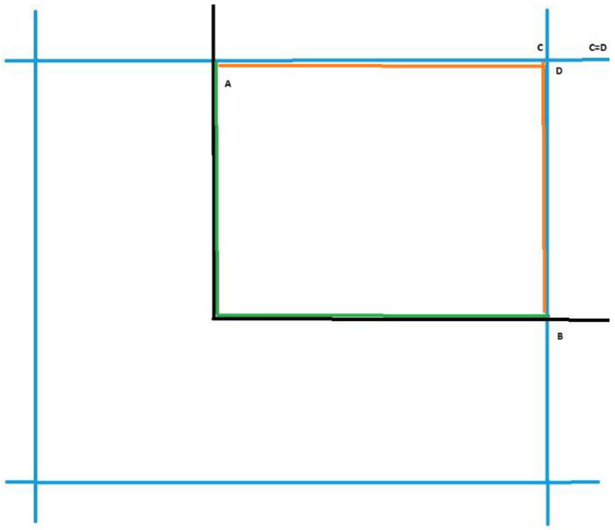


Figure 4-19 Connecting points A and B to opposing Euler faces

The dummy segments can be viewed on the coupling surface archive. By setting field `DUMMY` of `PARAM`, `AUTOCOUP` to `YES`, dummy segments are written to the coupling surface archive as shown in Figure 4-21.

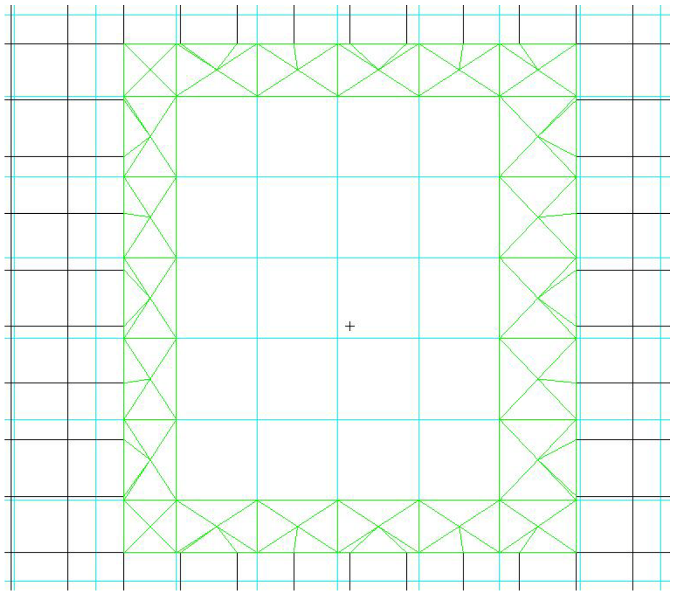


Figure 4-20 Dummy segments

The Mine Blast (EP4_8) (Ch. 4) in the *Dytran Example Problem Manual* has a structural surface with holes. Figure 4-21 and Figure 4-22 show the coupling surface without and with dummy segments. Figure 4-23 shows how structural segments and dummy segments fit into an Euler element.

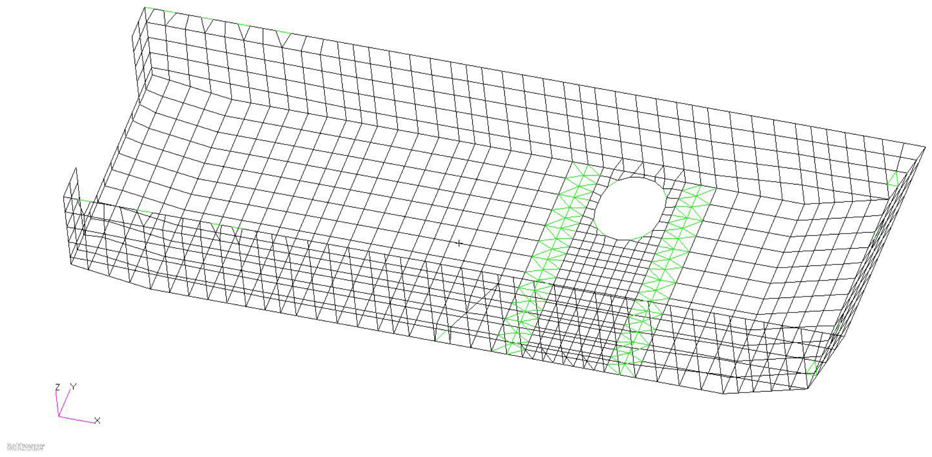


Figure 4-21 Coupling surface without dummy segments

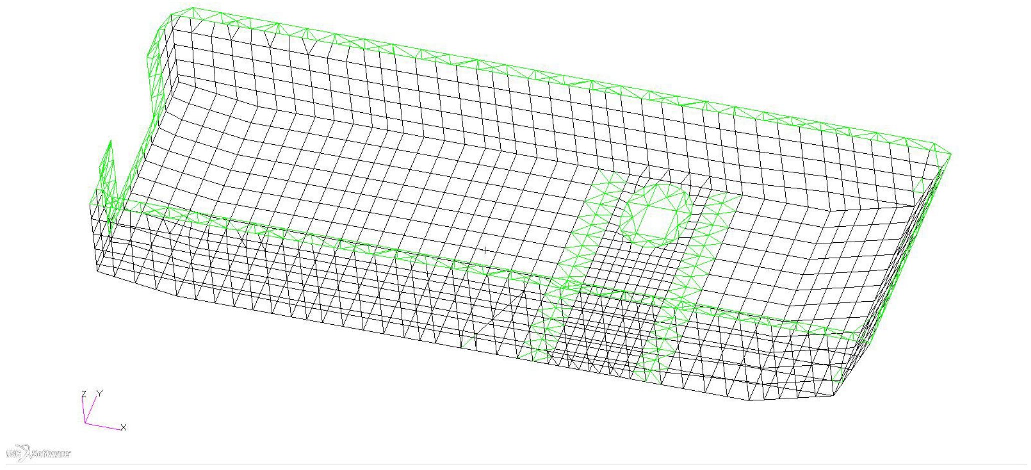


Figure 4-22 Coupling surface with dummy segments

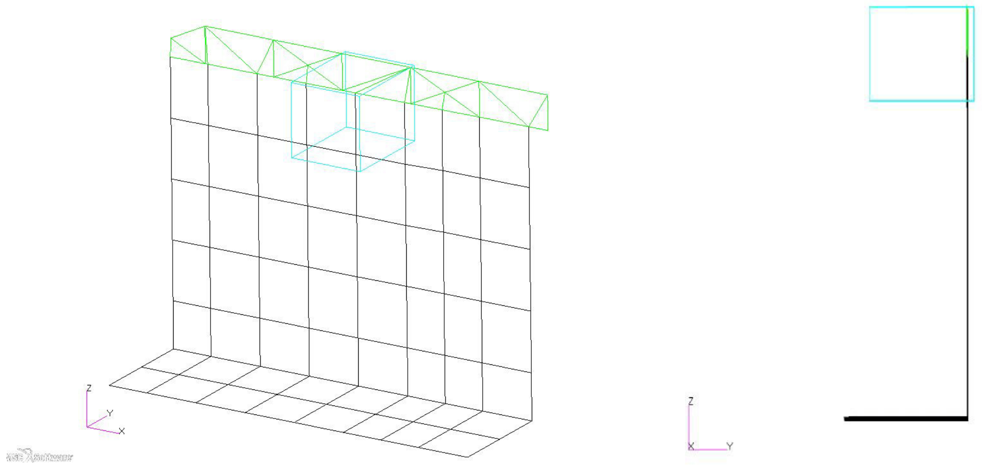


Figure 4-23 How the dummy segments fit into The Euler element

T-joints

Splitting Euler elements into sub elements also allows for T-joints. For the model given in Figure 4-24 the auto coupling approach give the sub elements shown in Figure 4-25.

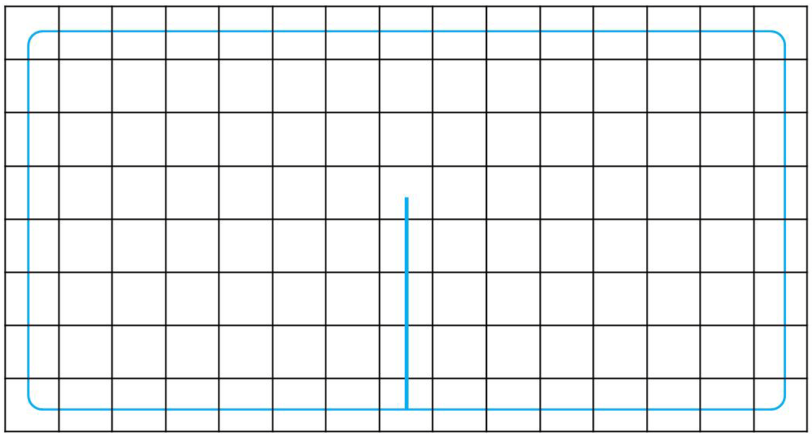


Figure 4-24 Model with baffle

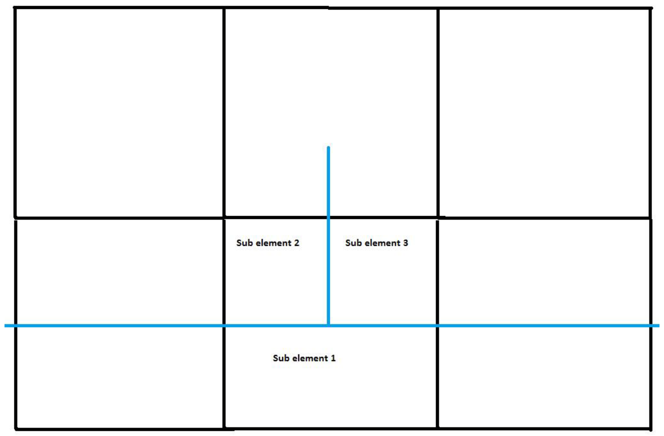


Figure 4-25 Sub elements for model with baffle

The sub surfaces of each sub element are illustrated in Figure 4-26.

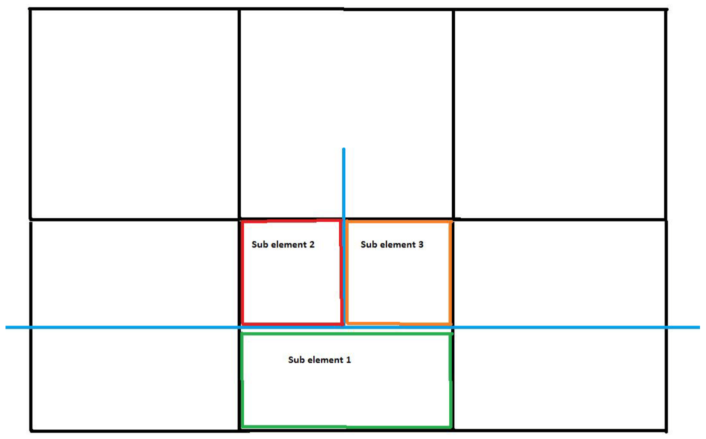


Figure 4-26 The three regions that define the sub elements

The sub element 1, 2 and 3 have to be connected to other elements and to sub elements. This is shown in Figure 4-27.

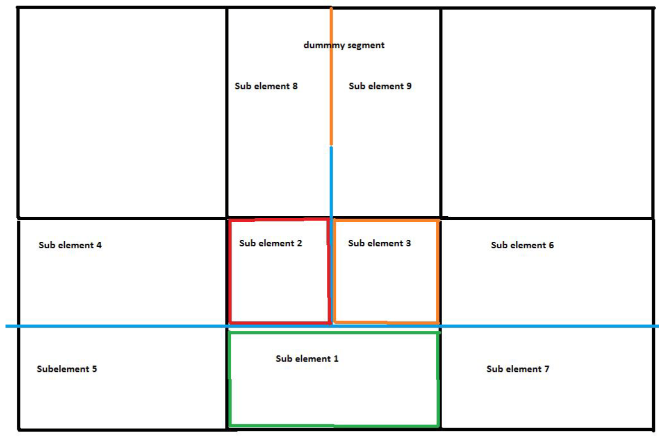


Figure 4-27 Connecting sub elements

The Euler element in the top middle has a hole and a dummy segment is created. In addition, the element is split into sub element 8 an 9. Several connections are made. Following table shows the connections/links between sub elements:

	Subelm 1	Subelm 2	Subelm 3	Subelm 4	Subelm 5	Subelm 6	Subelm 7	Subelm 8	Subelm9
Subelm1		Linked	Linked		Linked		Linked		
Subelm2	Linked		Linked	Linked				Linked	
Subelm3	Linked	Linked				Linked			Linked

If sub elements are linked there can fluid flow between them.

At baffles it is often needed to create dummy segments illustrated in Figure 4-28 and Figure 4-29.

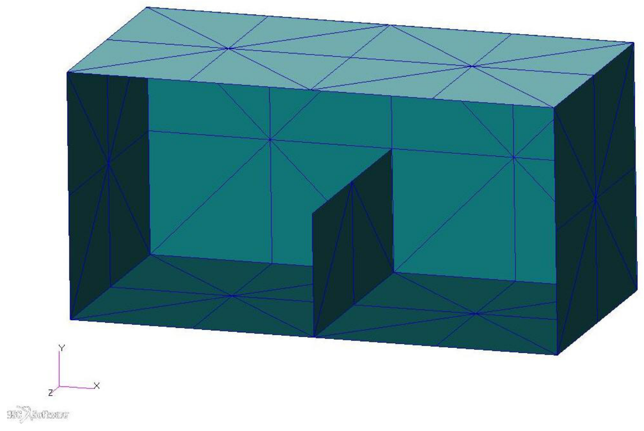


Figure 4-28 Baffle

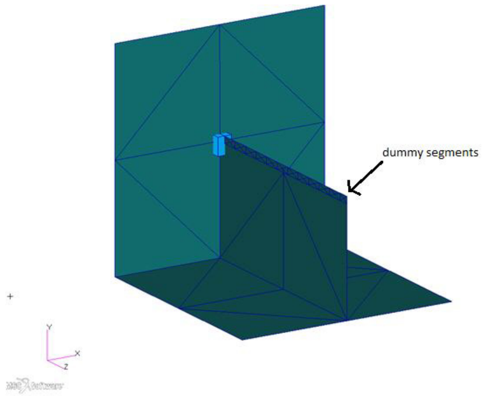
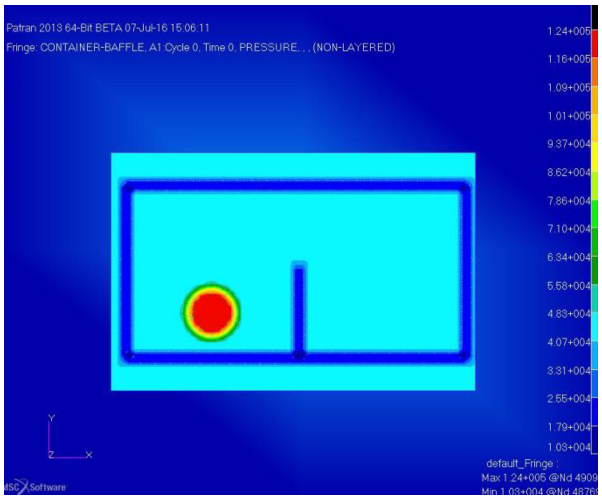


Figure 4-29 Adding dummy segments at the baffle

Euler archive output with auto coupling

Euler elements that are intersected by the structure are split into sub elements. On the Euler achieves these Euler elements will get by default zero values:



Patran 2013 64-Bit BETA 07-Jul-16 15:06:11
Fringe: CONTAINER-BAFFLE, A1:Cycle 0, Time 0, PRESSURE, , (NON-LAYERED)

default_Fringe :
Max 1.24+005 @Nd 4909
Min 1.03+004 @Nd 4876

Figure 4-30 Zero pressure at Euler elements intersected by the coupling surface

By using option `OUTPUT` of `PARAM, AUTOCOUP` actual sub element values can be seen.

Euler initialization

The field `COVER` of `COUPLE` defines which parts of the Euler domain can contain fluid. In tank sloshing the outer surface is often closed. In that case the tank surface divides the Euler domain into an inside and outside region. An example is the tank model shown in Figure 4-28. This model has holes, but the outer surface is closed. In that case, `COVER = OUTSIDE` can be used with `PARAM, AUTOCOUP` and only Euler elements inside the outer structure will contain fluid. If the coupling surface does not divide the Euler domain into an inside and outside, then `COVER = NONE` has to be used. Then there can be fluid on both sides of the coupling surface.

Limitations of auto coupling

The following items are not supported:

- Self-intersections of the coupling surface.
- Splitting an Euler element in more than three sub element.
- Segments on Euler faces.

In these three cases an error message will be given.

This implementation is not suitable for folded air bag simulations.

Features that are not supported are:

- Multiple Euler domains and multiple coupling surfaces
- Adaptive Euler
- Surface failure
- `COUPLE` with option `AIRBAG`
- Graded meshes
- Euler import
- Viscosity
- Porosity
- Dmp
- Using eulercubes
- Markers
- Interactive failure, `COUP1INT`
- `COUP1FL`, flow through failed segment to ambient.

Running with auto coupling

To run a fluid-structure interaction simulation with auto coupling the following has to be done:

1. Add `PARAM, AUTOCOUP`. This PARAM cannot be combined with `PARAM, FASTCOUP`.
2. Use only one Euler domain. The Euler domain can be defined by a MESH entry or by CHEXA’s.

3. Use only one coupling surface. All structure involved in the fluid-structure interaction can be put into one coupling surface. With auto coupling the coupling surface can consist of structures that are disconnected.

   Modeling with fast coupling can require multiple coupling surfaces and Euler domains. As explained above with auto coupling there is no longer any need for multiple Euler domains or multiple coupling surfaces.

There is no need to use the porosity models `PORFCPL` or `PORFLCPL`. They enable flow between coupling surfaces and are only meant for the multiple coupling surface approach. They have no use for auto coupling. With auto coupling flow through holes is taken into account automatically.

In the auto coupling method dummy segments and sub elements are created. For each sub element a coupling surface computation is done. In addition these sub elements have to be connected together.

This makes it a sophisticated method and the method can give errors for more complicated models. The following errors can occur:

- `%E-P4310103-V4_SUBELMS_DEFINE_SUBSURFS,,,`

  Joint type not implemented yet.
- `%E-P4310301-V4_SUBELMS_INSECT_SUBSURFS_TJ,,,`

  inconsistent coupling surface computation for element 27179.

There are no work arounds for these errors. These errors will be addressed in the next Dytran release.

With fast coupling, coupling surface segments can be on Euler faces. Auto coupling does not support this. If this occurs an error is given.

Also, free edges of segments on Eulerfaces can give following warnings:

- `%W-P4309901-V4_SUBELEM_CONNECT_EXTENDED_GPS,,,`

  Program error: please contact MSC.

  Dummy surface cannot be triangulated.
- `%W-P4308103-V4_CREATE_DUMMY_SEGMENTS,,,`

  Dummy surface cannot be triangulated. This is program error.

  Please contact MSC.Software.

Then, slightly changing the geometric properties of the Euler mesh can avoid the free edge/segment on Euler face problem.

For example, if using:

```
MESH,1,BOX,,,,,,,+
+,-1.5,-1.5,-1.5,3.0,3.0,3.0,,,+
+,20,20,20,,,,EULER,1
```

gives the dummy surface warning, then slightly changing the mesh as follows:

```
MESH,1,BOX,,,,,,,+
+,-1.501,-1.501,-1.501,3.002,3.002,3.002,,,+
        +,21,21,21,,,,EULER,1
```

can often solve the problem.

5 Application Sensitive Default Setting

Dytran is capable of handling an extensive variety of applications. Due to the variety, it is sometimes difficult to make the correct choice of default settings according to the application at hand. Application sensitive default settings make it easier to select the appropriate element formulations or numerical algorithms to achieve the best solution possible in terms of accuracy and CPU time. By default, Dytran attempts to provide you with the most accurate solution possible. This default setting can also be achieved by including a `SETTING`, `SID`, `STANDARD` entry in your input file. An overview is given in Hierarchy of the Scheme which settings are automatically done when the default applies.

Overview of Default Definition

Element Formulation

Shell elements:

- Key-Hoff elements are used with three integration points through the element thickness.
- The element thickness is strain dependent.
- The element transverse shear stresses are computed assuming a linear distribution of the stress.
- Shear-locking is avoided.

Solid elements:

- One-point Gauss integration (PSOLID)

Hourglass Suppression Method

Shell elements:

- Flanagan-Belytschko Stiffness (`FBV`) where the warping coefficient is equal to 0.1 and rigid body rotation correction is not active.

Solid elements:

- Flanagan-Belytschko Stiffness (`FBS`)

Method for Material Plasticity Behavior

Shell elements:

- Plasticity is treated by an iterative scheme, using as many iterations as necessary (the total number of iterations is limited to 20)

Note: The method for material plasticity behavior does not apply to all material models available. For example, the SHEETMAT material model applies a special algorithm that does not require an iterative method.

Application Type Default Setting

In addition to the STANDARD definition, four other application types are available to influence the default settings:

- CRASH – The defaults set for optimal crash-type analysis.
- SHEETMETAL – The defaults set for optimal sheet metal forming analysis.
- SPINNING – The defaults set for optimal fast rotating structures.
- FAST – The defaults set for optimal fast, but not necessarily the most accurate solutions.
- VERSION2 – The defaults set to pre-Version 3.0.

The resulting default settings are listed below for each of the above mentioned applications.

Crash

Element Formulation

Shell elements:

- BLT (Belytschko-Lin-Tsay) elements are used with three integration points through the element thickness.
- The element thickness is strain dependent.
- The element transverse shear stresses are assumed constant through the element thickness.
- Shear-locking is not avoided.

Solid elements:

- One-point Gauss integration (PSOLID)

Hourglass Suppression Method

Shell elements:

- Flanagan-Belytschko Stiffness (`FBV`) where the warping coefficient is equal to 0.1 and rigid body rotation correction is not active.

Solid elements:

- Flanagan-Belytschko Stiffness (`FBS`)

Method for Material Plasticity Behavior

Shell elements:

- Plasticity is treated by an iterative scheme, using as many iterations as necessary (the total number of iterations is limited to 20).

Sheet Metal

Element Formulation

Shell elements:

- BLT (Belytschko-Lin-Tsay) elements are used with five integration points through the element thickness.
- The element thickness is strain dependent.
- The element transverse shear stresses are computed assuming a constant distribution of the stress.
- Shear-locking is not avoided.

Solid elements:

- One-point Gauss integration (PSOLID)

Hourglass Suppression Method

Shell elements:

- Flanagan-Belytschko Stiffness (`FBV`) where the warping coefficient is equal to 0.1 and rigid body rotation correction is not active.

Solid elements:

- Flanagan-Belytschko Stiffness (`FBS`)

Method for Material Plasticity Behavior

Shell elements:

- Plasticity is treated by an iterative scheme, using as many iterations as necessary (the total number of iterations is limited at 20).

Spinning

Element Formulation

Shell elements:

- Key-Hoff elements are used three integration points through the element thickness.
- The element thickness is strain dependent.
- The element transverse shear stresses are computed assuming a linear distribution of the stress through the element thickness.
- Shear-locking is avoided.

Solid elements:

- One-point Gauss integration (PSOLID)

Hourglass Suppression Method

Shell elements:

- Dyna method with the warping coefficient set to zero and rigid body rotation correction is active.

Solid elements:

- Original Dyna suppression method.

Method for Material Plasticity Behavior

Shell elements:

- Plasticity is treated by an iterative scheme, using as many iterations as necessary (the total number of iterations is limited to 20).

Fast

Element Formulation

Shell elements:

- BLT (fast Belytschko-Lin-Tsay) elements are used with three integration points through the element thickness.
- The element thickness is constant.
- The element transverse shear stresses are assumed to be constant through the element thickness.
- Shear-locking is not avoided.

Solid elements:

- One-point Gauss integration (PSOLID)

Hourglass Suppression Method

Shell elements:

- Flanagan-Belytschko Stiffness (`FBV`) where the warping coefficient is equal to 0.1 and rigid body rotation correction is not active.

Solid elements:

- Original Dyna suppression method.

Method for Material Plasticity Behavior

Shell elements:

- Plasticity is treated as a one step radial scale back scheme.

Version2

Element Formulation

Shell elements:

- Bely (original Dyna Belytschko-Lin-Tsay) elements are used with three integration points through the element thickness.
- Element thickness is constant.
- Element transverse shear stresses are assumed to be constant through the element thickness.
- Shear-locking is not avoided.

Solid elements:

- One-point Gauss integration (PSOLID)

Hourglass Suppression Method

Shell elements:

- Flanagan-Belytschko Stiffness (`FBV`) where the warping coefficient is equal to 0.1 and rigid body rotation correction is not active.

Solid elements:

- Original Dyna suppression method

Method for Material Plasticity Behavior

Shell elements:

- Plasticity is treated as a one step radial scale back scheme.

Hierarchy of the Scheme

Dytran has many more ways to influence the setting of defaults and to select a certain numerical algorithm. For consistency, the application sensitive defaults work in a hierarchical order. This is explained in the following sections.

Global and Property Specific Default Definition

The application sensitive defaults can be specified on a global level; i.e., by including a SETTING entry with an application type definition in the input file, but also for specific properties. For example, if your application is `CRASH` but you have some spinning parts in your model, you can define the global defaults by including the entry SETTING,`SID1`,`CRASH` and the specific default setting for the property by SETTING, `SID2`, and `SPINNING`, `SHELL`, `PID2`. This results in a global setting of defaults according to `CRASH`, except for the shell elements that have property number PID2 that will use the defaults necessary for a SPINNING application.

Shell Formulation

The shell formulation can always be globally changed using the entry PARAM,SHELLFORM (formulation type) irrespective of the SETTING entries present in the input file. The ways of shell formulation definition in order of increasing priority is SETTING,PARAM,SHELLFORM, PSHELL1, or PCOMPA.

The thickness of the elements can be made strain independent by including the PARAM,SHTHICK,NO entry in the input file. All application types, except for SHEETMETAL, then use this as the default.

The method for material plasticity can be altered by including the entry PARAM,SHPLAST, (RAD,VECT,ITER) in the input file. All application types then apply this setting as the default except for VERSION2 which always applies the radial return method (RADIAL).

Hourglass Suppression Method

The method to prevent hourglass modes from occurring can also be defined using the HGSUPPR entry in the input file. If there are any HGSUPPR entries in the input file, these always prevail using the hierarchical order within the hourglass definition scheme. The same applies to the hourglass method constants that can also be specifically defined on a global or on a property level.

6 Air Bags and Occupant Safety

Porosity in Air Bags

Porosity is defined as the flow of gas through the air bag surface. There are two ways to model this:

1. Holes: The air bag surface contains a discrete hole.
2. Permeability: The air bag surface is made from material that is not completely sealed.

The same porosity models are available for both the uniform pressure air bag model as the Eulerian coupled air bag model. The porous flow can be either to and from the environment or into and from another uniform pressure model.

The following table shows the available porosity models and their usage:

Entry	Flow Through	Flow To/From
PORHOLE	hole	environment
PORLHOLE	large hole	environment
PERMEAB	permeable area	environment
PERMGBG	hole	another uniform pressure model air bag
PORFLGBG	large hole	another uniform pressure model air bag
PERMGBG	permeable area	another uniform pressure model air bag
PORFCPL	hole	one Eulerian air bag to another one
PORFLCPL	large hole	one Eulerian air bag to another one
PERMCPL		permeable flow between Eulerian air bags

The following entries are required to activate the different porosity models. The id's are arbitrarily chosen, but are unique for each cross-reference. See the individual manual pages for further explanation of the fields. The model incorporates a switch from the Eulerian coupled air bag model to the uniform pressure air bag model at 50 milliseconds.

Flow Through a Hole to the Environment

air-bag surface:	`SURFACE,1,.....`
porous area:	`SUBSURF,10,1,....`
uniform pressure model:	`GBAG,20,1,,,30`
porosity for subsurface:	`GBAGPOR,40,30,10,PORHOLE,50, ,<coeffv>`
Eulerian coupled model:	`COUPLE,60,1,OUTSIDE, ,70`
porosity for subsurface:	`COUPOR,80,70,10,PORHOLE,50, ,<coeffv>`
hole characteristics:	`PORHOLE,50,,,BOTH,<penv>,<rhoenv>,<sieenv>`
Euler to GBAG switch: (at 50.E-3 seconds)	`GBAGCOU,101,60,20,50.E-3,1.E20`

Flow Through Permeable Area to the Environment

air-bag surface:	`SURFACE,1,.....`
porous area:	`SUBSURF,10,1,....`
uniform pressure model:	`GBAG,20,1,,,30`
porosity for entire bag:	`GBAGPOR,40,30,0,PERMEAB,50`
porosity for subsurface:	`GBAGPOR,40,30,10,PERMEAB,50`
Eulerian coupled model:	`COUPLE,60,1,OUTSIDE,,,70`
porosity for entire bag:	`COUPOR,80,70,0,PERMEAB,50`
porosity for subsurface:	`COUPOR,80,70,10,PERMEAB,50`
permeab characteristics:	`PERMEAB,50,1.E-4,: TABLED1,90,` `,BOTH,<penv>,<rhoenv>,<sieenv>`
linear:	`PERMEAB,50, ,90,BOTH,<penv>,<rhoenv>,<sieenv>`
tabular:	`: TABLED1,90,.....`
Euler to GBAG switch: (at 50.E-3 seconds)	`GBAGCOU,101,60,20,50.E-3,1.E20`

Flow Through a Hole to Another Uniform Pressure Air Bag

air bag 1

air-bag surface:	`SURFACE,1,.....`
porous area:	`SUBSURF,10,1,....`
uniform pressure model:	`GBAG,20,1,,,30`
porosity for subsurface:	`GBAGPOR,40,30,10,PORFGBG,50, ,<coeffv>`
Eulerian coupled mode:	`COUPLE,60,1,OUTSIDE, , ,70`
porosity for subsurface:	`COUPOR,80,70,10,PORFGBG,50, ,<coeffv>`
hole characteristics:	`PORFGBG,50,,,BOTH,1020`
Euler to GBAG switch: (at 50.E-3 seconds)	`GBAGCOU,101,60,20,50.E-3,1.E20`

air bag 2

air-bag surface:	`SURFACE,1001,.....`
porous area:	`SUBSURF,1010,1001,....`
uniform pressure model:	`GBAG,1020,1001`

Note:	The porosity characteristics need to be defined for air bag 1 only. The gas automatically flows from bag 1 into bag 2 and vice versa.

Flow Through a Permeable Area to Another Uniform Pressure Air Bag

air bag 1

air-bag surface:	`SURFACE,1,.....`
porous area:	`SUBSURF,10,1,....`
uniform pressure model:	`GBAG,20,1,,,30`
porosity for entire bag:	`GBAGPOR,40,30,0,PERMGBG,50`
porosity for subsurface:	`GBAGPOR,40,30,10,PERMGBG,50`
Eulerian coupled model:	`COUPLE,60,1,OUTSIDE,,,70`
porosity for entire bag:	`COUPOR,80,70,0,PERMGBG,50`
porosity for subsurface:	`COUPOR,80,70,10,PERMGBG,50`
`PERMGB` characteristics:	
linear:	`PERMGBG,50,1.E-4,   ,BOTH,1020`

tabular:	`PERMGBG,50,   ,90,BOTH,1020`
	`: TABLED1,90,.....`

Euler to `GBAG` switch: (at 50.E-3 seconds)	`GBAGCOU,101,60,20,50.E-3,1.E20`

air bag 2

air-bag surface:	`SURFACE,1001,.....`
porous area:	`SUBSURF,1010,1001,....`
uniform pressure model:	`GBAG,1020,1001`

Note:	The porosity characteristics need to be defined for air bag 1 only. The gas automatically flows from bag 1 into bag 2 and vice versa.

Flow Through a Hole from an Eulerian Air Bag to Another One

air bag

air-bag surface	`SURFACE,1,.....`
porous area	`SUBSURF,10,1,....`
Eulerian coupled model	`COUPLE,60,1,OUTSIDE, , ,70`
porosity for subsurface	`COUPOR,80,70,10,PORFCPL,50, ,<coeffv>`
hole characteristics	`PORFCPL,50,,,BOTH,1020`
ignore part of the Euler element	`IGNORE,1,60,22`
list of Euler elements	`SET1,22,...`

air bag 2

air-bag surface	`SURFACE,2,.....`
Eulerian coupled model	`COUPLE,1020,2`
Ignore part of the Euler elements	`IGNORE,2,1060,11`
List of Euler elements	`SET1,11,...`

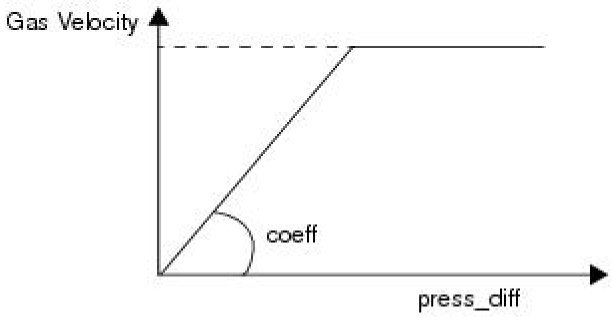
Note: The porosity characteristics need to be defined for air bag 1 only. The gas automatically flows from bag 1 into bag 2 and vice versa.

Permeability

Permeability is defined as the velocity of gas through a surface area depending on the pressure difference over that area.

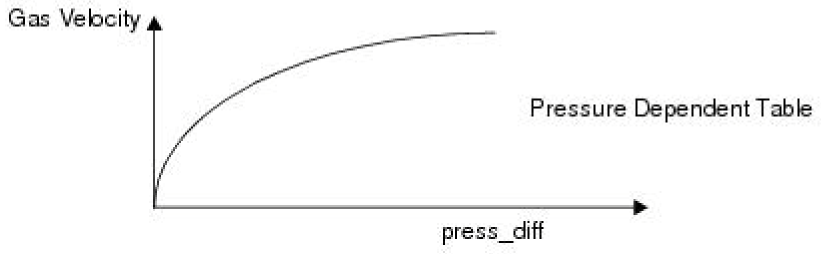
On the PERMEAB, PERMGBG, and PERMCPL entries, permeability can be specified by either a coefficient or a pressure dependent table:

a. Coefficient: Massflow = coeff*pressure_difference



$$coeff = \frac{\delta(massflow)}{\delta(pressdiff)}$$

b. Table:



The velocity of the gas flow can never exceed the sonic speed:

$$V_{max} = V_{sonic} = \sqrt{\gamma R T_{crit}}$$

where γ is the gas constant of in- or outflowing gas, and T_{crit} is the critical temperature.

The critical temperature can be calculated as follows:

$$\frac{T_{crit}}{T_{gas}} = \frac{2}{(\gamma + 1)}$$

where T_{gas} is the temperature of outflowing gas.

Holes

Flow through holes as defined on the PORHOLE, PORFGBG, or PORFCPL entries is based on the theory of one-dimensional gas flow through a small orifice. PORHOLE, PORFGBG, or PORFCPL entries define flow through a hole with the velocity method. The formulas to calculate the velocity of the gas are the same as for the PORFLOW with the pressure method. The formulas are given in Chapter 4 Fluid Structure Interaction, General Coupling in this manual.

The velocity method is only active for Eulerian air bags. When the PORHOLE, PORFGBG or PORFCPL is referenced from a GBAGPOR or COUPOR entry, the theory of one-dimensional gas flow through a small orifice is applied.

Contact Based Porosity

During the early stages of air bag deployment, internal layer contact might prevent air from escaping through holes or permeability. Also later during the inflation process, the air bag fabric will contact the occupant and/or instrument panel, potentially closing of holes or preventing membrane leakage.

In order to take this aspect of the unfolding process into account, contact based porosity in available. The algorithm is activated by the following parameter:

```
PARAM,CONTACT,COPOR,YES
```

When porosity is defined for an air bag by means of COUPOR or GBAGPOR options, a factor for each face of the surface or subsurface will be calculated. Depending on the connectivity, the porosity coefficient for an element will be set to 0/3, 1/3, 2/3 or 3/3 for triangular elements or 0/4, 1/4, 2/4, 3/4 or 4/4 for quad elements.

The total contact based porosity area coefficient for a SURFACE or SUBSURF can be visualized by requesting for the EFFAREA variable in an SURFOUT or SUBSOUT output request. The value of EFFAREA will be between zero and one. Zero representing full closing of porosity because contact and one representing no blocking of air flow through holes or leakage through air bag fabric.

Also the contact factor per node can be requested for each grid point of the air bag surface, by using PORFLG in the GPOUT (grid point output) request.

In a typical air bag simulation the influence of contact based porosity can be significant. Below two graphs of the same simulation: the first one being without and the second one with contact based porosity activated.

Contact based porosity is implemented for both the Uniform Pressure (GBAG) and the CFD (COUPLE) approach. Below an example of a simple air bag analysis that shows the difference between using or not using contact based porosity.

Figure 6-1 shows the area coefficient of the entire surface. In the beginning, the bag is completely folded, so the effective area is almost zero. When the bag starts to deploy, more area is exposed and the coefficient increases to about 0.7. This means that around 70% of the total air bag area is open. Through this area air can flow, however, 30% is still blocked and through this area no mass will be lost. Effectively, more mass remains inside the air bag. This shows by the higher level of pressure compared to the simulation where contact based porosity was not used. See Figure 6-2.

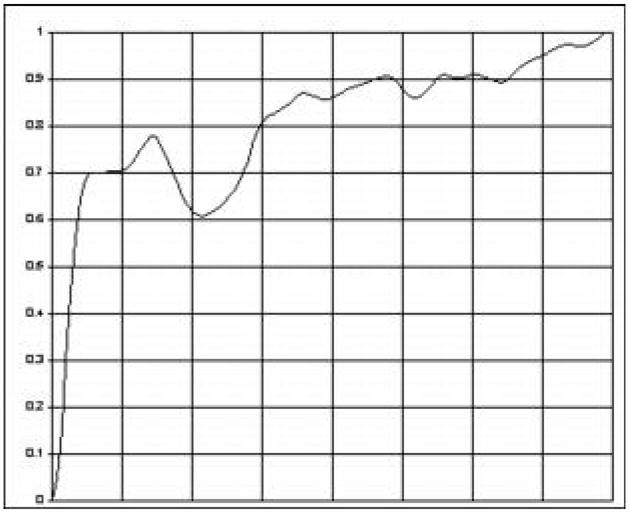


Figure 6-1 Effective Area Coefficient

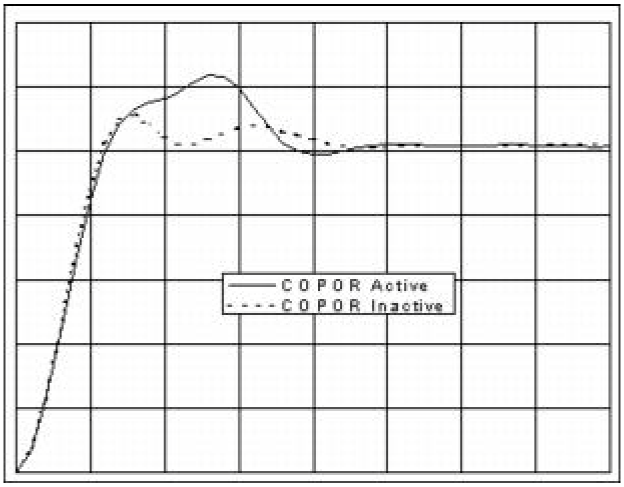


Figure 6-2 Air Bag Pressure

Inflator in Air Bags

There are several methods available to define an inflator in air bag analyses. The most general and extended inflator definitions are:

INFLATR	Standard inflator defined by mass flow rate and dynamic temperature of a single inflowing gas.	
INFLATR1	Standard inflator defined by mass flow rate and static temperature of a single inflowing gas.	
INFLHYB	Hybrid inflator defined by mass flow rate and dynamic temperature of multiple inflowing gasses.	
INFLHYB1	Hybrid inflator defined by mass flow rate and static temperature of multiple inflowing gasses.	
INFLCG	Cold gas inflator defined by initial conditions. Mass flow rate is computed from:	
	1	Total temperature in inflator container = total temperature at exit.
	2	Isentropic relations. Internal variables of the inflator decrease as mass flows out.

For both the uniform pressure model (GBAG) and the Euler coupled model (COUPLE), the inflator location and area are defined by means of a subsurface (SUBSURF), which must be part of the GBAG and/or COUPLE surface. The characteristics of the inflator are specified on an INFLATR entry. This entry references tables for the mass flow rate and the temperature of the inflowing gas.

A model can be defined containing both an Euler coupled model (COUPLE), as a uniform pressure model for the air bag (GBAG). It is possible to define these two options with identical inflator characteristics. This allows use of the GBAGCOU entry to switch from the Euler coupled model to the Uniform pressure model during the calculation. When the same air bag surface is referenced from both a COUPLE and a GBAG entry, the GBAGCOU switch must be present in the input file.

An inflator in an air bag analysis specified using the following input:

air bag surface:	`SURFACE,1,.....`
inflator area:	`SUBSURF,10,1,....`
uniform pressure model:	`GBAG,20,1,,,,40,....`
inflator for subsurface:	`GBAGINFL,50,40,10,INFLATR,50, ,<coeffv>`
Eulerian coupled model:	`COUPLE,60,1,OUTSIDE,,,,,,+`
	`+070`
inflator for subsurface:	`COUINFL,90,70,10,INFLATR,50,,<coeffv>`
inflator characteristics:	`INFLATR,50,130,,912.,1.4,286.`
Euler to GBAG switch: (at 50.E-3 secs)	`GBAGCOU,101,60,20,50.E-3,1.E20`

Note that it is possible to define multiple inflators per air bag module, by defining a set of COUINFL and/or GBAGINFL entries with the same value in the third field. This is the set ID. Each inflator can reference its own tables for mass flow rate and temperature.

Initial Metric Method for Air Bags

The Initial Metric Method is typically useful for air bag modeling. When using out-of-plane folding technique, the membrane elements can deform quite significantly. The final shape of the deformed bag can be negatively influenced. In order to overcome this problem, Dytran offers a way to initialize strains inside elements such that the final shape is preserved. It is called the Initial Metric Method, further called the IMM method. Elements can be initialized smaller than the original state, but also can be initialized larger.

- For elements that are initialized smaller, stresses only start to build up after the original state has been reached.
- Elements that are larger have a positive IMM strain. When growing larger, their Young's modulus is assumed to be twice as large during one time step. When shrinking, no stresses are applied until the original state is reached.

IMM can also be applied when scaling the model of the air bag such that the model fits inside the inflator housing.

IMM Methods

Three formulations are available in Dytran. Using the parameter IMM, they are:

FULL:	While elements are under IMM condition, they carry stresses when under compression. This is the default.
REDUCED:	While elements are under IMM condition, the carry a reduced stress when under compression. The relative area factor, SMDFER, is used to reduce Young's modulus.
ZERO:	While elements are under IMM condition, compressive stresses are not carried. They code relies on the material damping to avoid excessive nodal velocities. It has been shown that IMM formulation ZERO is most suitable when more that a couple of membrane elements with zero area exist in the initial state of the air bag model.

It should be noted that when elements are not under the IMM condition anymore (IMM strain tensor components are all zero), the elements start to behave like a regular membrane elements according to the material model attached.

IMM Recalculation

The IMM strains can be recalculated during the simulation. This is especially beneficial for models that have initially many elements with zero area. This recalculation results in more stable behavior of these elements and also results in an improved shape near the end of the calculation.

Usage

The IMM needs two models of the same air bag. One model is called Initial state and the second is called Original or Reference state.

- The Initial state model has to be part of the main input file. This state can be visualized in output requests.
- The Original state model has to be supplied to Dytran in a different file. Dytran reads this file and uses the data to initialize IMM strains on the elements of the Initial state model.

The following line activates the Initial Metric Method in Dytran (see Executing Dytran):

```
dytran jid=<dytran_input_file> imm=<original_input_file>
```

The Initial Metric Method is also activated when the IMMFILE = (filename) directive is present in the File Management section.

The Initial Metric Method can be used in combination with ATB and can be used in any contact type. IMM is only used for triangular membrane elements. The IMM-strains can be visualized in an archive file or time-history file. The variables are EXXIMM, EYYIMM and EXYIMM. A value of zero denotes that during the run, the original state was reached and IMM for that particular element is not active anymore.

Heat Transfer in Air Bags

For air bags with high temperature, energy is exchanged with the environment. There are two ways to define heat transfer in air bags, convection (HTRCONV) and radiation (HTRRAD).

The heat-transfer rates due to convection and radiation are defined by:

1. Convection:

   $$q_{conv} = h(t)A(T - T_{env})$$

   where $h(t)$ is the time-dependent heat-transfer coefficient, A is the (sub)surface area for heat transfer, T is the temperature inside the air bag, and T_{env} is the environment temperature.

2. Radiation:

   $$q_{rad} = eAs[T^A - T^A_{env}]$$

   where e is the gas emissivity, A the (sub)surface area for heat transfer, T is the temperature inside the air bag, and T_{env} the environment temperature.

Both types can be defined independently for the whole air bag surface, or for parts of the surface by means of SUBSURFs.

Example

air bag surface:	SURFACE,1,.....
subsurface with heat transfer:	SUBSURF,2,1,....
subsurface with heat transfer:	SUBSURF,10,1,....
uniform pressure model:	GBAG,20,1,,,,,30
convection for whole surface:	GBAGHTR,40,30, ,HTRCONV,50,,<coeffv>
radiation for whole surface:	GBAGHTR,41,30, ,HTRRAD,50,,<>
convection for subsurface 10:	GBAGHTR,42,30,10,HTRCONV,51,,<coeffv>
radiation for subsurface 10:	GBAGHTR,43,30,10,HTRRAD,51,,<coeffv>
radiation for subsurface 2:	GBAGHTR,44,30, 2,HTRRAD,52,,<coeffv>
Eulerian coupled model:	COUPLE,60,1,OUTSIDE,,,,,,+
	+,70
convection for whole surface:	COUHTR,80,70, ,HTRCONV,50,,<coeffv>
radiation for whole surface:	COUHTR,81,70, ,HTRRAD,50,,<coeffv>
convection for subsurface 10:	COUHTR,82,70,10,HTRCONV,51,,<coeffv>
radiation for subsurface 10:	COUHTR,83,70,10,HTRRAD,51,,<coeffv>
radiation for subsurface 2:	COUHTR,84,70, 2,HTRRAD,52,,<coeffv>
convection characteristics:	HTRCONV,50, 7.,,297.
convection characteristics:	HTRCONV,51,52.,,297.
radiation characteristics:	HTRRAD,50,.15,,297.,5.676-8

radiation characteristics:	`HTRRAD,51,.6,,297.,5.676-8`
radiation characteristics:	`HTRRAD,52,.4,,297.,5.676-8`
Euler to `GBAG` switch: (at 50.E-3 seconds)	`GBAGCOU,101,60,20,50.E-3,5.`

Seat Belts

A seat belt constraint system can be modeled within Dytran using a special belt element. The element has the following characteristics:

- Tension-only nonlinear spring with mass.
- User-defined loading and unloading path.
- Damping is included to prevent high-frequency oscillations.
- Possible to prestress and/or feed additional slack.

A special contact algorithm is available to model the contact between the belt elements and an occupant model.

Seat Belt Material Characteristics

You can specify the following material characteristics on a PBELT entry:

Loading and Unloading Curves

The loading/unloading curves are defined in a TABLED1 entry specifying the force as a function of strain. The strain is defined as engineering strain

$$\varepsilon^n = \frac{l^n - l^o}{l^o}$$

where l^n is the length at time n and l^o is the length at time zero.

The loading and unloading curves must start at (0, 0).

Upon unloading, the unloading curve is shifted along the strain axis until it intersects the loading curve at the point from which unloading commences. Figure 6-3 shows an example of a typical load, unload, and reload sequence.

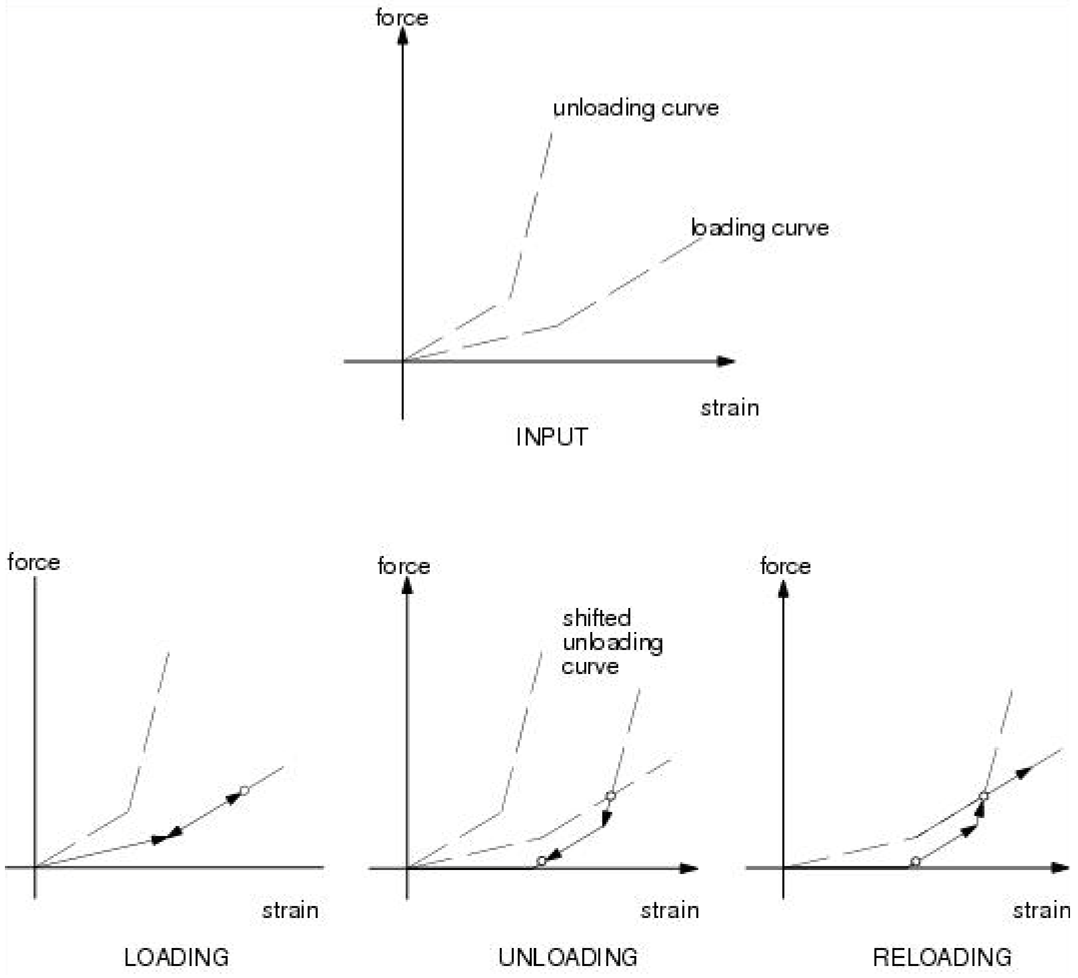


Figure 6-3 Seat Belt Loading and Unloading Characteristics.

The unloading table is applied for unloading and reloading until the strain again exceeds the point of intersection. At further loading, the loading table is applied.

Seat Belt Element Density

The density of the belt elements is entered as mass per unit length. The density is used during initialization to distribute the mass to the grid points. The grid points masses are used to calculate damping and contact forces.

Damping Forces

A damping force is added to the internal force to damp high-frequency oscillations. The damping force $\vec{F}_D$ is equal to

$$\vec{F}_D = \alpha_1 M \cdot \frac{\vec{V}_{G1} - \vec{V}_{G2}}{\Delta t}$$

where α_1 is the damping factor CDAMP1 as defined on the PBELT entry, $\backslash M$ is the element mass, $\vec{V}_{G1}$ and $\vec{V}_{G2}$ denote the velocity of grid point 1 and grid point 2 of the element respectively. Δt is the time step.

The damping force $\vec{F}_D$ is limited to

$$\vec{F}_D = max(\vec{F}_D, \alpha_2 \vec{F}_S)$$

where α_2 is the damping coefficient CDAMP2 as defined on the PBELT entry, and $\vec{F}_S$ is the internal force in the element.

Slack

Additional slack can be fed into the belt elements as a function of time. The slack is specified in engineering strain and will be subtracted from the element strain at time n as

$$\varepsilon^n = \varepsilon^n - e^n_{slack}$$

where ε^n_{slack} denotes the slack strain as found from the TABLED1 definition in the input file.

The force in the element is zero until the element strain exceeds the slack.

Prestress

The seat belt elements can be prestressed as a function of time. The prestress strain is specified in engineering strain and is added to the element strain at time n as

$$\varepsilon^n = \varepsilon^n + \varepsilon^n_{prestress}$$

where $\varepsilon^n_{prestress}$ is the prestress strain as found from the TABLED1 definition in the input file.

As a result, the elements build up a tensile force.

7 Interface to Other Applications

ATB Occupant Modeling Program

In order to perform analyses of occupant interaction with structures, the occupant modeling program ATB (Version 5.3.1) is included in Dytran (References 5., 8., 16., and 17.). This is not just a procedure for running one program, transferring results, and then running the other program. Dytran and ATB run concurrently, exchanging data as the analysis proceeds. ATB is built as standard in Dytran and is activated only when required.

ATB Input Specification

The input specification for ATB (Reference 16.) does not change. The ATB-input file name (with standard extension `.ain` or extension `.lin`) must be specified by the user. This is done on the command line of the Dytran script by using one of the keywords (`atb =`, `ain =` or `lin =`) followed by the ATB-input filename (without an extension). Both methods command Dytran to activate ATB.

The preprocessor GEBOD generates ATB input of various known dummy models like Hybrid II and Hybrid III. This interactive program also enables you to create dummy models with user-supplied dimensions. GEBOD Version V.1 is delivered with the Dytran installation. It is executed by typing `gebod` on the command line in a shell window.

Some useful options have been added to the ATB input file:

- If a nonzero value is set in the `NSJF` field (I4) right after the I3 field on entry G.1.a, initial joint forces are subtracted from internal forces. Thus, it is possible to create an initial equilibrium after positioning the dummy. Care should be taken when using this option.
- Dytran can control the timesteps of output generated by ATB. This achieved by changing flags on the ATB output generation card (Card A5) in the ATB input file:
    - **Field 3:** Frequency of output written to main ATB print-file. If the value is set to -1, the output is controlled by the Dytran parameter `PARAM,ATBAOUT,xxx`. The default for this parameter is 10 milliseconds.
    - **Field 26:** Frequency of output written to time-history files (H-cards and LU-numbers). If the value is set to -1, the output is controlled by the Dytran parameter `PARAM,ATBTOUT,xxx.` The default for this parameter is 1 millisecond.
- If extra output files of ATB are requested, the starting LU number of the ASCII time history files has been increased from 21 (which is the ATB standard) to 41 in order to avoid mixing with output files of Dytran.

Despite above-mentioned additions, any original ATB-input file is still valid.

Moreover, positioning of the dummy can be handled by Dytran. For a description, see the Bulk Data entries ATBJNT, ATBSEG, and the PARAM entry ATBSEGCREATE.

Additional copies of References 5-8, 16 and 17, where Reference 6 serves as *ATB User Manual*, may be purchased from:

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
UNITED STATES OF AMERICA

ATB Postprocessing

ATB results can be postprocessed with the standard ATB postprocessors. A specialized pre and post processing GUI can be obtained with Veridian at the following internet address:

http://www.veridian.com/products/atb.asp

Dytran Input Specification

Each segment contact ellipsoid used in the occupant model must be defined in the Dytran input file with the RELEX entry. The names of these ellipsoids must be the same as the segment names as defined in the ATB-input file on the B.2 entry. The use of the RELEX entry is described in the Bulk Data section.

Only the RELEX entry should be used. Internal ellipsoids defined by the RELLIPS entry cannot be used in combination with those defined by RELEX.

The interaction between the ATB ellipsoids and the structural parts of the Dytran model is achieved through the standard contact and connection entries CONTREL and RCONREL. This interaction applies to all structural elements: quadrilateral and triangular shells, membrane elements, Lagrangian solids, and rigid bodies.

Termination Conditions

All termination conditions selected in Dytran remain valid. If a termination condition occurs, ATB is forced into an end cycle, and Dytran terminates as usual.

ATB-termination conditions cannot be used to stop the analysis.

Dytran Pre- and Postprocessing

The dummy of ATB can be positioned with Patran. Use the session files and the Dytran Hybrid FEM file that is provided in the Demonstration directory. For a more detailed description, see the Bulk Data entries ATBJNT, ATBSEG, and the PARAM entry ATBSEGCREATE.

The data relating to the rigid ellipsoids can also be output from Dytran using the standard Case Control command RELOUT.

If the ellipsoid geometry is requested in an archive file, the ellipsoids and planes are covered with dummy shell elements at each requested output step. These dummy elements are written to the archive file. In this way the geometry, position, and orientation of the ellipsoids can be visualized within the deformed Dytran mesh.

8 Special Modeling Techniques

Prestress Analysis

In some cases, a prestressed initial state is needed for a structure in order to get the correct results in a transient follow-up problem; e.g., a bird striking a rotating turbine fan blade.

An efficient way to solve the static prestress problem is by using Nastran. From this solution, it is possible to initialize Dytran such that the correct initial, prestressed state is achieved for a transient dynamic analysis.

There are two ways of initializing a prestressed state:

1. Direct Nastran initialization.

   Both the displacement and the stress field are read from the Nastran output data and transferred directly into Dytran as an initial state for the structural elements. It is defined by including the NASTINP FMS command referring to the Nastran solution file.

2. Nastran initialization via an intermediate Dytran prestress analysis.

   The Nastran computed displacement field is used to obtain a stable prestressed state in Dytran (the prestress analysis). This Dytran solution is applied in the subsequent transient analysis and acts as a stable initial state for the structure.

   Entries involved in the prestress analysis are:

   - PRESTRESS
   - SOLUOUT
   - BULKOUT
   - NASINIT
   - NASTOUT

   The entry required to effect the initialization of the transient analysis:

   - SOLINIT

   The parameters involved are:

   - INITFILE
   - INITNAS

An Example Nastran Input Data

The following is an example of Nastran input data that can be used to obtain a solution that allows for geometric and material nonlinearities.

```
$
$
$ Example Input Deck for an Initialization for Centrifugal Loading         $
$                                                                          $
$ The Output will be an XL-database                                        $
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ Executive control statements
INIT DBALL LOGICAL=(DBALL(80000))
TIME 600
SOL 66
COMPILE SOL66 SOUIN=MSCSOU NOLIST NOREF $
```

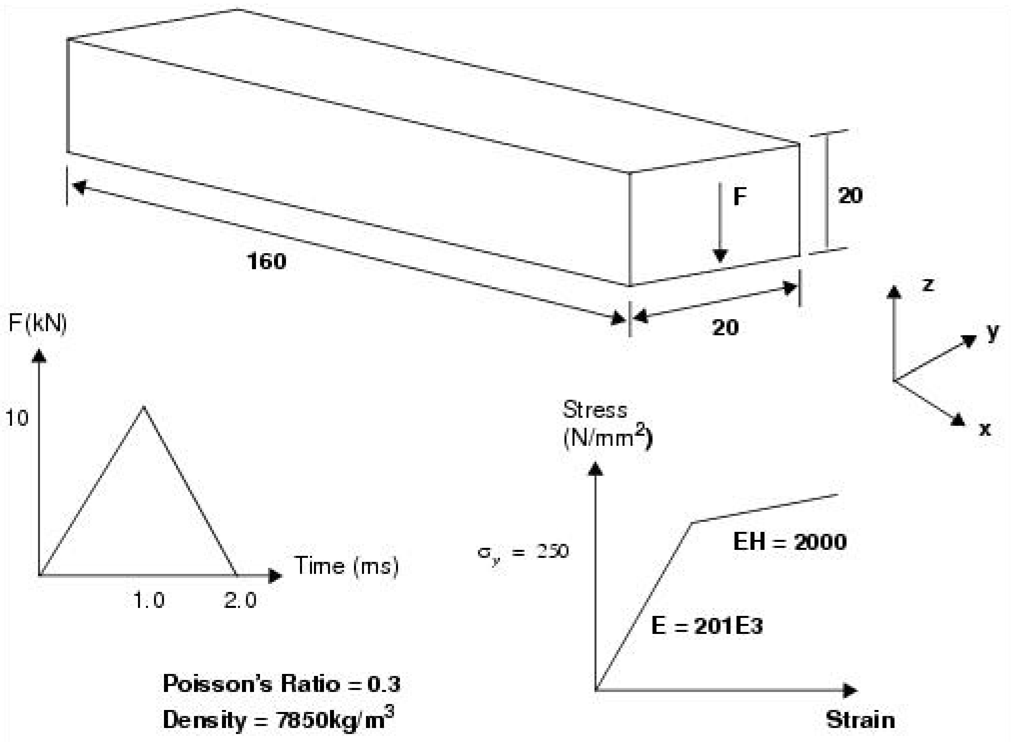
```
CEND
TITLE=       Example NASTRAN Input Deck
SUBTITLE=       PRE-STRESS CALCULATION
  SPC=1
  ECHO = NONE
  MAXLINES = 2000000
  SEALL=ALL
  DISP(PRINT,PLOT)=ALL
  STRESS(PRINT,PLOT)=ALL
  NLPARM = 10
SUBCASE 1
LABEL=CENTRIFUGAL LOADING OF 10000 RPM
  LOAD=1
BEGIN BULK
$
$ Include Geometry and Constraints From Files
INCLUDE SOLID.DAT
INCLUDE SPC1.DAT
$
$ Material definition
PSOLID,1,1,0
MAT1,1,1.156E11,,0.318,4527.
$
$ Gridpoint for the rforce loading...
GRID,100000,0,-.1,0.0,0.0,0,0
$
$ DATA DECK
PARAM,POST,0
$
$ Not Lumped But Coupled Masses
PARAM,COUPMASS,1
$
$ Define Large Displacements
PARAM,LGDISP,1
$
PARAM,DBDICT,2
PARAM,OUNIT,12
$
$ The extra point on the axis of rotation is fixed
SPC1,1,123546,100000
$
$ Define the Centrifugal Loading
RFORCE,1,100000,0,85.,0.0,0.0,1.0,2
$
$ Definition of the Solution Sequence and Convergence Criteria
NLPARM,10,10,,,,,UPW
$
ENDDATA
```

Cantilever Beam Example Input Data

This example shows the input file for a simple cantilever beam modeled with shell elements and subjected to a dynamic tip load.

The Problem

The diagram below summarizes the problem. (All dimensions are in millimeters)



The Model

The following mesh is used to model the beam. This mesh is purely for demonstrating how to create a Dytran input file; it is not intended to be indicative of a good modeling practice. The mesh was deliberately kept as simple as possible to limit the size of the input file. If this were a real problem that you were solving, then a considerably finer mesh would be required.

The grid-point and element numbers are shown on the plot in the illustration above.

Input File

The input file for this problem is listed below. The input is in free format, and the file has extensive comments to explain the operation of the various File Management Systems, Case Control commands, and the Bulk Data entries.

```
$ File Management Section
$ -----------------------
$
$ Define the type of output files to be used. We need one Archive file (logical
$ name ARC), one Time History file (logical name HIST) and one Restart file
$ (logical name RST).
$
TYPE(ARC) = ARCHIVE
TYPE(HIST) = TIMEHIS
TYPE(RST) = RESTART
```

```
$
$ CEND marks the end of the FMS and Executive Control Section
$
CEND
$
$ Case Control Section
$ --------------------
$
$ Set the termination time to 2.0ms
$
ENDTIME = 2.0E-3
$
$ Set CHECK to NO to switch of the data check and run the analysis
$
CHECK = NO
$
$ The Archive file will contain the effective plastic strain and the
$ effective stress for all elements at 0.2 and 2. msecs
$
TIMES(ARC) = 0.2E-3, END
SAVE(ARC) = 100000
ELEMENTS(ARC) = 10
SET 10 = 1,THROUGH,7
ELOUT(ARC) = EFFST01,EFFST03,EFFPL01,EFFPL03
$
$ The Time History file will contain the coordinates and velocities of grid
$ points 8 and 18. The results are stored at intervals of .1 msec.
$
TIMES(HIST) = 0 THROUGH END BY .1E-3
SAVE(HIST) = 100000

GRIDS(HIST) = 20
SET 20 = 8,18
GPOUT(HIST) = COORD,VEL
$
$ Write restart data at step 500
$
STEPS(RST) = 500
$
$ Select the constraints and loading to be used from the Bulk Data
$ section
$
SPC = 10
TLOAD = 11
$
$ Bulk Data Section
$ -----------------
$
$ The BEGIN BULK entry marks the end of Case Control and the start of
$ Bulk Data
$
BEGIN BULK
$
$ Define the grid points
$
GRID           1              0.      0.      0.
GRID           2            0.02      0.      0.
GRID           3            0.04      0.      0.
GRID           4            0.08      0.      0.
GRID           5            0.10      0.      0.
GRID           6            0.12      0.      0.
GRID           7            0.14      0.      0.
GRID           8            0.16      0.      0.
GRID          11              0.    0.02      0.
GRID          12            0.02    0.02      0.
GRID          13            0.04    0.02      0.
GRID          14            0.08    0.02      0.
GRID          15            0.10    0.02      0.
GRID          16            0.12    0.02      0.
GRID          17            0.14    0.02      0.
GRID          18            0.16    0.02      0.
$
$ Define the elements
$
CQUAD4         1     100       1       2      12      11
CQUAD4         2     100       2       3      13      12
CQUAD4         3     100       3       4      14      13
CQUAD4         4     100       4       5      15      14
CQUAD4         5     100       5       6      16      15
CQUAD4         6     100       6       7      17      16
CQUAD4         7     100       7       8      18      17
$
$ The properties of the elements are defined using a PSHELL entry.
$ This entry defines the thickness of the elements (20mm) and a material
$ identification number.
$
PSHELL       100     100    0.02
$
$ The bilinear elastic-plastic material properties are defined using a DMATEP
$ entry. The hourglass and bulk viscosity data is set to the default values
$ so the continuation line is omitted. The DMATEP defines the density and the
$ elastic properties of the material. The yield stress and hardening modulus
$ are defined on a YLDVM entry.
$
DMATEP, 100, 7850.0, 210.E9, 0.3, , , 100
YLDVM, 100, 250E6, 2000.E6
$
$ Use the SPC entry to constrain the grid points where the beam is built-in.
$ All the degrees of freedom are constrained for grid points 1 and 11.
$
SPC, 10, 1, 123456
SPC, 10, 11, 123456
$
$ The transient load is defined using three entries: TLOAD1, TABLED1 and FORCE.
$ The TLOAD1 entry gives the identification numbers of the other entries and
$ indicates that a force is being applied rather than enforced motion.
$
TLOAD1, 11, 11, , 0, 11
$
$ The TABLED1 entry gives the variation of load with time. Note that a
$ horizontal portion at zero load is added to the end of the curve. If the
$ analysis is continued past 2.0ms, this ensures that zero load will be applied,
$ since the table will be extrapolated from the last two points. The entry has
$ two continuation lines to define all the points and it is terminated with an
$ ENDT
$
TABLED1, 11, , , , , , , ,+
+, 0.0, 0.0, 1.0E-3, 10.0E3, 2.0E-3, 0.0, 3.0E-3, 0.0,+
+, ENDT
$
$ The position of the forces is specified using the FORCE entry. Since the load
$ is split between grid points 8 and 18, a scale factor of 0.5 is used. The load
$ is applied in the negative z direction so a vector of 0.0, 0.0, -1.0 is used.
$
FORCE, 11, 8, , 0.5, , , -1.0
FORCE, 11, 18, , 0.5, , , -1.0
$
$ The initial time step is set to 1 microsecond using the parameter   INISTEP.
$
PARAM, INISTEP, 1.E-6
$
$ The input must end with an ENDDATA entry.
$
ENDDATA
```

Mass Scaling

The explicit dynamics procedure of Dytran uses relatively small time steps dictated by the shortest natural period of the mesh: the analysis cost is in direct proportion to the size of the mesh. There are two types of problems where the cost-effectiveness of the analysis can be increased:

- If a mesh consists of a few, very small (or stiff) elements, the smallest (or stiffest) element determines the time step for all elements of the mesh.
- If a few severely distorted elements are obtained by the analysis, the most distorted element determines the time step for all elements of the mesh. This may even end up with a too small stable time step.

Speedup of those problems can be achieved by using mass scaling (PARAM,SCALEMAS). Mass scaling is based on adding numerical mass to an element so that its time step never becomes less than the minimum allowable time step defined by you. Note that mass scaling can be risky in areas where either inertia effects are relevant or contact with other parts is expected to occur.

Problems Involving a Few Small Elements

It is common practice that meshing of real-life problems may involve some relatively small elements: elements frequently localized in a kind of transition region and meant to connect large structural parts to each other. Those elements determine the time step of the whole calculation although they might be present in the model to a very limited extent. Speedup can be realized by using mass scaling. Some guidelines:

- Make a run for one cycle and retrieve the time step of all elements by requesting ELDLTH.
- By using a postprocessing program, see which elements are determining the time step and filter out the elements whose time steps exceed a user-defined minimum (DTMIN).
- See what the impact would be of specifying this new time step (DTMIN). Select the value of DTMIN such that hardly any elements would be scaled in the area of interest (for example, as much as possible outside the impact region in a crash simulation).

Problems Involving a Few Severely Distorted Elements

There are conceivable application areas where elements are distorted to such a high extent that a few of them determine the time step for all elements of the mesh. For example, crushing of a subfloor structure frequently involves failure modes associated with the occurrence of severely distorted elements. Modeling this kind of crushing behavior without including a failure mechanism might end up with a stable time step that is too small. Since those elements are often present in a relatively small region, the mass scaling method might be a good means to artificially speed up the calculation without losing the capability to model the global crushing behavior. Note that to prevent severely distorted elements, it is recommended that a proper failure mechanism be included, instead of coping with the distorted elements by making use of the mass scaling method.

Some guidelines:

- Since you do not know in advance which elements will become too distorted, you should first run the analysis as far as possible (without defining PARAM,SCALEMAS). You should request the time step of all elements (ELDLTH).
- If the problem ends up with a too small stable time step, the analysis finishes prematurely. See which elements are so severely distorted and decide what a reasonable minimum time step (DTMIN) might be without affecting elements in the area of interest. See the guidelines of the previous section.
- Rerun the analysis specifying PARAM,SCALEMAS if the region of highly distorted elements is relatively small compared to the whole model.
- If there is too much mass added to the grid points of those elements, the model might show significantly different inertia effects, and subsequently, different global structural response. In order to avoid this, no more mass is added if the numerically added mass exceeds a certain percentage (MXPERC).
- To limit the amount of overhead time spent on checking against its mass scaling criterion, the checking is done every specified number of STEPS.

Selective Mass Scaling (SMS)

The existing conventional mass scaling increases the mass in elements with a smaller time step than the user target time step. The results are reasonable in cases where dynamic effects are small or the mass increment is small. However, a large mass increment may cause a large difference in the results of the explicit simulation. This is a major limitation in conventional mass scaling.

A good mass increment scheme should preserve the lower frequency modes while reducing the higher frequency modes. Selective mass scaling is a technique which can accomplish this effectively. Generally, it is useful in simulations such as sheet metal forming, for instance stretch forming and punching.

A selectively scaled mass matrix M_s can be added to the diagonal mass matrix M_d to form the mass matrix M. The exposition below describes the general theory for any element type, with a specialization for shell and membrane elements. Mathematically, the lower frequency modes can be preserved by calculating:

$$M_s = \sum_{i-1}^{N} \frac{\beta m_i}{n} \left(I - \sum_{j-1}^{3} \bar{e}_j \bar{e}_j^T \right)$$

Here, N is the number of elements, m_i is the mass of element i, and β is the selective mass scale factor. It can be seen that without the summation term, the enhancement reduces to conventional mass scaling. e_j is a unit rigid body motion in the j-translational direction defined over n nodes in a typical element.

For instance, for a 4-noded element and for the y-direction, the vector can be written as

$$\overline{e_2} = \frac{e_2}{\sqrt{e_2^T e_2}}$$

in which

$$e_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

This can be written more concisely as

$$M_s = \sum_{i=1}^{N} \frac{\beta m_i}{n} P_i$$

where

$$P_i = I - \sum_{j-1}^{3} \overline{e_j}\,\overline{e_j}^T$$

The enhancements are uncoupled in the x-, y- and z-directions. For a typical n-noded element the selectively scaled contribution can be written as:

$$M_s = \beta \cdot \frac{m}{n} \begin{bmatrix} K & 0 & 0 \\ 0 & K & 0 \\ 0 & 0 & K \end{bmatrix}$$

in which m is the element mass.

For a 3-noded triangular element, the contribution is

$$K = \frac{1}{3}\begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

while for a 4-noded quadrilateral it is

$$K = \frac{1}{4}\begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix}$$

Selective mass scaling is controlled by the same input parameter, PARAM, SCALEMAS, as conventional mass scaling. There are three new fields to control selective mass scaling.

For example:

```
PARAM,SCALEMAS,1.0e-6,,,1,10001,4
...
SET1,10001,105,1005
```

1.0e-6 is the target time step, 1 activates selective mass scaling, 10001 indicates the SET1 id which must exist in the BULK DATA, and 4 is the number of threads to be used for the selective mass scaling calculation.

Usage limitations are outlined in the PARAM, SCALEMAS definition in the *Dytran Reference Manual*. The theoretical background is explained in Reference 1. The capability is only available in the serial run and dmp capability will be updated in the future release.

Drawbead Model

The success of the deep-drawing process strongly depends on the extent of slip of the blank at the interface, controlled by the blank holder force and the friction conditions at the interface between the blank, the blank holder, and die. Besides a blank holder, drawbeads are commonly used in sheet metal forming processes to provide an additional local control of plastic sheet deformation and, thereby, the amount of sheet material moving into the die cavity.

The modeling of drawbeads by a finite element mesh is often not feasible. Apart from the drawbead geometry, the region of the blank, which slides through the drawbead itself, would require a very fine FE mesh. This increases the total number of elements and decreases the time step of the calculation significantly.

The drawbead option in the contact is an efficient way to locate the grid points in the blank that are moving across the drawbead line, and therefore, need to be restrained by a drawbead force.

You can input a list of grid points to define the position of the drawbead. The list of grid points must be ordered along the drawbead line and is used to define a row of dummy rod elements. The dummy rod elements representing the drawbead must be connected to the tool from which the drawbead restraining force are applied on the blank. The connection between the drawbead grid points and the tool is achieved by using the rigid connection (RCONN entry).

The restraining force per unit of drawbead length must be supplied by the user and entered by means of the drawbead option in the contact entry. Dytran calculates the drawbead length associated with each drawbead grid point. At every time step, the appropriate drawbead force is applied as a localized restraining force on the blank.

Example of Modeling Procedure

CRODs to locate the drawbead line. The stiffness is chosen such that the rods do not determine the time step. The mass is chosen such that the rods do not add significant mass to the blank holder.

```
CROD, 501, 1, 5001, 5002
SET1, 51, 5001, 5002
PROD, 1, 5, 1.E-20
MAT1, 5, 1.E-20, , 0.3, 1.E-10
```

Define a rigid connection (RCONN) between the drawbead grid points (`GRIDset = 51`) and the tool (`SURFace ID = 11`). Note that gaps between the `GRIDset` and the tool are automatically closed (`CLSGAP = YES`).

```
RCONN, 1, GRID, SURF, 51, 11, , , ,+
+, , , , , , , , ,+
+, YES
```

Define the drawbead restraining force per unit length on the CONTACT entry. The force is applied via GRIDset = 51 on the blank (`SURFace ID = 1`). Note that the contact thickness is taken into account.

```
CONTACT, 1, GRID, SURF, 51, 1, , , ,+
+, DRAWBEAD, , , , 1.0, , , ,+
+, , , , , , , , ,+
+, , , , , , , , ,+
+, <force/length>
```

New Drawbead Model

Since original drawbead requires other combinations such as RCONN and it only has restraining forces, new drawbead option is added from Dytran 2021 release.

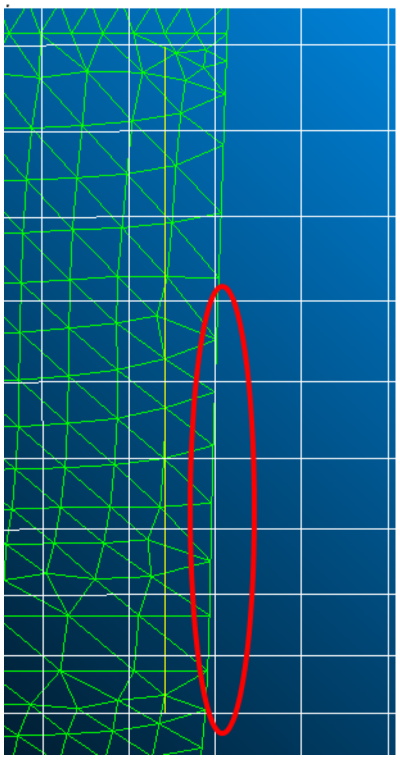
Instead of RCONN, new drawbead entry, DRAWBD is used for selecting drawbead `ROD` elements and rigid body surfaces which must be made by `MATRIG`. Using two definition in DRAWBD entry, the grid points in `ROD` elements are added into `MATRIG` and the drawbead orientation and length are calculated based on ROD elements. DRAWBD id is used in secondary contact body of CONTACT directly with V4 drawbead contact method, `DRAWBDV4`.

The drawbead depth is additionally added in DRAWBD entry. It can apply drawbead forces correctly although the gap between a work piece and tools are slightly larger than contact tolerance due to the oscillation during contact. Uplift forces of drawbead is also added in DRAWBD entry and a user can add uplift forces of drawbead during sheet metal forming simulation.

Currently, new drawbead model is only acceptable with `MATRIG` definition and `DRAWBDV4` contact.

It is recommended that the element size of a blank is less than distance between the edge of a blank holder and drawbead location. However sometimes drawbead is located to close to the edge of a blank holder and the element size of a blank becomes too small. In this case, Binding Contact option is useful when some grid points are not fully contacted by a blank holder and a die.

As the figure below (yellow - drawbead, white - blank and green - blank holder), the grid points in red circle are affected by drawbead contact but not fully contacted by a blank holder. The uplift forces on these grid points may generate strange deformation. Binding Contact option is used not to apply uplift forces on these grid points during the simulation.



Example of Modeling Procedure

```
CROD, 501, 1, 5001, 5002
SET1, 51, 501
PROD, 1, 5, 1.E-20
MAT1, 5, 1.E-20, , 0.3, 1.E-10
```

Attach the drawbead grid points in CROD elements (Elem set = 51) to the tool (`MATRIG ID = 11`). Note that gaps between the grid points in CROD elements and the tool are automatically closed without any option. Restraining and uplift forces of drawbead are defined in DRAWBD entry too.

```
DRAWBD,1,51,MR11,1.0,16.666+3,20.0+2
```

CONTACT between a drawbead (`DRAWBD id=1`) and a workpiece (Elem set =14) is applied with `DRAWBDV4` method.

```
CONTACT,9,DRAWBD,ELEM,1,14,,,,+
+       DRAWBDV4
```

Reference

1. Olovsson L., Simonsson K., Iterative solution technique in selective mass scaling, Communications in Numerical Methods in Engineering, 22, 77-82, 2006.

9 Running the Analysis

Analysis Sequence

The steps involved in running a successful analysis with Dytran are essentially the same as those involved in running a normal static analysis with any other finite element analysis code. The main difference is that a static analysis is usually run in one operation, whereas a Dytran analysis is often run in a number of stages.

The main steps involved are as follows:

1. Modeling
2. Data translation
3. Data check
4. Analysis
5. Results translation
6. Results postprocessing

Steps 4 through 6 are repeated for each stage of the analysis.

Using a Modeling Program with Dytran

You can produce a Dytran model in exactly the same way as you would produce any other finite element model, using Patran or another modeling package. There are a number of things to remember when modeling, and these are discussed in this section. Once your model is complete, you should use the translator supplied with your modeling package to write an Nastran input file. Since much of the input for Dytran is the same as Nastran, the Nastran style of input can be used with only minor modifications for Dytran. Virtually all modeling packages can write Nastran input files; if your modeling package does not do this, you must write your own translator to translate the data into a form that Dytran can understand.

The vast majority of the data for a Dytran analysis can usually be created by the modeling package including the following entries:

Grid points	GRID
1-D elements	CBAR, CBEAM, CROD
Solid elements	CHEXA, CPENTA , CTETRA
Shell and membrane elements	CQUAD4, CTRIA3
Properties	PSOLID, PSHELL
Single-point constraints	SPC, SPC1
Concentrated loads	FORCE, MOMENT
Pressure loads	PLOAD, PLOAD4

Since most modeling packages are usually set up to prepare data for linear elastic analyses, a complete Dytran input file cannot be created. You need to edit the file to add features, such as rigid walls or nonlinear material properties, that cannot be created by the modeling program. Such additions are usually quite minor.

When you are creating your Dytran model, there are some points you should remember. These points are discussed in the following section.

Grid Points

Dytran accepts any system of grid point numbering, so there is no need to renumber your model. Try to adopt a numbering scheme where a grid point's number is indicative of its position. Since Dytran does not have a bandwidth requirement, do not perform any bandwidth optimization on your model.

Elements

Remember that Dytran only has linear, solid, shell, and membrane elements, so do not create any elements with midside nodes. The elements must be given the correct property and material numbers. Choose the element numbers to help you decide where an element is. Gaps in the numbering system do not cause a problem. Since Dytran has no wavefront requirement, wavefront optimization is not necessary. The order in which you define the grid points on shell elements is important since it determines the top and bottom surfaces. All shell elements in a region should have the same top and bottom surfaces, so that when you plot the results you are looking at the same side of the real structure. Some modeling programs can plot the element coordinate system. Use this to check that the element z-axes are all pointing in the same direction.

The Lagrangian elements in Dytran can undergo large deformations during the analysis. Therefore, the analysis should be started with as little distortion as possible. Many of the highly automated mesh generations produce element shapes that can be viewed as distorted elements when the elements are used in highly nonlinear analyses. Use the automated mesh generators with care, and be prepared to modify the shapes of some of the elements that are produced.

Some automated mesh generators only produce triangles and tetrahedra. Avoid using such generators, since the tetrahedra have very poor performance. The triangular shells give correct answers in bending, but are stiffer than quadrilateral elements. Triangular elements tend to make the analysis more expensive since two triangles are necessary to model one quadrilateral.

Use the geometry checking option of the modeler to ensure that the elements have reasonable shapes, since Dytran analyzes elements of almost any shape. Use any other checking options available in the modeler because errors in the model can prove to be expensive.

Eulerian elements can be modeled in the same way as Lagrangian elements, since they can have general connectivity and arbitrary shape. Since material flows through an Eulerian mesh, it is important that the mesh extends far enough to accommodate the motion of the material.

Some actions in the modeling package can produce elements that have (mathematically) negative volumes. The GEOCHECK parameter performs a check on all three dimensional elements and corrects them if necessary.

Properties and Materials

Select an appropriate property type within the modeler. Depending on the modeler, it may be necessary to translate the available properties (normally PSHELL/PSOLID) into appropriate Dytran property types (e.g., PEULER,PEULER1). The Patran modeling package does enable the user to define interactively any of the Dytran property types and should be used if possible at this stage of the modeling process.

Constraints

Single-point constraints can be applied to any Lagrangian grid point in the model, but the enforced displacement must be zero. Nonzero displacements are not valid in Dytran. Eulerian grid points should not be constrained unless this is desired in an ALE calculation. A variety of different types of constraints (SPC1, SPC2, SPC3) are available, enabling you to define constraints with respect to both stationary and moving coordinate systems.

Loading

You can apply Lagrangian concentrated loads and pressures in the modeler, and they are translated to FORCE, MOMENT, and PLOAD4 entries. Most modeling packages only apply static loads, not dynamic ones. Therefore, you must add a TLOADn entry giving the variation of the load with time later. When you apply loads in the modeling package, the magnitude of the load or pressure that you specify is actually the scale factor for the load-time curve.

The faces of Eulerian elements with flow boundaries can be specified in the modeler (see Modeling of Surfaces and Faces in this chapter).

Modeling of Surfaces and Faces

The ability to model surfaces is an important part of preparing Dytran input data since surfaces (see SURFACE, SUBSURF) are used to model contact surfaces, coupling surfaces, rigid bodies of arbitrary shape, rigid connections between parts of the mesh, arbitrary Lagrange-Euler interfaces (known as ALE surfaces), gas bags (often used in air bag calculations).

These surfaces are built up from the faces of elements using one of three "element face" types: CSEG, CFACE, or CFACE1 entries. The only difference between these three types is the way in which the element face is defined. Of these three face types, the CFACE entry is the optimal entry resulting in the quickest data translation and processing.

Defining these faces is probably the greatest area of incompatibility between most modeling packages and Dytran. There are often a large number of CSEG, CFACE, or CFACE1 entries in a typical input file. It is important to be able to check graphically that the segments are all present and correct. Few modeling packages can write CSEG, CFACE, or CFACE1entries.

The exception is Patran, which can both generate and display CFACEs, enabling the user to visualize the surfaces on the screen. The Patran user can, therefore, interactively define CFACEs and visualize the surfaces/boundary conditions (e.g., FLOW boundary conditions) that refer to these CFACEs together with the entries, which in turn refer to the surfaces (e.g., RIGID, CONTACT, and COUPLE entries).

If the user wishes to use a modeling package other than Patran, Dytran contains several "tricks" enabling the user to define CSEGs and CFACE1s in a relatively straightforward manner:

CSEGs – Trick 1

Dytran converts CQUAD4 and CTRIA3 entries to CSEG entries if the following rules are followed:

1. The PID of the PSHELLn entry that is referenced by the CQUAD4 or CTRIA3 element is the set number (SID) of the CSEG entry produced.
2. The thickness (T) on the PSHELLn entry is set to 9999.

Note: A CQUAD4 or CTRIA3 with a thickness of 9999 only gives you a CSEG entry. It does not give a shell element as well. The following two forms of input, therefore, give identical results:

```
CSEG, 100, 10, 1, 2, 22, 21
CSEG, 101, 10, 2, 3, 23, 22
```

and

```
CQUAD4, 100, 10, 1, 2, 22, 21
CQUAD4, 101, 10, 2, 3, 23, 22
PSHELL, 10,, 9999
```

In the second case, you do not obtain any shell elements since the CQUAD4 entries are converted to CSEG entries as they are read in. This is because their thickness is 9999. Note that the material ID number of the PSHELL entry (field 3) is left blank and is no longer required data.

For example, to create the contact surface below, create CQUAD4 elements on the faces of all elements on the bottom surface of the upper block. Then give them a thickness of 9999.0. Next, create CQUAD4 elements on the faces of the elements on the top surface of the lower block and give them a thickness of 9999.0. Finally, add the following COUPLE and SURFACE entries to the translated input file. Your input file will contain:

```
$ Bottom surface of upper block
CQUAD4, 100, 10, 1, 2, 3, 4
CQUAD4, 101, 10, 5, 6, 7, 8
PSHELL, 10,, 9999.
$ Top surface of lower block
CQUAD4, 200, 20, 101, 102, 103, 104
CQUAD4, 201, 20, 105, 106, 107, 108
PSHELL, 20,, 9999.
CONTACT, 1, SURF, SURF, 1, 2
SURFACE, 1,, SEG, 10
SURFACE, 2,, SEG, 20
```

Although this may seem a little confusing at first, you will quickly become accustomed to it. All you need to remember is that a CQUAD4 or CTRIA3 element with a thickness of 9999 is not a shell element. Naturally, the other property data for such entries is irrelevant and is ignored.

CSEGs – Trick 2

Trick 1 enables the user to specify CSEGs for any kind of element (2-D or 3-D) via a PSHELL/PSHELL1 entry and corresponding CQUAD4/CTRIA3 entries using a thickness of 9999. In many applications, you only need to generate surfaces constructed from the faces of shell or membrane elements (an example is an air-bag analysis). In these cases, you can directly generate the surface by selecting the elements by property, material, or element number (see the SURFACE entry; TYPE=PROP, MAT or ELEM). Dytran then internally generates a CSEG for each of the specified shell/membrane elements, avoiding an explicit definition of each CSEG entry.

One drawback of this "trick" is the inability to visualize the faces and surface. Tricks 1 and 3 are the preferred methods since you can at least visualize the CSEGs and CFACE1s entries.

CFACE1 – Trick 3

When defining the boundary conditions of Eulerian meshes, it is more efficient to use CFACE1 than CSEG. Dytran converts PLOAD4 pressure entries to CFACE1 entries if the pressure is 9999. The following two forms of input are, therefore, identical:

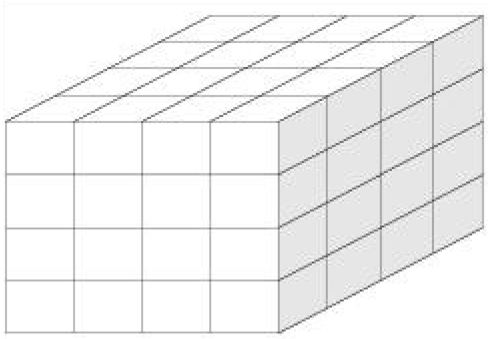
```
CFACE1, 1, 100, 1279, 7377,  7477
CFACE1, 2, 100, 3042, 6672, 5943
```

and

```
PLOAD4, 100, 1279, 9999., , , , 7377,  7477
PLOAD4, 100, 3042, 9999., , , , 6672,  5943
```

No pressure loads are applied since the pressure is 9999. This means that they are converted to CFACE1 entries as they are read in. Remember that PLOAD4 entries with a thickness of 9999. are not pressure loads but are CFACE1 entries.

In the following example, a nonreflecting boundary is applied to the shaded faces of a Eulerian mesh.



```
TLOAD1, 100, 110,  , 4
FLOW, 110,  200, MATERIAL, 10
PLOAD4, 200, 4, 9999., , , , 5, 115
PLOAD4, 200, 14, 9999., , , , 15, 125. . .
```

Since the PLOAD4 entries have a pressure of 9999., they are not pressure loads but are CFACE1 entries.

Translating the Data

After you have created your model using a modeling package, the data must be converted to a form that Dytran can understand. Use the translator in the modeling program to write a Nastran input file (virtually all modeling packages can write this type of file).

We recommend that you use Patran with the Dytran preference as the modeling package. The product is tailored to be used with Dytran. It allows you to address almost all capabilities that Dytran offers.

The exact form and completeness of the resulting input file varies depending on the modeler used. If your modeler adds Nastran Executive and Case Control to the top of the input file, delete them since the commands are different for Dytran. If you do not delete them, Dytran ignores them since they are valid only for Nastran. Similarly, Dytran ignores any PARAM entries written in the Bulk Data Section for use with Nastran DMAP.

If you are not using Patran, you must add some additional information to the file. First, if you are using any of the following, add the File Management Section:

- Output file types.
- User subroutines (if required).
- Restart control (if required).

Next, add a Case Control Section giving the following information:

- Termination time/time step for the analysis.
- Sets of constraints and loading to be used.
- Type and frequency of the results to be output.
- Other Case Control options.

Finally, add any additional features that could not be created during the modeling stage. These are problem dependent, but will usually include some of the following:

- Material properties (DMATxx)
- Eulerian properties (PEULER, PEULER1)
- Contact surfaces (CONTACT)
- Coupling surfaces (COUPLE)
- ALE surfaces (ALE)
- Rigid bodies (RIGID, RELLIPS, MATRIG, RBE2-FULLRIG)
- Rigid walls (WALL)
- Dynamic loading (TLOAD1, TLOAD2)
- Eulerian boundary conditions (FLOW, WALLET)
- Initial conditions (TIC, TICGP, TICEL)
- Cross sections (SECTION)

After these modifications are completed, you have a ready-to-run Dytran input file.

Checking the Data

Once you have an Dytran input file, always run a data check before running the main analysis. A data check performs the following tasks:

- Reads and checks the input file.
- Sorts out the data and prints any error or warning messages.
- Performs two time steps.

No action is required to carry out a data check, since it is the default option for all new analyses. Carefully check the output to ensure that the model is correct.

The importance of checking your input data cannot be overemphasized. The nonlinear, dynamic nature of Dytran analyses requires large amounts of computer resources. Dytran can also analyze models that might be rejected by other codes. Input errors can be both expensive and time consuming. A little extra time spent thoroughly checking the data is well worth it in the long term.

Executing Dytran

The input for Dytran is the same regardless of the computer being used. However, the way you actually run the program is machine dependent. The release notes (part of the on-line help system) describe the computer architectures that this version of Dytran has been ported to and verified on.

In some instances, the execution of Dytran may be customized to a particular machine configuration. If you have not used Dytran on a particular machine before, contact your system administrator for details on how to run it.

Running Dytran Using the `dytran` Command

Dytran can also be run using the command procedure called `dytran`. The command line procedure processes the arguments that you supply and creates a second command procedure, which is submitted to a batch queue, or is executed interactively. To view all the possible command line options, type `dytran` without any arguments and a listing of all available commands with an explanation of what they mean is shown.

Below are some examples on how to use the command line procedure:

```
% dytran  jid=mydata  exe=my_exe.exe
% dytran  jid=mydata  bat=no ( interactive run )
% dytran  jid=mydata  bat=no print=printout
% dytran  jid=mydata
```

Stopping Dytran on Linux

There are several methods of stopping a Dytran job after it is running. The method used depends on how the job is running (batch or interactive) and whether the contents of the output files is important.

When Output Files are Unimportant

If the program is running interactively, a <control-c> command may be typed at the terminal. If the program is running in batch, then the following command may be issued:

```
%kill -9 pid
```

where `<pid>` is the process ID of the Dytran job. The process ID can be obtained using the UNIX command `ps`.

When Output Files are Important

To provide a systematic, forced shutdown, Dytran traps the UNIX signal `SIGUSR1`. When this signal is caught, the program writes its restart files when requested by means of the keyword `END`, and terminates when the current time step is complete. To terminate the program, enter the following command:

```
%kill -USR1 <pid>
```

where `<pid>` is the process ID of the Dytran job. If the job is running interactively, you have to do this from another terminal.

When a Restart File is Important

In some cases, it may be necessary to terminate an analysis prematurely. In this case, it is still possible to force a restart file to be written upon termination regardless of any restart output requests. This is done on receipt of the UNIX signal `SIGUSR2`. To terminate the program, enter the following command:

```
%kill -USR2 <pid>
```

where `<pid>` is the process ID of the Dytran job. If the job is running interactively, you will have to do this from another terminal.

User Defined Services

Starting from version 2019 the Dytran "user subroutines" have been replaced by the SCA based User Defined Services (UDS), also used by MSC Nastran. In order to use UDS a FORTRAN and/or C/C++ compiler and an installation of the MSC Software Development Kit (SDK) are required. For the 2021 release the correct version numbers can be found in Table 3-1 of the Release Guide.

First a directory tree needs to be created holding the source- and definition files for the UDS services to be used. The easiest way to accomplish this is probably copying an example and making the appropriate modifications in the C/C++ or FORTRAN files.

Next the UDS libraries need to be created from the source tree. This can be done manually, by using the Dytran run scripts, or DytranExplorer, of which the first is the most complicated. For more detailed information about using Dytran Explorer to build UDS jobs, please refer to Running Dytran Using Dytran Explorer.

Building UDS Services Manually

To create the UDS libraries by hand, the following items should be known:

1. The location of the SCONS executable. SCONS comes with the Software Development Kit (SDK) and can be found in the "Tools" subdirectory. For Windows it will be something like …\SDK\20210\Tools\scons.cmd and for Linux …/SDK/20210/Tools/scons.

   In the example variable SCONS will be used for it.
2. The location of the SCA System. SCA also comes with the SDK and can be found in the "SCAKernel" subdirectory. On Windows it will be something like …\SDK\20210\SCAKernel and for Linux …/SDK/20210/SCAKernel. In the example variable SCASYSTEM will be used for it.
3. The root of the target directory for the SCA objects to be created. Any (empty) directory will do, for example a subdirectory called "scaobj" in the working directory.

In the example variable SCAOBJ will be used for it.

4. The names of the UDS services to be included in the build. These are the services defined by the CONNECT SERVICE entries in your input deck.

   In the example variable SCATARGETS will be used for it. When there are more than one services, they should be separated by a space.

5. The location of the "sdk" subdirectory in the Dytran installation. On Windows this will be something like ...\Dytran\2021\sdk.

   In the example variable DYTRANSDK will be used for this.

6. The root of the directory tree containing the UDS source files. This is the directory containing the four Scons files SConopts, SConopts.delivery, SConscript and SConstruct.

To start the build change directory to the root of the tree containing the UDS source files and issue the command:

```
<SCONS>  <SCATARGETS>  APPS_LOCAL=<SCAOBJ>  SCA_OBJECT=<SCAOBJ>
          APPS_SYSTEM=<SCASYSTEM>  APPS2_SYSTEM=<DYTRANSDK>
```

Building UDS Services Using Dytran Run Command

The wrapper for the Dytran run command has two new parameters for UDS support, "udsdir" and "scaobj". "udsdir=" is used to define the root of the directory tree with the UDS source files and "scaobj=" defines the root of the tree with the UDS library and can be used both to specify the target for a build from "udsdir", as well as to specify the location of pre-built UDS libraries.

The way the run command handles UDS builds is as follows. First it checks if there are any "CONNECT SERVICE" entries in the input deck. If so it checks if any of the parameters "udsdir=" or "scaobj=" was specified. If parameter "udsdir" was used, a build run is assumed targeted to the directory specified by the "scaobj" parameter. If the "scaobj" parameter is missing, the directory "scaobj" in the "udsdir" directory will be used by default. Just like with the manual build, the location of the "scons" executable and the SCAKernel directory need to be known. They can be specified directly with the environment variables SCATOOLS and SCASYSTEM, with SCATOOLS the path of the directory containing the scons executable (on Windows something like ...`\SDK\20210\Tools` and for Linux .../`SDK/20210/Tools`) and SCASYSTEM the root of the SCA system tree (on Windows something like ...`\SDK\20210\SCAKernel` and for Linux .../`SDK/20210/SCAKernel`). If only SCATOOLS was given, the run command wrapper tries to find the location of the SCA system directory from that of the "Tools" directory, assuming both come from the SDK with SCAKernel installed parallel to Tools. If also SCATOOLS is undefined, it is checked if the scons executable can be found in one of the directories in the PATH environment variable. If so, this directory will be used for SCATOOLS and again the SCASYSTEM will be assumed to be in a parallel subdirectory.

In summary

1. Use the values of environment variables SCATOOLS and SCASYSTEM when both are set.
2. Derive SCASYTEM from SCATOOLS, if only SCATOOLS was set.
3. Derive SCATOOLS from PATH when the scons executable is in PATH, and derive SCASYSTEM from SCATOOLS.

The SCons build may take some time and will be done before the execution of the Dytran job during the interactive session of the wrapper no matter the value of the “bat=” parameter.

Running Dytran with UDS Using Run Command

There are two options for running Dytran with UDS services with the run command: with or without a build. When the “udsdir” parameter is specified a run with build will be performed. See above. If not, a run with pre-built UDS libraries is assumed.

For a run with pre-built libraries there are also two options: using the environment variables SCA_SERVICE_CATALOG and SCA_LIBRARY_PATH, or specifying the locations of the libraries with the “`scaobj=`” parameter.

When the environment variables are used, they should be defined as follows:

```
SCA_SERVICE_CATALOG path to <scaobj>/res/SCAServiceCatalog.xml
SCA_LIBRARY_PATH path to <scaobj>/<SC_HOST>/lib
```

With `<scaobj>` the same directory as used for the builds and `<SC_HOST>` either WIN8664 on Windows or something like LX8664_RHE71 on Linux.

When using the “`scaobj=`” parameter, it should specify the same directory as used for the build.

The command may look like:

```
<DYTRANDIR>/bin/dytran jid=myjob scaobj=myobjects  …
```

With `<DYTRANDIR>` the root directory of the Dytran 2021 installation.

Files Created by Dytran

Dytran creates a number of files during the analysis. The names of these files are generally identical for all computer systems. Some differences may exist for other systems and are given in the *Dytran Installation and Operations Guide*.

In the main sections of this manual, generic names are used when referring to a particular Dytran file. These generic reference names and the actual generic file names are given below:

File	Generic Reference Name	Generic Name
Input	`dat`	`file_name.dat`
Messages	`MSG`	`FILE_SUMMARY.MSG`
Output	`PRT`	`file_name.OUT`
Archive	`ARC`	`output_file_#.ARC`
Time History	`THS`	`output_file_#.THS`
Restart	`RST`	`output_file_#.RST`
Warning and Errors	`ERR`	`ERROR_SUMMARY.MSG`

File	Generic Reference Name	Generic Name
Data Ignored	`IGN`	`NASTRAN_IGNORE.MSG`
Neutral Input	`NIF`	`file_name.NIF`
Job Status	`INFO`	`JOB.INFO`

Input File

The input file contains all the input data and must be present in order to run Dytran. It is a text file with up to 80 characters on each line.

Message File

The message file contains information every time data is written to an archive or restart file.

Output File

The output file is a text file suitable for printing on a line or laser printer, or viewing with a standard editor. It contains messages produced by Dytran as well as a summary of the calculation at every time step.

Archive Files

Dytran can create any number of archive files containing results at times during the analysis. Archive files are binary files. They contain a complete description of the geometry and connectivity of the analysis model as well as the requested results. Patran can read archive files for postprocessing.

Time-History Files

Dytran can also create any number of time-history files containing results for particular grid points and elements during the calculation. They are also binary files. Time-history files only contain results. With Dytran Explorer, you can quickly plot the results contained in the time-history files. It also allows you to combine results, as well as importing and exporting of results to other formats. For more detailed information, please refer to Running Dytran Using Dytran Explorer.

Restart Files

Restart files are binary files that contain the information necessary to restart the analysis. Any number can be created; however, since a full representation of the analysis is stored every time restart data is written, restart files can become very large. You may run out of disk space if you write restart data too often.

Neutral Input File

This is an internal file used to store the input data after initial sorting. Usually, it can be deleted.

Error File

Dytran produces a single error file containing a summary of all warnings and errors issued during reading and subsequent data processing.

Data Ignored File

In many cases, Dytran issues messages indicating differences between Nastran entries and the corresponding Dytran interpretation for example. If these are not fatal, they are written to this file.

Job Status

This is an internal file and will be used by the Dytran Explorer to update the progress bar.

Outputting Results

Input Commands

Dytran is very flexible in the way results are output for postprocessing. Any number of output files can be created, and you can choose exactly what entities, such as grid points and elements, and which results are stored in each file. Data can be written at any time or time step during the analysis.

You need to specify the following for a complete output specification:

1. Type of the file
2. How often it is saved
3. How often data is written
4. What entities (e.g., grid points, elements) are stored
5. What results are output

Type of the File - TYPE Entry

There are six types of files selected using the TYPE FMS statement:

`Archive:` This is usually used for storing connectivity and results for all or part of the model at particular times during the analysis. It can then be used to plot deformed shapes, contour plots, or arrow plots.

```
TYPE (logical_file) = ARCHIVE
```

`TimeHis:` This is usually used for storing only results from a few key entities during the analysis. It does not contain the connectivity capability, and so it can be used only to create time history plots of the results.

`TYPE (logical_file) = TIMEHIS`

`Restart:` This is not a results file but is used for restarting Dytran and in fact is similar to an archive file.

`TYPE (logical_file) = RESTART`

`StepSum:` A one-line time step summary is written to the output (file). It is useful for checking the characteristics of the analysis.

`TYPE (logical_file) = STEPSUM`

`MatSum:` Output of the state of a material at selected time steps.

`TYPE (logical_file) = MATSUM`

`MRSUM:` Information about MATRIG and RBE2-`FULLRIG` assemblies at selected time steps.

`TYPE (logical_file) = MRSUM`

`EBDSum:` Information about the Eulerian boundaries at selected time steps.

`TYPE (logical_file) = EBDSUM`

The file that a particular command refers to is identified by a logical filename. This logical filename forms part of the actual filename on most computers (see the *Dytran Installation and Operations Guide*).

How Often It Is Saved - `SAVE` Entry

Every time data is stored, it is usually stored in the same file. However, after data is stored for a number of times as specified by the SAVE FMS statement, a new file is opened and the old file is closed and saved. For example, `SAVE (logical_file) = 20` means that twenty sets of results are stored in the file before it is closed and a new file is opened.

On most types of computers the time-step number identifies the files when data was first written to the file. See the *Dytran Installation and Operations Guide* for details of the filenames on your computer.

Once a file has been saved and closed, the results stored are available for postprocessing even when the analysis is still running. When the file is not yet closed, you may have access to the data already. It may be however, that the last record is incomplete as it may still reside in the output buffer. An error may occur upon reading the data from the file. No data is lost, and the data will be available the next time the buffer is flushed.

How Often Data Is Written – STEPS/TIMES Entry

Results can be output at intervals during the analysis. You may specify the intervals in terms of the analysis time (using the TIMES entry), or in terms of the time-step number (by using the STEPS entry). You specify a list of the times or time steps at which output is required. For example:

```
STEPS (logical_file) = 100, 200, 300
```

A range can be specified as `<start>, THRU, <end>, BY, <increment>`. The end of the calculation can be identified by the word `END`. For example,

```
TIMES (logical_file) = 0, THRU, END, BY, 1.0E-3
```

Grid Points, Elements Stored/Results Output

The items for which variables are to be stored are specified through a SET entry and one of the following:

GRIDS	Grid points to be stored
ELEMENTS	Elements to be stored
RIGIDS	Rigid bodies to be stored
RELS	Rigid ellipsoids to be stored
MATS	Materials to be stored
GBAGS	Gas bags to be stored
CONTS	Contact surfaces to be stored
CSECS	Cross sections to be stored
CPLSURFS	Coupling surfaces to be stored
CPLSUBS	Coupling subsurfaces to be stored
SURFACES	Surfaces to be stored
SUBSURFS	Subsurfaces to be stored
WALLS	Rigid walls to be stored
EBDS	Eulerian boundary conditions to be stored

The required results for these entities are specified using the following commands:

GPOUT	Grid-point results
ELOUT	Element results
RBOUT	Rigid body results
RELOUT	Rigid ellipsoid results
MATOUT	Material results
GBAGOUT	Gas bag results
CONTOUT	Contact surface results
CSOUT	Cross-section results
CPLSOUT	Coupling-surface results
CPLSBOUT	Coupling subsurface results
SURFOUT	Surface results
SUBSOUT	Subsurface results
WALLOUT	Rigid wall results
EBDOUT	Eulerian boundary data results

For example, the following commands store the xx-stress for elements 200 through 210:

```
ELEMENTS (FILE7) = 20
SET 20 = 200, THRU, 210
ELOUT (FILE7) = TXX
```

The following commands store the x-velocity and y-force components for grid points 101, 209, and 1005.

```
GRIDS(ARC1) = 40
SET 40 = 101, 209, 1005
GPOUT(ARC1) = XVEL, YFORCE
```

The results available for output are listed below in Result Types. Results for rigid bodies, rigid ellipsoids, gas bags, materials, contact surfaces, cross sections and surface gauges can only be stored in time-history files. Results for coupling surfaces can only be stored in archive files.

Result Types

The data stored and available for output varies with the element type or, in the case of grid points, the type of element to which the grid points are attached. The following fundamental element types are available in Dytran:

- One-dimensional elements
- Lagrangian solid elements
- Quadrilateral shell elements
- Triangular shell elements
- Triangular membrane elements
- Dummy quadrilateral and triangular elements
- Eulerian solid elements (hydrodynamic)
- Eulerian solid elements (with shear strength)
- Eulerian solid elements (multimaterial hydrodynamic)
- Eulerian solid elements (multimaterial with shear strength)
- Rigid bodies

A particular archive or time-history file can contain information for the grid points or elements of one or more of these element types.

Below you find a list with all of the available results types. The keyword is the description you use in the Case Control section (GPOUT, ELOUT etc.), to write the data to archive or time-history files.

GPOUT — Grid Point Results

One-Dimensional Elements

Keyword	Description
XPOS	x-coordinate
YPOS	y-coordinate
ZPOS	z-coordinate
XVEL	z-translational velocity
YVEL	y-translational velocity
ZVEL	z-translational velocity
XFORCE	x-force = external + internal
YFORCE	y-force = external + internal
ZFORCE	z-force = external + internal
XDIS	x-translational displacement
YDIS	y-translational displacement
ZDIS	z-translational displacement
PMASS	Grid-point mass
PMOMI	Grid-point inertia
XAVEL	x-angular velocity
YAVEL	y-angular velocity
ZAVEL	z-angular velocity
XMOMENT	x-moment = external + internal
YMOMENT	y-moment = external + internal
ZMOMENT	z-moment = external + internal
EXRVEL	Initial enforced x-angular velocity (Nastran Initialization)
EYRVEL	Initial enforced y-angular velocity (Nastran Initialization)
EZRVEL	Initial enforced z-angular velocity (Nastran Initialization)
EXVEL	Initial enforced x-velocity (Nastran Initialization)
EYVEL	Initial enforced y-velocity (Nastran Initialization)
EZVEL	Initial enforced z-velocity (Nastran Initialization)
XFCON	Constraint x-force (SPC3 and FORCE3 only)
YFCON	Constraint y-force (SPC3 and FORCE3 only)
ZFCON	Constraint z-force (SPC3 and FORCE3 only)
XMCON	Constraint x-moment (SPC3 and FORCE3 only)

Keyword	Description
YMCON	Constraint y-moment (SPC3 and FORCE3 only)
ZMCON	Constraint z-moment (SPC3 and FORCE3 only)
DLTPNT	Nodal time step
XACC	x-translational acceleration
YACC	y-translational acceleration
ZACC	z-translational acceleration
RPOS	Resultant coordinate
RVEL	Resultant velocity
RACC	Resultant acceleration (see Remark 3.)
RFORCE	Resultant force = external + internal
RDIS	Resultant translational displacement
RAVEL	Resultant angular velocity
RMOMENT	Resultant moment = external + internal
RFCON	Resultant constraint force
RMCON	Resultant constraint moment

Remarks

1. The constraint forces and moments (XFCON, YFCON, ZFCON, XMCON, YMCON, and ZMCON) are only output when the constraint is an SPC3 or a FORCE3. In all other cases the result is zero.
2. All variables starting with R (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

3. If filtering of the acceleration is required, it should be done on the individual components (XACC, YACC, ZACC) and not on the resultant (RACC). This is because the resultant value is always positive, while the components can be either positive or negative.

Quadrilateral and Triangular Shells

Keyword	Description
XPOS	x-coordinate
YPOS	y-coordinate
ZPOS	z-coordinate
XDIS	x-translational displacement
YDIS	y-translational displacement

Keyword	Description
ZDIS	z-translational displacement
XVEL	x-translational velocity
YVEL	y-translational velocity
ZVEL	z-translational velocity
XAVEL	x-angular velocity
YAVEL	y-angular velocity
ZAVEL	z-angular velocity.
XFORCE	x-force = external + internal
YFORCE	y-force = external + internal
ZFORCE	z-force = external + internal
XMOMENT	x-moment = external + internal
YMOMENT	y-moment = external + internal
ZMOMENT	z-moment = external + internal
EXVEL	Initial enforced x-velocity (Nastran Initialization)
EXRVEL	Initial enforced x-angular velocity (Nastran Initialization)
EYRVEL	Initial enforced y-angular velocity (Nastran Initialization)
EZRVEL	Initial enforced z-angular velocity (Nastran Initialization)
EYVEL	Initial enforced y-velocity (Nastran Initialization)
EZVEL	Initial enforced z-velocity (Nastran Initialization)
XFCON	Constraint x-force (SPC3 and FORCE3 only)
YFCON	Constraint y-force SPC3 and FORCE3 only)
ZFCON	Constraint z-force (SPC3 and FORCE3 only)
XMCON	Constraint x-moment SPC3 and FORCE3 only)
YMCON	Constraint y-moment (SPC3 and FORCE3 only)
ZMCON	Constraint z-moment (SPC3 and FORCE3 only)
DLTPNT	Nodal time step
PMASS	Grid-point mass
PMOMI	Grid-point inertia
XACC	x-translational acceleration
YACC	y-translational acceleration
ZACC	z-translational acceleration
RPOS	Resultant coordinate
RVEL	Resultant velocity

Keyword	Description
RACC	Resultant acceleration (see Remark 3.)
RFORCE	Resultant force = external + internal
RDIS	Resultant translational displacement
RAVEL	Resultant angular velocity
RMOMENT	Resultant moment = external + internal
RFCON	Resultant constraint force
RMCON	Resultant constraint moment

Remarks

1. The constraint forces and moments (XFCON, YFCON, ZFCON, XMCON, YMCON, and ZMCON) are only output when the constraint is an SPC3 or a FORCE3. In all other cases the result is zero.
2. All variables starting with R (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

3. If filtering of the acceleration is required, it should be done on the individual components (XACC, YACC, ZACC) and not on the resultant (RACC). This is because the resultant value is always positive, while the components can be either positive or negative.

Triangular Membranes

Keyword	Description
XPOS	x-coordinate
YPOS	y-coordinate
ZPOS	z-coordinate
XDIS	x-displacement
YDIS	y-displacement
ZDIS	z-displacement
XVEL	x-velocity
YVEL	y-velocity
ZVEL	z-velocity
XFORCE	x-force = external + internal
YFORCE	y-force = external + internal
ZFORCE	z-force = external + internal
PMASS	Grid-point mass

Keyword	Description
XFCON	Constraint x-force (SPC3 and FORCE3 only)
YFCON	Constraint y-force (SPC3 and FORCE3 only)
ZFCON	Constraint z-force (SPC3 and FORCE3 only)
DLTPNT	Nodal time step
XACC	x-translational acceleration
YACC	y-translational acceleration
ZACC	z-translational acceleration
RPOS	Resultant coordinate
RVEL	Resultant velocity
RACC	Resultant acceleration (see Remark 3.)
RFORCE	Resultant force = external + internal
RDIS	Resultant translational displacement
RFCON	Resultant constraint force

Remarks

1. The constraint forces (XFCON, YFCON, and ZFCON) are only output when the constraint is an SPC3 or a FORCE3. In all other cases the result is zero.
2. All variables starting with R (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

3. If filtering of the acceleration is required, it should be done on the individual components (XACC, YACC, ZACC) and not on the resultant (RACC). This is because the resultant value is always positive while the components can be either positive or negative.

Dummy QUAD4s and TRIA3s – Rigid Bodies

Keyword	Description
XPOS	x-coordinate
YPOS	y-coordinate
ZPOS	z-coordinate
XVEL	x-velocity
YVEL	y-velocity
ZVEL	z-velocity
XFORCE	x-force = external + internal

Keyword	Description
YFORCE	y-force = external + internal
ZFORCE	z-force = external + internal
XLOCAL	x-coordinate in the rigid body coordinate system
YLOCAL	y-coordinate in the rigid body coordinate system
ZLOCAL	z-coordinate in the rigid body coordinate system
PMASS	Grid-point mass
DLTPNT	Nodal time step
XACC	x-translational acceleration
YACC	y-translational acceleration
ZACC	z-translational acceleration
RPOS	Resultant coordinate
RVEL	Resultant velocity
RACC	Resultant acceleration (see Remark 2.)
RFORCE	Resultant force = external + internal

Remarks

1. All variables starting with R (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

2. If filtering of the acceleration is required, it should be done on the individual components (XACC, YACC, ZACC) and not on the resultant (RACC). This is because the resultant value is always positive while the components can be either positive or negative.

Lagrangian Solids

Keyword	Description
XPOS	x-coordinate
YPOS	y-coordinate
ZPOS	z-coordinate
XVEL	x-velocity
YVEL	y-velocity
ZVEL	z-velocity
XFORCE	x-force = external + internal
YFORCE	y-force = external + internal
ZFORCE	z-force = external + internal
PMASS	Grid-point mass
EXVEL	Initial enforced x-velocity (Nastran Initialization)
EYVEL	Initial enforced y-velocity (Nastran Initialization)
EZVEL	Initial enforced z-velocity (Nastran Initialization)
XFCON	Constraint x-force (SPC3 and FORCE3 only)
YFCON	Constraint y-force (SPC3 and FORCE3 only)
ZFCON	Constraint z-force (SPC3 and FORCE3 only)
DLTPNT	Nodal time step
XACC	x-translational acceleration
YACC	y-translational acceleration
ZACC	z-translational acceleration
RPOS	Resultant coordinate
RVEL	Resultant velocity
RACC	Resultant acceleration (see Remark 3.)
RFORCE	Resultant force = external + internal
RFCON	Resultant constraint force
For all Lagrangian grid points that belong to a rigid surface, use XFCON, YFCON, and ZFCON to store the local coordinates in the rigid body system.	

Remarks

1. The constraint forces (XFCON, YFCON, and ZFCON) are only output when the constraint is an SPC3 or a FORCE3. In all other cases, the result is zero.
2. All variables starting with R (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

3. If filtering of the acceleration is required, it should be done on the individual components (XACC, YACC, ZACC) and not on the resultant (RACC). This is because the resultant value is always positive, while the components can be either positive or negative.

Eulerian Solids (Hydrodynamic, Multi-material Hydrodynamic, and Multi-material with Shear Strength)

Keyword	Description
XPOS	x-coordinate
YPOS	y-coordinate
ZPOS	z-coordinate
RPOS	Resultant coordinate
XVG	x-velocity
YVG	y-velocity
ZVG	z-velocity
RVG	Resultant velocity
APOS	Weight factor alpha
BPOS	Weight factor beta
CPOS	Weight factor gamma
XAG	x-acceleration
YAG	y-acceleration
ZAG	z-acceleration
RAG	Resultant acceleration
RACC	Resultant acceleration (see Remark 2.)

Remarks

1. All variables starting with R (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

2. If filtering of the acceleration is required, it should be done on the individual components (XACC, YACC, ZACC) and not on the resultant (RACC). This is because the resultant value is always positive, while the components can be either positive or negative.

Eulerian Solids (Shear Strength)

Keyword	Description
XPOS	x-coordinate
YPOS	y-coordinate
ZPOS	z-coordinate
RPOS	Resultant coordinate
RACC	Resultant acceleration (see Remark 2.)

Remarks

1. All variables starting with R (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

2. If filtering of the acceleration is required, it should be done on the individual components (XACC, YACC, ZACC) and not on the resultant (RACC). This is because the resultant value is always positive, while the components can be either positive or negative.

ELOUT — Element Results

One-Dimensional Elements

Keyword	Description
MASS	Element mass
YHAT1X	x-component of the y-axis of the local element coordinate system at end 1
YHAT1Y	y-component of the y-axis of the local element coordinate system at end 1
YHAT1Z	z-component of the y-axis of the local element coordinate system at end 1
ZHAT1X	x-component of the z-axis of the local element coordinate system at end 1
ZHAT1Y	y-component of the z-axis of the local element coordinate system at end 1
ZHAT1Z	z-component of the z-axis of the local element coordinate system at end 1
YHAT2X	x-component of the y-axis of the local element coordinate system at end 2
YHAT2Y	y-component of the y-axis of the local element coordinate system at end 2
YHAT2Z	z-component of the y-axis of the local element coordinate system at end 2
ZHAT2X	x-component of the z-axis of the local element coordinate system at end 2
ZHAT2Y	y-component of the z-axis of the local element coordinate system at end 2
ZHAT2Z	z-component of the z-axis of the local element coordinate system at end 2
XFORCE	x-force resultant in local element coordinate system

Keyword	Description
YFORCE	y-force resultant in local element coordinate system
ZFORCE	z-force resultant in local element coordinate system
XMOMENT	x-moment resultant in local element coordinate system
YMOMENT	y-moment resultant in local element coordinate system
ZMOMENT	z-moment resultant in local element coordinate system
FIBL1	Area properties - variable 1
FIBL2	Area properties - variable 2
FIBL3	Area properties - variable 3
FIBL4	Area properties - variable 4
EDIS	Element distortional energy
FAIL	Failure time
ELDLTH	Stable time step of element (according to CFL-criteria)
ELTIME	Time at which element is updated (subcycling)
ALPSTBL	Stable alpha of element (subcycling)
ELGROUP	Subcycle group of element
EXUSER1	User variable 1
EXUSER2	User variable 2
STRAIN	Element strain (belt elements only).
RFORCE	Resultant force.
RMOMENT	Resultant moment.
The Hughes-Liu beam can have variables at each integration point (layer) requested for output. For a list of valid variable names see Sublayer Variables Keywords and Descriptions.	

Remarks

1. For co-rotational `CELASx`/CDAMP1, the direction vector is stored in `ZHAT2X`, `ZHAT2Y`, and `ZHAT2Z`.
2. All variables starting with R (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

Quadrilateral Shell Elements

Keyword	Description
`MASS`	Element mass
`THICK`	Thickness
`AREA`	Area
`FAIL`	Failure time
`EDIS`	Element distortional energy
`QHOUR1`	Hourglass force 1
`QHOUR2`	Hourglass force 2
`QHOUR3`	Hourglass force 3
`QHOUR4`	Hourglass force 4
`QHOUR5`	Hourglass force 5
`EHRG`	Hourglass energy
`ELDLTH`	Stable time step of element (according to CFL criteria)
`ELTIME`	Time at which element is updated (subcycling)
`ALPSTBL`	Stable alpha of element (subcycling)
`ELGROUP`	Subcycle group of element
`EXUSER1`	User variable 1
`EXUSER2`	User variable 2
`Q1`	Cosine of angle between element and material coordinate systems
`Q2`	Sine of angle between element and material coordinate systems

Triangular Shell Elements

Keyword	Description
`MASS`	Element mass
`THICK`	Thickness
`AREA`	Area
`FAIL`	Failure time
`EDIS`	Element distortional energy
`ELDLTH`	Stable time step of element (according to CFL criteria)
`ELTIME`	Time at which element is updated (subcycling)
`ALPSTBL`	Stable alpha of element (subcycling)
`ELGROUP`	Subcycle group of element

Keyword	Description
EXUSER1	User variable 1
EXUSER2	User variable 2
Q1	Cosine of angle between element and material coordinate systems
Q2	Sine of angle between element and material coordinate systems
Note:	Both the triangular and the quadrilateral shell elements use integration points through the element thickness in the solution sequence. The results at the integration points (sublayers) can be individually requested for output. For a list of valid variable names see Sublayer Variables Keywords and Descriptions.

Sublayer Variables

The variable name consists of the normal name of the variable with the number of the integration layer added to the end.

TXX03 - TXX of third integration sublayer

TXZ32 - TXZ of thirty-second integration sublayer.

For convenience, the inner, middle, and outer sublayers can be referred to as -IN, -MID, and -OUT, rather than the specific integration sublayer number. Thus, TXX-IN refers to TXX at the inner sublayer (sublayer 1) and TXX-OUT refers to TXX at the outer sublayer (sublayer "max"). Note that the numbering of sublayers is in the positive direction of the element normal.

The program verifies whether the layer number at which output is requested (either by means of the -IN, -MID, or -OUT definition or by a sublayer number) is a valid one according to the property definition of the elements. If a sublayer number is not valid, the resulting value is zero.

DMATEP, YLDVM, and FAILMPS/FAILEX entries are used to specify the material data for the Hughes-Liu beam. The following specifications indicate which of the variables is relevant depending on the YLDVM entry. If the variable is not relevant but requested for output, the stored value is zero.

Hughes-Liu beam (form = HUGHES on PBEAM1 entry) and

- No YLDVM or FAILMPS/FAILEX entry

  TXX, TYY, TZZ, TXY, TYZ, and TZX
- YLDVM with only YIELD, EH fields (simple bilinear stress-strain curve)

  TXX, TYY, TZZ, TXY, TYZ, TZX, EFFPL, EFFST, and FAIL
- YLDVM with TABLE, TYPE fields (piecewise linear stress-strain curve) and/or TABY/ D, P fields (strain-rate-dependent yield stress)

  TXX, TYY, TZZ, TXY, TYZ, TZX, EFFPL, EFFST, FAIL, and EFFSR

For shell elements (CQUAD4 CTRIA3) with isotropic elastoplastic material (DMATEP definition) the default set of sublayer variables that are available for output are the stress and strain components, the effective stress, and the effective plastic strain. A restricted set can be obtained by setting the PARAM,STRNOUT to NO. In that case, the sublayer variables for output are the stress components and effective plastic strain.

Sublayer Variables Keywords and Descriptions

Keyword	Description
EFFPL	Effective plastic strain
EFFST	Effective stress
TXX	xx-stress
TYY	yy-stress
TZZ	zz-stress
TXY	xy-stress
TYZ	yz-stress
TZX	zx-stress
EPSXX	xx-strain
EPSYY	yy-strain
EPSZZ	zz-strain
EPSXY	xy-strain
EPSYZ	yz-strain
EPSZX	zx-strain
FAIL	Sublayer-failure flag
EFFSR	Strain rate
EDIS	Distortional energy
FBCFI	Fiber-compression index
FBTFI	Fiber-tension index
MXCFI	Matrix-compression index
MXTFI	Matrix-tension index
SHRFI	Shear-failure index
Q1XX	Composite-fiber orientation
Q2XX	Composite-matrix orientation
USR1L	User variable 1
USR2L	User variable 2
USR3L	User variable 3
USR4L	User variable 4
USR5L	User variable 5
USR6L	User variable 6
USR7L	User variable 7

Keyword	Description
USR8L	User variable 8
USR9L	User variable 9
USR10L	User variable 10
FIBFL	Fiber-failure flag
MTXFL	Matrix-failure flag
SHRFL	Shear-failure flag
TYZLIN	Linear stress in yz-direction
TZXLIN	Linear stress in zx-direction
EPSMX	Major principal strain
EPSMN	Minor principal strain
EPZZ	Effective plastic strain in transverse direction
FLP	Forming limit parameter
YLDRAD	Yield radius
Q1AFIB	FABRIC: Direction cosines between the element coordinate system and the warp ends
Q2AFIB	FABRIC: Direction sines between the element coordinate system and the warp ends
Q1BFIB	FABRIC: Direction cosines between the element coordinate system and the weft picks
Q2BFIB	FABRIC: Direction sines between the element coordinate system and the weft picks
SGMA	FABRIC: Stress in fabric parallel to the warp ends
SGMB	FABRIC: Stress in fabric parallel to the weft picks
SGFRIC	FABRIC: Stress due only to shear in the weave of the fabric
EPSFA	FABRIC: Strain in fabric parallel to the warp ends
EPSFB	FABRIC: Strain in fabric parallel to the weft picks
ANGLE	FABRIC: Crossover angle between warp ends and weft picks

Triangular Membrane Elements

Keyword	Description
MASS	Element mass
THICK	Thickness
TXX	xx-stress (element coordinate system)

Keyword	Description
TYY	yy-stress (element coordinate system)
TXY	xy-stress (element coordinate system)
EFFPLS	Effective plastic strain
EFFSTS	Equivalent stress
AREA	Original area
SLEN21	Side length 21
SLEN32	Side length 32
SLEN31	Side length 31
X2HAT	x-coordinate point 2 in co-rotational frame
Y2HAT	y-coordinate point 2 in co-rotational frame
X3HAT	x-coordinate point 3 in co-rotational frame
Y3HAT	y-coordinate point 3 in co-rotational frame
EPSXX	xx-membrane strain
EPSYY	yy-membrane strain
EPSXY	xy-membrane strain
EXXIMM	xx-membrane IMM strain
EYYIMM	yy-membrane IMM strain
EXYIMM	xy-membrane IMM strain
FAC1	Mass fraction of the first grid point
FAC2	Mass fraction of the second grid point
FAC3	Mass fraction of the third grid point
ECF3X	x-direction of co-rotational base-vector 3
ECF3Y	y-direction of co-rotational base-vector 3
ECF3Z	z-direction of co-rotational base-vector 3
SMDFER	Ratio of current and original area
EXUSER1	User variable 1
EXUSER2	User variable 2
ELTIME	Time at which element is updated
ALPSTBL	Stable alpha of element (subcycling)
ELGROUP	Subcycling group of element

Dummy Quads and Trias

Keyword	Description
ZUSER	User-specified

Lagrangian Solid Elements

Keyword	Description
TXX	xx-stress
TYY	yy-stress
TZZ	zz-stress
TXY	xy-stress
TYZ	yz-stress
TZX	zx-stress
EFFSTS	Effective stress
PRESSURE	Pressure
EFFPLS	Effective plastic strain
FAIL	Failure time
SIE	Specific internal energy
EDIS	Distortional energy
TDET	Detonation time
MASS	Element mass
VOLUME	Volume
HALFQ	Half of the artificial viscosity
SNDSPEED	Sound speed
EXUSER1	User variable 1
EXUSER2	User variable 2
ELTIME	Time at which element is updated
ALPSTBL	Stable alpha of element (subcycling)
ELGROUP	Subcycle group of element

Lagrangian Solid Elements — Nonhydrodynamic Materials Only

Keyword	Description
EPSXX	xx-centroidal strain
EPSYY	yy-centroidal strain
EPSZZ	zz-centroidal strain
EPSXY	xy-centroidal strain
EPSYZ	yz-centroidal strain
EPSZX	zx-centroidal strain

Lagrangian Solid Elements — Orthotropic Materials Only

Keyword	Description
AX	x-component of material axis a
AY	y-component of material axis a
AZ	z-component of material axis a
BX	x-component of material axis b
BY	y-component of material axis b
BZ	z-component of material axis b
CX	x-component of material axis c
CY	y-component of material axis c
CZ	z-component of material axis c
EXX	xx-Young's modulus
EYY	yy-Young's modulus
EZZ	zz-Young's modulus
EXY	xy-Young's modulus
EYZ	yz-Young's modulus
EZX	zx-Young's modulus
GXY	xy-shear modulus
GYZ	yz-shear modulus
GZX	zx-shear modulus
EMAX	Maximum Young's modulus

Lagrangian Solid Elements — IG Explosive Materials (EOSIG) Only

Keyword	Description
SIE-E	Specific internal energy per unit mass of un-reacted explosive part
SIE-P	Specific internal energy per unit mass of reaction products part
FBURN	Burn fraction
FMAT	Volume fraction
RHO-E	Density of un-reacted explosive part
RHO-P	Density of reaction products part
MASS-E	Mass of un-reacted explosive part
MASS-P	Mass of reaction products part

Eulerian Solid Elements — Hydrodynamic

Keyword	Description
XVEL	x-velocity
YVEL	y-velocity
ZVEL	z-velocity
VOLUME	Element volume
MASS	Mass of material 1
DENSITY	Density of material 1
SIE	Specific internal energy of material 1
PRESSURE	Pressure
Q	Artificial viscosity
ENERGY	Total energy (kinetic + internal) of material 1
DIV	Divergence
VOID	Void fraction
FMAT	Material fraction of material 1
FMATPLT	Material fraction of material times the volume uncovered fraction
TEMPTURE	Temperature
SSPD	Speed of sound
FBURN	Burn fraction
TDET	Detonation time
BFTIME	Burn time
DEFMAT	Indicates the region of material that can burn. Only used for EOSDEF

Keyword	Description
DEFBURN	Burn fraction for EOSDEF
AFTERBURN	After burn fraction. Used only for EOSJWL
XMOM	x-momentum
YMOM	y-momentum
ZMOM	z-momentum
FVUNC	Volume uncovered fraction
QDIS	Characteristic element length
FVUOLD	Old volume uncovered fraction
EXUSER1	User variable 1
EXUSER2	User variable 2
MFLR	Total mass-flow rate (sum of MFLR-POR, MFLR-PER and MFLR-INF)
MFL	Total mass-flow (sum of MFLR-POR, MFLR-PER and MFLR-INF)
MFLR-POR	Mass-flow rate by PERMEAB (flow between the Euler element and GBAGs through holes) or PORHOLE (flow between the Euler element and the environment through holes)
MFL-POR	Mass-flow by PORFGBG (flow between the Euler element and GBAGs through holes) or PORHOLE (flow between the Euler element and the environment through holes)
MFLR-PER	Mass-flow rate by PERMGBG (flow between the Euler element and GBAGs through permeable (SUBSURF) (SURFACEs) or PERMEAB (flow between the Euler element and the environment through permeable (SUBSURF) (SURFACEs)
MFL-PER	Massflow by PERMGBG (flow between the Euler element and GBAGs through permeable (SUBSURF) (SURFACEs) or PERMEAB (flow between the Euler element and the environment through permeable (SUBSURF) (SURFACEs)
MFLR-INF	Massflow rate by inflators (inflow by INFLATRs and EXFLOW,INFLATR3)
MFL-INF	Massflow by inflators (inflow by INFLATRs and EXFLOW,INFLATR3)
MFLR-GBG	Massflow rate by PORFGBG or PORFLGBG (flow between GBAGs).
MFL-GBG	Massflow by PORFGBG or PORFLGBG (flow between GBAGs).
MFLR-CPL	Massflow rate by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFL-CPL	Massflow by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFLR-FAIL	Massflow rate by COUP1INT through failed segments
MFL-FAIL	Massflow by COUP1INT through failed segments
DQDT	Total heat transfer rate (sum of DQDT-CNV and DQDT-RAD)
DQ	Total heat transfer (sum of DQ-CNV and DQ-RAD)
DQDT-CNV	Heat transfer rate by HTRCONV (transfer from the Euler element to the environment through (SUBSURF) (SURFACEs)

Keyword	Description
`DQ-CNV`	Heat transfer by HTRCONV (transfer from the Euler element to the environment through (SUBSURF) (SURFACEs)
`DQDT-RAD`	Heat transfer rate by HTRRAD (transfer from the Euler element to environment through (SUBSURF) (SURFACEs)
`DQ-RAD`	Heat transfer by HTRRAD (transfer from the Euler element to the environment through (SUBSURF) (SURFACEs)

Remarks

1. All variables starting with `MFL` (MassFLow) have following sign conventions:

   `MFLxx > 0` means inflow of mass in the Euler element.
   `MFLxx < 0` means outflow of mass from the Euler element.

2. All variables starting with `DQ` have following sign conventions:

   `DQxx > 0` means inflow of heat in the Euler element.
   `DQxx < 0` means outflow of heat from the Euler element.

Eulerian Solid Elements — Viscous Hydrodynamic

Keyword	Description
`XVEL`	x-velocity
`YVEL`	y-velocity
`ZVEL`	z-velocity
`VOLUME`	Element volume
`MASS`	Mass of material 1
`DENSITY`	Density of material 1
`SIE`	Specific internal energy of material 1
`PRESSURE`	Pressure
`Q`	Artificial viscosity
`ENERGY`	Total energy (kinetic + internal) of material 1
`DIV`	Divergence
`VOID`	Void fraction
`FMAT`	Material fraction of material 1
`SSPD`	Speed of sound
`FBURN`	Burn fraction
`TDET`	Detonation time
`BFTIME`	Burn time

Keyword	Description
DEFMAT	Indicates the region of material that can burn. Only used for EOSDEF
DEFBURN	Burn fraction for EOSDEF
AFTERBURN	After burn fraction. Used only for EOSJWL
XMOM	x-momentum
YMOM	y-momentum
ZMOM	z-momentum
FVUNC	Volume uncovered fraction
QDIS	Characteristic element length
SXX	xx-deviatoric stress
SYY	yy-deviatoric stress
SZZ	zz-deviatoric stress
TXX	xx-stress
TYY	yy-stress
TZZ	zz-stress
TXY	xy-stress
TYZ	yz-stress
TZX	zx-stress
EFFSTS	Effective stress
DUDX	x-velocity gradient in x-direction
DUDY	x-velocity gradient in y-direction
DUDZ	x-velocity gradient in z-direction
DVDX	y-velocity gradient in x-direction
DVDY	y-velocity gradient in y-direction
DVDZ	y-velocity gradient in z-direction
DWDX	z-velocity gradient in x-direction
DWDY	z-velocity gradient in y-direction
DWDZ	z-velocity gradient in z-direction
FVUOLD	Old volume uncovered fraction
EXUSER1	User variable 1
EXUSER2	User variable 2
MFLR	Total mass-flow rate (sum of MFLR-POR, MFLR-PER and MFLR-INF)
MFL	Total mass-flow (sum of MFLR-POR, MFLR-PER and MFLR-INF)

Keyword	Description
MFLR-POR	Mass-flow rate by PORFGBG (flow between the Euler element and GBAGs through holes) or PORHOLE (flow between the Euler element and the environment through holes).
MFL-POR	Mass-flow by PORFGBG (flow between the Euler element and GBAGs through holes) or PORHOLE (flow between the Euler element and the environment through holes).
MFLR-PER	Mass-flow rate by PERMGBG (flow between the Euler element and GBAGs through permeable (SUBSURF) (SURFACEs) or PERMEAB (flow between the Euler element and the environment through permeable (SUBSURF) (SURFACEs).
MFL-PER	Mass-flow by PERMGBG (flow between the Euler element and GBAGs through permeable (SUBSURF) (SURFACEs) or PERMEAB (flow between the Euler element and the environment through permeable (SUBSURF) (SURFACEs).
MFLR-INF	Mass-flow rate by inflators (inflow by INFLATRs and EXFLOW,INFLATR3).
MFL-INF	Mass-flow by inflators (inflow by INFLATRs and EXFLOW,INFLATR3).
MFLR-GBG	Massflow rate by PORFGBG or PORFLGBG (flow between GBAGs).
MFL-GBG	Massflow by PORFGBG or PORFLGBG (flow between GBAGs).
MFLR-CPL	Massflow rate by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFL-CPL	Massflow by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFLR-FAIL	Massflow rate by COUP1INT through failed segments
MFL-FAIL	Massflow by COUP1INT through failed segments
DQDT	Total heat transfer rate (sum of DQDT-CNV and DQDT-RAD).
DQ	Total heat transfer (sum of DQ-CNV and DQ-RAD).
DQDT-CNV	Heat transfer rate by HTRCONV (transfer from the Euler element to the environment through (SUBSURF) (SURFACEs))
DQ-CNV	Heat transfer by HTRCONV (transfer from the Euler element to the environment through (SUBSURF) (SURFACEs).
DQDT-RAD	Heat transfer rate by HTRRAD (transfer from the Euler element to environment through (SUBSURF) (SURFACEs).
DQ-RAD	Heat transfer by HTRRAD (transfer from the Euler element to the environment through (SUBSURF) (SURFACEs).

Remarks

1. All variables starting with MFL (MassFLow) have following sign conventions:

   MFLxx > 0 means inflow of mass in the Euler element.
   MFLxx < 0 means outflow of mass from the Euler element.

2. All variables starting with DQ have following sign conventions:

   DQxx > 0 means inflow of heat in the Euler element.
   DQxx < 0 means outflow of heat from the Euler element.

Eulerian Solid Elements — with Shear Strength

Keyword	Description
XVEL	x-velocity
YVEL	y-velocity
ZVEL	z-velocity
TXX	xx-stress (basic coordinate system)
TYY	yy-stress (basic coordinate system)
TZZ	zz-stress (basic coordinate system)
TXY	xy-stress (basic coordinate system)
TYZ	yz-stress (basic coordinate system)
TZX	zx-stress (basic coordinate system)
EFFSTS	Effective stress
PRESSURE	Pressure
SXX	xx-deviatoric stress (basic coordinate system)
SYY	yy-deviatoric stress (basic coordinate system)
SZZ	zz-deviatoric stress (basic coordinate system)
SXY	xy-deviatoric stress (basic coordinate system)
SYZ	yz-deviatoric stress (basic coordinate system)
SZX	zx-deviatoric stress (basic coordinate system)
SINWX	x-stress rotation angle
SINWY	y-stress rotation angle
SINWZ	z-stress rotation angle
EFFPLS	Effective plastic strain
EPSXXD	xx-strain rate
EPSYYD	yy-strain rate
EPSZZD	zz-strain rate
EPSXYD	xy-strain rate
EPSYZD	yz-strain rate
EPSZXD	zx-strain rate
VOLUME	Element volume
MASS	Element mass
DENSITY	Element density
SIE	Specific internal energy

Keyword	Description
EDIS	Distortional energy
ENERGY	Total energy (internal + kinetic)
Q	Artificial viscosity
DIV	Divergence
VOID	Void fraction
FMAT	Material fraction
FMATPLT	Material fraction times the volume uncovered fraction
SSPD	Speed of sound
FBURN	Burn fraction
TDET	Detonation time
BFTIME	Burn time
XMOM	x-momentum
YMOM	y-momentum
ZMOM	z-momentum
FVUNC	Volume uncovered fraction
QDIS	Characteristic element length
FVUOLD	Old uncovered fraction
EXUSER1	User variable 1
EXUSER2	User variable 2
VOLPLS	Volumetric plastic strain; used for snow model
SOFTE	Softening; used for snow model

Eulerian Solid Elements — Multimaterial Hydrodynamic

Keyword	Description
XVEL	x-velocity
YVEL	y-velocity
ZVEL	z-velocity
VOLUME	Element volume
MASS	Mass of material 1
DENSITY	Density of material 1
SIE	Specific internal energy
PRESSURE	Pressure

Keyword	Description
Q	Artificial viscosity
ENERGY	Total energy (kinetic + internal) of material 1
DIV	Divergence of material 1
VOID	Void fraction
FMAT	Material fraction of material 1
FMATPLT	Material fraction times the volume uncovered fraction
TEMPTURE	Temperature
SSPD	Speed of sound
FBURN	Burn fraction
TDET	Detonation time
BFTIME	Burn time
XMOM	x-momentum
YMOM	y-momentum
ZMOM	z-momentum
FVUNC	Volume uncovered fraction
QDIS	Characteristic element length
FVUOLD	Old volume uncovered fraction
MASST	Total mass of element
EXUSER1	User variable 1
EXUSER2	User variable 2
MFLR	Total mass-flow rate (sum of MFLR-POR, MFLR-PER and MFLR-INF)
MFL	Total mass-flow (sum of MFLR-POR, MFLR-PER and MFLR-INF)
MFLR-POR	Mass-flow rate by PORFGBG (flow between the Euler element and GBAGs through holes) or PORHOLE (flow between the Euler element and the environment through holes).
MFL-POR	Mass-flow by PORFGBG (flow between the Euler element and GBAGs through holes) or PORHOLE (flow between the Euler element and the environment through holes).
MFLR-INF	Mass-flow rate by inflators (inflow by INFLATRs and EXFLOW,INFLATR3).
MFL-INF	Mass-flow by inflators (inflow by INFLATRs and EXFLOW,INFLATR3).
MFLR-CPL	Massflow rate by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFL-CPL	Massflow by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFLR-FAIL	Massflow rate by COUP1INT through failed segments
MFL-FAIL	Massflow by COUP1INT through failed segments

Keyword	Description
HMAT	Internal material number with the largest volume fraction
`VOLPLS`	Volumetric plastic strain; used for snow model
`SOFTE`	Softening; used for snow model
`DAMAGE`	Damage; used for `FAILJC`

The different materials in an element are in pressure equilibrium, but they have their own density, specific internal energy, and material volume fraction. The variables stored in element data storage relate to the material in the element that is stored as the first one internally. Multi-material editing can be performed for a material with material user number xx using the following variables:

Keyword	Description
`MASSxx`	Mass of material xx.
`DENSITYxx`	Density of material xx.
`SIExx`	Specific internal energy of material xx.
`ENERGYxx`	Total energy of material xx.
`DIVxx`	Divergence of material xx.
`FMATxx`	Volume fraction of material xx.
`TEMPTURExx`	Temperature of Materials xx.
`SIE-Exx`	Specific internal energy per unit mass of unreacted explosive part for material xx.
`SIE-Pxx`	Specific internal energy per unit mass of reaction products part for material xx.
`IGBURNxx`	Burn fraction for material xx.
`FMAT-Pxx`	Volume fraction of reaction product for material xx.
`RHO-Exx`	Density of unreacted explosive part for material xx.
`RHO-Pxx`	Density of reaction products part for material xx.
`MASS-Exx`	Mass of unreacted explosive part for material xx.
`MASS-Pxx`	Mass of reaction products part for material xx.

Note: Leading zeros in material user numbers are not accepted as correct input.

The multi-material variables `TEMPTURExx ... MASS-Pxx` cannot be read with Patran version 2008r1 or earlier.

Eulerian Solid Elements — Multi-material with Shear Strength

Keyword	Description
XVEL	x-velocity
YVEL	y-velocity
ZVEL	z-velocity
VOLUME	Element volume
MASS	Mass of material 1
DENSITY	Density of material 1
SIE	Specific internal energy of material 1
PRESSURE	Pressure
Q	Artificial viscosity
ENERGY	Total energy (kinetic + internal) of material 1
DIV	Divergence
VOID	Void fraction
FMAT	Material fraction of material 1
FMATPLT	Material fraction times the volume uncovered fraction
FBURN	Burn fraction
TDET	Detonation time
BFTIME	Burn time
DEFMAT	Indicates the region of material that can burn. Only used for EOSDEF
DEFBURN	Burn fraction for EOSDEF
AFTERBURN	After burn fraction. Only used for EOSJWL
EPSXXD	xx-strain rate
EPSYYD	yy-strain rate
EPSZZD	zz-strain rate
EPSXYD	xy-strain rate
EPSYZD	yz-strain rate
EPSZXD	zx-strain rate
SINWX	x-stress rotation angle
SINWY	y-stress rotation angle
SINWZ	z-stress rotation angle
SXX	xx-deviatoric stress
SYY	yy-deviatoric stress

Keyword	Description
SZZ	zz-deviatoric stress
SXY	xy-deviatoric stress
SYZ	yz-deviatoric stress
SZX	zx-deviatoric stress
TXX	xx-stress
TYY	yy-stress
TZZ	zz-stress
TXY	xy-stress
TYZ	yz-stress
TZX	zx-stress
EFFPLS	Effective plastic strain
EFFSTS	Effective stress
EDIS	Distortional energy of material 1
VOLOLD	Old volume
EXUSER1	User variable1
EXUSER2	User variable2
TYY-VIS	Viscous yy-stress.
TZZ-VIS	Viscous zz-stress.
TXY-VIS	Viscous xy-stress.
TYZ-VIS	Viscous yz-stress.
TZX-VIS	Viscous zx-stress.
EFFSTS-VIS	Viscous effective stress.
DUDX	x-velocity gradient in x-direction
DUDY	x-velocity gradient in y-direction
DUDZ	y-velocity gradient in z-direction
DVDX	y-velocity gradient in x-direction.
DVDY	y-velocity gradient in y-direction.
DVDZ	y-velocity gradient in z-direction.
DWDX	z-velocity gradient in x-direction.
DWDY	z-velocity gradient in y-direction.
DWDZ	z-velocity gradient in z-direction.
MFLR	Total mass-flow rate (sum of MFLR-POR, MFLR-PER and MFLR-INF)
MFL	Total mass-flow (sum of MFLR-POR, MFLR-PER and MFLR-INF)

Keyword	Description
MFLR-POR	Mass-flow rate by PORFGBG (flow between the Euler element and GBAGs through holes) or PORHOLE (flow between the Euler element and the environment through holes).
MFL-POR	Mass-flow by PORFGBG (flow between the Euler element and GBAGs through holes) or PORHOLE (flow between the Euler element and the environment through holes).
MFLR-INF	Mass-flow rate by inflators (inflow by INFLATRs and EXFLOW,INFLATR3).
MFL-INF	Mass-flow by inflators (inflow by INFLATRs and EXFLOW,INFLATR3).
MFLR-CPL	Massflow rate by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFL-CPL	Massflow by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFLR-FAIL	Massflow rate by COUP1INT through failed segments
MFL-FAIL	Massflow by COUP1INT through failed segments
HMAT	Internal material number with the largest volume fraction
VOLPLS	Volumetric plastic strain; used for snow model
SOFTE	Softening; used for snow model
DAMAGE	Damage; used for FAILJC

Eulerian Solid Elements — IG Explosive Materials (EOSIG) Only

Keyword	Description
SIE-E	Specific internal energy per unit of mass of un-reacted explosive part
SIE-P	Specific internal energy per unit of mass of reaction products part
IGBURN	Burn fraction
FMAT-P	Volume fraction of reaction product
RHO-E	Density of un-reacted explosive part
RHO-P	Density of reaction products part
MASS-E	Mass of un-reacted explosive part
MASS-P	Mass of reaction products part

The different materials in an element are in pressure equilibrium, but they have their own density, specific internal energy, and material volume fraction. The variables stored in element data storage relate to the material in the element that is stored as the first one internally. Multi-material editing can be performed for a material with material user number xx using the following variables:

Keyword	Description
`MASSxx`	Mass of material xx.
`DENSITYxx`	Density of material xx.
`SIExx`	Specific internal energy of material xx.
`ENERGYxx`	Total energy of material xx.
`DIVxx`	Divergence of material xx.
`TEMPTURExx`	Temperature of Materials xx.
`EIDSxx`	Distortional energy material xx.
`DAMAGExx`	Damage fraction material xx.
`SOFTExx`	Softening material xx.
`VOLPSxx`	Specific internal energy per unit mass of unreacted explosive part for material xx.
`SIE-Exx`	Specific internal energy per unit mass of unreacted explosive part for material xx.
`SIE-Pxx`	Specific internal energy per unit mass of reaction products part for material xx.
`IGBURNxx`	Burn fraction for material xx.
`FMAT-Pxx`	Volume fraction of reaction product for material xx.
`RHO-Exx`	Density of unreacted explosive part for material xx.
`RHO-Pxx`	Density of reaction products part for material xx.
`MASS-Exx`	Mass of unreacted explosive part for material xx.
`MASS-Pxx`	Mass of reaction products part for material xx.
`FLMATxx`	Volume fraction of material xx.
`DEFBURNXX`	Burn fraction for EOSDEF of material xx
`AFTERBURNXX`	After burn fraction. Only used for EOSJWL of material xx

Note: Leading zeros in material user numbers are not accepted as correct input.

The multi-material variables `TEMPTURExx ... MASS-Pxx` cannot be read with Patran version 2008r1 or earlier.

MATOUT — Material Results

Keyword	Description
XMOM	x-component of momentum
YMOM	y-component of momentum
ZMOM	z-component of momentum
EKIN	Kinetic energy
EINT	Internal energy
EDIS	Distortional energy
VOLUME	Volume
MASS	Mass
EHRG	Hourglass energy
ETOT	Total energy = Kinetic energy + Internal energy + Hourglass energy

EBDOUT —Eulerian Boundary Results

Keyword	Description
XMOM	x-component of transported momentum
YMOM	y-component of transported momentum
ZMOM	z-component of transported momentum
FX	x-component of force acting on boundary
FY	y-component of force acting on boundary
FZ	z-component of force acting on boundary
MFL	Total mass flow through boundary
MFLR	Mass flow rate through boundary
ENERGY	Energy transport through boundary
XVEL	x-component of velocity through boundary
YVEL	y-component of velocity through boundary
ZVEL	z-component of velocity through boundary
XIMP	x-component of impulse acting on boundary
YIMP	y-component of impulse acting on boundary
ZIMP	z-component of impulse acting on boundary

RBOUT — Rigid Body Results

Keyword	Description
XCG	x-coordinate of center of gravity
YCG	y-coordinate of center of gravity
ZCG	z-coordinate of center of gravity
INERTIA1	First principal moment of inertia
INERTIA2	Second principal moment of inertia
INERTIA3	Third principal moment of inertia
A11	x-component of the first principal axis
A21	y-component of the first principal axis
A31	z-component of the first principal axis
A12	x-component of the second principal axis
A22	y-component of the second principal axis
A32	z-component of the second principal axis
A13	x-component of the third principal axis
A23	y-component of the third principal axis
A33	z-component of the third principal axis
XVEL	x-translational velocity
YVEL	y-translational velocity
ZVEL	z-translational velocity
XAVEL	x-angular velocity
YAVEL	y-angular velocity
ZAVEL	z-angular velocity
XLAVEL	x-angular velocity in the local coordinate system
YLAVEL	y-angular velocity in the local coordinate system
ZLAVEL	z-angular velocity in the local coordinate system
XFORCE	x-force
YFORCE	y-force
ZFORCE	z-force
XMOMENT	x-moment
YMOMENT	y-moment
ZMOMENT	z-moment
XACC	x-translational acceleration

Keyword	Description
YACC	y-translational acceleration
ZACC	z-translational acceleration
RCG	Resultant coordinate of center of gravity
RVEL	Resultant translational velocity
RACC	Resultant acceleration
RFORCE	Resultant force = external + internal
RAVEL	Resultant angular velocity
RMOMENT	Resultant moment = external + internal
RLAVEl	Resultant angular velocity in the local coordinate system

Remark

1. All variables starting with `R` (Resultant) are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}$$

RELOUT — Rigid Ellipsoid Results

Keyword	Description
bordered	The ellipsoid is covered with dummy elements, which are written to an archive file so the ellipsoid can be visualized when postprocessing.
XIMP	x-impulse constraint
YIMP	y-impulse constraint
ZIMP	z-impulse constraint
WORK	Work done by constraint
XPULSE	x-pulse acting on ellipsoid
YPULSE	y-pulse acting on ellipsoid
ZPULSE	z-pulse acting on ellipsoid
XPUMOM	x-pulse moment acting on ellipsoid
YPUMOM	y-pulse moment acting on ellipsoid
ZPUMOM	z-pulse moment acting on ellipsoid
XCGEO	x-coordinate of the geometrical center
YCGEO	y-coordinate of the geometrical center
ZCGEO	z-coordinate of the geometrical center
A11	x-component of the first principal axis (a-axis)

Keyword	Description
A21	y-component of the first principal axis (a-axis)
A31	z-component of the first principal axis (a-axis)
A12	x-component of the second principal axis (b-axis)
A22	y-component of the second principal axis (b-axis)
A32	z-component of the second principal axis (b-axis)
A13	x-component of the third principal axis (c-axis)
A23	y-component of the third principal axis (c-axis)
A33	z-component of the third principal axis (c-axis)
INERTIA1	First principal moment of inertia
INERTIA2	Second principal moment of inertia
INERTIA3	Third principal moment of inertia
MASS	Mass
XVEL	x-component of translational velocity of geometrical center
YVEL	y-component of translational velocity of geometrical center
ZVEL	z-component of translational velocity of geometrical center
OMEGAA	x-component of angular velocity in the local coordinate system (around a-axis)
OMEGAB	y-component of angular velocity in the local coordinate system (around b-axis)
OMEGAC	z-component of angular velocity in the local coordinate system (around c-axis)
A	Half length of the local x-axis
B	Half length of the local y-axis
C	Half length of the local z-axis
CI	“Stiffness” of ellipsoid, used in contact logic
TXPULS	Accumulated xpulse (used for coupling to the external program)
TYPULS	Accumulated ypulse (used for coupling to the external program)
TZPULS	Accumulated zpulse (used for coupling to the external program)
TXPMOM	Accumulated xpumom (used for coupling to the external program)
TYPMOM	Accumulated ypumom (used for coupling to the external program)
TZPMOM	Accumulated zpumom (used for coupling to the external program)
XAVEL	x-component of angular velocity in the global coordinate system
YAVEL	y-component of angular velocity in the global coordinate system
ZAVEL	z-component of angular velocity in the global coordinate system
XCG	x-coordinate of the center of gravity
YCG	y-coordinate of the center of gravity

Keyword	Description
ZCG	z-coordinate of the center of gravity.
XACC	x-translational acceleration.
YACC	y-translational acceleration.
ZACC	z-translational acceleration.
RCG	Resultant coordinate of center of gravity.
RVEL	Resultant translational velocity.
RACC	Resultant acceleration.
RAVEL	Resultant angular velocity in the global coordinate system.
OMEGAR	Resultant angular velocity in the local coordinate system.
RIMP	Resultant impulse constraint.
RPULSE	Resultant pulse acting on ellipsoid.
RPUMOM	Resultant pulse moment acting on ellipsoid.
RCGEO	Resultant coordinate of the geometrical center.

Remark

1. All variables starting with `R` (Resultant), including `OMEGAR`, are calculated as follows:

$$R_{xx} = \sqrt{X_{xx}^2 + Y_{xx}^2 + Z_{xx}^2}\,.$$

GBAGOUT — Gas Bag Results

Keyword	Description
CDEX	Discharge coefficient for the exhaust openings
AEX	Total area of the exhaust openings
PEX	Pressure level above, which the flow out of the air bag through the holes starts
CDLEAK	Discharge coefficient for the permeability of the bag
ALEAK	Effective leak area
PSTOP	Pressure level below, which the flow out of the gas bag stops
PENV	Environmental pressure
RGAS	Specific gas constant
FLGAS	Inflow of gas from inflator
TGAS	Temperature of the inflowing gas
CPGAS	Specific heat capacity of the gas at a constant pressure
VOLUME	Volume inside the gas bag

Keyword	Description
VOLY	Volume inside the gas bag, calculated by the y-plane projection
VOLZ	Volume inside the gas bag, calculated by the z-plane projection
VFLUX	Rate of change of the gas-bag volume
TEMP	Temperature inside the gas bag
TFLUX	Rate of change of temperature inside the gas bag
PRESSURE	Pressure of the gas bag during the previous steps
VOLPREV	Volume at previous time step
MASS	Mass inside the gas bag
MFLR	Total mass-flow rate (sum of MFLR-POR, MFLR-PER, and MFLR-INF)
MFL	Total mass flow (sum of MFLR-POR, MFLR-PER, and MFLR-INF)
MFLR-POR	Mass-flow rate by PORFGBG (flow between the GBAG and another GBAG through holes) or PORHOLE (flow between the GBAG the environment through holes)
MFL-POR	Mass flow by PORFGBG (flow between the GBAG and another GBAG through holes) or PORHOLE (flow between the GBAG the environment through holes)
MFLR-PER	Mass-flow rate by PERMGBG (flow between the GBAG and another GBAG through permeable (SUBSURF) (SURFACEs) or PERMEAB (flow between the GBAG and the environment through permeable (SUBSURF) (SURFACEs)
MFL-PER	Mass flow by PERMGBG (flow between the GBAG and another GBAG through permeable (SUBSURF) (SURFACEs) or PERMEAB (flow between the GBAG and the environment through permeable (SUBSURF) (SURFACEs)
MFLR-INF	Mass-flow rate by inflators (inflow by INFLATRs and EXFLOW,INFLATR3)
MFL-INF	Mass flow by inflators (inflow by INFLATRs and EXFLOW,INFLATR3)
DQDT	Total heat transfer rate (sum of DQDT-CNV and DQDT-RAD)
DQ	Total heat transfer (sum of DQ-CNV and DQ-RAD)
DQDT-CNV	Heat transfer rate by HTRCONV (transfer from the GBAG to the environment through (SUBSURF) (SURFACEs)
DQ-CNV	Heat transfer by HTRCONV (transfer from the GBAG to the environment through (SUBSURF) (SURFACEs)
DQDT-RAD	Heat transfer rate by HTRRAD (transfer from the GBAG to environment through (SUBSURF) (SURFACEs)
DQ-RAD	Heat transfer by HTRRAD (transfer from the GBAG to the environment through (SUBSURF) (SURFACEs)
TEMPTURE	Temperature for fluid-filled container (only for FFCONTR).
VOLGAS	Volume of gas inside of a fluid-filled container (only for FFCONTR).

Keyword	Description
VOLFLUID	Volume of fluid inside a fluid-filled container (only for FFCONTR).
GAUGEPRES	Difference between pressure of the gas inside a fluid-filled container and ambient pressure (only for FFCONTR).
RHOFLUID	Density of fluid (only for FFCONTR).

Remarks

1. All variables starting with `MFL` (MassFLow) have following sign convention:

   `MFLxx > 0` means inflow of mass in the Euler element.
   `MFLxx < 0` means outflow of mass from the Euler element.

2. All variables starting with DQ have following sign convention:

   `DQxx > 0` means inflow of heat in the Euler element.
   `DQxx < 0` means outflow of heat from the Euler element

CONTOUT — Contact Surface Results

Keyword	Description
DMIN	Overall smallest distance between a primary contact face and a secondary contact node
XFORCE	x-component of contact force
YFORCE	y-component of contact force
ZFORCE	z-component of contact force
FMAGN	Magnitude of contact force
XACC	x-component of acceleration
YACC	y-component of acceleration
ZACC	z-component of acceleration
AMAGN	Magnitude of acceleration

The following variables are available as archive file data:

Keyword	Description
NFORCE	Normal component of contact force
SHFORCE	Shear component of contact force
FRFORCE	Friction component of contact force
PRESSURE	Pressure on the contact surface

Remarks

1. The contact surface data in archive format is saved to a new file for every time step that it is requested.
2. One archive file can contain data for one contact surface only. You can of course define multiple output definitions.
3. Output for contact surface data in archive format is available only when (a) you are using contact version V4, and (b) when both primary and secondary contacts have been defined as a surface.

CSOUT — Cross Section Results

Keyword	Description
XFORCE	x-component of cross-section force
YFORCE	y-component of cross-section force
ZFORCE	z-component of cross-section force
FMAGN	Magnitude of cross-section force

Cross sections are arbitrarily defined by lists of grid points and elements by the SECTION entry as explained in the *Dytran Reference Manual*. The output that can be requested for the cross section consists of the x-, y-, and z-components of the total force on the cross section. The force components are defined in the basic coordinate system. The total force (the magnitude of the force) is also available for output. See *Dytran Reference Manual*, Case Control Section CSOUT and CSECS entries for the definition.

The list of grid points is used to define the geometry of the cross section. The list of elements is used to define the orientation of the cross section. The element list may contain different element types. Each grid point must be attached to one of the elements. Cross-section data is available in the form of a time-history data.

CPLSOUT — Coupling-Surface Results

Coupling surface data can be written in archive and time-history data format. When you request an archive file for coupling surface output any Eulerian variable is available on the coupling surface output.

The output of a variable on a coupling-surface element is computed as the sum of the variable of the Eulerian elements intersected by the coupling surface segment, multiplied by the ratio of the intersection area and the coupling-surface segment area.

The following variables are available in time-history format on a coupling surface

Keyword	Description
XFORCE	x-component of coupling force
YFORCE	y-component of coupling force
ZFORCE	z-component of coupling force
RFORCE	Resultant coupling force

CPLSBOUT — Coupling Subsurface Results

Coupling subsurface data can be written in time history data format.

The following variables are available in time history format on a coupling subsurface

Keyword	Description
XFORCE	x-component of coupling force
YFORCE	y-component of coupling force
ZFORCE	z-component of coupling force
RFORCE	Resultant coupling force

SURFOUT — Surface Results

A SURFACE, referenced from a COUPLE entry, encloses a number of Eulerian elements completely or partly. The enclosed Euler elements have certain variables available for output (e.g., the material MASS, DENSITY, TEMPTURE, etc.). The sum, or the average of these values, is available as SURFACE output. The table below lists which variables are available as a sum or as an average.

This option is only available for the Single Material Hydrodynamic Euler solver and multi-material Euler solver (defined by a PEULER or PEULER1 entry with TYPE set to HYDRO, MMHYDRO, and MMSTREN).

For the multi-material Euler solver, multi-material variables can also be requested as SURFACE output. For example to monitor the mass of a specific material inside the coupling surface MASSxx can be used. Here xx denotes the material number. Likewise, the outflow of a specific material can be obtained by requesting MFLxx, MFL-CPLxx, MFL-PORxx, or MFL-FAILxx.

The same variables are made available as SURFACE output when the SURFACE is referenced from a GBAG entry. This makes the output transparent in case of air bag analyses where a switch is made from a Euler coupled model to a GBAG model during the calculation. When a variable is not used in the GBAG approach (e.g., XVEL, XMOM) its value is zero.

Apart from these variables related to Eulerian elements, also the total AREA of the SURFACE is available.

Furthermore, some variables that apply to the GBAG logic only (e.g., MFLR-GBG as described in Gas Bag Results (GBAGOUT) are available.

The table below lists which variables are available as a sum or as an average. Please refer to the following sections for a more detailed description of each variable.

ELOUT — Element Results and Eulerian Solid Elements (Hydrodynamic)

GBAGOUT — Gas Bag Results

Some variables of Eulerian elements are not listed here. They are not useful for output and, therefore, not listed.

Keyword	Description
AREA	Total Area of the SURFACE
VOLUME	Total Volume enclosed by the SURFACE
MASS	Sum of MASS in the intersected Euler elements or GBAG
ENERGY	Sum of ENERGY in the intersected Euler elements or GBAG
XMOM	Sum of XMOM in the intersected Euler elements or GBAG
YMOM	Sum of YMOM in the intersected Euler elements or GBAG
ZMOM	Sum of ZMOM in the intersected Euler elements or GBAG
MFLR	Sum of MFLR in the intersected Euler elements or GBAG
MFLR-INF	Sum of MFLR-INF in the intersected Euler elements or GBAG
MFLR-POR	Sum of MFLR-POR in the intersected Euler elements or GBAG
MFLR-PER	Sum of MFLR-PER in the intersected Euler elements or GBAG
MFL	Sum of MF in the intersected Euler elements or GBAG
MFL-INF	Sum of MFL-INF in the intersected Euler elements or GBAG
MFL-POR	Sum of MFL-POR in the intersected Euler elements or GBAG
MFL-PER	Sum of MFL-PER in the intersected Euler elements or GBAG
MFLR-GBG	Sum of MFLR-GBG in the GBAG
MFL-GBG	Sum of MFL-GBG in the GBAG
MFLR-CPL	Massflow rate by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFL-CPL	Massflow by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFLR-FAIL	Massflow rate by COUP1INT through failed segments
MFL-FAIL	Massflow by COUP1INT through failed segments
DQDT	Sum of DQDT in the intersected Euler elements or GBAG
DQDT-CNV	Sum of DQDT-CNV in the intersected Euler elements or GBAG
DQDT-RAD	Sum of DQDT-CNV in the intersected Euler elements or GBAG
DQ	Sum of DQ in the intersected Euler elements or GBAG
DQ-CNV	Sum of DQ-CNV in the intersected Euler elements or GBAG
DQ-RAD	Sum of DQ-RAD in the intersected Euler elements or GBAG
DENSITY	Average value of DENSITY in the intersected Euler elements or GBAG
PRESSURE	Average value of PRESSURE in the intersected Euler elements or GBAG
TEMPTURE	Average value of TEMPTURE in the intersected Euler elements or GBAG
SIE	Average value of SIE in the intersected Euler elements or GBAG
Q	Average value of Q in the intersected Euler elements or GBAG

Keyword	Description
DIV	Average value of DIV in the intersected Euler elements or GBAG
FMAT	Average value of FMAT in the intersected Euler elements or GBAG
SSPD	Average value of SSPD in the intersected Euler elements or GBAG
FVUNC	Average value of FVUNC in the intersected Euler elements or GBAG
QDIS	Average value of QDIS in the intersected Euler elements or GBAG
XVEL	Average value of XVEL in the intersected Euler elements or GBAG
YVEL	Average value of YVEL in the intersected Euler elements or GBAG
ZVEL	Average value of ZVEL in the intersected Euler elements or GBAG

Remarks

1. All variables starting with MFL (MassFLow) have the following sign conventions:

   MFLxx > 0 means inflow of mass in the Euler element.
   MFLxx < 0 means outflow of mass from the Euler element.

2. All variables starting with DQ have the following sign conventions:

   DQxx > 0 means inflow of heat in the Euler element.
   DQxx < 0 means outflow of heat from the Euler element.

SUBSOUT — Subsurface Results

A SUBSURF that belongs to a SURFACE referenced from a COUPLE entry intersects a number of Eulerian elements. The intersected Euler elements have certain variables available for output (e.g., the material MASS, DENSITY, TEMPTURE, etc.). The sum or the average of these values is available as SUBSURF output. The table below lists which variables are summed up and which variables are averaged.

This option is only available for the Single Material Hydrodynamic Euler solver (defined by the PEULER entry with TYPE set to HYDRO).

The same variables are available as SUBSURF output when the SURFACE it belongs to is referenced from a GBAG entry. This makes the output transparent in case of air bag analyses where you switch from an Euler coupled model to a GBAG model during the calculation. When a variable is not used in the GBAG approach (e.g., XVEL, XMOM) its value is zero.

Apart from these variables related to the Eulerian elements, the total AREA of the SUBSURF is also available.

Furthermore, some variables that apply to the GBAG logic only (e.g., MFLR-GBG as described in Gas Bag Results (GBAGOUT) are available.

The table below lists which variables are summed up, and which variables are averaged. Please refer to the following sections for a more detailed description of each variable:

Element Results (ELOUT) and Eulerian Solid Elements (Hydrodynamic)

Gas Bag Results (GBAGOUT)

Some variables of the Eulerian elements are not listed here. These variables are not useful for output. Therefore, they are not listed.

Keyword	Description
AREA	Total Area of the SUBSURF
PRESDIFF	Average value of PRESSURE difference between intersected Euler elements or GBAG and environment
TEMPDIFF	Average value of TEMPERATURE difference between intersected Euler elements or GBAG and environment
VALPMB	Total PERMEABILITY value of subsurface (VOLUME/AREA/SECOND)
MASS	Sum of MASS in the intersected Euler elements or GBAG
ENERGY	Sum of ENERGY in the intersected Euler elements or GBAG
XMOM	Sum of XMOM in the intersected Euler elements or GBAG
YMOM	Sum of YMOM in the intersected Euler elements or GBAG
ZMOM	Sum of ZMOM in the intersected Euler elements or GBAG
MFLR	Sum of MFLR in the intersected Euler elements or GBAG
MFLR-INF	Sum of MFLR-INF in the intersected Euler elements or GBAG
MFLR-POR	Sum of MFLR-POR in the intersected Euler elements or GBAG
MFLR-PER	Sum of MFLR-PER in the intersected Euler elements or GBAG
MFL	Sum of MFL in the intersected Euler elements or GBAG
MFL-INF	Sum of MFL-INF in the intersected Euler elements or GBAG
MFL-POR	Sum of MFL-POR in the intersected Euler elements or GBAG
MFL-PER	Sum of MFL-PER in the intersected Euler elements or GBAG
MFLR-GBG	Sum of MFLR-GBG in the GBAG
MFL-GBG	Sum of MFL-GBG in the GBAG
MFLR-CPL	Massflow rate by PORFCPL and PORFLCPL (flow between coupling surfaces)
MFL-CPL	Massflow by PORFCPL and PORFLCPL (flow between coupling surfaces)
DQDT	Sum of DQDT in the intersected Euler elements or GBAG
DQDT-CNV	Sum of DQDT-CNV in the intersected Euler elements or GBAG
DQDT-RAD	Sum of DQDT-CNV in the intersected Euler elements or GBAG
DQ	Sum of DQ in the intersected Euler elements or GBAG
DQ-CNV	Sum of DQ-CNV in the intersected Euler elements or GBAG
DQ-RAD	Sum of DQ-RAD in the intersected Euler elements or GBAG
DENSITY	Average value of DENSITY in the intersected Euler elements or GBAG
PRESSURE	Average value of PRESSURE in the intersected Euler elements or GBAG

Keyword	Description
TEMPTURE	Average value of TEMPTURE in the intersected Euler elements or GBAG
SIE	Average value of SIE in the intersected Euler elements or GBAG
Q	Average value of Q in the intersected Euler elements or GBAG
DIV	Average value of DIV in the intersected Euler elements or GBAG
FMAT	Average value of FMAT in the intersected Euler elements or GBAG
SSPD	Average value of SSPD in the intersected Euler elements or GBAG
FVUNC	Average value of FVUNC in the intersected Euler elements or GBAG
QDIS	Average value of QDIS in the intersected Euler elements or GBAG
XVEL	Average value of XVEL in the intersected Euler elements or GBAG
YVEL	Average value of YVEL in the intersected Euler elements or GBAG
ZVEL	Average value of ZVEL in the intersected Euler elements or GBAG

Remarks

1. All variables starting with MFL (MassFLow) have the following sign conventions:

   MFLxx > 0 means inflow of mass in the Euler element.
   MFLxx < 0 means outflow of mass from the Euler element.

2. All variables starting with DQ have following sign conventions:

   DQxx > 0 means inflow of heat in the Euler element.
   DQxx < 0 means outflow of heat from the Euler element.

WALLOUT — Rigid Wall Results

Keyword	Description
XFORCE	x-component of force
YFORCE	y-component of force
ZFORCE	z-component of force
FMAGN	Magnitude of force
XACC	x-component of acceleration
YACC	y-component of acceleration
ZACC	z-component of acceleration
AMAGN	Magnitude of acceleration

Rigid wall data can only be written in the form of a time-history file.

Time History Output Using Markers

Markers enable time history output of Eulerian properties at a specific location. Markers can be stationary, moving along with the Eulerian material or Lagrangian grid points. For markers, any variable that is valid for Eulerian elements can be requested. Markers are defined by the CMARKN1 option. It fully supports multiple coupling surfaces. Markers can move from one Euler domain to another. In addition, multi-material variables can be requested as part of the Eulerian output of markers.

A CMARKN1 references a grid point that specifies the location. Time history output for the CMARKN1 is obtained from the Euler element that contains this grid point.

To define markers, the following steps must be taken:

1. Define a number of CMARKN1s. Each CMARKN1 refers to one grid point. If the CMARKN1 does not refer to an existing Lagrangian grid point, then a new grid point has to be defined.
2. Each CMARKN1 refers to a PMARKER property that specifies whether it is stationary or moving along with Eulerian material. Define the PMARKER properties that are referenced by the CMARKN1s. If the grid point refers to a Lagrangian grid point, the PMARKER ID is ignored.
3. Output request for marker results can be made by CMARKS and CMARKOUT. Output results refer to the grid points that are associated with the CMARKN1s.

The example below illustrates the use of CMARKN1:

```
CMARKN1,10,12,1001
CMARKN1,11,12,1002
CMARKN1,12,11,1003
CMARKN1,13,11,1004
GRID,1001,,0.1,0.57,0.3
GRID,1002,,0.1,0.517,0.3
GRID,1003,,0.1,0.57,0.3
GRID,1004,,0.1,0.517,0.3
PMARKER,12,FIXED
PMARKER,11,MOVING
```

With output request

```
CMARKS(MARKER) = 17
SET 17 = 10,11,12,13
CMARKOUT(MARKER) =TEMPTURE,XPOS,YPOS,ZPOS,XVEL,YVEL,ZVEL,
RVEL,XVELMARK,YVELMARK,ZVELMARK,
RVELMARK FMAT4 FMAT3
STEPS(MARKER) = 0 THRU END BY 1
TYPE(MARKER) = TIMEHIS
SAVE(MARKER) = 100000000
$
```

Only `TIMEHIS` output is supported and `ARCHIVE` output is not supported. The motion of moving marker grid points are available through `XPOS,..,ZPOS` and `XVELMARK,..,ZVELMARK`.

With Lagrange elements, it is easy to visualize deformation and material flow because grid points and elements move along with the material. This is less straightforward for Euler elements because they are stationary. To enable easy visualization of Eulerian flow, marker bars available. The two grid points of a marker bar move along with the Eulerian material and shows how material flows and deforms. These bars can be defined by the CMARKB2 option. They also refer to a PMARKER property, but they are only useful when this type is set to moving. For CMARKB2, archive and time history output is available.

An example is given below

```
PMARKER,11,MOVING
CMARKB2,22,11,1102,1112
CMARKB2,23,11,1103,1113
GRID,1102,,0.5,0.4,0.15
GRID,1112,,0.5,0.6,0.15
GRID,1103,,0.5,0.4,0.25
GRID,1113,,0.5,0.6,0.25
```

With output request

```
CMARKS(MARKERB) = 18
CMARKOUT(MARKERB) = XPOS,YPOS,ZPOS,XVEL,YVEL,ZVEL,PRESSURE,
                    RVEL,TEMPTURE,XVELMARK,YVELMARK,ZVELMARK,
                    RVELMARK,FMAT4,FMAT3
STEPS(MARKERB) = 0 THRU END BY 5
TYPE(MARKERB) = ARCHIVE
SAVE(MARKERB) = 100000000
SET 18= 22,23
```

Time History Output for Boxes

There are several possibilities for making time histories. One can make a material time history but that would be for all the material and can be too global. Another option would be to make time histories of certain elements. But this can be too local. To allow for time history that are neither too global nor too local, the MATBX entry can be used.

MATBX enables time history output for elements inside user defined boxes. These boxes are stationary. For **MATBX**, any variable that is valid for Eulerian elements can be requested. It fully supports multiple coupling surfaces. In addition multi-material variables can be requested as part of the Eulerian output of **MATBX**.

An example is given by

```
TYPE    (ARCMATBX) = TIMEHIS
MATBOUT (ARCMATBX)  = FMAT,XVEL,YVEL,ZVEL,MASS,DENSITY
MATBX   (ARCMATBX) = 60
SET 60 = 210,220
STEPS   (ARCMATBX) = 0,THRU,END,BY,1
SAVE    (ARCMATBX) = 99999

BOX,210,,0.0345,-1,-1,0.002,2,2
BOX,220,,0.1085,-1,-1,0.002,2,2
```

As result time histories are available for the two boxes between x=0.0345 and x=0.0365 and between x=0.1085 and x=1.105. For more details see example 3-7 in the Dytran example manual.

By choosing the size of the box sufficiently small MATBX can function as marker. For meshes allowing only general coupling, markers are not supported and **MATBX** provides a good alternative.

Restarts

A restart is used when the analysis is proceeding correctly and you want to run the next stage. You must ensure that restart files are available from the initial run. The following FMS statements and Case Control commands write restart data to a file (logical name RST1) at 0.5E–3 and 1.0E–3 seconds.

```
TYPE(RST1) = RESTART
CEND
TIMES(RST1) = 0.5E-3, 1.0E-3
```

It is not necessary to define which grid points and elements or which data is stored in a restart file. In the example above, all the restart information is stored in one file. The FMS statement SAVE can be used to create more than one file. The STEPS Case Control command can also be used to write restart data on a time-step basis. The following input writes restart data to a file (logical name `RST2`) every 100 time steps, and a new file is created each time data is written.

```
TYPE(RST2) = RESTART
SAVE(RST2) = 1
CEND
STEPS (RST2) = 0, THRU, END, BY, 100
```

On most UNIX systems a restart file will also be generated upon receipt of the `USRSIG2` signal to terminate the analysis, regardless of any restart output requests. This restart file contains restart information for the last complete analysis cycle. The filename of this restart file will contain the identifier `USR_RESTART` as well as the cycle number for which the restart information is written. For example, if an analysis with `job-id RUN1` was terminated at cycle 5131, the restart file would be `RUN1_USR_RESTART_5131.RST`.

Restarting a Previous Analysis

To restart an analysis, you need an input file that only contains the FMS, Executive, and Case Control Sections for the job and includes the FMS statement RESTART. Only those Case Control options that you want to change must be included. The Bulk Data Section must be empty.

The restart file from the previous analysis is specified using the FMS statement RSTFILE, and the step from which the analysis is to continue is specified using the RSTBEGIN statement. To continue an analysis to a new termination time, your input file is:

```
RESTART
RSTFILE = filename
RSTBEGIN = step number
CEND
ENDTIME = new finish time
BEGIN BULK
ENDDATA
```

For example,

```
RESTART
RSTFILE = RUN1_RST1_322.RST
RSTBEGIN = 636
CEND
ENDTIME = 2.0E-3
BEGIN BULK
ENDDATA
```

It is possible to remove certain types of elements from the calculation when restarting. The RSTDROP parameter allows all of the Eulerian elements, Lagrangian elements, and coupling surfaces to be removed from the calculation. All of the elements of that type are removed. For example, it is not possible to remove only a few Lagrangian or Eulerian elements. Otherwise, no data can be changed when restarting except the ENDTIME, ENDSTEP, `STEPMIN`, RSTDROP parameters and parameter options.

Prestress Analysis

A prestress analysis based on a Nastran solution can be performed as follows:

```
PRESTRESS
BULKOUT = file-name-1
SOLUOUT = file-name-2
NASDISP = file-name-3
CEND
BEGIN BULK
NASINIT, ..., ..., ...,
....
ENDDATA
```

Controlling the Analysis

Executing the transient dynamic analysis is very simple since most of the control is automatic. For most analyses, you need to do very little. The analysis is performed, and the time step is continually adjusted by Dytran to ensure a stable solution with a minimum use of computer resources. However, there are many ways to override the automatic control and manually select the parameters that control the analysis. This is done using Case Control commands or the PARAM entry. Details for each parameter that can be set using the PARAM entry are provided in Chapter 6: Parameters in the *Dytran Reference Manual*. A brief description is given below.

Modifying the Time Step

Perhaps the most critical factor affecting the solution is the time step. By default, the time step is calculated by Dytran so that it is smaller than the time taken for a stress wave to cross the smallest element. This ensures a stable solution. The time step is recalculated every iteration. The automatic time step works well for the vast majority of analyses, and you should change it only when you have a very good reason to do so.

The time step can be scaled using the STEPFCT parameter. The internally calculated time step is multiplied by the scale factor that you specify to get the time step actually used. The scale factor cannot be greater than 1.0, otherwise, the solution becomes unstable.

The initial time step must be specified using the parameter INISTEP. The specified time step is used for the first iteration; thereafter, the internally calculated time step is used.

The MAXSTEP parameter allows you to specify a maximum time step for the analysis. The time step cannot exceed this value.

The MINSTEP parameter lets you specify a minimum time step. When the calculated time step falls below this value, the analysis terminates.

Blending of Eulerian Elements

When using coupling surfaces, the surface may cut through Eulerian elements so that only a very small proportion of the element is uncovered. To prevent such elements from controlling the time step, the Eulerian elements are blended with adjacent elements. The FBLEND parameter allows you to specify the uncovered fraction at which elements are blended when they would otherwise control the time step.

Coupling Subcycling

Subcycling techniques are used to improve the efficiency of the coupling algorithms. The geometry of the coupling surface is only updated when required, which depends on the motion of the surface.

The COSUBMAX parameter lets you specify the maximum number of time steps before the geometry of the coupling surface is forced to be updated. The COSUBCYC parameter allows you to control the rate of growth of the subcycling interval. When either of these parameters is specified, the subcycling is activated.

Element Subcycling

The element subcycling algorithm partitions the elements into groups of equal time steps. Each group of elements is then updated with the group time step. Especially when a few small elements determine the time step, large CPU savings can be achieved. The element subcycling algorithm can be activated with the `ELSUBCYC` parameter.

Limits

Various limits can be set that affect the analysis. The RHOCUT parameter defines a minimum density for Eulerian elements. When the density of an element is less than the minimum density, the element is considered empty. Each of the Eulerian solvers has its own density cutoff. These can be defined separately, although in most cases, the automatic setting is sufficient. The VELCUT parameter sets a velocity cutoff. The VELMAX parameter allows you to specify a maximum velocity in Eulerian meshes. This can be useful for near-vacuous flows. Finally, the SNDLIM parameter specifies the minimum value for the speed of sound.

Terminating the Analysis

The analysis terminates when any of the following conditions occur.

Termination Time Reached

When the time reaches the time specified by the ENDTIME Case Control command, the analysis terminates.

Termination Step Reached

When the step number reaches the number specified by the ENDSTEP Case Control command, the analysis terminates.

Insufficient CPU Time

If the CPU time (in minutes) specified on the TIME statement in Executive Control is exceeded, the analysis terminates.

Time Step Too Small

If the time step falls below the value specified by the PARAM option MINSTEP, the analysis terminates.

User Signal

On most UNIX systems, the user can send a signal to force a wrap-up of the analysis. The analysis terminates with the normal output as requested at `END`. It is also possible to generate a restart file for the last analysis cycle regardless of any restart output requests. For more details, see Stopping Dytran on Linux .

Postprocessing

You can postprocess your results using Patran, Dytran Explorer or the postprocessor you normally use for standard finite element analyses. Essentially, the usual techniques can be used, but you may consider the following suggestions.

Plot the Time Variation of Results

Use the results from the time-history files to see how particular parameters vary during the analysis. Time variation plots can be used to identify the key times during the analysis that demand more detailed postprocessing.

Use Real Displacements

When plotting deformed shapes, set the magnification factor for the displacement to 1.0, so that real displacements, not magnified ones, are plotted. Since Dytran is a large displacement code, you can get some very odd-looking plots if you scale the deformations. In particular, contact surfaces may appear to be penetrated and parts of the mesh may seem to overlap.

Plot Contours on the Deformed Shape

Try to plot the contours of the results on the deformed shape for Lagrangian elements. This produces much more meaningful results when the deformations are large.

Plot Material Contours

The location of material within an Eulerian mesh can be determined by plotting contours of the material fraction (FMAT).

Plot the Effective Plastic Strain

Dytran outputs the effective plastic strain, a scalar measure of how much permanent deformation has occurred. The quantity is very useful for showing the amount of deformation in a component and the areas that have yielded.

Plot the Velocity Fields

Use the vector or arrow plots to view the velocity fields in the structure. Arrow plots give you a rapid indication of the way in which a structure is moving. Arrow plots are essential in understanding the flow characteristics in Eulerian meshes.

Animate the Analysis

Some postprocessing programs, such as Patran and Paraview, have the facility to animate the progression of the analysis. This can be very useful in obtaining an overall impression of what is happening during the analysis.

Dytran Utility Tools

Dytran distribution comes with a set of utility applications which are used to perform certain operations on Dytran output files.

ExtractSteps

The ExtractSteps tool extracts the result associated to a cycle(s) from an ARC file and dump it as a new ARC file. Usage:

Case 1: Extracting a list of cycle(s)

```
{Dytran_dir}\Dytran\{Dytran_year}\bin\exe\extractsteps.exe {FOOIN.ARC}
{FOOOUT.ARC} -l {n_i} {n_j} …
```

`FOOIN.ARC`	Input ARC file name
`FOOOUT.ARC`	Output ARC file name
`n_i, n_j, …`	Ascending sequence numbers on which the cycles appeared on the archive file, NOT the cycle numbers.

The bellow command extracts the 5th, 10th, and 15th cycles from `FOOIN.ARC` and dumps them into `FOOOUT.ARC`.

```
C:\Program Files\MSC.Software\Dytran\2021\bin\exe\extractsteps.exe FOOIN.ARC
FOOOUT.ARC -l 5 10 15
```

Case 2: Extracting a range of cycles

```
{Dytran_dir}\Dytran\{Dytran_year}\bin\exe\extractsteps.exe {FOOIN.ARC}
{FOOOUT.ARC} {n_first} {n_last}
```

`FOOIN.ARC`	Input ARC file name
`FOOOUT.ARC`	Output ARC file name
`n_first` and `n_last`	The sequence numbers specifying the beginning and the end of the range. `n_first` and `n_last` are sequence numbers on which the cycles appeared on the archive file, NOT the cycle numbers.

The below command extracts the 2nd to 10th cycles from `FOOIN.ARC` and dumps them into `FOOOUT.ARC`.

```
C:\Program Files\MSC.Software\Dytran\2021\bin\exe\extractsteps.exe FOOIN.ARC
FOOOUT.ARC 2 10
```

Tip:	If you do not know the sequence numbers of the cycles within the ARC file, you may run this program with very high `n_first` and `n_last` to get a log of cycle in the ARC file.

ARC2TXT

The ARC2TXT tool dumps the content of ARC files in the ASCII format. Usage:

1. Go to the directory where the ARC files are located.
2. Create a file named `Arc2Txt_inp.txt` with the list of ARC file(s) that need to be converted, each on a new line.
3. Run `Arc2Txt` tool without any arguments (it will automatically read `Arc2Txt_inp.txt` file and converts all the ARC files listed there)

4. The ASCII dumps are written to files with the same name as the original ARC files with appended `.2.txt`. So, `FOOIN.ARC` will be dumped into `FOOIN.ARC.2.txt`.

Note: Log information about the conversions is written to a file named `Arc2Txt_msg.txt`.

For example, in Linux:

```
$ ls
FOO_0.ARC
$ touch Arc2Txt_inp.txt
$ echo FOO_0.ARC >> Arc2Txt_inp.txt
$ \MSC.Software\Dytran\2021\bin\exe\Arc2Txt
$ ls
FOO_0.ARC.2.txt FOO_0.ARC Arc2Txt_msg.txt Arc2Txt_inp.txt
```

THS2TXT

The THS2TXT tools dumps the content of THS files in the ASCII format. Usage:

1. Go to the directory where the THS files are located.
2. Create a file named `Ths2Txt_inp.txt`. with the list of THS file(s) that need to be converted, each on a new line.
3. Run `Ths2Txt` tool without any arguments (it will automatically read `Ths2Txt_inp.txt` file and converts all the ARC files listed there)
4. The ASCII dumps are written to files with the same name as the original THS files with appended `.t2t`. So, `FOOIN.ARC` will be dumped into `FOOIN.THS.t2t`.

Note: Log information about the conversions is written to a file named `Ths2Txt_msg.txt`.

For example, in Linux:

```
$ ls
FOO_0.THS
$ touch Ths2Txt_inp.txt
$ echo FOO_0.THS >> Ths2Txt_inp.txt
$ \MSC.Software\Dytran\2021\bin\exe\Trc2Txt
$ ls
FOOIN.THS.t2t FOO_0.THS Ths2Txt_msg.txt Ths2Txt_inp.txt
```

Running Dytran Using Dytran Explorer

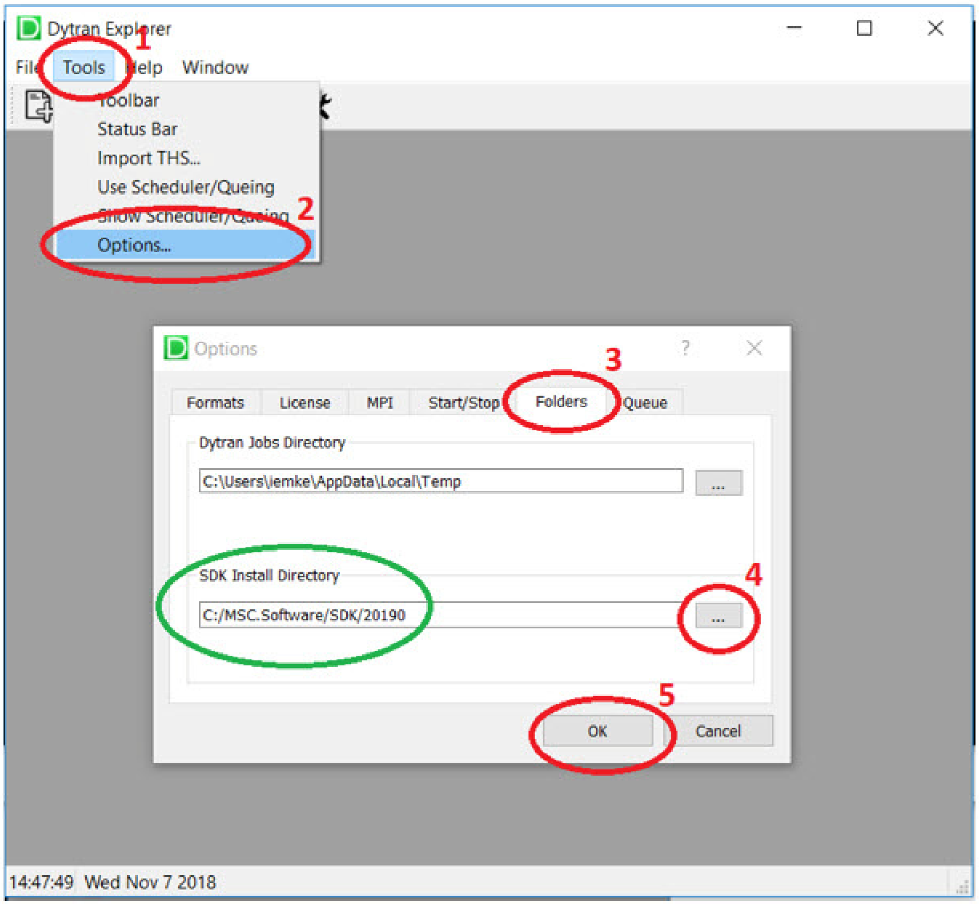
Dytran analyses can also be submitted using Dytran Explorer. Dytran Explorer offers an easy way to control your jobs, manage your data, and visualize time history data.

User Defined Services with DytranExplorer

Starting from version 2019 the "user-subroutines" of Dytran have been replaced by the SCA based User Defined Services (UDS) also in use by MSC Nastran. To be able to use UDS, also the "Software Development Kit" (SDK) needs to be installed on the system. The SDK can be obtained separately from the MSC Software download website.

To use UDS with DytranExplorer, a directory tree should be set up containing the (modified) files required by the specific service. The easiest way is to copy an example and make the modifications for the purpose the UDS is intended for. When this is done DytranExplorer can be used to build the modified UDS libraries and run Dytran.

First the location of the SDK needs to be defined, so DytranExplorer knows where to find it.



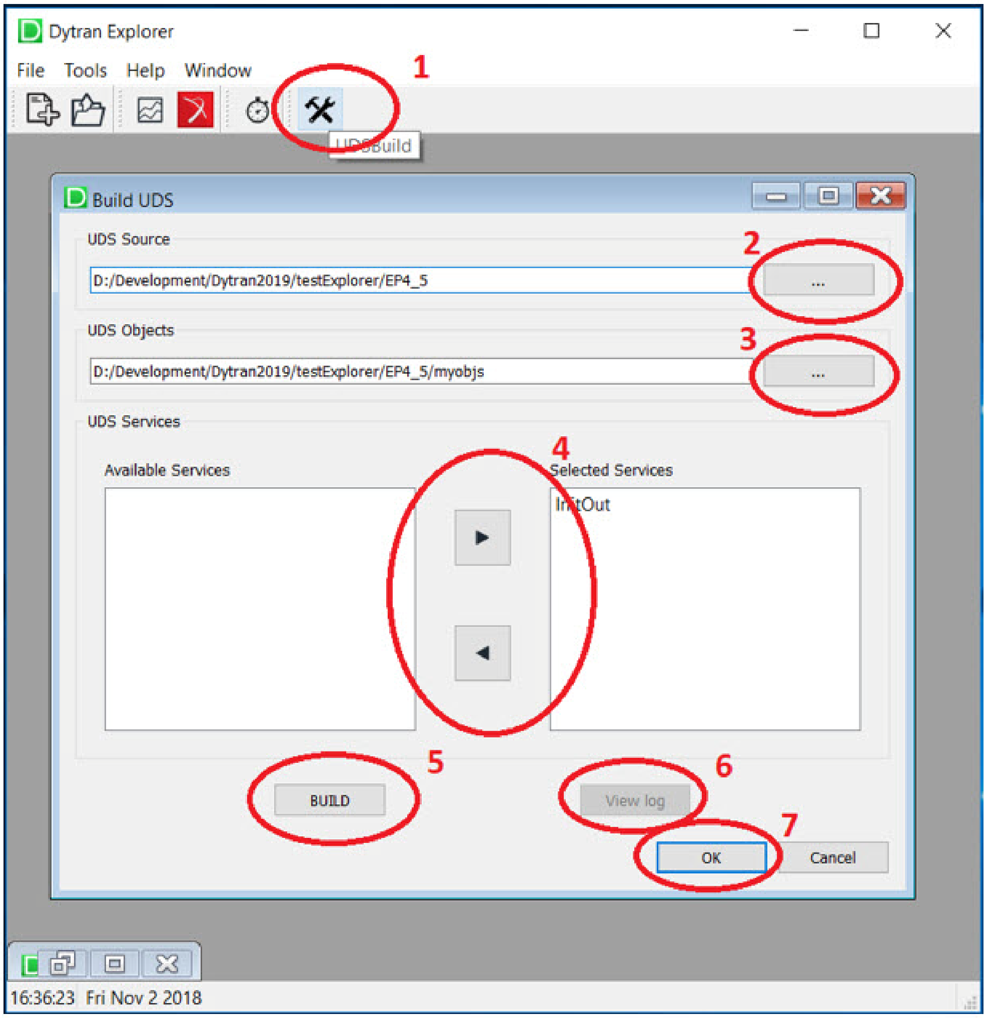
Dytran Jobs Directory: C:\Users\iemke\AppData\Local\Temp

SDK Install Directory: C:/MSC.Software/SDK/20190

1. Click **Tools** to open the Tools menu.
2. Click **Options** to open the Options window.
3. Select the **Folders** tab.

4. Use the directory browser to select the SDK installation directory
5. Click **OK** when done.

Next the UDS services need to be build.



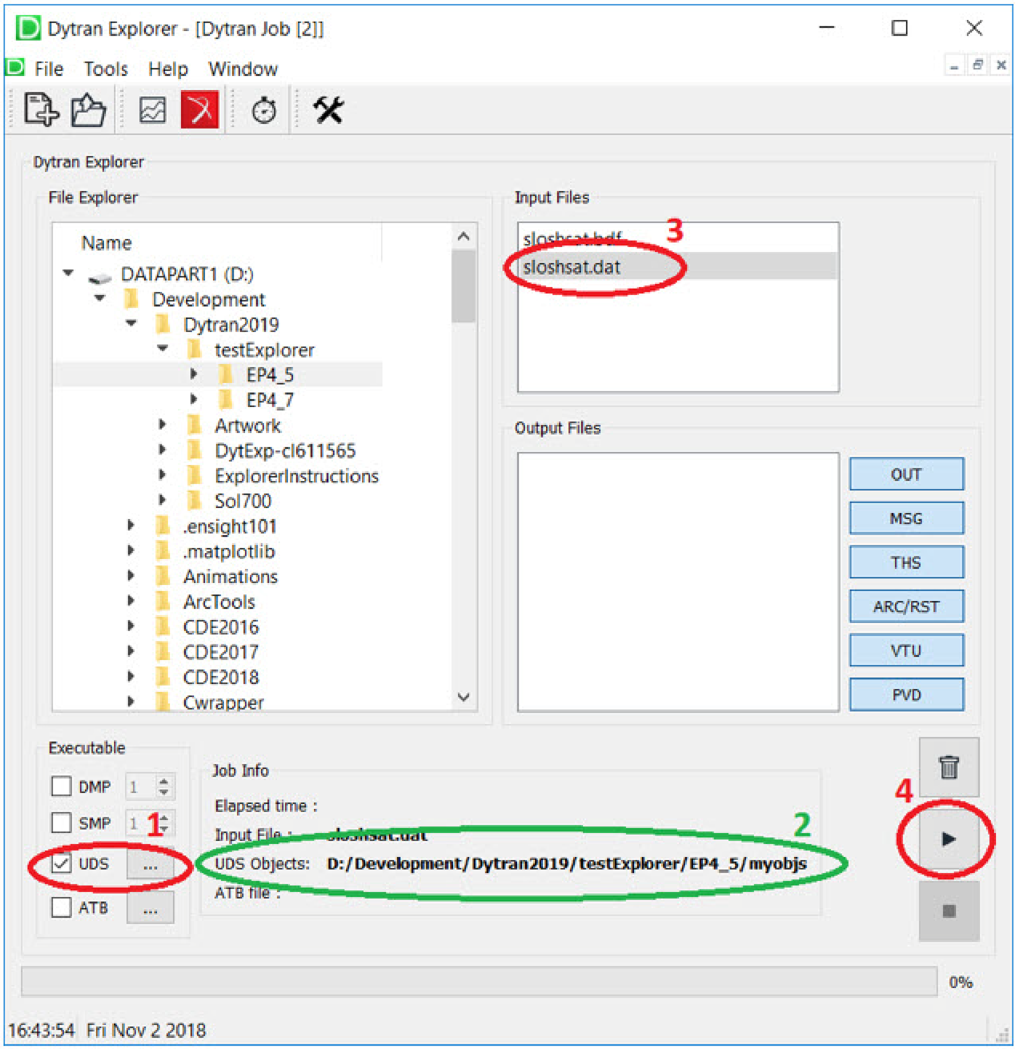
UDS Source: D:/Development/Dytran2019/testExplorer/EP4_5

UDS Objects: D:/Development/Dytran2019/testExplorer/EP4_5/myobjs

1. Open the **Build UDS** tool.
2. Select the root of the directory tree containing the UDS service to be build. This is the directory that contains the four SCons files: SConopts, SConopts.delivery, SConscript and SConstruct.
3. Select the root of the directory tree to receive the UDS objects. An empty directory is sufficient.
4. Select the services to be included in the build. All services found in the source tree will be listed. If there are any that should not be included in the build, they can be moved in or out the selection with the right and left arrow buttons.
5. Click **Build** to start the build. This may take some time.
6. Use **View Log** to inspect the log file. Check that it ends with “scons: done building targets” to make sure the build ended successfully.

7. Click **OK** when done.

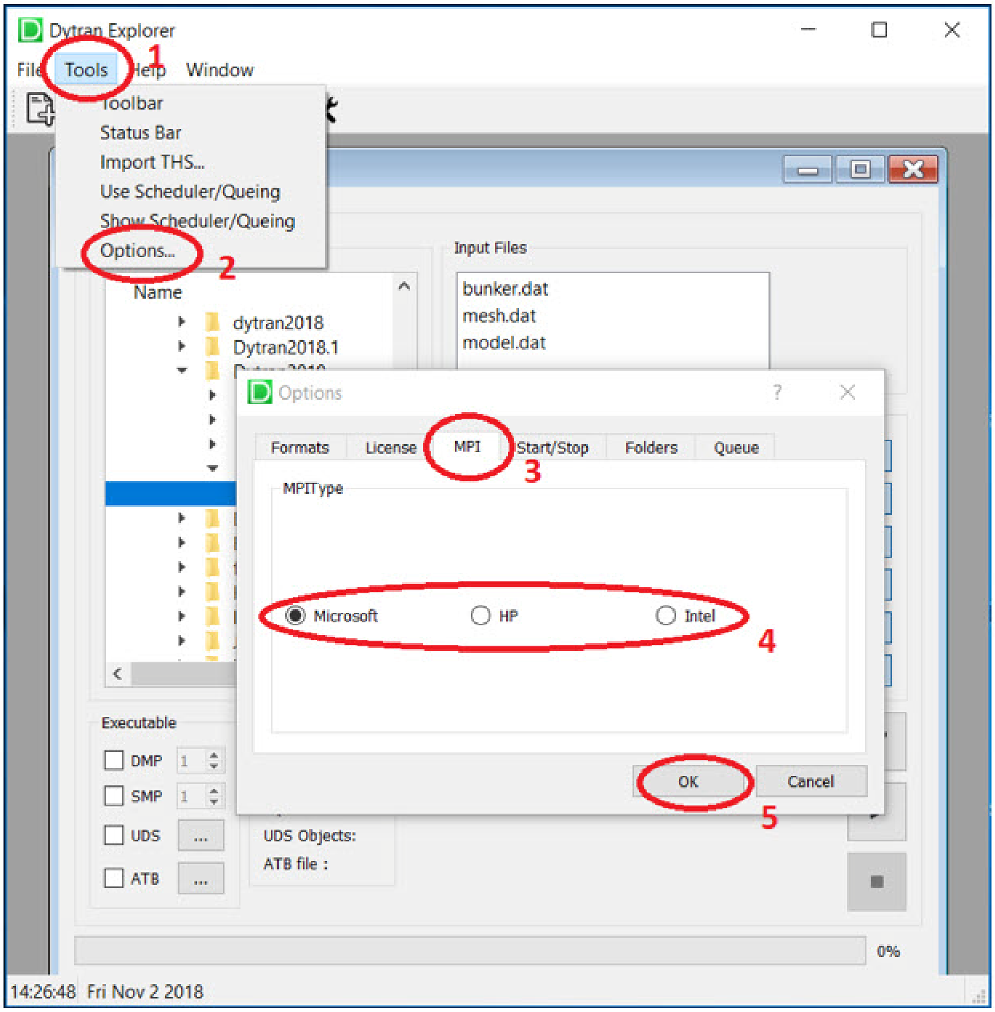
To start a run with UDS services applied:



1. Open a "New Dytran Job" window as usual.
2. Select the **UDS** check box and select the root of the UDS objects tree. This is the directory defined in Step (3) of the UDS Build. The location of this root is displayed in the **Job Info** pane.
3. Select the **Input File** as usual.
4. Click ▶ to the start run with UDS.

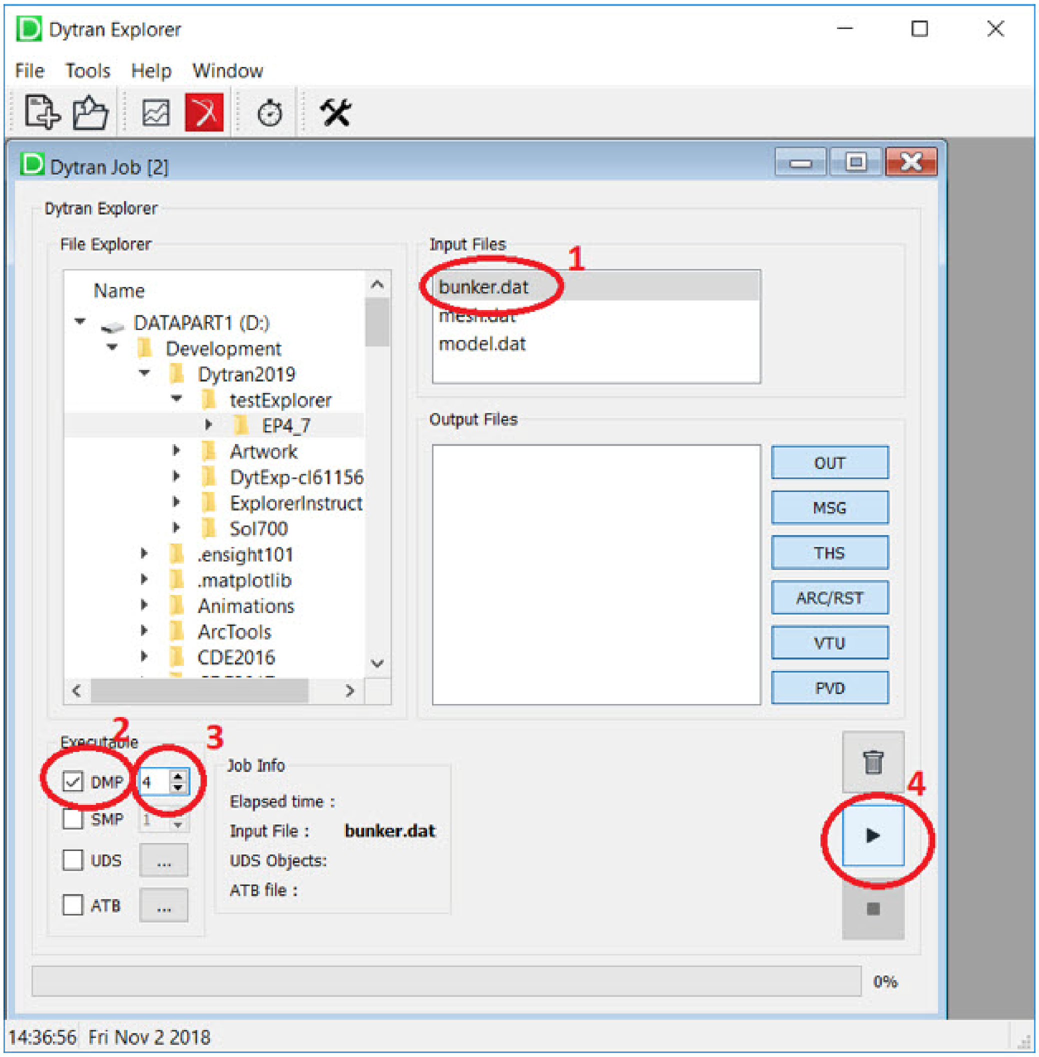
Running DMP and SMP Dytran jobs with DytranExplorer

For DMP runs, first the MPI versions needs to be selected with “Tools/Options/MPI”



1. Open the **Tools** menu.
2. Open the **Options** window.
3. Select the **MPI** tab.
4. Select the **MPI** type to be used.
5. Click **OK** when done.

To start a DMP or/and SMP run, open the "New Dytran Job" window as usual.

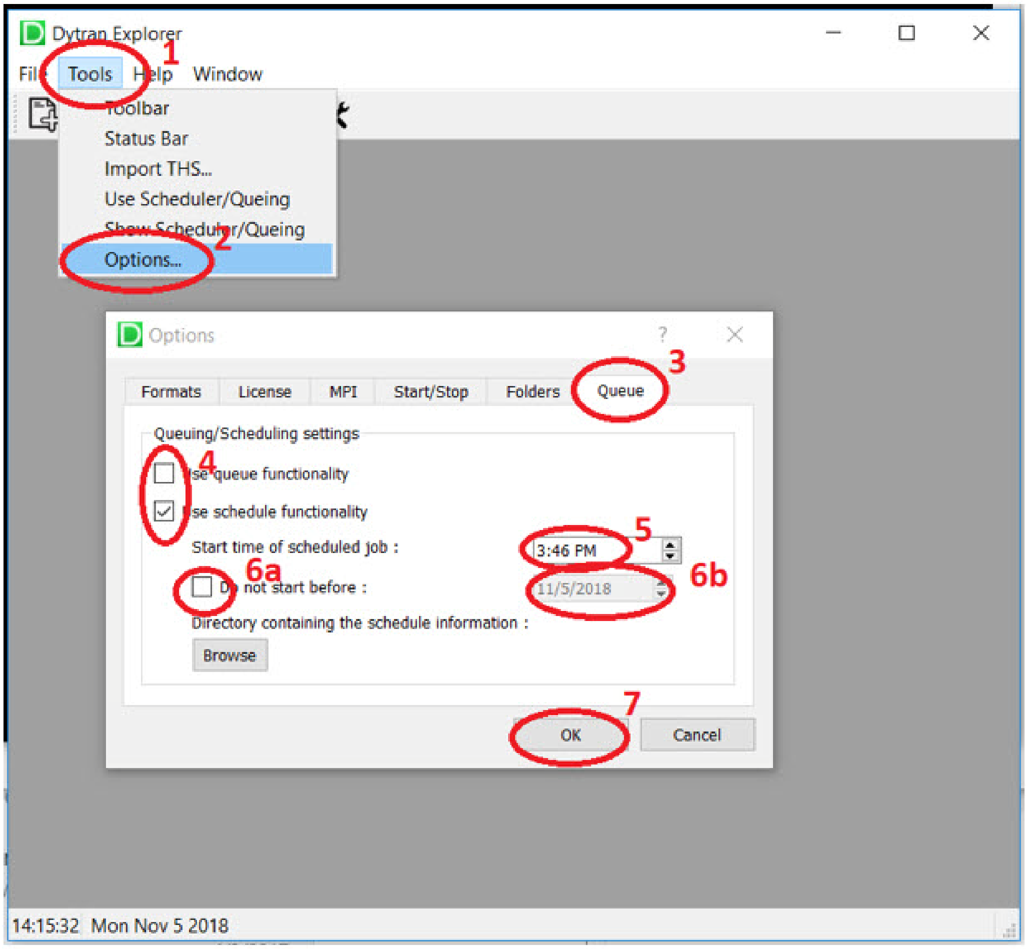


1. Select the **Input File**.
2. Select **DMP** check box.
3. Set the number of processors to be used
4. Start the job as usual.

For SMP jobs, step (2) and (3) should be applied for the SMP selector. Depending on the Dytran options used on the input deck, it is possible to apply DMP and SMP simultaneously.

Using Queue/Scheduler in DytranExplorer

DytranExplorer offers the option to run multiple jobs in a queue, or start jobs at a selected time. The Queue/Scheduler can be enabled with the tools/options menu:



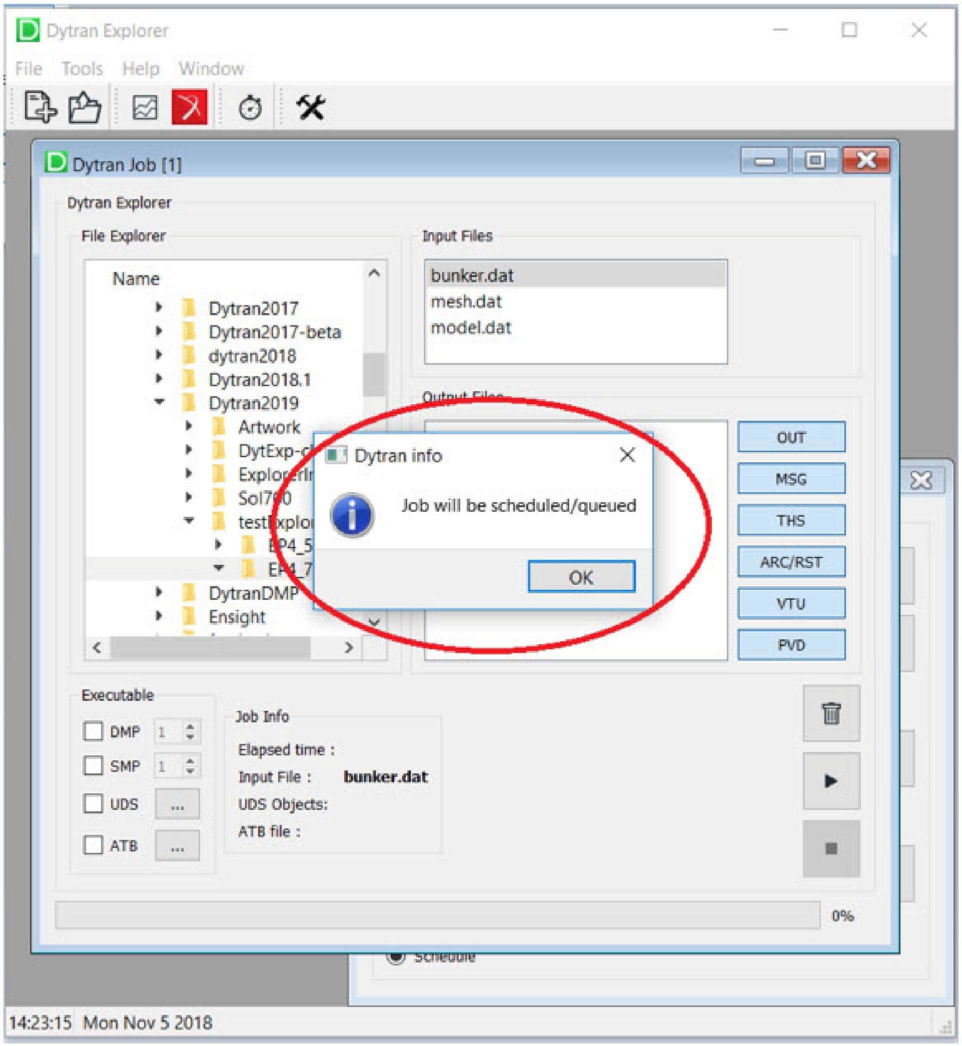
Queuing/Scheduling settings
☐ Use queue functionality
☑ Use schedule functionality
Start time of scheduled job : 3:46 PM
☐ Do not start before : 11/5/2018
Directory containing the schedule information :

14:15:32 Mon Nov 5 2018

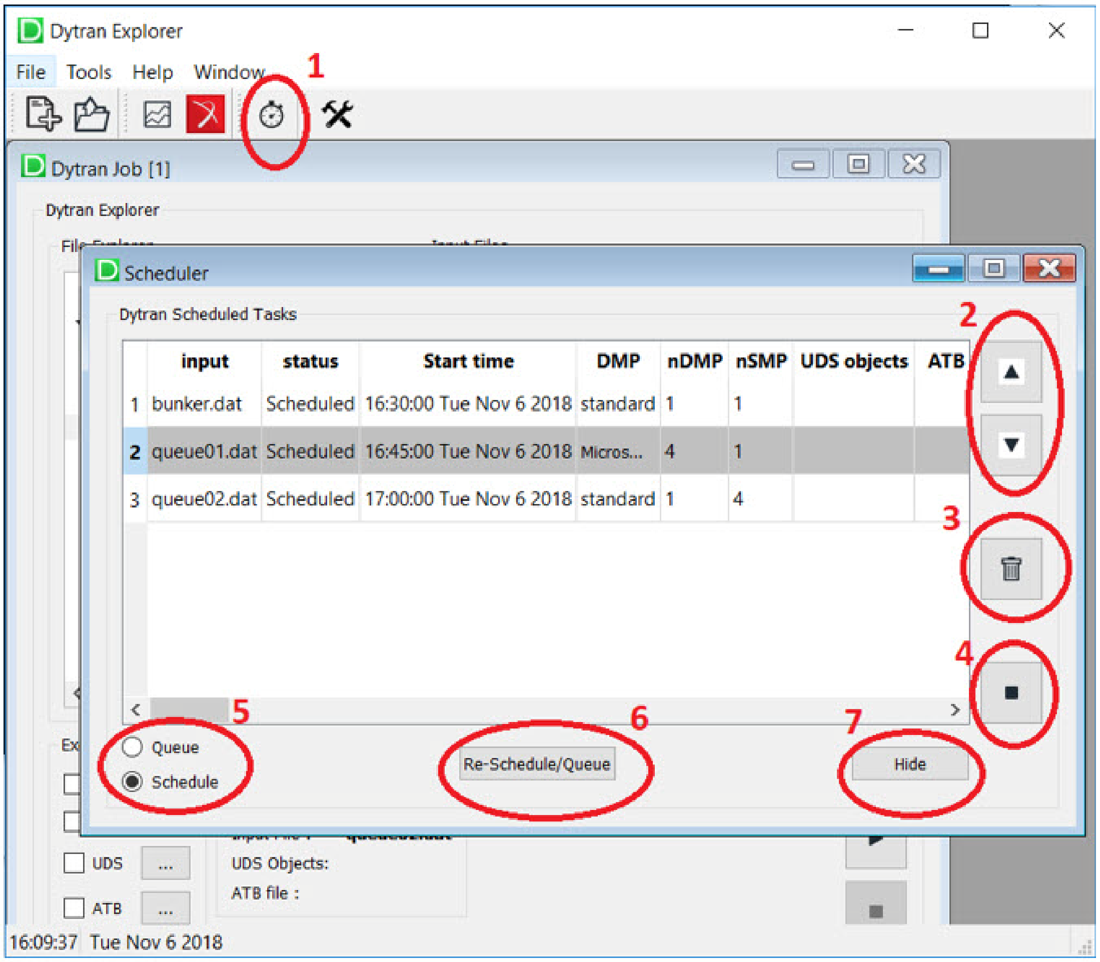
1. Click **Tools** to open the Tools menu.
2. Click **Options** to open the Options window.
3. Select the **Queue** tab.
4. Select either **queue functionality** or **schedule functionality.**
5. If **schedule** is used, set the start time here.
6. Optionally also a start date can be set.
7. Click **OK** when done.

Now new jobs can be setup in the same manner as regular jobs, except that when the start button is pressed, the job is not submitted directly, but instead it's added to the queue or scheduled depending on the functionality selected in tools/options/queue.

A pop-up window will warn users about the job not being started immediately.



If not already active, the **Scheduler** window will pop-up.



Dytran Scheduled Tasks

	input	status	Start time	DMP	nDMP	nSMP	UDS objects	ATB
1	bunker.dat	Scheduled	16:30:00 Tue Nov 6 2018	standard	1	1		
2	queue01.dat	Scheduled	16:45:00 Tue Nov 6 2018	Micros...	4	1		
3	queue02.dat	Scheduled	17:00:00 Tue Nov 6 2018	standard	1	4		

Queue [✓ Schedule]

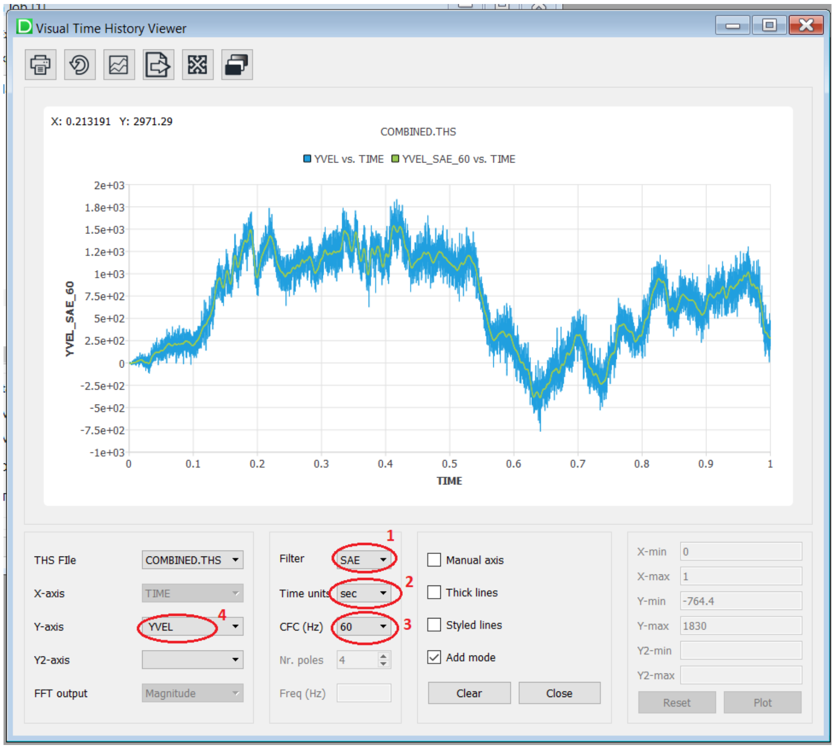
UDS Objects:
ATB file :

16:09:37 Tue Nov 6 2018

1. The **Clock** can be used to activate the **Scheduler** window, or make it visible again when it’s hidden.
2. The up- and down arrows can be used to move the selected job up and down in the list, so it will get a higher or lower priority in the queue. It will not affect the start time of scheduled jobs.
3. The “waste basket” can be used to remove jobs permanently from the queue/scheduler.
4. The stop ■ can be used to kill running jobs.
5. The **Queue** and **Schedule** selectors can be used to switch between the two types of functionality.
6. The **Re-Schedule/Queue** button will re-enter killed or finished jobs. When schedule functionality is active, the start time currently set in tools/options/queue will be used.
7. The **Hide** button will minimize the **Scheduler** window, for example to create some room for the new Dytran job window. To make it visible again, press the “Clock” button (1).

SAE-filtering

To filter out higher frequencies, an SAE filter has been added.



X: 0.213191 Y: 2971.29

THS FIle	COMBINED.THS
X-axis	TIME
Y-axis	YVEL
Y2-axis	
FFT output	Magnitude
Filter	SAE
Time units	sec
CFC (Hz)	60
Nr. poles	4
Freq (Hz)	
Manual axis	☐
Thick lines	☐
Styled lines	☐
Add mode	✓
X-min	0
X-max	1
Y-min	-764.4
Y-max	1830
Y2-min	
Y2-max	

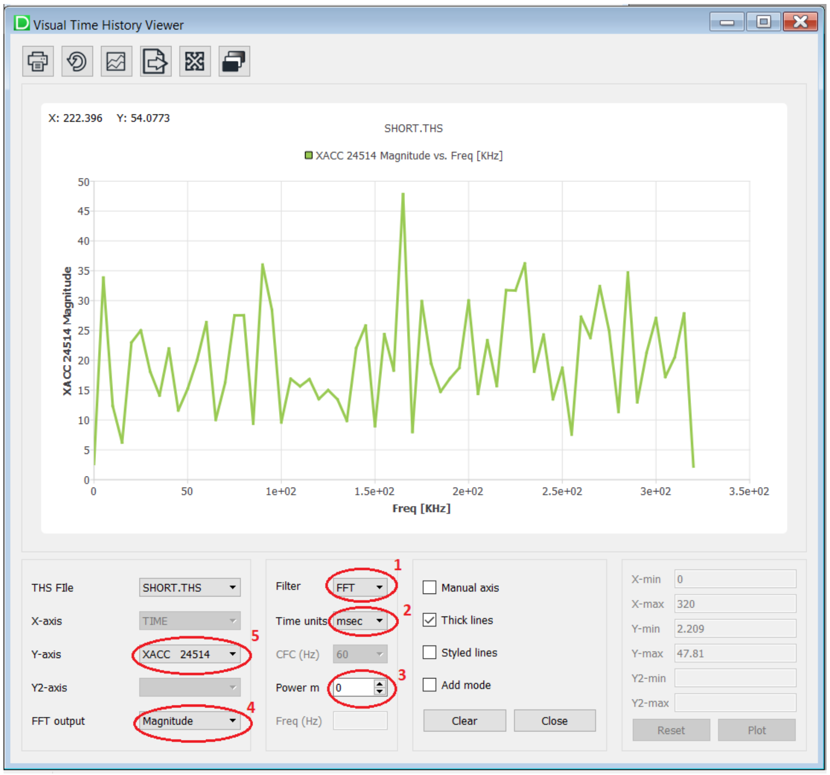
To activate the filter:

1. Set the "Filter" to SAE.
2. Set the time units used for your analysis (seconds, milliseconds, or microseconds).
3. Set the desired filter frequency.
4. Select variable to be filtered. It will be plotted automatically.

In the "Add mode", it's possible to draw both the filtered and unfiltered data on the same chart.

FFT-filtering

To do a Fast Fourier Transform analysis on a curve:



THS FIle	SHORT.THS
X-axis	TIME
Y-axis	XACC 24514
Y2-axis	
FFT output	Magnitude
Filter	FFT
Time units	msec
CFC (Hz)	60
Power m	0
Freq (Hz)	
Manual axis	☐
Thick lines	☑
Styled lines	☐
Add mode	☐
X-min	0
X-max	320
Y-min	2.209
Y-max	47.81
Y2-min	
Y2-max	

1. Set the "Filter" to FFT.
2. Set the time units used for your analysis (seconds, milliseconds, or microseconds).
3. Set the number of probes for the analysis as power of 2.

   Special values for which m will be calculated are 0 and 1. The m is calculated for 0 as the largest power of two that is smaller than the number of data points on the curve and for 1 as the smallest power of two that is larger than the number of data points.
4. Select the type of output, magnitude or phase.
5. Select variable to be analyzed. It will be plotted automatically.

10 Executing Dytran Using DMP

Eulerian and Lagrangian DMP

From Dytran 2021 release, Dytran DMP is extended to Lagrangain Solver. The existing DMP option for Eulerian solver and FSI capability is still available as the default setting too. A user can control DMP option using `PARAM,DMPOPT` entry. When `PARAM,DMPOPT,1` (default) is used, Dytran distributes the Euler elements across the available CPUs, while performing all Lagrangian solver calculations on the parent CPU.

The Eulerian DMP capability includes:

- Single and Multi-Material Euler
- Roe Solver
- Failed elements in coupling surface
- Graded Mesh
- Biased meshing
- Coupling surface output and markers
- Geometric boundary conditions
- Viscosity
- Automatic coupling

The Eulerian DMP capability does not support the following:

- ALE Method

The Lagrangian DMP capability includes:

- All element/point calculation
- MATRIG/RBE2/RIGID
- CONTACT/RCONN/CONTREL/RCONREL
- SPC/BJOIN/KJOIN/JOIN/PWELD/RBC3
- PLOADx/FORCEx/MOMENTx/TLOADx
- Failure of elements
- COG
- WALL

The limitations of Lagrangian DMP capability:

- Sublayer information in out is not correct (child cpu information is not stored): requires redesigning the OUT file to print out the info in each cpu
- If number of neighbor hexa or tetra nodes connected to kjoin node is larger than 6, the results may be different from serial run. (Warning message)
- V2 version contact is not available.
- CONTINI is not available.
- BELT/BELT1/DRAWBEAD options in contact is not available.
- ATB option is not available.
- SOLINIT is not available.

- SOLUOUT OR BULKOUT/NASINIT (NASINIT is required for SOLUOUT and BULKOUT) is not available.
- IMM option is not available.

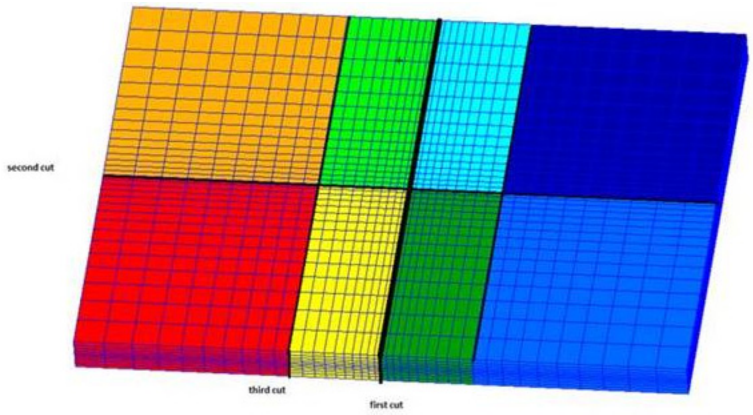
Eulerian DMP

The partitioning of the Euler domain into sub domains is done automatically. Each of these sub domains is put on a different CPU. This partitioning can be visualized by requesting the variable `PARTITION` on Euler archive requests. The value of `PARTITION` equals the CPU number.

The default method of partitioning the Euler mesh is the ORB partitioner. This partitioner implements the orthogonal recursive bisection method. This method takes an Element mesh and makes a split into two parts that gives the smallest communication costs (smallest number of faces along the split). Here, the split is made along a plane that is orthogonal to one of the coordinate axes. This gives two mesh parts. Then the method does exactly the same for the two mesh parts. This makes the algorithm recursive. It is a simple algorithm but the recursion makes it quite useful.

Consider for example, a mesh with 200, 150, and 50 elements in, respectively, the x-, y-, and z-direction. When split in x-direction (cut along a YZ-plane), the number of Euler faces at the split will be 150*50 = 7500. When splitting across the y-direction, it will be 200*50=10000; for the z-direction, it will be 200*150=30000. Since the x-split has the smallest number of faces at the split, the x-split has the smallest communication cost. Therefore, the ORB scheme will select the x-split. At the x-split, there are two mesh parts. Both have 100*150*50 elements. Now, a split in y-direction will give the minimal communication costs. This process of bisecting is continued until the number of sub meshes equals the number of CPU's.

An example of orthogonal recursive bisection of a 40*30*10 mesh for eight CPU's for a biased mesh is shown in the following figure:



The figure shows a fringe plot of the Euler element variable `PARTITION`. This shows how ORB has partitioned the Euler mesh across CPU's. The cuts of the Euler mesh are also shown.

The DMP in Dytran is extended to include the support of adaptive Euler. The adaptive meshing technology automatically generates the Eulerian meshes as needed during the analysis. When the structure undergoes severe deformation, the adaptive mesh is expanded to follow the movement of coupling surfaces, preventing the creation of extraneous elements at the start of the analysis and, therefore, resulting in significant reduction in simulation runtime. The DMP support for adaptive Euler capability will allow the users to run CPU intensive FSI applications on multiple cores further improving the performance runtimes. There are no GUI requirements for this new capability.

One important feature in DMP support for adaptive meshing is the load balancing scheme. During the DMP simulation with adaptive mesh, the Eulerian mesh is decomposed into many domains, and each domain is spawned onto different processors. The load balancing is intended to ensure that all cores have about the same amount of work for an efficient computation as the meshes are transported from one core to the next during the simulation. There are many FSI applications that can result in poor performance due to lack of proper load balancing. For optimal speed-ups, the Euler mesh has to be re-partitioned across the cores. For example, during druid-filled bottle drop test on two cores, the fluid moves through the Euler mesh leaving one core and entering the second core. As a result, one core is off-loaded as the second core is loaded.

Load balancing ensures that both cores have a balanced amount of processing and is also a major step towards good scalability on multiple cores.

Adaptive meshing can result in shorter run times for tank sloshing simulations especially when there is significant movement of the tank through the Euler mesh. Running these simulations with adaptive meshing in combination with DMP may result in poor performance in some cases. This happens because of inefficient load unbalancing across the processors. When the tank moves, the adaptive mesher creates Euler elements at the front of the tank and deletes the elements at the back of the tank.

Consequently, elements are deleted on one processor and are created on another one. This causes load unbalance that might lead to poor performance. To overcome load unbalancing, Euler cubes can be moved from one processor to the other processor when there is a sufficient number of cubes present in the simulation. But if the number of Euler cubes is not sufficient, the re-allocation of cubes is not possible and causes load unbalance.

Whether or not load unbalance problems have occurred, it can be checked by the message in the `OUT` file:

```
**********************************************************************
*                                                                    *
* MESH (ID=101) HAS BEEN REDISTRIBUTED ACROSS CPUS AT CYCLE 191400 *
*                                                                    *
* NUMBER OF FINITE VOLUME ELEMENTS ON EACH TASK:                    *
*                                                                    *
* TASK 0 : 197772                                                   *
* TASK 1 : 52013                                                    *
*                                                                    *
**********************************************************************
```

In the example above, there are only two cubes for the entire tank sloshing that were used, and the message shows the load unbalance. By adding more cubes in the direction of tank motion, there will be more cubes to move around and will resolve the load unbalance problems. Caution must be exercised in creating excessive number of cubes since it can lead to overhead costs. For this reason, the number of cubes should be limited to about 100. The overhead costs for simulations with adaptive meshing on 8 or more cores may results in degraded performance.

The number of cubes can be set by using `PARAM,EULERCUB`. The number of Euler cubes created are shown in the `OUT` file as follows:

```
********************************************************************
*                                                                  *
* SUMMARY OF CUBE MESH (ID=1) FOR CYCLE 0                          *
*                                                                  *
* TOTAL ELEMENTS INCL PHANTOM : 64064                              *
* PHANTOM ELEMENTS : - 2464                                        *
* TOTAL ELEMENTS EXCL PHANTOM : 61600                              *
*                                                                  *
* TOTAL CUBES INCL PHANTOM : 4                                     *
* PHANTOM CUBES : - 2                                              *
* TOTAL CUBES EXCL PHANTOM : 2 NBX * NBY * NBZ                     *
* CHECK: NBX*NBY*NBZ : = 2 2 1 1                                   *
```

Phantom cubes consist of one layer of elements and are used to communicate Euler element data from one CPU to the other.

Lagrangian DMP

In this release, the default of DMP is set only to Eulerian and FSI DMP. To activate Lagrangian DMP, `PARAM,DMPOPT,2` or `PARAM,DMPOPT,3` must be set. Partitioning Lagrangian DMP uses Metis Partitioner which already used in Nastran and Marc. There are 7 different options to partition Lagrangian elements in `PARAM,LAGPR` but generally the default option is enough to show the good load balance.

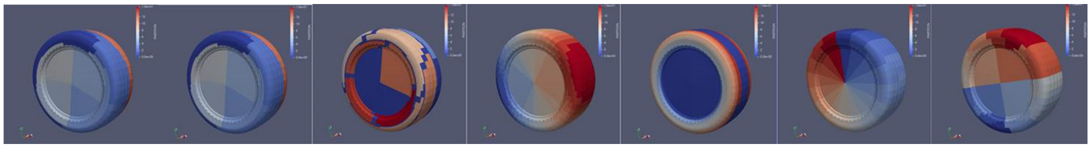


Figure 10-1 Different partition option from 1 (left) to 7 (right)

The Figure 10-1 shows the partitioning contour of each partitioning option which is set in `PARAM,LAGPR`.

Currently, contact is working only in parent cpu. So when the simulation has quite severe loading in contact calculation, the Lagrangian DMP will not show the good performance. In addition, there are couple of the reasons why the load balance is not good such as different types of elements, different types of materials, different number of integration points of shells. When a model has many different types of elements such as shells, hexa elements, tetra elements, etc, the DMP performance is not good as expected. The limitations will be improved in the future.

The partition in each cpu of Lagrangian DMP is printed out in the OUT file.

For example,

```
**** AN OVERVIEW OF THE MESH FOR PARTITION    0 IS GIVEN BELOW:

         .............................................................
         .    ELEMENT TYPE AND NAME      .  POINTS .  ZONES   .  FACES  .
         .............................................................
```

```
          .                                .          .          .          .
          .TYPE 2 - ELEM1D (ROD,BAR,BEAM).        19 .          9 .          0 .
          .                                .          .          .          .
          ..............................................................
          .                                .          .          .          .
          .TYPE 3 - SHELLS (TRIA)          .     64781 .     127371 .     127371 .
          .                                .          .          .          .
          ..............................................................
          .                                .          .          .          .
          .TYPE 4 - SHELLS (QUAD)          .      1661 .       1637 .       1637 .
          .                                .          .          .          .
          ..............................................................

**** AN OVERVIEW OF THE MESH FOR PARTITION    1 IS GIVEN BELOW:

          ..............................................................
          .     ELEMENT TYPE AND NAME      .  POINTS .   ZONES   .   FACES   .
          ..............................................................
          .                                .          .          .          .
          .TYPE 2 - ELEM1D (ROD,BAR,BEAM).        19 .          9 .          0 .
          .                                .          .          .          .
          ..............................................................
          .                                .          .          .          .
          .TYPE 3 - SHELLS (TRIA)          .     64856 .     127371 .     192113 .
          .                                .          .          .          .
          ..............................................................
          .                                .          .          .          .
          .TYPE 4 - SHELLS (QUAD)          .      3294 .       1637 .       4911 .
          .                                .          .          .          .
          ..............................................................
```

System Requirements

Running Dytran on multiple CPUs requires MPI to be installed on every machine used. This is true even for single processor machines.

Dytran expects hardware-specific native MPI to have been installed at default locations. When MPI is not properly installed on your Unix/Linux machine or is not installed at the expected default location, a job submission exits with an error message to this effect. To avoid problems of this nature or problems caused by different versions of MPI on several of the supported platforms, the MPI version is now part of the release and is installed at a defined location. See Table 10-1 for details.

Table 10-1 MPI Version and Expected Location

Platform	MPI Version	MPI Location
Windows 64	Microsoft v8.1 (Default) Intel MPI 2017 Update 2	$MSMPI_BIN[1]
Linux 64	Intel MPI 2017.0.5.249	

[1] Exact location is not important. The Dytran execution scripts rely on the definition of the environment variable MSMPI_BIN which is defined by the Microsoft HPC PACK installer

Hardware and Software Requirements

For the Linux platforms an installation of the correct version of IntelMPI is required on all the machines that are clustered for the job submission. The installation of these MPI versions is made available and is installed with the Dytran installation.

Although no specific hardware requirements exist for Dytran to run in distributed memory parallel mode, it is preferable to have fast network connections between the machines if more than one machine is used. It is recommended that the network should have a speed of at least 100 MBs per second. The appropriate licenses are required in all machines used.

For Windows platforms the correct version of Microsoft MPI is required on all machines. Also IntelMPI can be used to run on multiple platforms and requires an installation on all platforms. The correct version of these MPI packages are included in the Dytran installation.

The head node is used to check your licenses. So make sure the head node has correctly defined license settings.

If only two machines are to be used, you can use a hub or a cross-over cable to connect them. If more than two machines are to be used, a switch is preferable. TCP/IP is used for communications.

Windows Specific Fixes

If you are running an older version of Windows on a system that is not connected to a network and want to run a parallel job on that system, you may have to install the Microsoft Loopback Adapter. Follow these steps:

1. Go to Control Panel, Add/remove Hardware.
2. Select the hardware task you want to perform:

   Add/Troubleshoot a device
3. Choose a Hardware Device:

   Add a new device
4. Do you want Microsoft Windows to search for your new hardware?

   No, I want to select the hardware from a list
5. Select the type of hardware you want to install:

   Network adapters
6. Select Network Adapter:

   Manufacturers: Microsoft

   Network Adapter: Microsoft Loopback Adapter

It will now install the loopback adapter. You will have to enable/disable the loopback adapter as you remove/connect your machine to the network.

Compatibility

Two machines are compatible if they can both use the same executables. Some examples of compatible machines are:

1. Several machines with exactly the same processor type and O/S.

Definition

1. Root machine: The machine on which the job is started.
2. Remote machine: Any machine other than the root machine that is part of a distributed parallel run on the network.

User Notes

This section assumes that Dytran has been successfully installed on at least one of the machines that are to be used in a distributed analysis, and that the appropriate Dytran licenses are in order. Assume that `host1` is the host name of the machine on which the job is to be started (the root machine). The host names of the other machines (remote machines) are `host2`, `host3`, etc.

Dytran always uses one of the CPU's of the root machine. Additional child nodes can be used in a single calculation by using a host file. The host files for windows and Linux are the same:

```
hostname1 [number 1 of CPUs] [working directory] [executable location 1]
hostname2 [number 2 of CPUs] [working directory] [executable location 2]
hostname3 [number 3 of CPUs] [working directory] [executable location 3]
```

`hostnamei`	=	name of the machine in the network. Each new `hostname` must be on a new line.
`number i CPUs`	=	number of CPUs to be used on each machine. Default is 1.
`working directory`	=	working directory on each machine. This is dummy input for this version.
`executable location`	=	location of executable on each machine. Default for `executable location 1` is the script to figure out MSC installation on `hostname1`. Default for other executable location is `executable location 1`.

Examples:

Linux:

Start run on four CPU's: two on the root machine (for example named "headnode") and two on the remote machine, named "secondnode". Both machines have exactly the same location of executable. Then host file has the following lines:

```
headnode 2
secondnode 2
```

If Dytran is installed in different locations between the root machine and the remote machine:

`headnode 2`
`secondnode 2 /tmp /app2/msc/dytran/Dytran2021/dytranexe/dytran.exe` (note the blank after /tmp)

Windows:

```
headnode 2 C:\scratch C:\"Program Files"\MSC.Software\Dytran\2021\dytranexe\dytran.exe
secondnode 2 D:\scratch D:\"Program Files"\MSC.Software\Dytran\2021\dytranexe\dytran.exe
```

How to Run Dytran in Parallel on UNIX and Linux

To submit DMP jobs, you must specify the number of processors:

- `dytran jid={job-name}   dmp=yes  ncpus=2`
- `dytran jid={job-name}   dmp=yes  ncpus=4  bat=no`

Currently, it is not possible to create customized executables for DMP.

Running Dytran on Windows

On Windows, submit a Dytran analysis by double clicking the Dytran icon. The icon should be available on your desktop. Alternatively, you can use the **Start Menu** to locate Dytran under the `Programs Folder`. Once you picked either the icon or the menu entry, the Dytran user environment appears on your screen.

To submit Dytran jobs using the command line, open Start/run command window, and type:

```
{Dytran directory}\Dytran\2021\bin\dytran.exe jid={job-name}
dmp={yes or both} nproc={number of processors} intelmpi={yes
or no}
```

For instance:

1. Name input model is `bunker.dat`.
2. Dytran is installed at `C:\Program Files\MSC.Software`.
3. Model needs to run on 4 CPUs in DMP mode.

   The correct command would be:

   ```
   C:\Program Files\MSC.Software\Dytran\2021\bin\dytran.exe jid=bunker dmp=yes nproc=4
   ```

Additional Information

Linux x8664

1. First check that `rsh` is installed by doing:

   ```
   rpm -qa | grep rsh
   ```

   The answer should be:

   ```
   rsh-server-0.17-14
   rsh-0.17-14
   ```

2. If installed then check the following two files:

   ```
   /etc/xinetd.d/rsh
   /etc/xinetd.d/rlogin
   ```

   There is a line that should say:

   `disable` = no

   If it says yes, change it to no.

3. Check the following file:

   ```
   /etc/nsswitch.conf
   ```

   It should say:

   ```
   hosts:  files dns nis
   ```

   `files` should come before anything else

4. Restart after making changes:

   ```
   /etc/init.d/xinetd restart
   ```

5. Add in `.rhosts` file to your home directory that contains the following entries:

   ```
   {machine1} {user}
   {machine1 full name} {user}
   {machine2} {user}
   {machine2 full name} {user}
   +
   ```

   For example:

   ```
   murano walter
   murano.adams.com walter
   usher walter
   usher.adams.com walter
   miller walter
   miller.adams.com walter
   +
   ```

6. Make sure that `rsh` does not show any echo's from the `.cshrc` file. You must modify your `.cshrc` file such with `if (! $?prompt)` that no echo's or clear's are done. You can check that by:

   ```
   rsh {remote system} date
   ```

7. Set the path such that `hboot` can be executed. This means that the path must be set correctly by the `.cshrc` on the remote machine. You can test this by:

   ```
   rsh {remote system} 'echo $path'
   rsh {remote system} 'which hboot'
   ```

Do not omit the quotes from the command line. Without them, the actual path of the machine you are on will be echoed.

The same working directory as used on the root machine does not have to exist on the remote machines.

Windows

Dytran supports two MPI options:

- Microsoft MPI
- Intel MPI

When a single machine is used with multiple cores, Microsoft MPI needs an installation. This installer is not included in a Dytran distribution, but needs to be downloaded from Microsoft directly. For Intel MPI running on a single machine, the Dytran installation already provides the proper libraries and a special Intel MPI installation is not required.

In order to run Intel MPI on multiple Windows machine the following considerations need to be met:

1. The user must have Administrator privileges on all machines.
2. Start the MPI service on all machines with:

   ```
   C:\Program Files\MSC.Software\Dytran\2021\dytranexe\intelmpi\bin\hydra_service.exe -install
   ```

   The response should be:

   ```
   Intel(R) MPI Library Hydra Process Manager installed.
   ```

3. Register the user id on all machines:

```
C:\Program Files\MSC.Software\Dytran\2021\dytranexe\intelmpi\bin\mpiexec.exe -register
```

The response should be:

```
account (domain\user) [XX\yyyy]:
password:
confirm password:
Password encrypted into the Registry.
```

Now the cluster will be ready to run Intel MPI on multiple machines.

11 Postprocessing Dytran Results with ParaView

How to obtain ParaView

ParaView is a free open source data visualization program developed by Kitware Inc. It is based on the Visualization Toolkit (VTK), also developed and maintained by Kitware Inc. ParaView can be downloaded from http://www.paraview.org. The installation is straight forward.

Arc2Vtk Translator

There are a number of readers available in ParaView that can handle a various data formats. MSC has developed the Arc2Vtk translator to convert the Dytran archive files into "VTK" files that can be read by the VTK-toolkit, which is part of ParaView.

In the case of Dytran ARC files, the "unstructured grid" variant of VTK with file extension `.vtu` has to be used, either in ASCII or base64 encoded binary format. The `.vtu` file format itself does not support cycle or problem time information, but this can be provided to ParaView by a meta-file with file extension. PVD that describes sets of `.vtu` file. The Arc2Vtk translator creates them automatically. For this purpose, time and cycle numbers are written in the VTU files as XML-comment.

ParaView can also create time history plots from a.o. "comma separated values" files. Arc2Vtk has an option to create these `.CSV` files from Dytran Time History files (`.THS`).

There is no graphical user interface available yet for Arc2Vtk, so the program has to be run from the command line. On Windows, you need to open a **cmd** window (**Start/All Programs/Accessories/Command** prompt); on Linux/Unix, you need to open a **"**terminal**"**. The easiest way to translate a set of Dytran archive files and prepare an appropriate PVD file is:

```
arc2vtk.exe my_file_name_XXX.ARC
```

with `my_file_name_XXX.ARC` the name of an archive file as created by Dytran. Arc2Vtk searches the current working directory for all `.ARC` files with the same prefix, but different cycle numbers 'XXX', translate all of them to VTU files and create a PVD file called "`my_file_name_XXX.pvd`".

The PVD file can be opened in ParaView.

To translate a single Dytran/SOL700 archive file use:

```
arc2vtk.exe [-binary|-ascii] -nonauto my_file_name.ARC
```

Arc2Vtk creates one or more `.VTU` files, depending on the number of time steps found on the archive file. The same naming convention is used as in Dytran, which means that when cycle `XXX` is found on archive file `my_file_name_YYY.ARC`, the corresponding output file will be named `my_file_name_XXX.VTU`.

It is strongly recommended to use the default settings: `-xml` and `-binary`.

To create a PVD-file from a set of VTU files already created by Arc2Vtk use:

```
arc2vtk.exe my_file_name_XXX.vtu
```

This will create a PVD-file for the collection of all VTU-files matching the name `my_file_name_XXX.vtu`, where `XXX` are cycle numbers. Note that the argument should be the name of an existing `VTU` file.

The resulting output file is named `my_file_name_XXX.PVD`, so for example `MY_FILE_0.VTU` results in the output file `MY_FILE_0.PVD`.

To create a set of CSV-files (comma separated values) from a Dytran time history file (`THS`) use:

```
arc2vtk.exe my_file_name.THS
```

This creates a CSV-file for each of the "items" (gridpoint, element, face or material, depending on the type of THS-file) on the time history file, named `my_file_name_X_NNN.csv`. With "`X`" replaced by the type of item (`N` for node, E for element, `F` for face and `G` for global data) and `NNN` its identifier number. CSV files can be processed by ParaView, but also be used as input for spreadsheet programs like Excel.

ARC2H5 Translator

ARC2H5 translator is used to convert ARC files into H5 file format. H5 is a data file format in the Hierarchical Data Format (HDF) that is widely used to store multidimensional arrays of scientific data. H5 files have shown a good performance in reading/writing data and capability of compressing data using built-in and third-party filters. The current version of ARC2H5 uses the HDF5 built-in "Scale-offset" compression tool. The compression ratio may vary depends on many factors such as ARC file size, data structure, number of cycles, scale-offset factor, etc. Users should expect a compression ratio between 1:1 (no compression) to 5:1 (20% of ARC file/s size).

ARC2H5 can be run using either Dytran Explorer or command line:

- **Run ARC2H5 using command line:** First the user needs to change the directory to the location where the ARC files are located, then execute ARC2H5.

  ```
  cd {path-to-ARC-files-dir}
  ```

  ```
  {Dytran-installation-dir}/bin/exe/arc2h5.exe {arc-file-name}
  [options]
  ```

  where the ARC file must follow its default name format:

  ```
  {job-id}_{outpu-tag}_{cycle-number}.ARC
  ```

  and the options are:

`-pv=0|1`	paraview output request		
	`0`	Eq. no (default)	
	`1`	Eq. yes	
`-vt=0|1`	vector/tensor output request		
	`0`	Eq. no	
	`1`	Eq. yes (default)	
`-na=i`	max number of arc files translated into one h5 file		
	(default is 9999)		
`-sf=0|i`	hdf5 scale-offset filter scale factor which denotes MinBits		
	`0`	Eq. the filter calculates MinBits	

	i	positive integer Eq. MinBits number higher values correspond to higher precision (default is 9)
-h	to see valid options.	

Example:

```
"C:\Program Files\MSC.Software\Dytran\2021\bin\exe\arc2h5.exe"
CYLPAN_PANEL_0.ARC -pv=1
```

Dytran may store `ARCHIVE` output request into multiple files depending on the `SAVE` entry specified in the input file. If there are multiple ARC files that belong to the same `ARCHIVE` request, only one of them needs to be passed as argument to arc2h5 translator. The translator automatically finds and combine the ARC files in the working directory that corresponds to that ARCHIVE request, i.e. ARC files with the same prefix but different cycle numbers. It should be noted that ARC2H5 does not combine ARC files of different `ARCHIVE` request.

For example:

```
ls *.ARC
CYLPAN_PANEL_0.ARC
CYLPAN_PANEL_100.ARC
CYLPAN_PANEL_200.ARC
arc2h5.exe CYLPAN_PANEL_0.ARC
```

The resulting `CYLPAN_PANEL_0.H5` file will contain the entire output of steps 0, 100, and 200 in one single file.

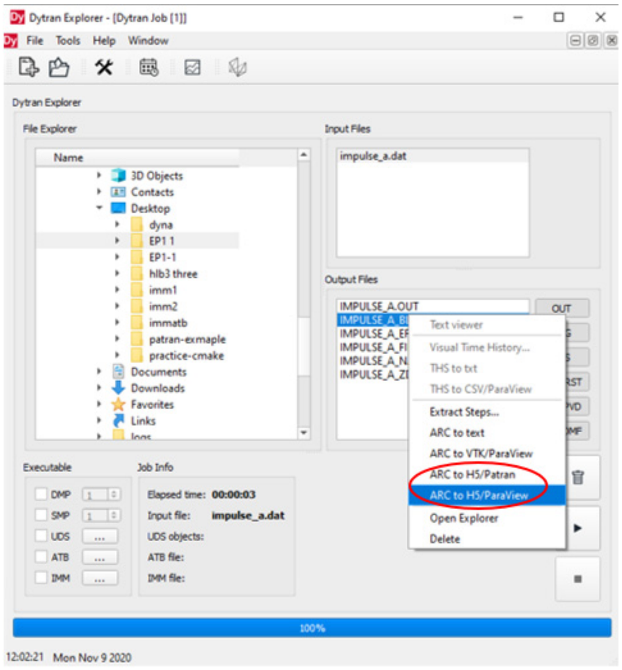
Since the current version of Patran is able to load only one h5 file in a session, the `-na` option should be kept as default (`-na=9999`). This case allows to convert multiple ARC files belong to an `ARCHIVE` request into a single H5 file.

- **Run ARC2H5 using Dytran Explorer:** Users can run ARC2H5 from the ARC file action menu. In this case ARC2H5 tools will be run with the default options, which are equivalent to the below commands:

  ARC to H5/Patran: `arc2h5.exe {arc-file-name} -vt=1   -na=9999 -sf=9`

  ARC to H5/ParaView: `arc2h5.exe {arc-file-name} -vt=1   -na=9999 -sf=9 -pv=1`

  If there are multiple ARC files that belong to the same `ARCHIVE` request, the actions should be done only one of them. ARC2H5 will automatically fetch other ARC files that belong to the `ARCHIVE` request.



impulse_a.dat

IMPULSE_A.OUT
IMPULSE_A_BI
IMPULSE_A_EF
IMPULSE_A_FI
IMPULSE_A_N
IMPULSE_A_ZI

Elapsed time: 00:00:03
Input file: impulse_a.dat
UDS objects:
ATB file:
IMM file:

12:02:21 Mon Nov 9 2020

After executing ARC2H5, it generates H5 file/s and XDMF file/s (if ParaView option is selected). H5 files and XDMF files have the same name as the input ARC file/s. If multiple files are combined into a single H5 file, the H5 file name will be the same as the ARC file with the lowest cycle number.

ParaView can be used to postprocess Dytran H5 files. ParaView locates data arrays, i.e. mesh info and results, in H5 file using the XDMF metadata file that is generated during ARC-to-H5 translation.

The content of H5 files can also be viewed in a tabular form using HDFView. Dytran H5 files may have the following data structure:

```
/RESULT
    /EXPLICITOUT
        /DOMAIN_{cycle-number}
            /ELEMENT
                -ZONE_CONN: element connectivity
                -ZONE_MAT: element material id
                -ZONE_UID: element user id
            /NODE
                -GRID_UID: node user id
                -GRID_XYZ: element coordinate
            /RESULT
                /ELEMENTAL
                    /SCALAR
                        -ALLEL-flt: float scalar variables
                        -ALLEL-int: integer variables
                    /VECTOR
                        -ALLEL-flt: float vector variables
                    /TENSOR
                -ALLEL-flt: float tensor variables
        /NODAL
            /SCALAR
                -ALLEL-flt: float scalar variables
                -ALLEL-int: integer scalar variables
            /VECTOR
                -ALLEL-flt: float vector variables
            /TENSOR
                -ALLEL-flt: float tensor variables
```

where:

- "/" and "-" denote groups and datasets in H5 file respectively.
- Variable names are stored in dataset's attributes.
- `Domain` group is repeated for each cycle that output is requested.
- Elemental (and nodal) result are stored in the same order as the element (and nodes) are defined.

Limitations

The current version of ARC2H5 file has the following limitations:

- ARC files that were created before Dytran 2019 cannot be processed.
- ARC files that were created using ATB ellipse covered option, cannot be processed.

Running ParaView

The user interface of ParaView is rather intuitive and creation of a simple animation is not too difficult. The program has a lot of capabilities and describing all of them here is not possible. Furthermore. there is also a Python interface.

To startup ParaView go to **Start/Programs** as usual.

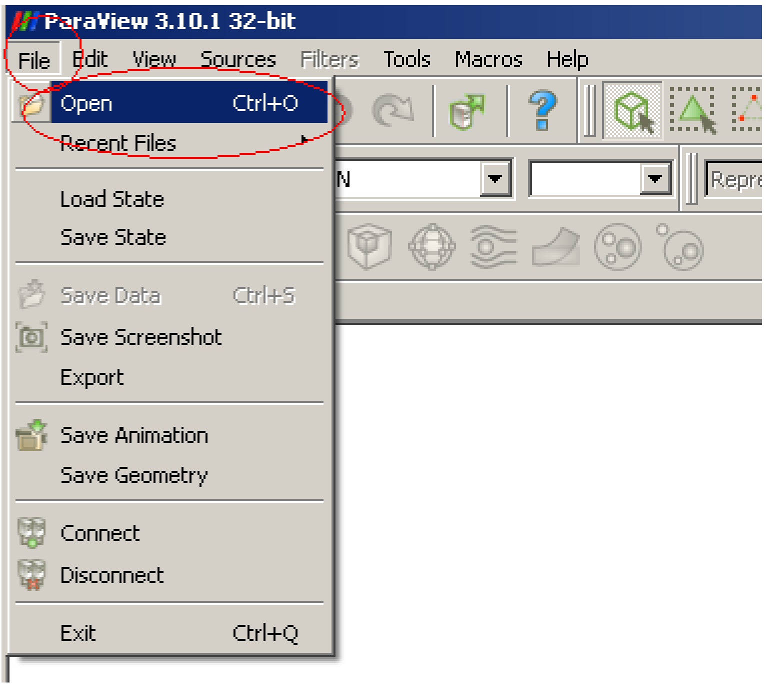
In the following example we will demonstrate:

- How to open files in ParaView
- How to select results variables to display
- How to apply a clipping plane
- How to open a series of VTU (translated .ARC) files
- How to apply a filter to convert cell data to point data
- How to create an isosurface
- How to make an object transparent
- How to start an animation
- How to save an animation

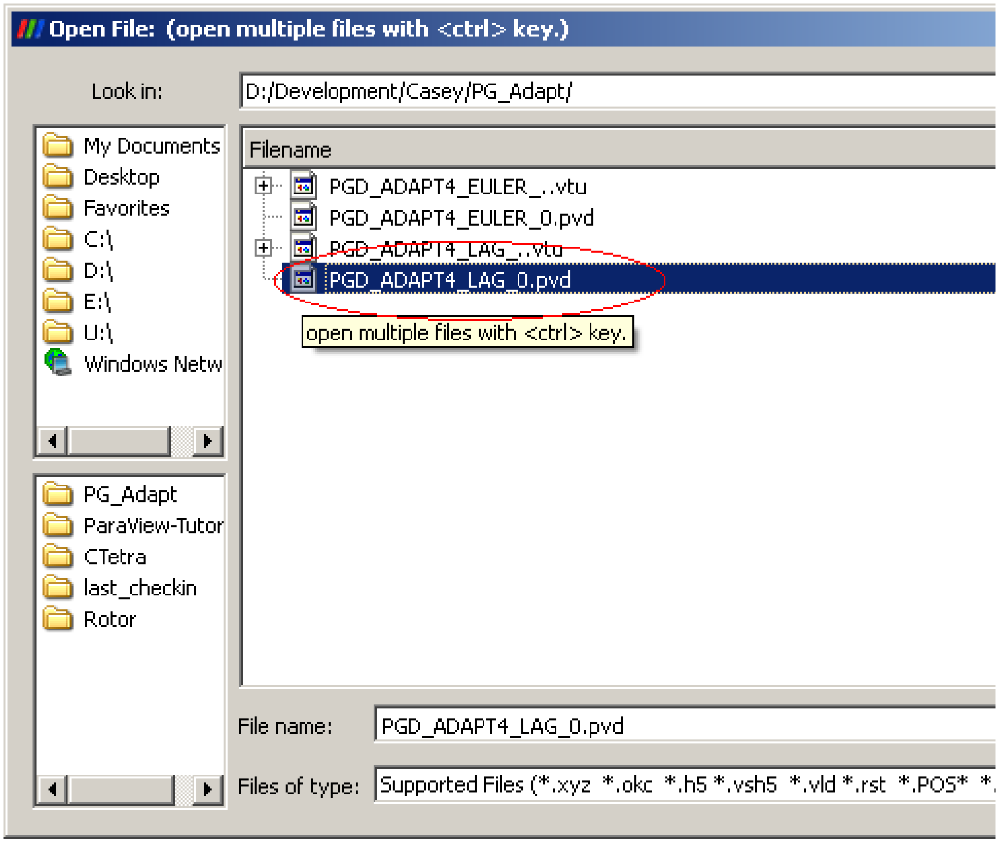
1. Open a file

   One can open either single .vtu files (click on the "+" sign to expand the list when there are more .vtu files with the same base name), a whole series of .vtu files by selecting the first of the list without expanding it, or the preferred way, by selecting the .pvd meta-file. When the results of more than one series of archive files have to combined, for example when there are separate Euler and Lagrange data, using the .pvd file is the only way to get the results synchronized.

   - Click on file in the menu bar at the top of the ParaView window.
   - Click **Open**.



- Use the Explorer window to browse to the file(s) you want to open and click **OK** The name of the file will appear in the **Pipeline Browser**.



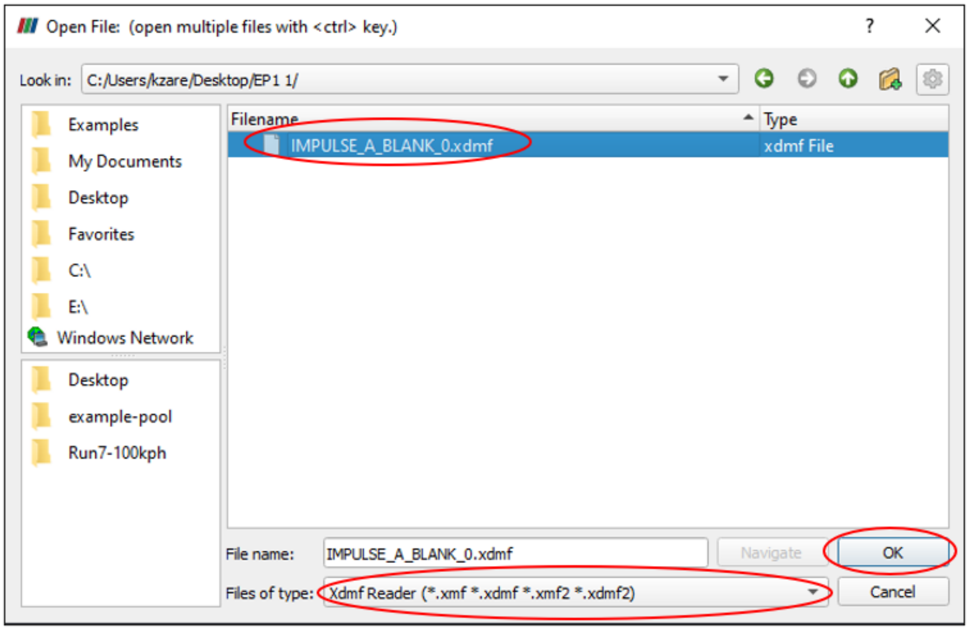
D:/Development/Casey/PG_Adapt/

PGD_ADAPT4_EULER_..vtu
PGD_ADAPT4_EULER_0.pvd
PGD_ADAPT4_LAG_..vtu
PGD_ADAPT4_LAG_0.pvd

File name: PGD_ADAPT4_LAG_0.pvd
Files of type: Supported Files (*.xyz *.okc *.h5 *.vsh5 *.vld *.rst *.POS* *.

The file name will appear in the **Pipeline Browser** on the left side of the ParaView window. You need to click **Apply** on the **Properties** tab of the **Object Inspector** pane that is positioned below the **Pipeline Browser**.

In case of loading H5 file in ParaView, open or drag and drop the XDMF file/s into ParaView. Make sure that the H5 file/s and the corresponding XDMF file/s are in the same directory.



Open File: (open multiple files with <ctrl> key.)

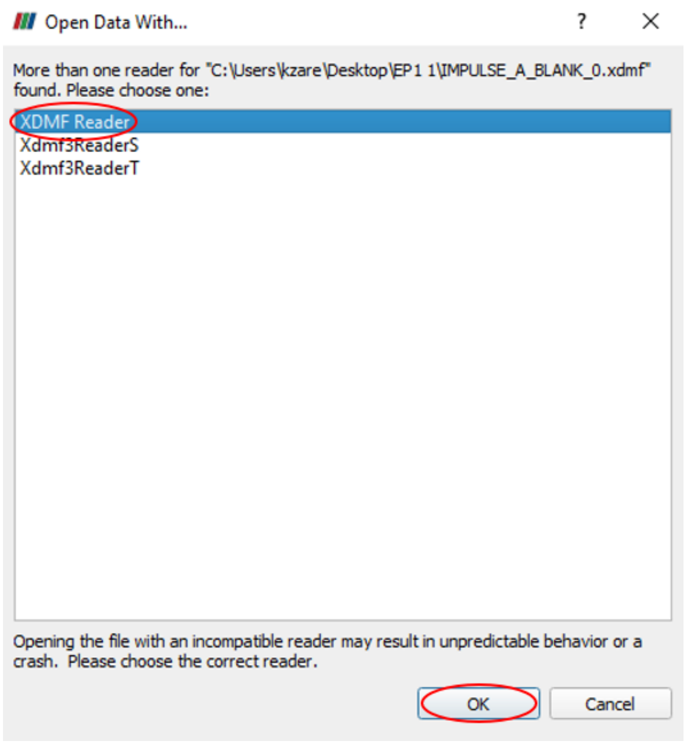
Look in: C:/Users/kzare/Desktop/EP1 1/

Filename	Type
IMPULSE_A_BLANK_0.xdmf	xdmf File

File name: IMPULSE_A_BLANK_0.xdmf

Files of type: Xdmf Reader (*.xmf *.xdmf *.xmf2 *.xdmf2)

Then, in the pop-up menu, select XDMF Reader.


Open Data With...

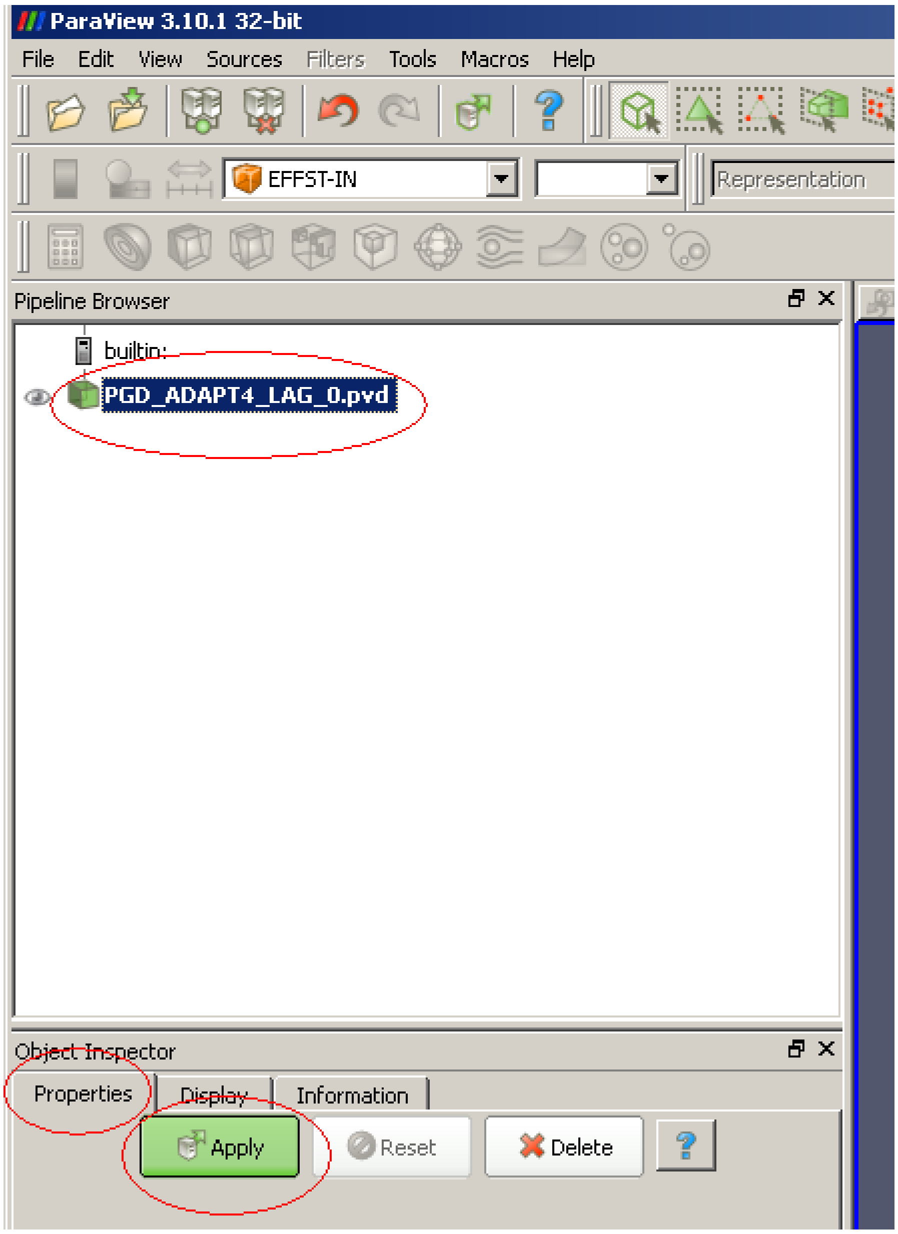
More than one reader for "C:\Users\kzare\Desktop\EP1 1\IMPULSE_A_BLANK_0.xdmf" found. Please choose one:

XDMF Reader
Xdmf3ReaderS
Xdmf3ReaderT

Opening the file with an incompatible reader may result in unpredictable behavior or a crash. Please choose the correct reader.

Reading and processing of the data may take some time.

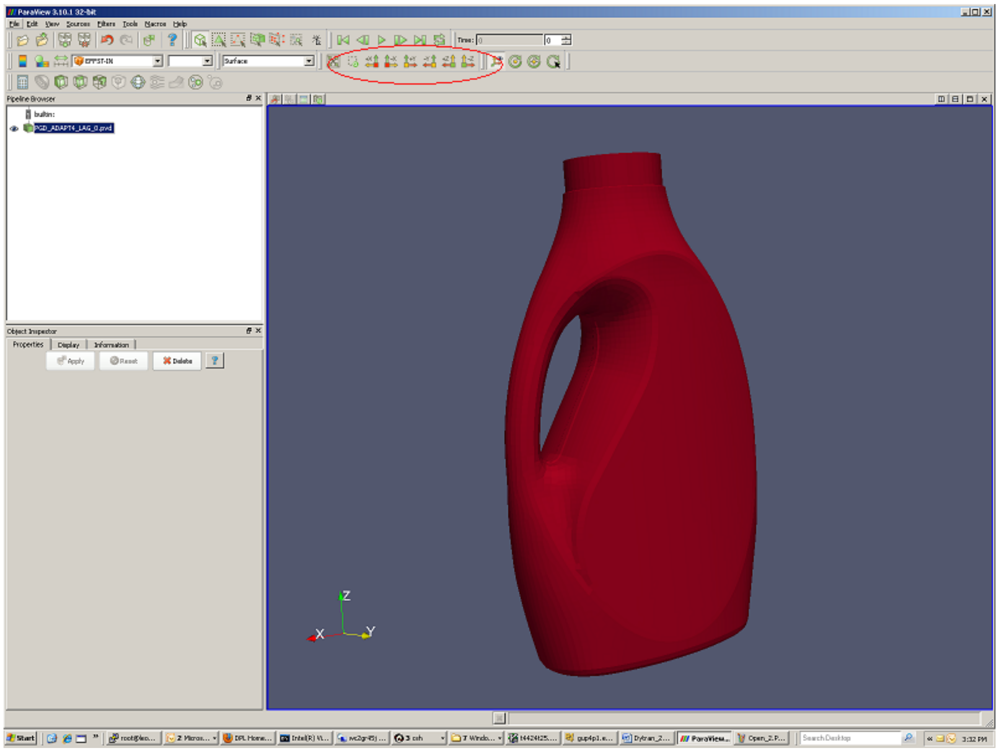
- Click on **Apply** on the **Properties** tab of the **Object Inspector**.



2. Positioning of the object

   You can use the buttons with the axis icons in the toolbar to position the object in the display pane. Furthermore, clicking one of the mouse buttons and moving the mouse when the cursor is in the display window will enable you to:

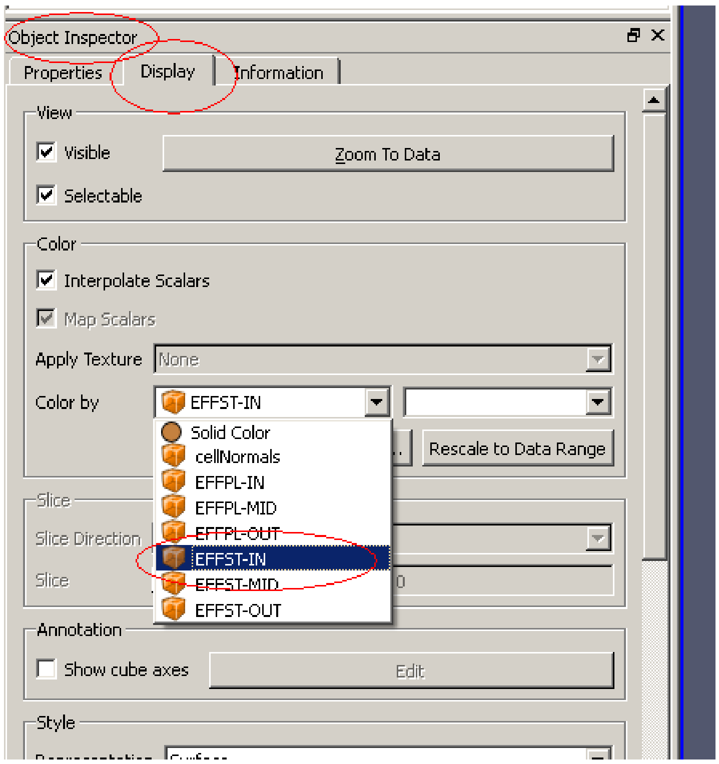
   - left mouse button: rotate the object
   - middle mouse button: move the object in horizontal and vertical directions
   - right mouse button: zoom in and out
   - scroll wheel: zoom in and out



3. Displaying results

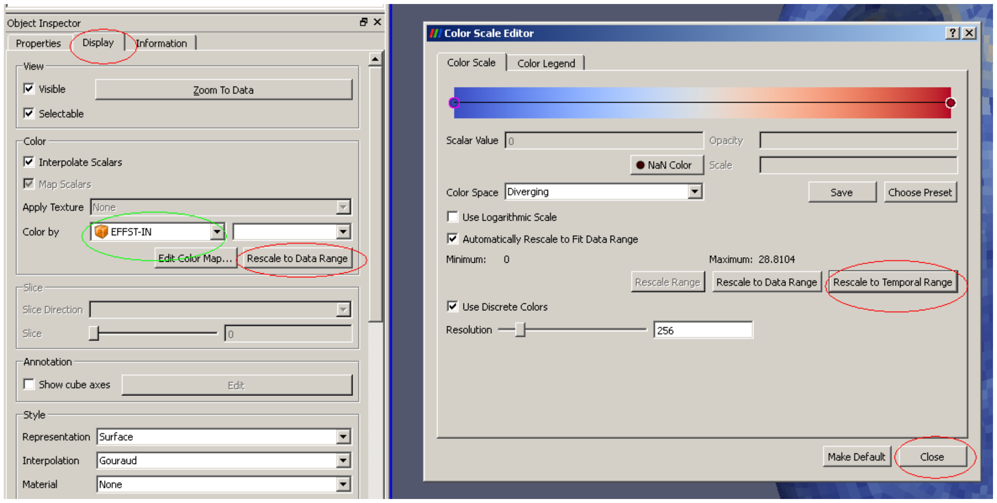
To create fringe plots of results:

- Make sure the object on which you want to make the plot is selected in the **Pipeline Browser**.
- Open the **Display** tab of the **Object Inspector** and select the variable you want to use for the fringe plot.
- Select the variable you want to display.
- Open the **Properties** tab of the object inspector and click the (green) **Apply** button.



To change the color map:

- Open the **Display** tab of the object browser.
- Click **Edit Color Map**.
- Click either **Rescale to Range Data** if you want to apply the color map to the current time step only, or
  click **Rescale to Temporal Range** if you want a color map that will cover the range in all time steps. The latter may take some time.
- **Click Close** when finished.



View: ☑ Visible, ☑ Selectable
Color: ☑ Interpolate Scalars, ☑ Map Scalars
Apply Texture: None
Color by: EFFST-IN
Slice Direction
Slice: 0
Annotation: ☐ Show cube axes
Style:
Representation: Surface
Interpolation: Gouraud
Material: None

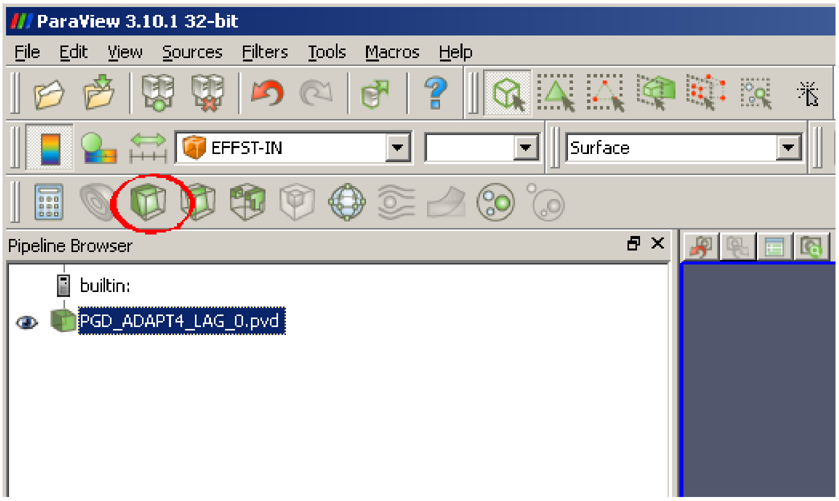
Scalar Value: 0
Opacity
Scale
Color Space: Diverging
☐ Use Logarithmic Scale
☑ Automatically Rescale to Fit Data Range
Minimum: 0
Maximum: 28.8104
☑ Use Discrete Colors
Resolution: 256

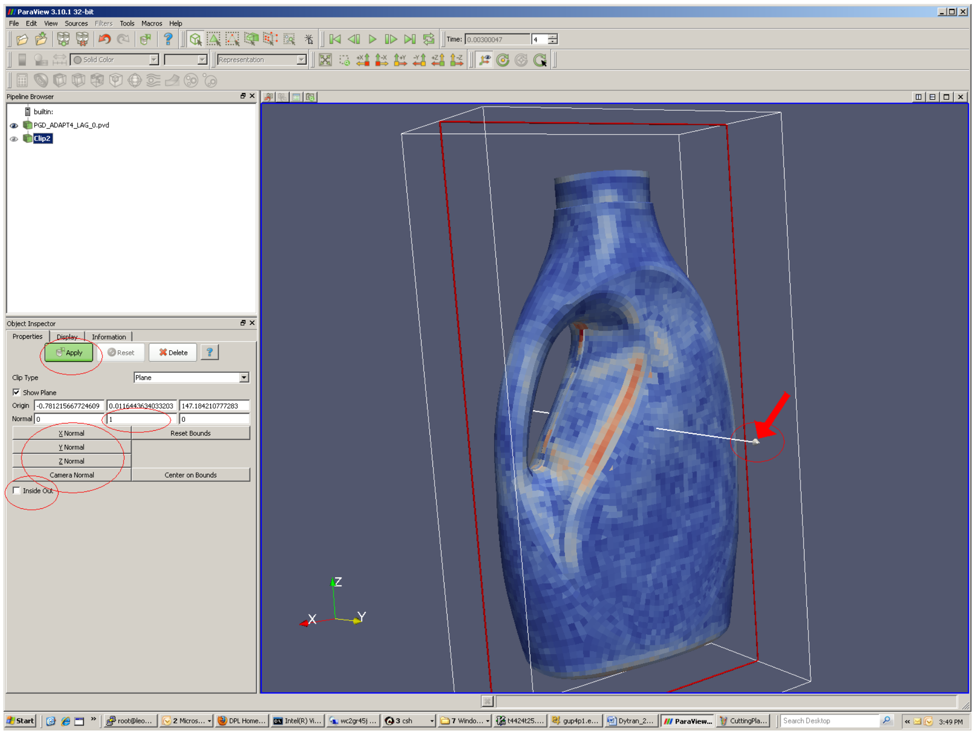
4. Applying a clipping plane.

   To look inside objects, you can apply one or more cutting planes.

   - Make sure the object on which you want to apply the cutting plane is the active one in the **Pipeline Browser**.
   - Click the **Cutting Plane** icon on the toolbar.
   - Position the cutting plane, either with the buttons and data fields on the **Properties** tab of the **Object Browser**
     or by manipulating the origin of the cutting plane, its normal or the borders with the mouse cursor in the display window.

- Click **Apply** when finished.





Clip Type: Plane

[✓] Show Plane

Origin	-0.781215667724609	0.0116443634033203	147.184210777283
Normal	0	1	0

[ ] Inside Out

5. Opening more than one series of results files.

   Opening a second or higher series of results files, is similar to the way the first series was opened, just use **File/Open** on the menu bar and browse to the `.pvd` file you want to add to your visualization.

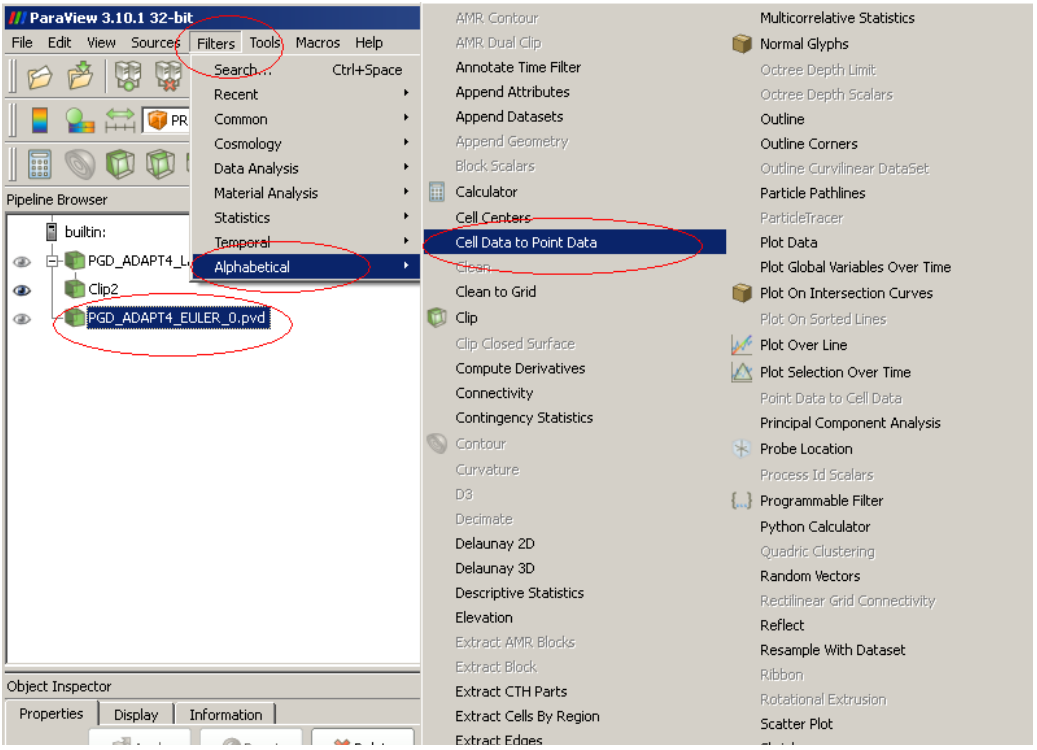
   The file appears as another object in the pipeline browser.

6. Creation of contour plots

   Most Dytran results are zone variables, that are translated to VTK "Cell data" and will look like a pile of blocks in ParaView. They can be examined with cutting planes, but most of the time you will also want to make contour plots on isosurfaces, vector plots or generate streamlines. Unfortunately, this can only be done with "Point data" in ParaView. However, there is a filter that can transfer "Cell data" into "Point data".

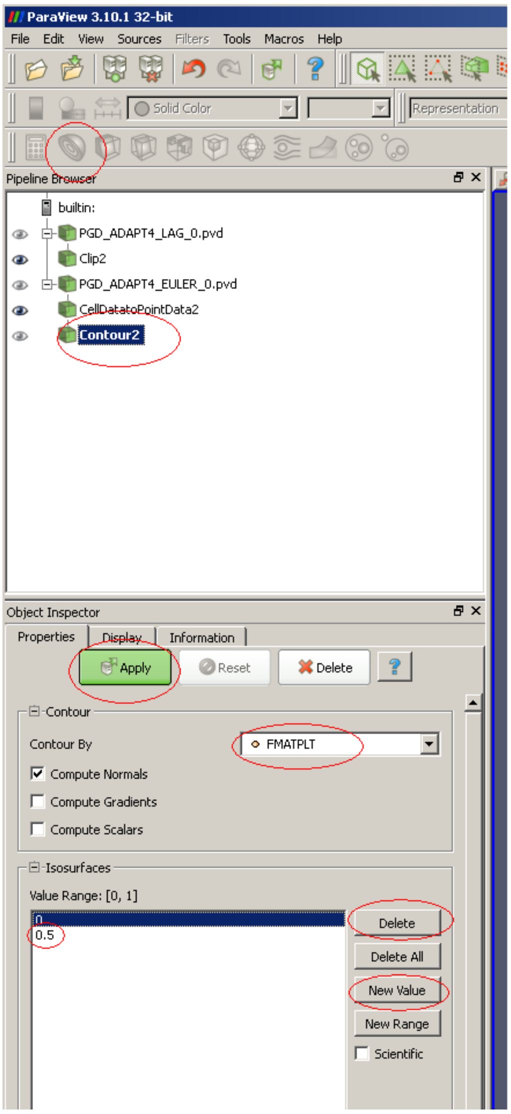
   - Make sure the object on which you want to apply the filter is selected in the **Pipeline Browser**.
   - Click on **Filters** in the menu bar.
   - Click **Alphabetical**.
   - Click **Cell to Point Data.**

   The translation of the "Cell data" and the creation of the "CellDatatoPointData" object may take some time.



Now we can create contour plots:

- Make sure the newly created **CellDatatoPointData** object is selected.
- Click on the **Contour** icon on the toolbar.
- Select the variable to be used for the creation of the isosurface in the **Object Selector** (here FMATPLT),
  use **New Value** to create the value for which you want to create the isosurface (here 0.5) and use **Delete** to remove the value Paraview did set as default (here 0).
- Click **Apply** to create the isosurface

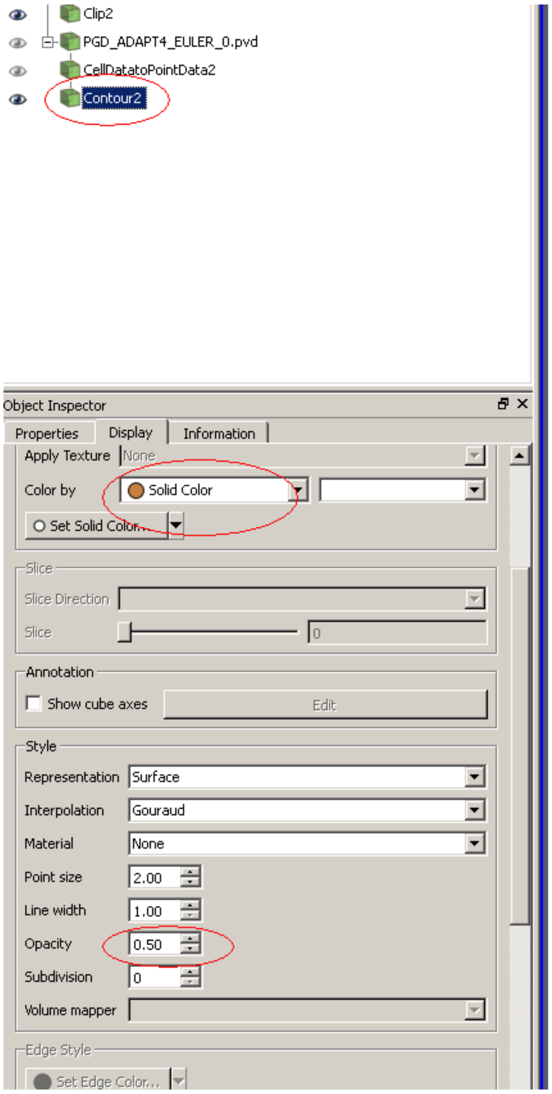


7. Transparency

The combination of mesh plot of the structure with a cutting plane and the iso-surface of the Euler mesh does not give information that is very clear in this situation.

You can apply cutting planes on the iso-surfaces as well and making them transparent.

- Select the object of the structure.
- Lower the value for **Opacity** on the **Display** tab of the **Object Inspector**.



Clip2
PGD_ADAPT4_EULER_0.pvd
CellDatatoPointData2
Contour2

Object Inspector

Properties | Display | Information

Apply Texture: None
Color by: Solid Color
Set Solid Color...

Slice
Slice Direction:
Slice: 0

Annotation
Show cube axes
Edit

Style
Representation: Surface
Interpolation: Gouraud
Material: None
Point size: 2.00
Line width: 1.00
Opacity: 0.50
Subdivision: 0
Volume mapper:

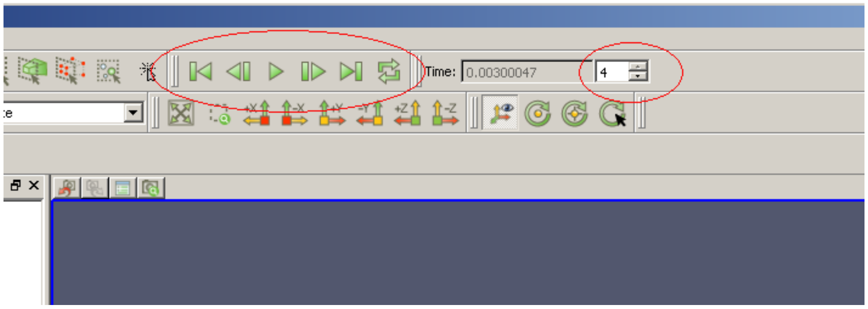
Edge Style
Set Edge Color...

8. Animation

   Use the "animation" buttons on the menu bar to play/stop animations of series of timesteps.

   The functions of the buttons are from left to right:

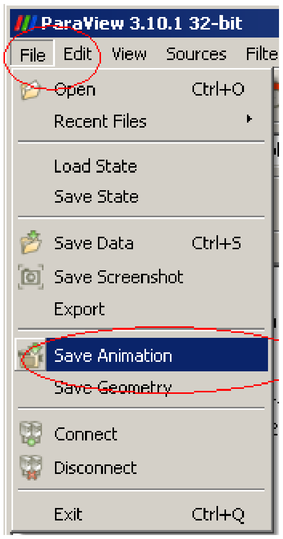
   - Jump to first frame
   - Go one frame backward
   - Start/Stop the animation
   - Go to the next frame
   - Jump to the last frame
   - Loop

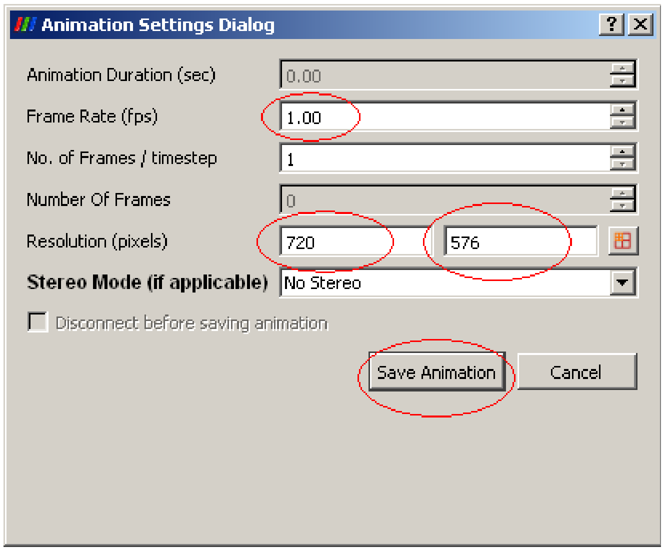


It is also possible to save animations as AVI files:

Click File on the menu bar and select **Save Animation**.

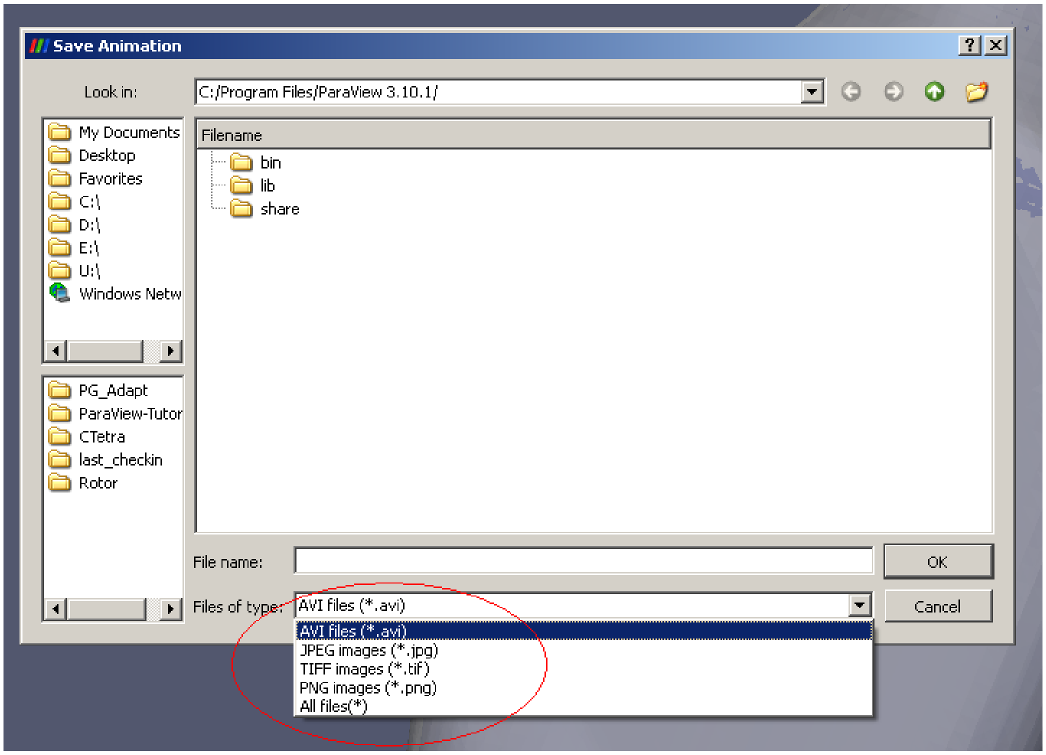
This will bring up the **Animation Settings Dialog** where you can set things like **Frame Rate** (frames per second), **No. of Frames / timestep**, and **resolution (pixels)**. For smaller numbers of time steps (10-25), the defaults look appropriate. If you have larger numbers of timesteps (say 100-500), you could consider to increase the number of frames per second. For professional videos, 12.5 (PAL) or 15 (NTSC) are appropriate values. Also, be aware that larger is not always better since many media players cannot handle large or/and non-standard sized frames without hickups. Standard PAL (720 x 576) or NTSC (640 x 480) are safe choices if you want to be sure that the result will be playing smoothly on any ones computer.





Animation Duration (sec) 0.00
Frame Rate (fps) 1.00
No. of Frames / timestep 1
Number Of Frames 0
Resolution (pixels) 720 576
Stereo Mode (if applicable) No Stereo
Disconnect before saving animation

Animations can be saved as AVI (unfortunately this AVI variant is not the most optimal with respect to file size), or separate frames in JPEG, PNG or TIFF format, that can be used by third party products like ImageMagick to create MPEGs.



Time History Plots

ParaView cannot read Dytran Time history files (THS) directly. They need to be translated to CSV files (comma separated value) first by arc2vtk.
Use:

```
arc2vtk.exe my_file_name.THS
```

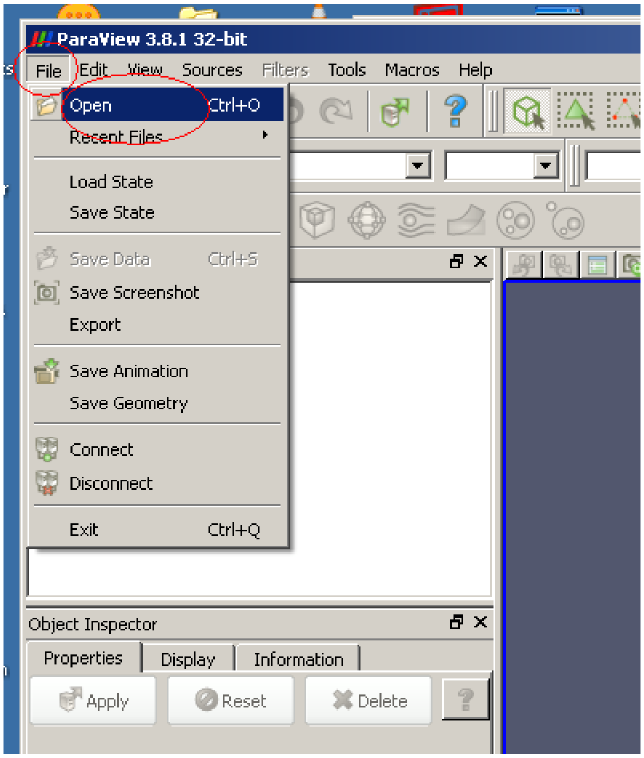
To create one or more CSV-files (comma separated values) from a Dytran time history file. For each of the "items" (gridpoint, element, face or material, depending on the type of THS-file), a CSV-file will be created named `my_file_name_X_NNN.csv`, with `X` replaced by the type of item (`N` for node, `E` for element, `F` for face, and `G` for global data) and `NNN` its identifier number. CSV files can be processed by ParaView, but also be used as input for spreadsheet programs like Excel.

A. Opening CSV-files in ParaView

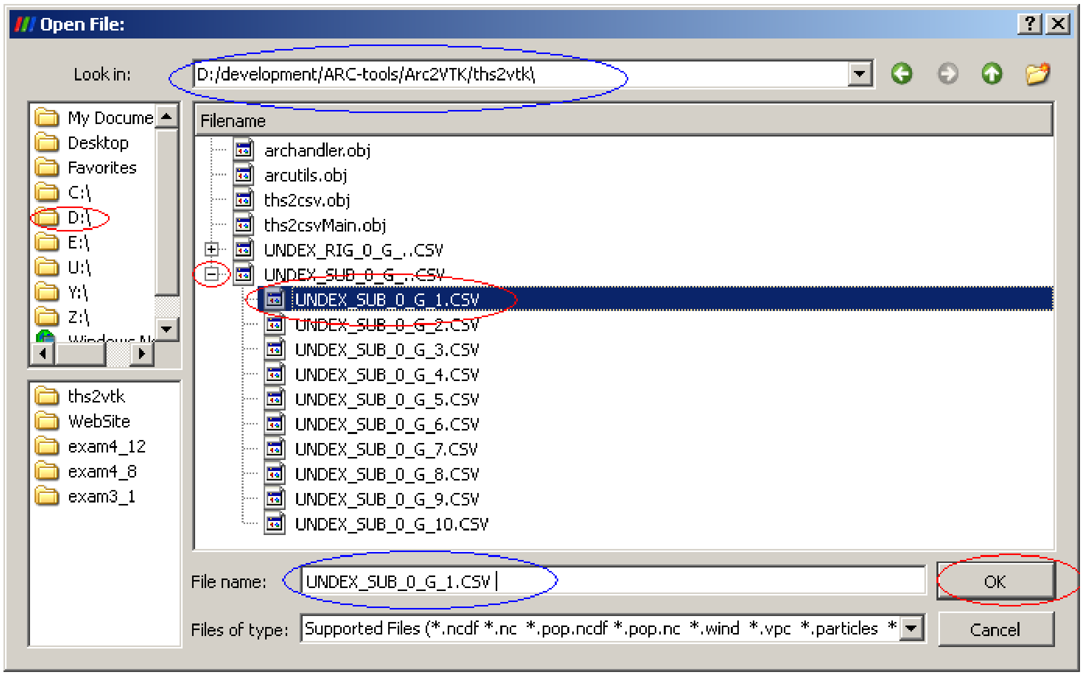
To open a CSV-file in ParaView use "File / Open" in the menu bar

Opening CSV-files in ParaView

To open a CSV-file in ParaView use **File/Open** in the menu bar:



This will pop-up an explorer window where you can select the file to be opened as usual. When there are multiple files with similar file names, only differing in a number, they will be displayed as a series. To select a single file, you need to expand the list first by clicking on the "+"-sign.


Open File:

Look in: D:/development/ARC-tools/Arc2VTK/ths2vtk\

My Docume
Desktop
Favorites
C:\
D:\
E:\
U:\
Y:\
Z:\

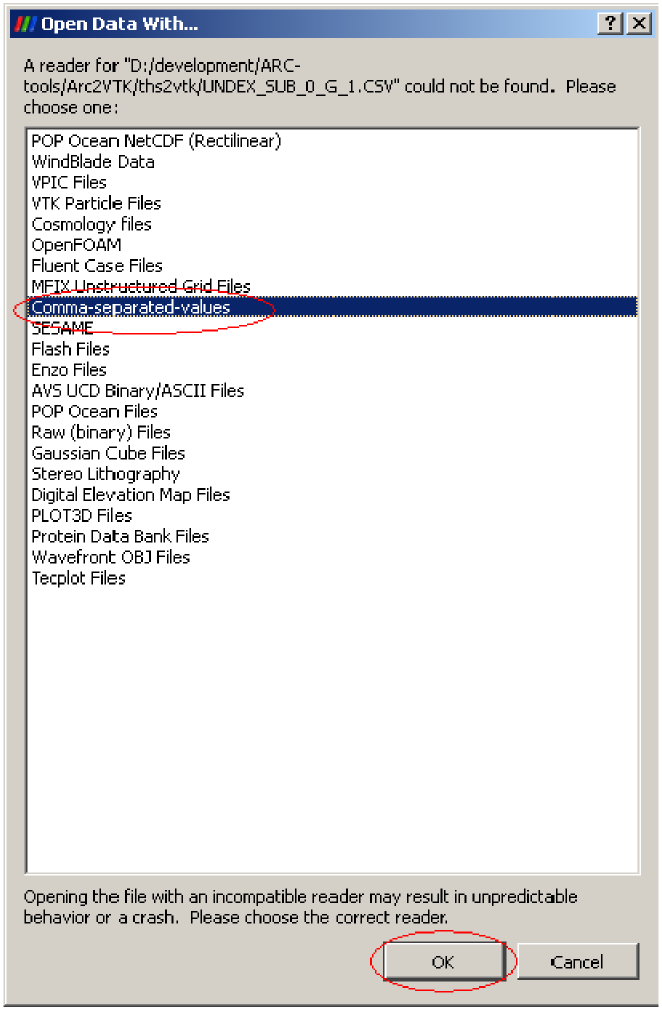
ths2vtk
WebSite
exam4_12
exam4_8
exam3_1

Filename
archandler.obj
arcutils.obj
ths2csv.obj
ths2csvMain.obj
UNDEX_RIG_0_G_..CSV
UNDEX_SUB_0_G_..CSV
UNDEX_SUB_0_G_1.CSV
UNDEX_SUB_0_G_2.CSV
UNDEX_SUB_0_G_3.CSV
UNDEX_SUB_0_G_4.CSV
UNDEX_SUB_0_G_5.CSV
UNDEX_SUB_0_G_6.CSV
UNDEX_SUB_0_G_7.CSV
UNDEX_SUB_0_G_8.CSV
UNDEX_SUB_0_G_9.CSV
UNDEX_SUB_0_G_10.CSV

File name: UNDEX_SUB_0_G_1.CSV

Files of type: Supported Files (*.ncdf *.nc *.pop.ncdf *.pop.nc *.wind *.vpc *.particles *

For some reason, ParaView is aware of `.csv` being the extension of a supported file type, but it apparently doesn't know, what reader has to be used. Select the **Comma-separated-values** reader in the window that pops-up.

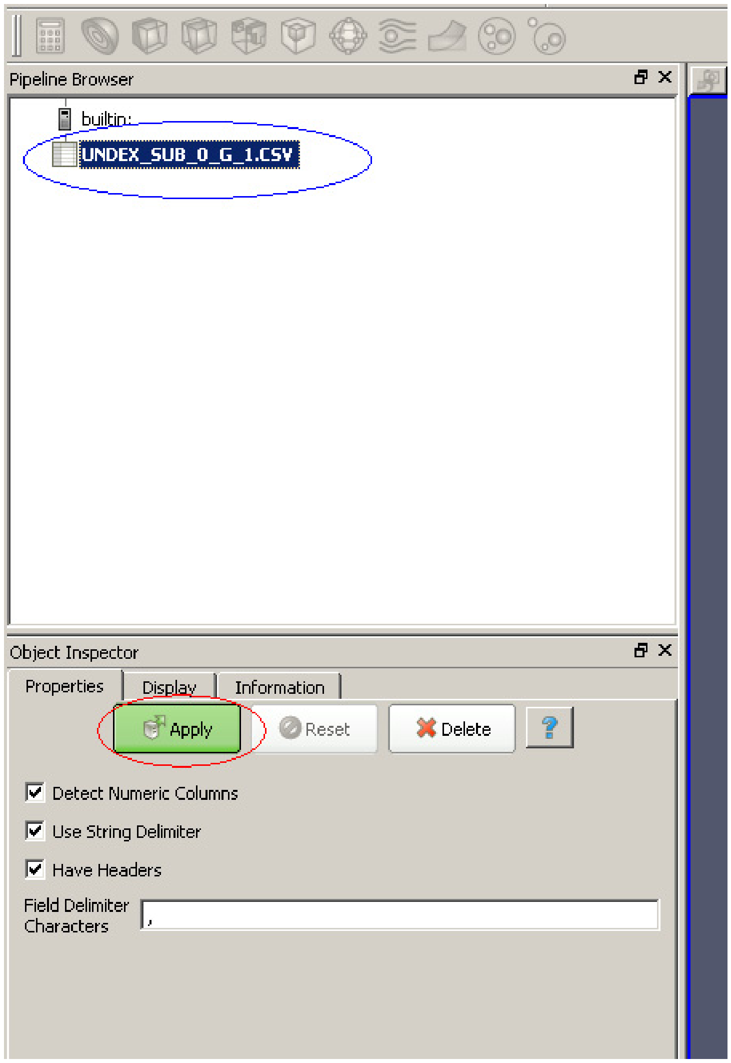

Open Data With...

A reader for "D:/development/ARC-tools/Arc2VTK/ths2vtk/UNDEX_SUB_0_G_1.CSV" could not be found. Please choose one:

POP Ocean NetCDF (Rectilinear)
WindBlade Data
VPIC Files
VTK Particle Files
Cosmology files
OpenFOAM
Fluent Case Files
MFIX Unstructured Grid Files
Comma-separated-values
SESAME
Flash Files
Enzo Files
AVS UCD Binary/ASCII Files
POP Ocean Files
Raw (binary) Files
Gaussian Cube Files
Stereo Lithography
Digital Elevation Map Files
PLOT3D Files
Protein Data Bank Files
Wavefront OBJ Files
Tecplot Files

Opening the file with an incompatible reader may result in unpredictable behavior or a crash. Please choose the correct reader.

The name of your CSV file will be listed in the **Pipeline Browser, but** you still need to click the **Apply** button, to start the actual read.



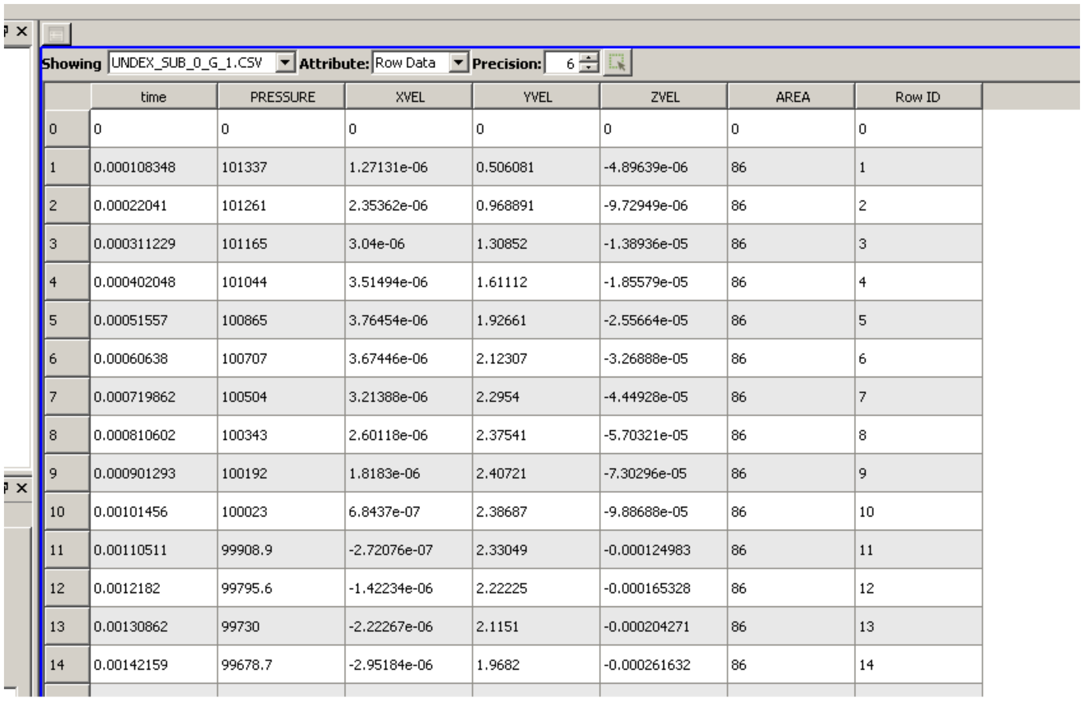
☑ Detect Numeric Columns

☑ Use String Delimiter

☑ Have Headers

Field Delimiter Characters ,

The contents of the CSV file are being displayed as a spreadsheet:

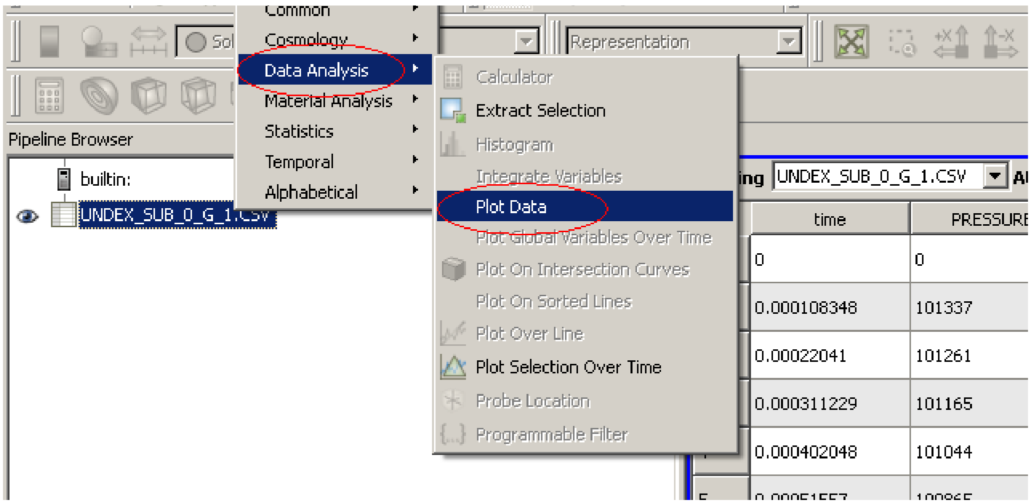


Showing UNDEX_SUB_0_G_1.CSV Attribute: Row Data Precision: 6

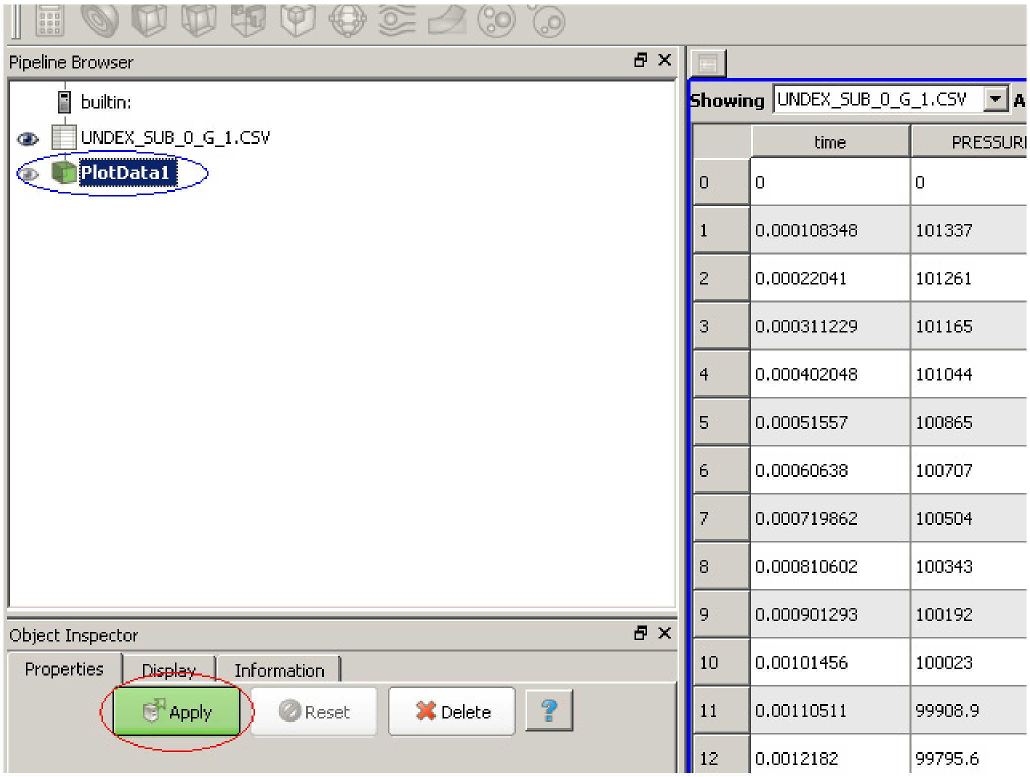
	time	PRESSURE	XVEL	YVEL	ZVEL	AREA	Row ID
0	0	0	0	0	0	0	0
1	0.000108348	101337	1.27131e-06	0.506081	-4.89639e-06	86	1
2	0.00022041	101261	2.35362e-06	0.968891	-9.72949e-06	86	2
3	0.000311229	101165	3.04e-06	1.30852	-1.38936e-05	86	3
4	0.000402048	101044	3.51494e-06	1.61112	-1.85579e-05	86	4
5	0.00051557	100865	3.76454e-06	1.92661	-2.55664e-05	86	5
6	0.00060638	100707	3.67446e-06	2.12307	-3.26888e-05	86	6
7	0.000719862	100504	3.21388e-06	2.2954	-4.44928e-05	86	7
8	0.000810602	100343	2.60118e-06	2.37541	-5.70321e-05	86	8
9	0.000901293	100192	1.8183e-06	2.40721	-7.30296e-05	86	9
10	0.00101456	100023	6.8437e-07	2.38687	-9.88688e-05	86	10
11	0.00110511	99908.9	-2.72076e-07	2.33049	-0.000124983	86	11
12	0.0012182	99795.6	-1.42234e-06	2.22225	-0.000165328	86	12
13	0.00130862	99730	-2.22267e-06	2.1151	-0.000204271	86	13
14	0.00142159	99678.7	-2.95184e-06	1.9682	-0.000261632	86	14

To create XY-graphs, you need to apply a “Plotdata” filter:

- Open the filter menu on the menubar on top of the ParaView window
- Select the **Data Analysis** submenu and select **Plot Data**



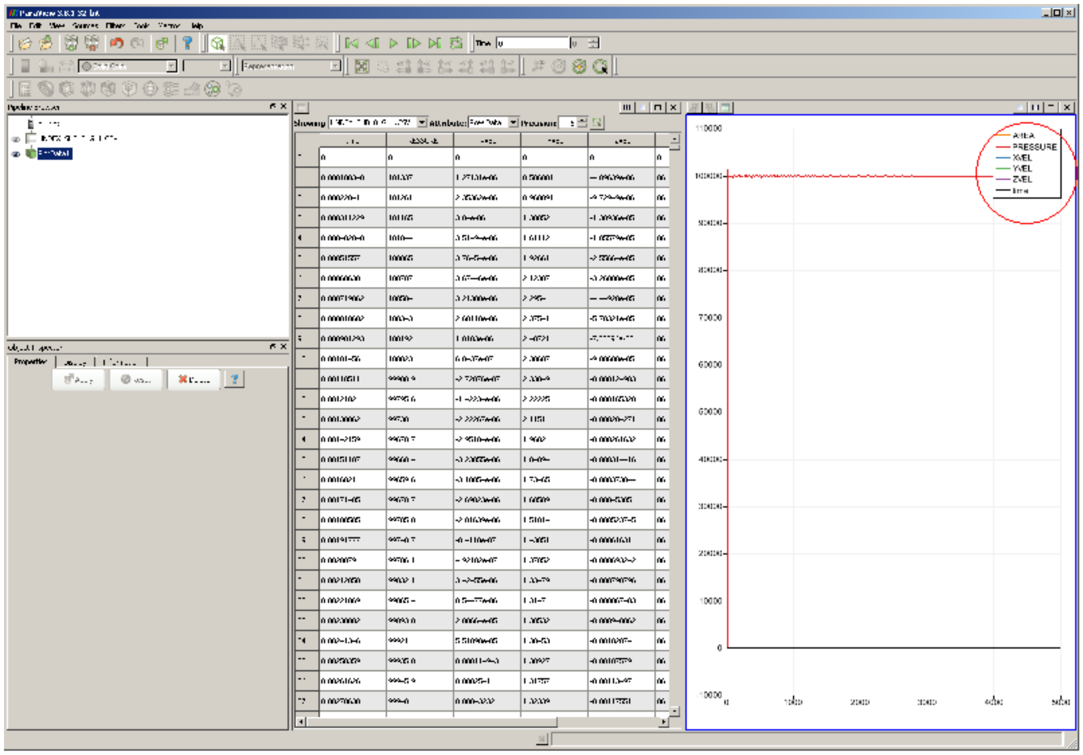
In ParaView, the **Apply** button has to be clicked to start the actual work.



	time	PRESSURI
0	0	0
1	0.000108348	101337
2	0.00022041	101261
3	0.000311229	101165
4	0.000402048	101044
5	0.00051557	100865
6	0.00060638	100707
7	0.000719862	100504
8	0.000810602	100343
9	0.000901293	100192
10	0.00101456	100023
11	0.00110511	99908.9
12	0.0012182	99795.6

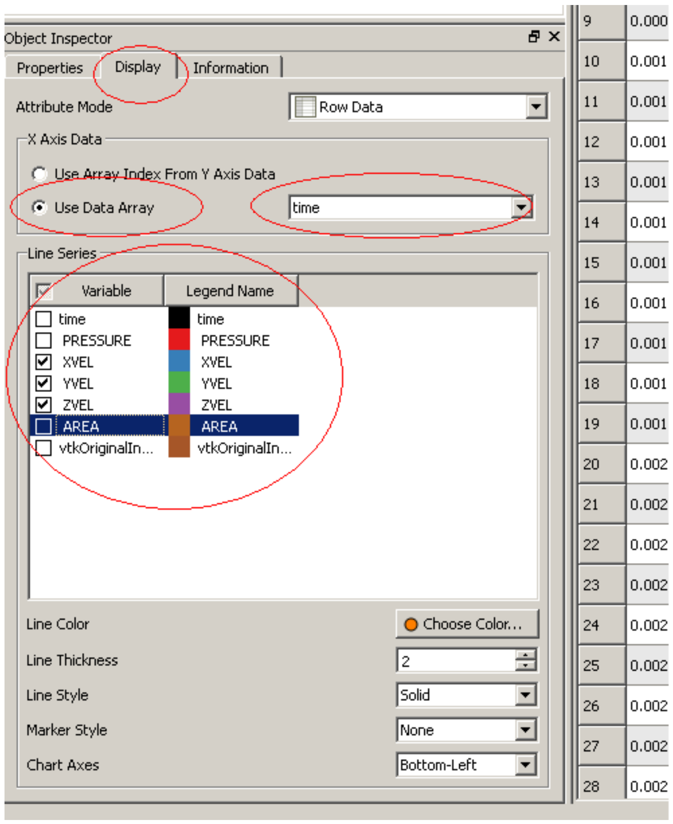
Now the screen looks like this:

The usual **Pipeline Browser** and **Object Inspector** on the left, a spreadsheet in the middle, and a narrow XY-graph at the right.



The graph contains curves of all variables found on the CSV file, plotted against the same scale. This is most likely not what you want. First deselect the variables you do not want in the current graph:

- Make sure the window with the graph is selected (border colored blue).
- Open the **Display** tab on the **Object Inspector**
- Deselect the variables you don't want
- Probably you also want to change the X-Axis to **time**:
  Select **Use Data Array**
  Sometimes it's necessary to toggle variable **time** on and off to force a redraw of the graph.



Attribute Mode: Row Data

X Axis Data
- Use Array Index From Y Axis Data
- [✓ Use Data Array] time

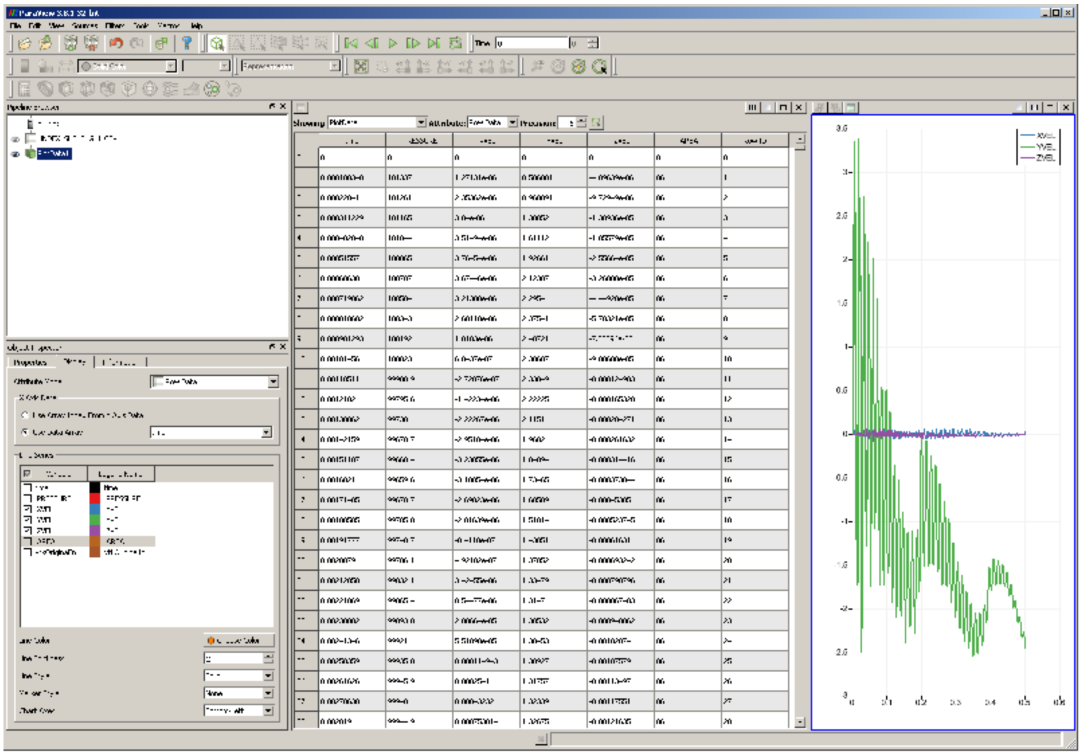
Line Series

Variable	Legend Name
☐ time	■ time
☐ PRESSURE	🟥 PRESSURE
☑ XVEL	🟦 XVEL
☑ YVEL	🟩 YVEL
☑ ZVEL	🟪 ZVEL
☐ AREA	🟧 AREA
☐ vtkOriginalIn...	🟫 vtkOriginalIn...

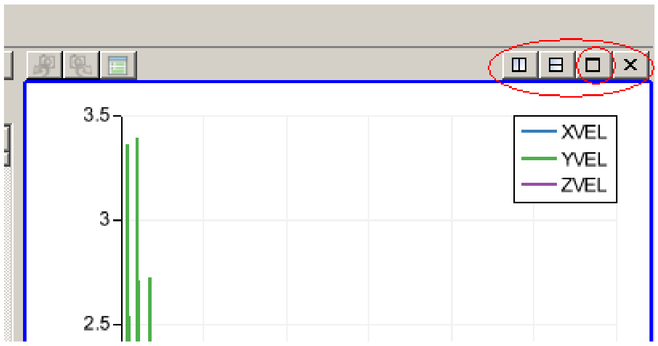
Line Color: Choose Color...
Line Thickness: 2
Line Style: Solid
Marker Style: None
Chart Axes: Bottom-Left

9	0.000
10	0.001
11	0.001
12	0.001
13	0.001
14	0.001
15	0.001
16	0.001
17	0.001
18	0.001
19	0.001
20	0.002
21	0.002
22	0.002
23	0.002
24	0.002
25	0.002
26	0.002
27	0.002
28	0.002

The ParaView window will look like this:



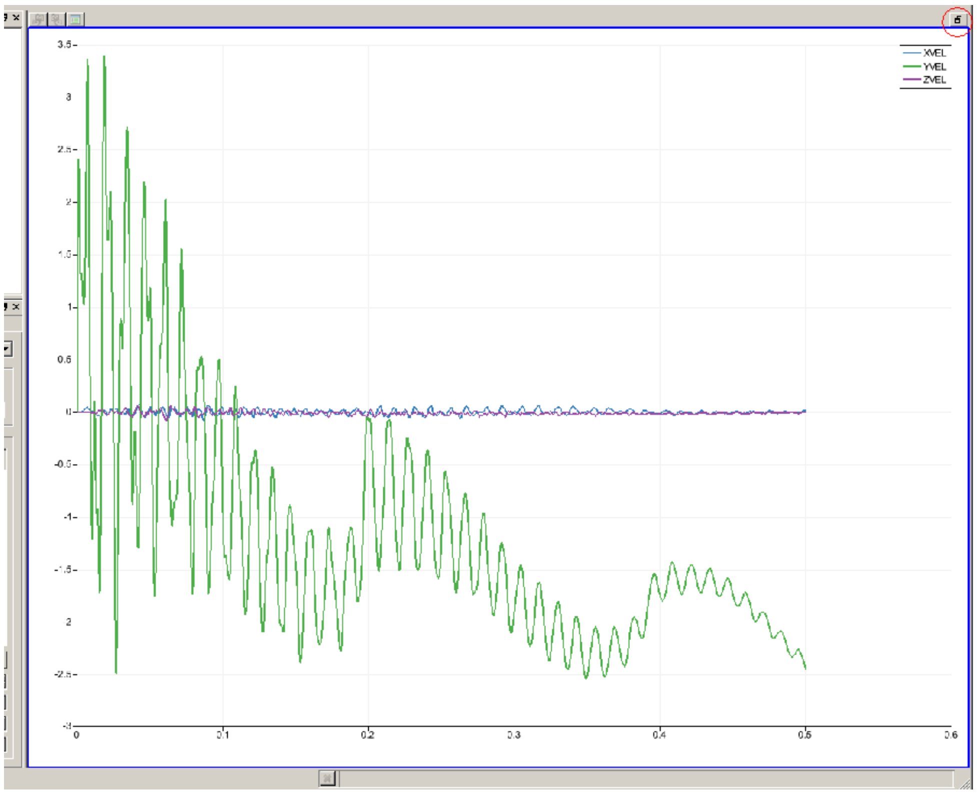
Use the **Resize Window** buttons at the right top of the graph window to expand it to full size.



Giving the final result below.

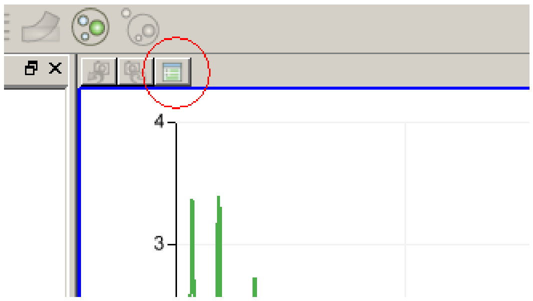
You can play with the size and aspect ratio of the Paraview window to get the format you want.

Use the **Resize Window** button on the top right to go back to the initial situation.



Adding text to XY-plots

Click the **View Settings** button on the left side on top of the graph window to add headers or/and axis labels to a plot or to change the appearance of the axes.



This brings up the **View Settings** menu where you can add a title or add labels to the axes.

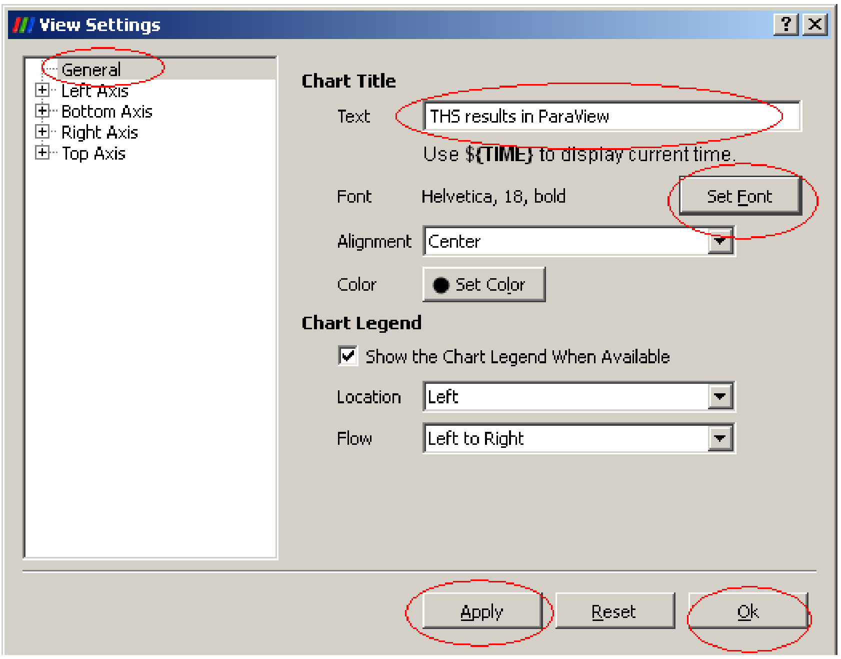


Chart Title

Text: THS results in ParaView

Use ${TIME} to display current time.

Font: Helvetica, 18, bold

Alignment: Center

Color: ● Set Color

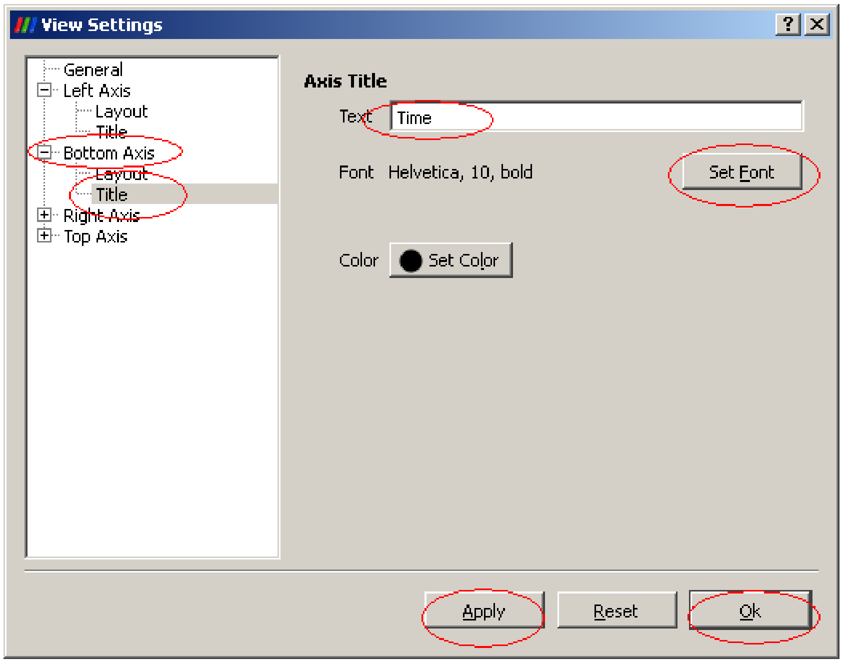
Chart Legend

[✓] Show the Chart Legend When Available

Location: Left

Flow: Left to Right

With **Layout**, it is also possible to define the axis spacing manually or switch to logaritmic scale.

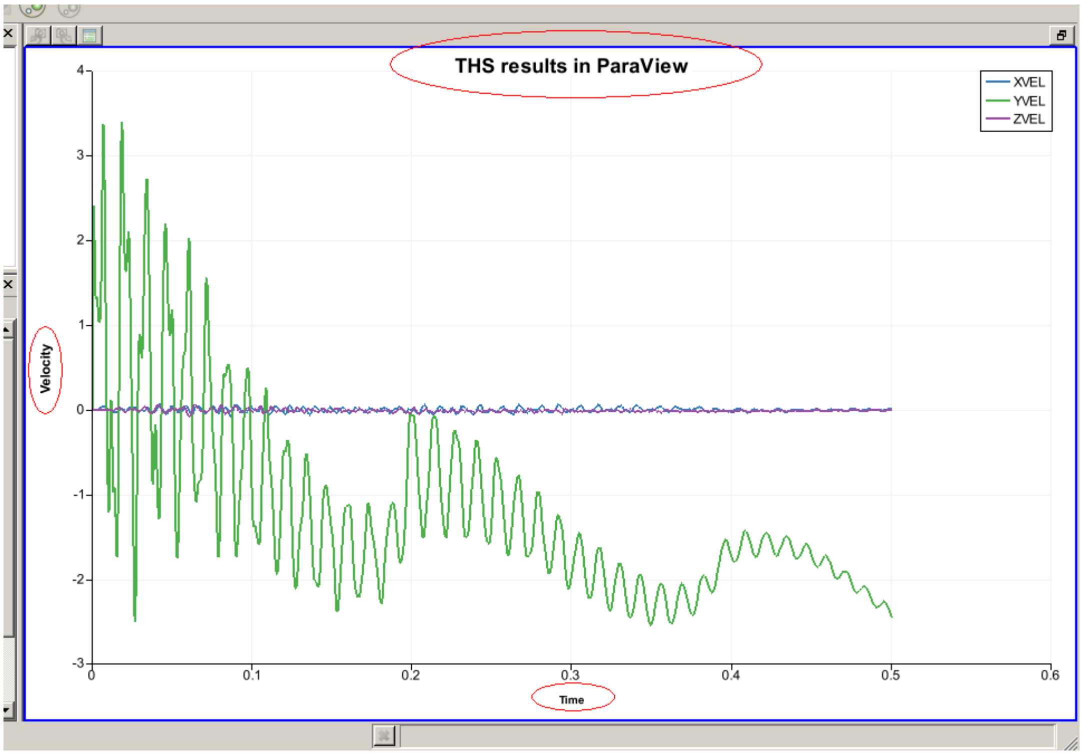


Text Time

Font Helvetica, 10, bold

Color ● Set Color

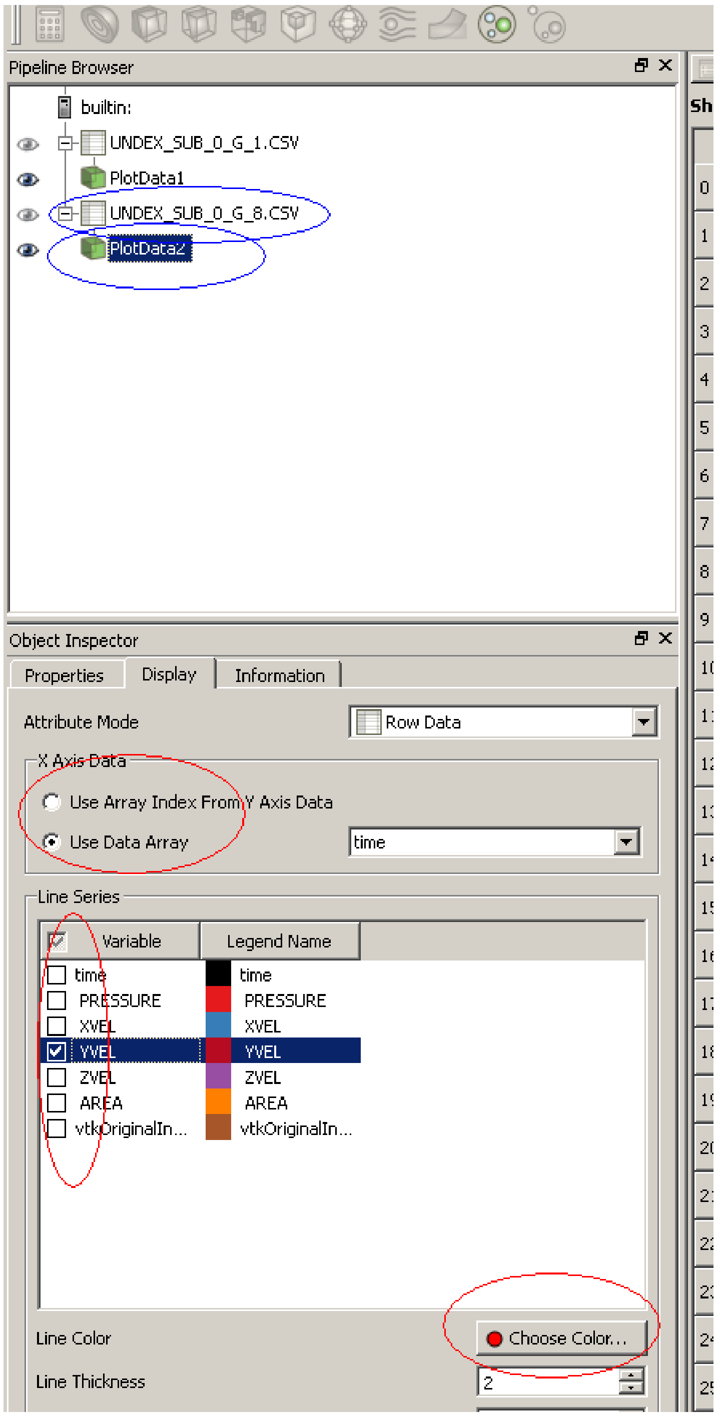
The final result may look like this:



Processing Data from Multiple CSV Files

It is possible to combine the results from multiple files in one graph.

- First create a graph with the results of one CSV file as described above.
- Also open the second file and apply a **PlotData** filter to it.
- Make sure the window with the graph of the first file is active (blue border).
- Select the second **PlotData** filter in the **Pipeline Browser** and change the settings on the **Display** tab of the **Object Inspector**.
- The curve also appears in the active window.
- Use **Choose Color..** to open a menu where you can change the color of the curve of the selected (blue background) variable.



Pipeline Browser

- builtin:
  - UNDEX_SUB_0_G_1.CSV
    - PlotData1
  - UNDEX_SUB_0_G_8.CSV
    - PlotData2

Object Inspector

Properties | Display | Information

Attribute Mode: Row Data

X Axis Data
- ( ) Use Array Index From Y Axis Data
- (●) Use Data Array: time

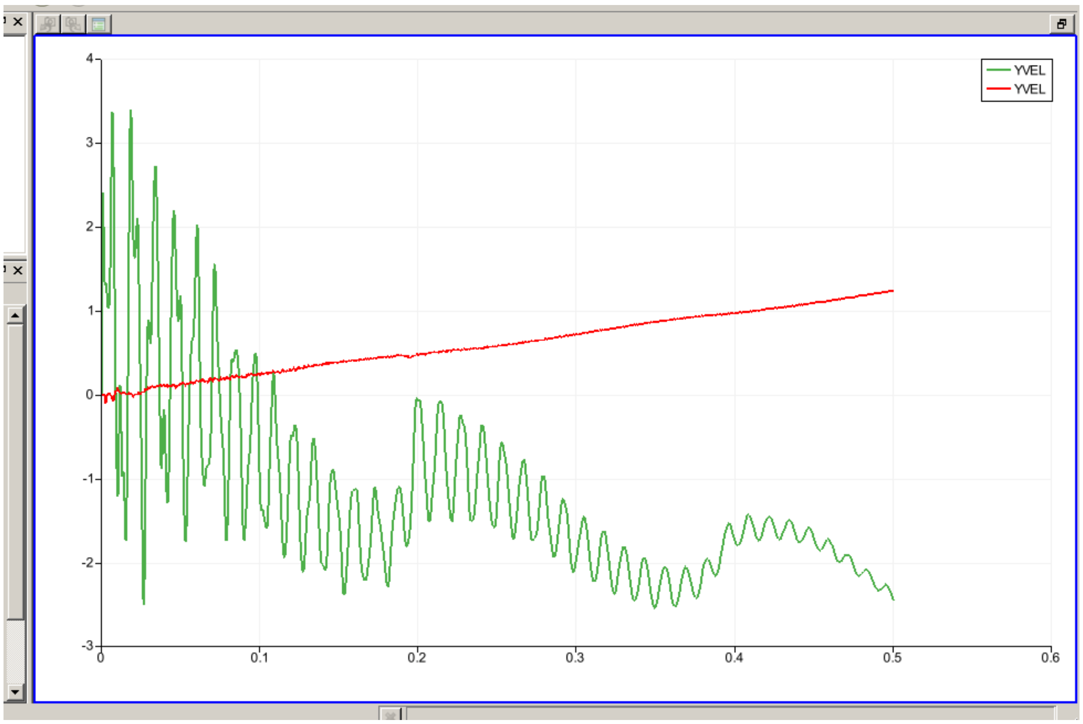
Line Series

	Variable		Legend Name
☐	time	⬛	time
☐	PRESSURE	🟥	PRESSURE
☐	XVEL	🟦	XVEL
☑	YVEL	🟥	YVEL
☐	ZVEL	🟪	ZVEL
☐	AREA	🟧	AREA
☐	vtkOriginalIn...	🟫	vtkOriginalIn...

Line Color: ● Choose Color...

Line Thickness: 2

The final result may look like this:



A References

1. *MSC/DYNA User's Manual*, The MacNeal-Schwendler Corporation.
2. *MSC/PISCES-2DELK User's Manual*, The MacNeal-Schwendler Corporation.
3. E. L. Lee et al., "Adiabatic Expansions of High Explosive Detonation Products", UCRL-50422, May 1968, Lawrence Livermore National Laboratory, Livermore, California.
4. *MADYMO User's Manual 3-D Version 4.3*, The Dutch Institute of Applied Scientific Research - Road/Vehicles Research Institute, Department of Injury Prevention, October 1988.
5. Louise A. Obergefell, Thomas R. Gardner, Ints Kaleps, and John T. Fleck, *Articulated Total Body Model Enhancements*, Volume 1: "Modifications", (NTIS No. ADA198726).
6. Louise A. Obergefell, Thomas R. Gardner, Ints Kaleps, and John T. Fleck, *Articulated Total Body Model Enhancements*, Volume 2: "User's Guide", (NTIS No. ADA203566).
7. Louise A. Obergefell, Thomas R. Gardner, Ints Kaleps, and John T. Fleck, *Articulated Total Body Model Enhancements*, Volume 3: "Programmer's Guide", (NTIS No. ADA197940).
8. J. T. Fleck, F. E. Butler, and N. J. Deleys, "Validation of the Crash Victim Simulator", Calspan Report Nos. ZS-5881-V-1 through 4, DOT-HS-806-279 through 282, 1982, Volumes 1 through 4, (NTIS No. PC E99, PB86-212420).
9. P. L. Roe, *Approximate Riemann Solvers, parameter vectors, and difference schemes,* Journal of Computational Physics, 43, 357-372, 1981.
10. P. L. Roe and J. Pike, *Efficient construction and utilisation of approximate Riemann solutions,* Computing Methods in Applied Sciences and Engineering (VI), R. Glowinski and J. L. Lions (Editors), Elsevier Publishers B.V. (North Holland), INRIA 1984.
11. Bram van Leer, Chang-Hsien Tai, and Kenneth G. Powell, *Design of Optimally Smoothing Multi-Stage Schemes for the Euler Equations,* AIAA-89-1933-CP, 1989.
12. Ch. Hirsch, *Numerical Computation of Internal and External Flows, Fundamentals of Numerical Discretization, 1*, Wiley, Chichester, 1988.
13. Ch. Hirsch, *Numerical Computation of Internal and External Flows, Computational Methods for Inviscid and Viscous Flows, 2,* Wiley, Chichester, 1990
14. Louise A. Obergefell, Huaing Cheng, Annnette L. Rizer, *Articulated Total Body Model Version V*: "User's Manual", (NTIS No. ASC-98-0807), 1998.
15. Louise A. Obergefell, Huaing Cheng, Annnette L. Rizer, *Input Description for the Articulated Total Body Model ATB-V.1*, 1998.
16. Joseph A. Pellettiere, Huaing Cheng, Annnette L. Rizer, *The Development of GEBOD Version V*, 2000.
17. S.Tanimura, K.Mimura, and W.H. Zhu, *Practical Constitutive Models Covering Wide Ranges of Strain Rates, Strains and Temperatures,* Key Engineering Materials Vols. 177-180,189-200, 2000
18. W.A. van der Veen, "Simulation of a compartmented airbag deployment using an explicit, coupled Euler/Lagrange method with adaptive Euler Domains", Presented at: Nafems 2003, Orlanda. Available at the Dytran section of www.mscsoftware.com.