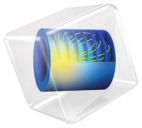


COMSOL Multiphysics

Mesh Import and Export Guide

6.2

COMSOL

COMSOL Multiphysics Mesh Import and Export Guide

Version: COMSOL 6.2

Contact Information

Visit the Contact COMSOL page at www.comsol.com/contact to submit general inquiries or search for an address and phone number. You can also visit the Worldwide Sales Offices page at www.comsol.com/contact/offices for address and contact information.

If you need to contact Support, an online request form is located on the COMSOL Access page at www.comsol.com/support/case. Other useful links include:

- Support Center: www.comsol.com/support
- Product Download: www.comsol.com/product-download
- Product Updates: www.comsol.com/product-update
- COMSOL Blog: www.comsol.com/blogs
- Discussion Forum: www.comsol.com/forum
- Events: www.comsol.com/events
- COMSOL Video Gallery: www.comsol.com/videos
- Support Knowledge Base: www.comsol.com/support/knowledgebase

Part number: CM020015

Contents

Importing and Exporting Mesh and Interpolation Data

In COMSOL Multiphysics, you can import mesh and interpolation data created by other software for use in your simulations. You can also export data created in COMSOL Multiphysics to use in other COMSOL models and applications as well as other software. This document and accompanying example files give an overview of the import and export functionality available in COMSOL Multiphysics and a detailed description of the import, export, and native COMSOL formats. For more detailed information on how to use COMSOL Multiphysics for import and export of data, see the *COMSOL Multiphysics Reference Manual* in the installed documentation set.

Overview of Data Formats

Table 1 shows the mesh and interpolation data file types that COMSOL Multiphysics can import and export. The table also gives an overview of what type of data the files support (volume mesh and/or surface mesh) as well as what type of data can be imported into and exported from COMSOL Multiphysics on the respective formats.

TABLE 1: FILE FORMATS SUMMARY

FILE FORMAT	EXTENSION	VOLUME MESH	SURFACE MESH	INTERPOLATION DATA IMPORT	IMPORT AS GEOMETRY	IMPORT AS MESH	EXPORT	SELECTIONS TRANSFER
3MF	.3mf	No	Yes	No	Yes	Yes	Yes	No
COMSOL native binary	.mphbin	Yes	Yes	No	Yes	Yes	Yes	Import and export
COMSOL native text	.mphtxt	Yes	Yes	No	Yes	Yes	Yes	Import and export
COMSOL sectionwise	.txt	Yes	Yes	Yes	Yes	Yes	Yes	No
COMSOL spreadsheet	.txt	No	No	Yes	Yes[1]	No	Yes	No
COMSOL grid	.txt	Yes	No	Yes	No	No	Yes	No
glTF	.glb	No	Yes	No	No	No	Yes	No
NASTRAN	.nas .bdf .nastran .dat	Yes	Yes	No	Yes	Yes	Yes	Import
PLY	.ply	No	Yes	No	Yes	Yes	Yes	No
STL	.stl	No	Yes	No	Yes	Yes	Yes	Import
VRML, v1	.wrl, .vrml	No	Yes	No	Yes	Yes	No	No
VTK	.vtu	Yes	Yes	No	No	No	Yes	No

[1] Can be imported as geometry curve objects

The COMSOL native binary and text file formats can also be used for storing CAD data, which is not covered by this document. For a mesh data file, there will be exactly one mesh defined in the file. Mixing geometry and mesh

data in the same file is not supported. The COMSOL sectionwise, spreadsheet, and grid data interpolation formats can be used for exporting and importing simulation and geometric data.

Note that Table 1 only includes file formats for mesh and interpolation data and that many other types of file formats are supported by COMSOL Multiphysics as well as by various add-on products. For a complete list of file formats, see *Introduction to COMSOL Multiphysics* or the COMSOL web page.

CONVERSIONS ON IMPORT

The import functionality includes the capability of partitioning the imported mesh data into multiple domains and boundary entities, either automatically or by manual partition operations. Imported mesh data can be used in two different ways: directly as a mesh used for analysis or indirectly as a geometry object. This is indicated in Table 1 in the columns *Import as Geometry* and *Import as Mesh*. It is recommended to work with the mesh directly, using the Operations for Editing Imported Meshes to repair, combine, and remesh the imported meshes. As an alternative, when importing volumetric or surface mesh data, COMSOL Multiphysics can convert the surface part of the data into a solid COMSOL geometry object that can further be used for generating a new mesh that is different from the original mesh.

When importing volumetric mesh data directly, COMSOL Multiphysics can retain the original mesh, such as a tetrahedral mesh, and use the mesh data in any type of physics analysis, regardless of the intent of the original imported mesh. For example, a mesh generated by another software, originally intended for structural analysis, can be imported into COMSOL Multiphysics and used for fluid flow, acoustics, electromagnetics, etc. The functionality in COMSOL Multiphysics allows the user to change the element type, also called the shape function, defined on an imported mesh. For example, the imported mesh may have originally been used for a structural analysis based on so-called serendipity elements. After being imported to COMSOL Multiphysics, the same mesh can be used in the RF Module for an electromagnetic wave propagation analysis, where a completely different type of element is needed; so-called vector elements. The conversion of the element type, or shape function, is done automatically. In addition, mixed elements can be used for a multiphysics analysis where each participating physics defines its own type of element. See Second-Order Element Data for more information about how the element order is handled.

USING MESH AND SIMULATION DATA SIMULTANEOUSLY

For simultaneous use of simulation and mesh data, you import the simulation data as one or more *Interpolation* functions and the mesh data in a model tree *Mesh>Import* node in the software.

Overview of Data Export

Mesh data can be exported from the model tree *Mesh* node and from the *Export* node under *Results*. From the *Mesh* node, you can export 2D meshes and volumetric meshes, such as a tetrahedral mesh, to a COMSOL native format, the COMSOL sectionwise file, or a NASTRAN file. In addition, you can export the surface part of a volumetric mesh to the STL, PLY, or 3MF file format.

From the model tree *Results>Export* node, you can export the mesh, simulation, and other data in several ways, including mesh data used for visualization that is not directly related to the mesh used for simulation, as described in the section Exporting Data from the Results Node.

EXPORTING MESH DATA FROM THE MESH NODE

The different file types present different capabilities of what type of data to export from the model tree *Mesh* node.

Exporting to the COMSOL Native Formats

For export of a mesh to a COMSOL native binary (.mphbin) or text (.mphtxt) file, you can export all types of elements: domain elements, boundary elements, edge elements (available in 3D), and vertex elements (available in 2D and 3D). You can also export geometric entity information. The export operation then also writes information on the corresponding geometric entity index for each element. See the section Mesh Elements for 1D, 2D, and 3D

Meshes and Complete Mesh Data on the COMSOL Native Format for more information. You can include selections that are defined on the current geometry in the mesh export. Such selections can then be included when importing a mesh, as described in the section Selections. The exported data can include second-order element node information. If the mesh to export is an imported linear mesh, the information about how to place the second-order nodes will be estimated from the linear mesh. See also Second-Order Element Data. The COMSOL native formats always store data with double precision. See Specification of COMSOL Formats for Mesh Data for more information.

For the sectionwise (.txt) format, you specify one type of elements to export: domain, boundary, or edge mesh data.

Exporting to the NASTRAN Format

Export of a 2D or 3D mesh to a NASTRAN file can include the following types of elements: domain elements and boundary elements (available in 3D). By selecting to export the geometric entity information, the export operation also writes information on each element's corresponding geometric entity index to the corresponding property identification field of the resulting NASTRAN file.

The output NASTRAN file can be stored in the small field format (single precision), large field format (double precision), or free field format (comma separated). These options are specific to the NASTRAN format. For more information, see the various online resources for the NASTRAN format. By default, second-order elements are exported. If the mesh to export is an imported linear mesh, the information about how to place the second-order nodes will be estimated from the linear mesh. Exporting the linear, or first-order, element information is also supported. Second-order information is then ignored.

Exporting to the STL, PLY, and 3MF Formats

You can export the boundary elements of a 3D mesh to a STL, PLY, or 3MF file. For STL and PLY files, you can choose binary or text format. For any of the STL, PLY, or 3MF file formats, second-order elements are converted to first-order elements. Because the STL and 3MF formats only support triangles, quadrilateral elements (four-sided elements) are exported as pairs of triangles. The PLY format supports both triangles and quadrilateral elements.

The 3MF format supports two modes of export: the entire surface mesh as one 3MF object of Surface type, or export of the surface mesh of each domain as a separate 3MF object of Model type and the remaining surface mesh, corresponding to faces not adjacent to any domain, as a 3MF object of Surface type.

EXPORTING DATA FROM THE RESULTS NODE

From the model tree *Results>Export* node, you can export data in a number of ways, as described below.

Exporting Mesh Data

Mesh data can be exported to the COMSOL native file format. For a solution in 3D it is also possible to export to an STL file (surface mesh part only). This is useful when exporting the mesh data from a deformed configuration, the result of a topology optimization, or a (geometric) parameter configuration from a parametric sweep.

Exporting Mesh and Simulation Data

You can export mesh simulation data to the VTK format or one of the sectionwise, spreadsheet, or grid interpolation formats. Note that data sampled on a grid can be exported to either the spreadsheet format or the grid format. The spreadsheet format is used for both unstructured mesh and structured grid export. Data can also be exported from a point set defined by a coordinate file. For more information on the interpolation file formats, see Interpolation Data Formats.

Exporting Mesh and Interpolation Plot Data

You can export the data contained in a plot to the STL, PLY, 3MF, VTK, sectionwise, or spreadsheet format. When exporting plot data to the sectionwise format, the program will export triangular, edge, or vertex element data, depending on the plot type. For example, an isosurface plot will be exported as triangle elements and a line plot will be exported as edge elements.

Exporting Image Data

You can export image data to the 3D glTF format and a number of 2D image formats. In addition, image data can be exported to a PowerPoint presentation.

Overview of Data Import

Depending on the format used, a data file can be imported and used as a computational mesh, geometry, or interpolation data. See Table 1. Imported mesh data can be used in two different ways: directly as a mesh used for analysis (recommended) or indirectly as a geometry object.

The following tutorial series showcases how imported STL data can be used in the software:

https://www.comsol.com/model/stl-import-tutorial-series-30951

IMPORTING MESH DATA AS A COMPUTATIONAL MESH

If you wish to use an external mesh directly as the computational mesh in COMSOL Multiphysics, you can import volumetric mesh data from a file in the following formats: COMSOL native binary, COMSOL native text, COMSOL sectionwise, or NASTRAN. The length unit specified in COMSOL sectionwise files is used to scale the mesh during the import. For modeling in 2D, the same formats are supported when importing 2D mesh data. The third coordinate must be the same for all GRID entries when importing 2D NASTRAN mesh data.

The following tutorial showcases NASTRAN import in 3D:

https://www.comsol.com/model/eigenvalue-analysis-of-a-crankshaft-986.

Alternatively, you can import a surface mesh on any of the formats STL, PLY, 3MF, or VRML. The length unit specified in 3MF files is used to scale the mesh during the import. Use Operations for Editing Imported Meshes to improve the quality of the surface mesh, intersect the mesh, form domains, and fill the domains with a volume mesh.

Mesh Partitioning

When a mesh is imported into COMSOL Multiphysics, it is automatically partitioned into geometric domains, boundaries, edges, and points. If the automatically performed partitioning does not match the requirements, you can modify the boundary partitioning options or perform manual partitioning using Operations for Editing Imported Meshes. The additional domains and boundaries formed by this type of operation can be used, for example, to define other material properties or boundary conditions (loads). See Mesh Elements for 1D, 2D, and 3D Meshes for more information.

Combining Several Imported Meshes

It is possible to combine meshes from several mesh files using several *Import* nodes. Then, COMSOL Multiphysics adds the elements and vertices of the newly imported mesh data to the existing mesh. Mesh data from different *Import* nodes form an assembly with potentially overlapping elements. Mesh data with overlapping elements has very limited use in computations. For intersecting meshes with overlapping elements, use the *Union* operation. For disjoint sets of mesh data that are overlapping only at touching surfaces, you can either use the *Merge Entities* operation to merge the surfaces, or define pair continuity conditions and proceed with computations as usual. See also Conforming, Nonconforming, and Nonmatching Meshes.

Operations for Editing Imported Meshes

There are several operations available if you need to repair, clean up, edit, and remesh imported surface meshes and fill them with a volumetric mesh in the software:

- Adapt
- Boundary Layers
- Collapse Entities

- Convert
- Create Vertices
- Create Edges
- Create Faces
- Create Domains
- Delete Entities
- Detect Faces
- Fill Holes
- Free Triangular
- Free Quad
- Free Tetrahedral
- Imprint
- Intersect with Plane (Intersect with Line in 2D)
- Join Entities
- Mapped
- Merge Entities
- Partition with Ball / Box / Cylinder / by Expression
- Refine
- Remesh Faces (3D)
- Remesh Domains (2D)
- Remesh Edges (3D and 2D)
- Swept
- Union

See the following blog post for examples of editing and repairing imported meshes:

https://www.comsol.com/blogs/editing-and-repairing-imported-meshes-in-comsol-multiphysics/

Selections

Selections are used for grouping geometric entities in order to organize the model setup better. They are used for assigning material properties, volume forces and sources, boundary conditions, edge conditions, and point conditions to domains, boundaries, edges, and points. See Table 1 for a list of file formats that support the import and export of selections.

IMPORTING MESH DATA AS GEOMETRY OBJECTS

Any surface or volumetric mesh can be used to form geometry objects under the *Geometry* node. Smooth geometric faces are formed based on the surface part of the imported mesh. If the mesh data is used to define a geometry object, COMSOL Multiphysics allows for solid operations on the imported mesh using a mix of surface mesh and CAD representations. For more information, see the *COMSOL Multiphysics Reference Manual*. Note that it is more robust to work with imported meshes directly under the *Mesh* node using the Operations for Editing Imported Meshes.

IMPORTING SIMULATION DATA

Data on the sectionwise, spreadsheet, and grid formats can be imported to an existing model for comparing a COMSOL simulation with a simulation done in another software or for use as input; for example, a source term or boundary load in a new COMSOL Multiphysics simulation. In addition, the sectionwise and spreadsheet formats can be imported and used to create curve geometry objects in 2D and 3D. You can, for example, export a set of

contour curves or streamlines on the sectionwise format and then import the set of curves as a curve geometry object.

Importing Numerical Data and Comparing with Simulated Data

Simulation data stored on the sectionwise, spreadsheet, and grid formats can be imported as interpolation data.

The example in this section is a modified version of the example in the document *Introduction to COMSOL Multiphysics*, available in the installed documentation set.

Files corresponding to the example below can be downloaded from:

https://www.comsol.com/model/72351

The filenames are:

```
busbar_box_fully_coupled_export_from_normal_mesh.mph
busbar_box_fully_coupled_compare.mph
busbar_box_T_data_grid.txt
busbar_box_T_data_spreadsheet.txt
busbar_box_T_data_sectionwise.txt
```

In this case, temperature data in the computational volume of a fluid flow with heat transfer simulation is being imported. The data is computed on a coarser mesh than that of the current simulation.

Imported data can be visualized on an existing mesh, automatically applying data interpolation and extrapolation where needed, or against a regular grid. Figure 1 shows the temperature data visualized on a regular grid.

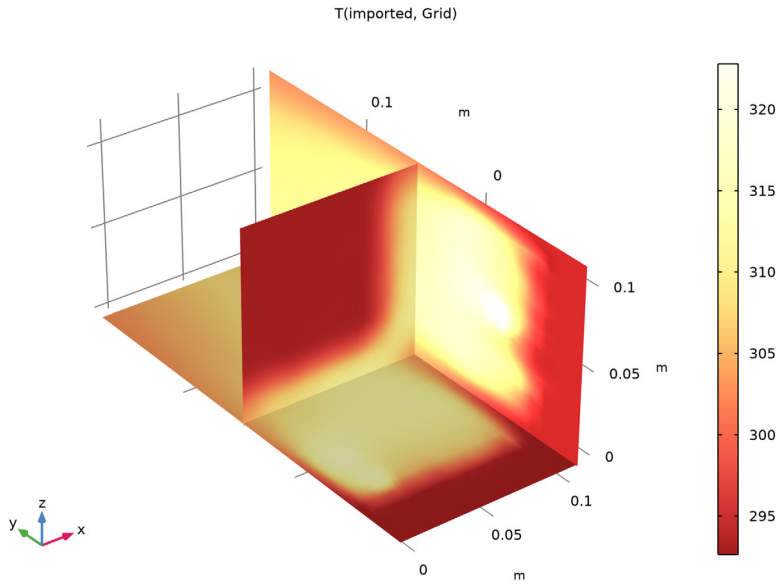


Figure 1: Imported temperature data visualized on a regular grid.

Figure 2 shows the same data visualized on the unstructured tetrahedral mesh of the current simulation. The two visualizations are very similar in appearance.

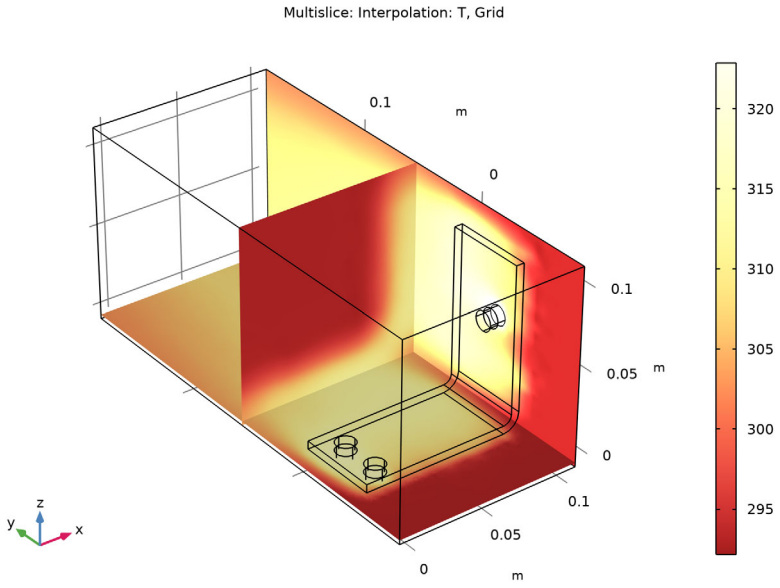


Figure 2: Imported temperature data visualized on the tetrahedral mesh of the simulation.

Figure 3 shows a comparison between the temperature field of the current simulation, using a finer mesh, and the imported data, previously simulated on a mesh with the default size.

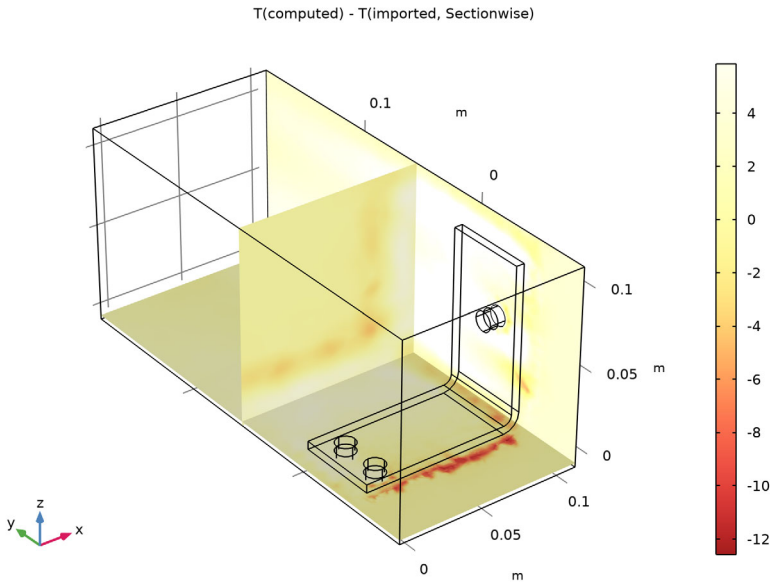


Figure 3: Comparison between the computed temperature field and imported temperature data.

Note that the same type of comparison can be done by using two different meshes within the user interface without using file export and import. This example illustrates the use of file export and import.

The mesh in the example above is generated in COMSOL Multiphysics, but an externally created mesh could also be used with the functionality provided when importing the mesh under the model tree *Mesh* node.

Figure 4 shows a similar comparison, but where the imported data is originally defined on a regular grid with 50-by-50-by-50 grid points.

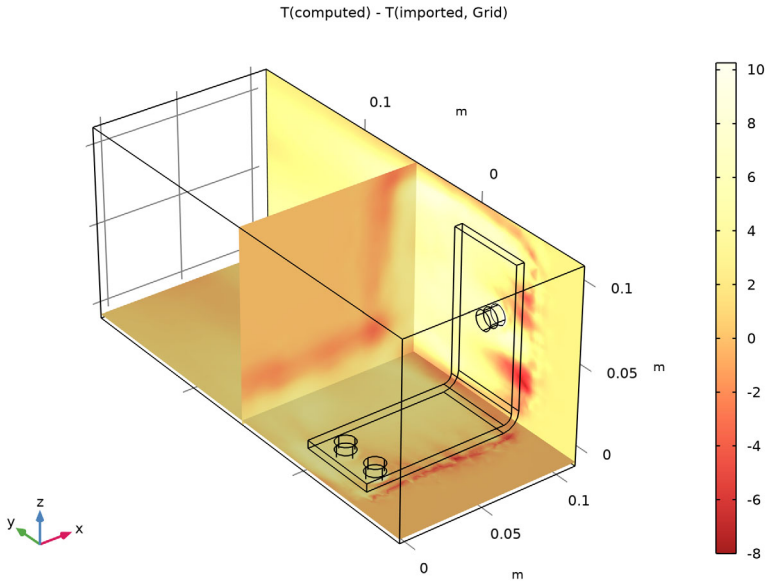


Figure 4: Comparison between the computed temperature field and imported temperature data defined on a regular grid with 50-by-50-by-50 grid points.

In this case, the discrepancy is a bit higher due to the mismatch between the regular grid and the unstructured mesh.

Importing Data on Surfaces

When interpolating data sampled on a surface, only the sectionwise and spreadsheet formats are available. The sectionwise format is faster, but the spreadsheet format is recommended, since it gives higher accuracy in interpolation.

Files corresponding to the example below can be downloaded from:

https://www.comsol.com/model/72351

The filenames are:

```
non_newtonian_flow_compare_surface.mph
non_newtonian_flow_export_from_normal_mesh_surface.mph
non_newtonian_flow_normal_sectionwise_w.txt
non_newtonian_flow_normal_spreadsheet_w.txt
```

Figure 5 shows a flow field in the z direction visualized by slices.

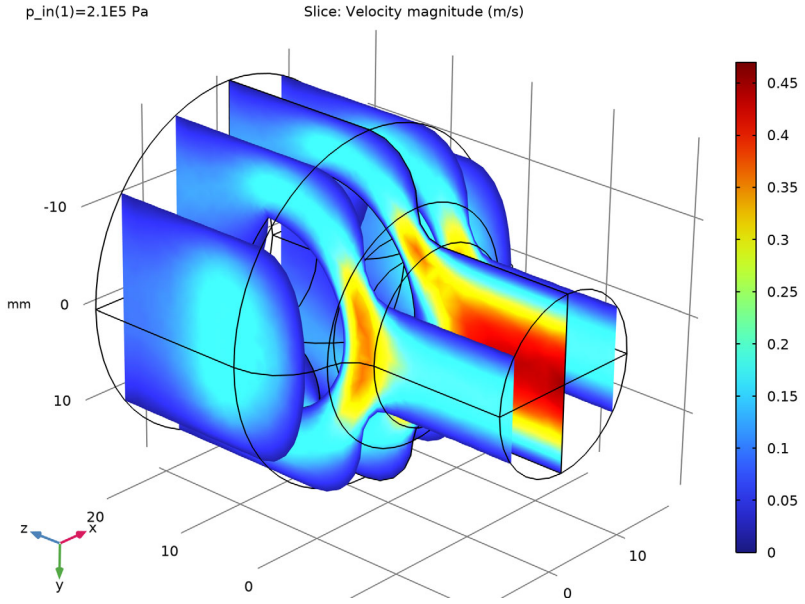


Figure 5: A slice plot showing the flow field in the z direction.

This simulation uses a mesh with the default size. Figure 6 shows the same flow field in the z direction visualized on the surface. In this visualization, the color range is set to a minimum value of `0` and a maximum value of `0.1` (m/s).

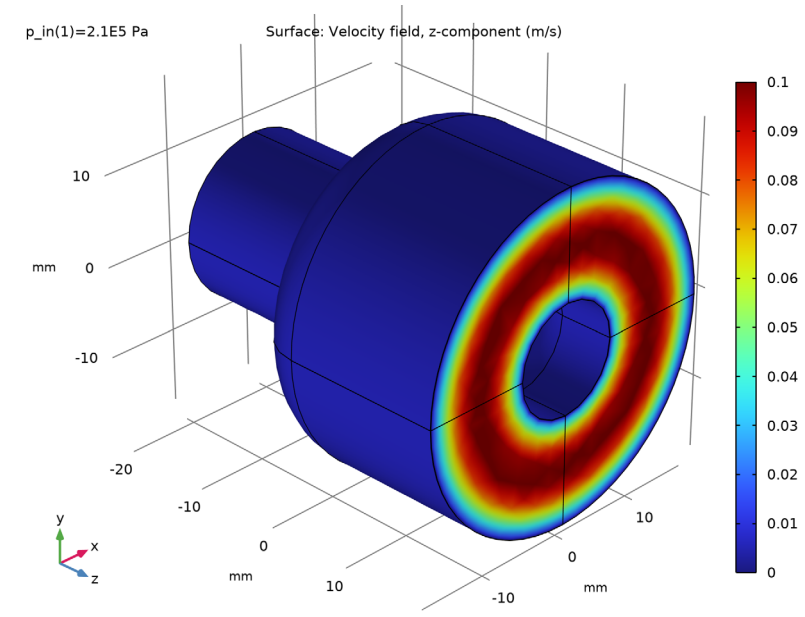


Figure 6: A surface plot showing the computed flow field in the z direction, using a mesh with a default size.

Figure 7 shows the corresponding mesh.

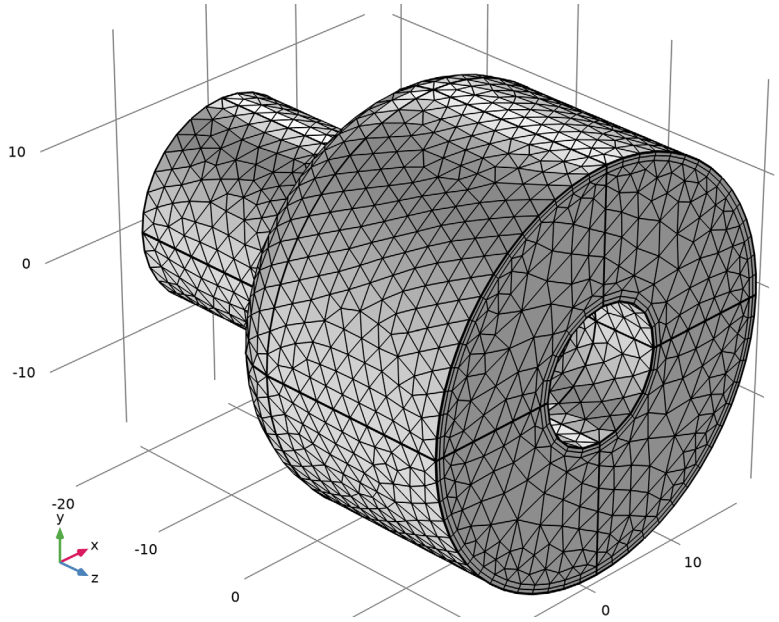


Figure 7: Mesh with a default size used for simulation.

Figure 8 shows the simulation with a mesh with a finer size. The same manual color range is used.

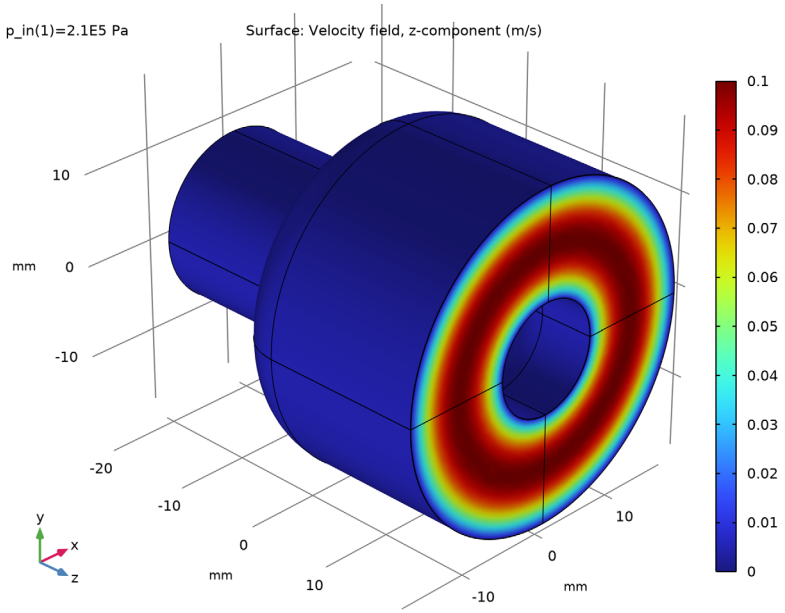


Figure 8: Surface plot showing the z component of the computed velocity field, using a mesh with a finer size.

Figure 9 shows the corresponding mesh.

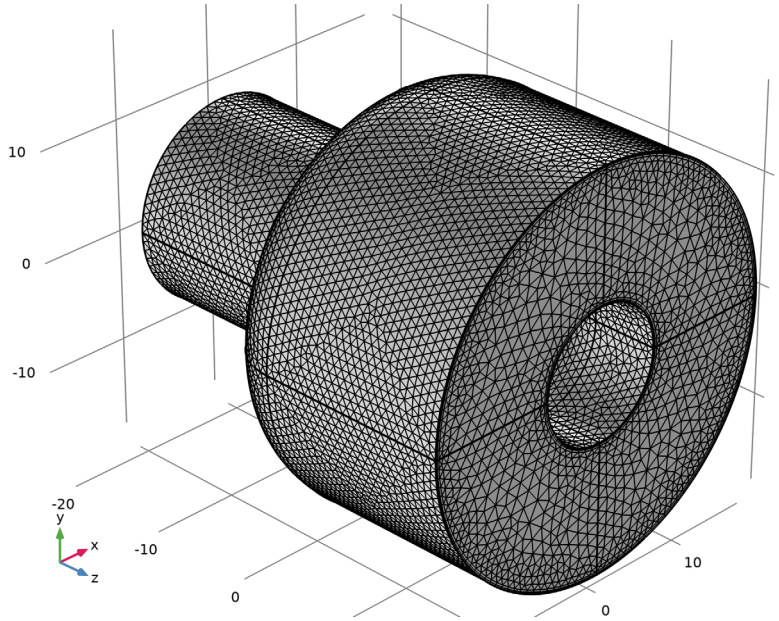


Figure 9: Mesh with a finer size, used for the second simulation.

Figure 10 shows the z-directional flow data imported as an interpolation function in the spreadsheet format. In this example, the data is first exported from a simulation based on the mesh with the default size and then imported to a model where a mesh with a finer size is used. The simulation data based on the coarser mesh is automatically interpolated to the finer mesh and can be used, for example, as input to subsequent simulations. The same manual color range is used.

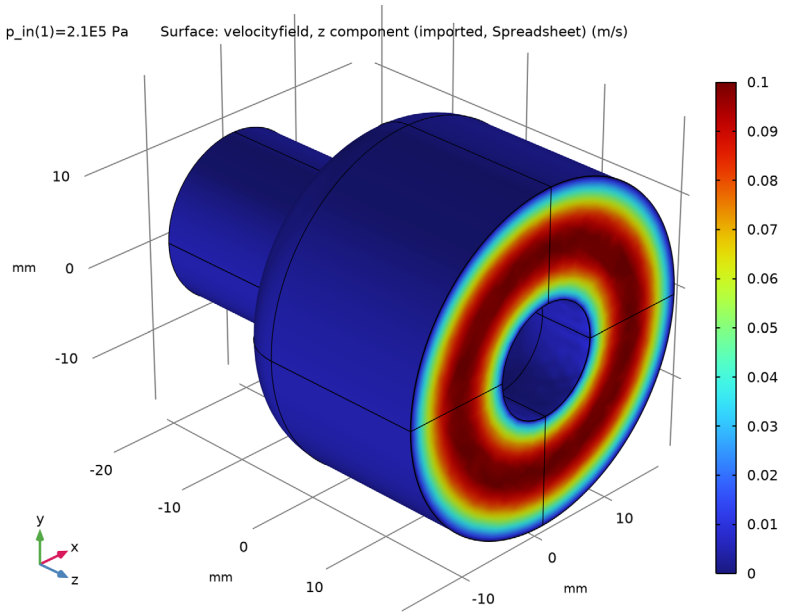


Figure 10: Surface plot showing the z component of the velocity field, imported from file in the spreadsheet format.

It is difficult to see when comparing Figure 8 with Figure 10, but the interpolated data is not as smooth as the original simulated data, which is expected, since some of the original information is lost due to interpolation error.

Figure 11 shows the difference `abs(w-int1(x,y,z))`, where `w` is the computed z-directional velocity and `int1(x,y,z)` is the imported spreadsheet data. The color scale is changed to run between `0` and `0.01` (instead of `0` and `0.1`).

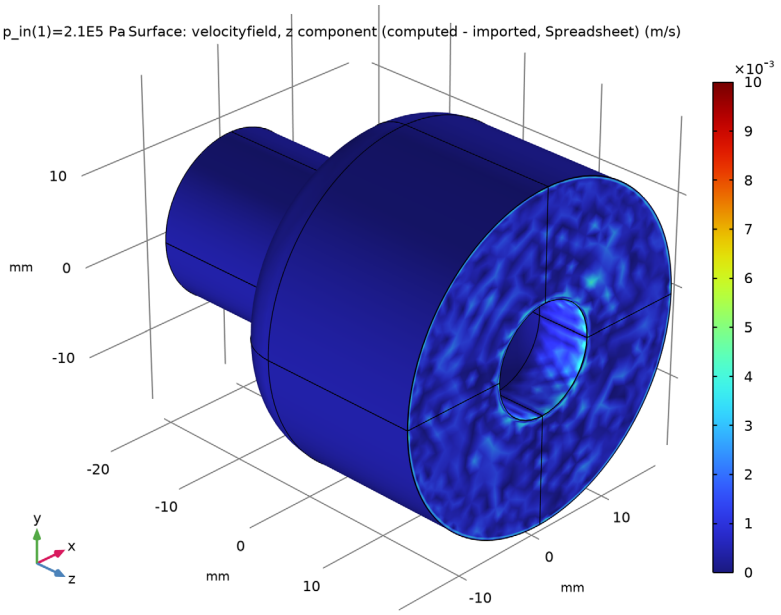


Figure 11: The absolute difference between the computed and the imported velocity field data (z direction).

Interpolation Data Formats

Data on the sectionwise, spreadsheet, and grid formats can be imported to an existing model for comparing a COMSOL simulation with a simulation done in another software or for use in its own right as input; for example, a source term or a boundary load in a new simulation. For examples of such import, see Importing Simulation Data. The sectionwise and spreadsheet formats can be used for creating geometry curve objects of the types. In addition, the sectionwise format can be used for importing volumetric mesh data and for the indirect creation of geometry.

The grid format is a structured format and contains only point coordinates and data values with no topological information. The grid format allows for more extrapolation options than the other formats. Interpolation is faster when using the grid data format than for the other formats.

The spreadsheet format may contain either structured or unstructured data and contains only point coordinates and data values with no topological information. This means that extrapolated data in regions without point coordinates will be extrapolated to a fixed, user-defined constant value. In order to perform interpolation of the spreadsheet format, the software creates an auxiliary unstructured mesh (not visible to the user). For this reason, using the sectionwise format generally results in faster interpolation than the spreadsheet format.

The sectionwise format is an unstructured format and contains detailed topological information that can be used to carefully design both interpolation and extrapolation. For example, holes in the geometry could be filled with data for the purpose of controlling the values for imported data that would otherwise be extrapolated to a constant value.

For more details on the available interpolation and extrapolation options, see the *COMSOL Multiphysics Reference Manual*.

The Grid Interpolation Format

The grid interpolation format is a compact format that does not list the coordinates of each data point but instead lists the unique x-, y-, and z-coordinate values defining the grid together with the associated data. The syntax of the grid interpolation format is as follows:

```
%Grid
x grid point coordinate values
y grid point coordinate values (optional)
z grid point coordinate values (optional)
%Data
Data values
```

Lines starting with `%` at the beginning of the file are ignored. Values can be separated by space, tab, comma, or semicolon.

The grid coordinate section lists the unique coordinate values that form the grid. In the data section, each row contains values for different x grid points for fixed values of y and z. In a particular column of the data section, the data values in the rows correspond to grid points that first increase in the y-grid coordinate value and then the z-grid coordinate value. The grid points do not have to be coordinates and can represent another independent variable that the data values depend on. For example, the grid points can be temperature values and the data values can be the thermal conductivity at these temperatures. It is possible to include more than one function in the file, as long as the function data is separated by a line `%Data`. There is no limit to the number of functions defined in this way. For grid points that are outside of the computational mesh, the data values will be `NaN`.

The example below shows interpolation data on the grid format with a 2-by-3-by-4 grid of temperature values for x-, y-, and z-coordinates, respectively.

```
% Model:              busbar_box.mph
% Version:            COMSOL 6.2.0.266
```

```
% Date:              Oct 13 2023, 10:08
% Dimension:         3
% Nodes:             24
% Expressions:       1
% Description:       Temperature
% Length unit:       m
% Grid
0.01875                 0.07625
-0.045                  0.045                   0.135
0.01                    0.04                    0.07                    0.1
% Data
% T (K)
300.4134823983414       304.1458294322944
303.61176146559325      308.41125258438797
301.0107309335149       307.0349404910426
293.1495739459461       293.224152079104
293.19134447218363      296.20256368929404
293.17171166142396      297.59018698728636
293.15006403092355      293.0444409082566
293.15148214481064      294.9969430341624
293.15360362519647      294.8358739228041
293.1499833079099       293.3016314358608
293.1504532755099       294.031314528188
293.1505419055767       294.42024600759163
```

The grid section lists the unique coordinate values for x (first row), y (second row), and z (third row).

The columns of the data section correspond to the two x-coordinate values `0.01875` and `0.07625`, respectively. The length of each column is `12=3*4`, where the first 3 rows of a column correspond to the data associated with the y-coordinate values `-0.045, 0.045, 0.135`, and the z-coordinate value `0.01`. The next 3 rows correspond to the y-coordinate values `-0.045, 0.045, 0.135`, the z-coordinate value `0.04`, and so on.

An example file showcasing the grid data format can be downloaded from:

https://www.comsol.com/model/72351

The filename is:

```
busbar_box_T_data_grid.txt
```

The Spreadsheet Interpolation Format

The spreadsheet format can be used for exporting results and importing numerical data defined on an unstructured set of point coordinates. It cannot be used directly for defining a mesh or indirectly as a solid geometry. A spreadsheet file contains coordinates and function data for space-dependent functions or, instead of coordinates, general input variable values and function values for functions of one or more variables.

The syntax of the spreadsheet interpolation format is as follows:

```
%Header (optional)
Columns containing x, y (optional), and z (optional) coordinate values, or any other input
variable values, followed by columns containing function data values.
```

Lines starting with `%` at the beginning of the file are ignored. Values can be separated by space, tab, comma, or semicolon.

There is no limit to the number of function data columns, but the number of input variable data columns can be at most 3. The number of input variable columns is implicit and needs to be specified by the user before import.

The example below shows the first few lines of a file containing interpolation data in the spreadsheet format for a 2D model with x- and y-coordinates, with two function data columns for the temperature and the temperature gradient, respectively.

```
% Model:             heat_convection_2d.mph
% Version:           COMSOL 6.2.0.266
% Date:              Oct 13 2023, 10:11
```

```
% Dimension:          2
% Nodes:              160
% Expressions:        2
% Description:        Temperature, Gradient of T, x-component
% Length unit:        m
% X                       Y                       T (K)                   Tx (K/m)
0                         0                       373.15                  -2.7284841053187847E-12
0.06666666666666662       0                       373.15                  1.8189894035458565E-12
0                         0.06666666666666667     362.85733360233905      0.016296245209559856
0.06666666666666662       0.06666666666666667     362.7335662634441       -3.710334001333649
0                         0.13333333333333333     352.80688221894144      0.02772268940861977
...
```

The following example shows the first few lines of a file containing interpolation data in the spreadsheet format for a 3D model with x-, y-, and z-coordinates, with one function data column for the temperature.

```
% Model:              busbar_box.mph
% Version:            COMSOL 6.2.0.266
% Date:               Oct 13 2023, 10:08
% Dimension:          3
% Nodes:              1861
% Expressions:        1
% Description:        Temperature
% Length unit:        m
% x                       y                       z                       T (K)
0.05621906254245827       0.1669952915832881      0.09803256635402077     293.1943545749328
0.04782791857244105       0.18000000000000002     0.09052024778162253     293.20080218975176
0.059                     0.18000000000000005     0.11500000000000002     293.29372333205174
0.06685101713939509       0.18000000000000002     0.08907407172684315     293.4285639892069
0.10500000000000001       0.18000000000000002     0.067                   305.7374583411461
...
```

Note that a function data column need not represent a continuous field, as in the example above, but can represent discrete quantities. For example, if you export a set of streamlines, the first function data column will contain the number of each streamline that the data belongs to.

Example files showcasing the spreadsheet data format can be downloaded from:

https://www.comsol.com/model/72351

The filenames are:

```
busbar_box_T_data_spreadsheet.txt
non_newtonian_flow_normal_spreadsheet_w.txt
```

THE COORDINATE FILE FORMAT

The coordinate file format is a special case of the spreadsheet format and contains only the point coordinate value columns. The coordinate file format can be used to specify the point coordinates for data export and for creating curve geometry objects.

The example below shows the first few lines of a coordinate file containing 3D point coordinate values.

```
% Coordinate values
-15.588426130362178   -8.999992011840579    3.000000000000001
-14.562300137244211   -10.580140343409553   3.000000000000001
-15.097478588358936   -9.80133359671242     4.64656422933223
-13.376581989138662   -12.044339345370581   3.000000000000001
-13.989436345998962   -11.326767885025419   4.646420660346455
...
```

The Sectionwise Interpolation Format

The sectionwise format can be used for exporting results and importing numerical data defined on any one of a linear tetrahedral, triangular, or edge element mesh. It can be used directly for defining a mesh and indirectly for creating a geometry. See Overview of Data Import for more information. A sectionwise file contains unstructured

mesh and function data for space-dependent functions. The functions need not be functions of the space coordinates x, y, or z, but they can be functions of some other quantities, such as deformation or temperature.

```
%Coordinates
One, two, or three columns containing x, y (optional), and z (optional) coordinate values
or other input variable values
%Elements
Triangulation where each row contains the row indices (1-based) of the points in the
Coordinates section for one element (edge, triangular, or tetrahedral)
%Data (funname)
Column of function data values for each point
```

Lines starting with `%` are ignored for interpolation data import. However, the length unit is taken into account if the file is imported under the *Mesh* node. Values and indices can be separated by a space, tab, comma, or semicolon.

It is possible to include more than one function in the file as long as the function data is separated by a line `%Data.` There is no limit to the number of functions defined in this way. However, the number of independent variables can be at most 3, such as in the case of functions of space coordinates (x, y, z). Note that in the sectionwise format, rows are indexed starting from 1.

The abbreviated example below shows temperature data from a 3D heat transfer model with tetrahedral elements. The temperature values are defined for each coordinate; the number of coordinate lines is the same as the number of temperature data lines.

```
% Model:              busbar_box.mph
% Version:            COMSOL 6.2.0.266
% Date:               Oct 13 2023, 10:08
% Dimension:          3
% Nodes:              1861
% Elements:           9120
% Expressions:        1
% Description:        Temperature
% Length unit:        m
% Coordinates
0.05621906254245827       0.1669952915832881       0.09803256635402077
0.04782791857244105       0.18000000000000002      0.09052024778162253
0.059                     0.18000000000000005      0.11500000000000002
0.06685101713939509       0.18000000000000002      0.08907407172684315
0.10500000000000001       0.18000000000000002      0.067
...
0.10500000000000001       -0.0230117893210171      0.056949328388214815
0.10499999999999998       -0.023506233446282956    0.053559015131106474
0.105                     -0.026400036416763993    0.05663458881955498
0.10249999998293331       -0.02751267010274674     0.05306405946148493
0.105                     -0.026933687000386404    0.05311865209250027
% Elements (tetrahedra)
4          3         2         1
8          7         6         5
12         11        10        9
13         9         12        10
11         14        9         12
...
1852       831       1854      1855
832        1852      1819      1855
831        1828      1854      1855
1856       625       1852      1859
1860       852       1859      1852
% Data (T (K))
293.1943545749328
293.20080218975176
293.29372333205174
293.4285639892069
305.7374583411461
...
```

The abbreviated example below shows fluid velocity data for a 3D fluid flow model with values defined on the triangular surface elements.

```
% Model:              non_newtonian_flow_3d.mph
% Version:            COMSOL 6.2.0.266
% Date:               Oct 13 2023, 10:02
% Dimension:          3
% Nodes:              3687
% Elements:           7370
% Expressions:        1
% Description:        Surface
% Coordinates
-14.56230598174954        -10.580134427024147      3.0000000000000004
-13.376608866077628       -12.044348684921628      3.0000000000000013
-13.989442409223177       -11.326760396469421      4.646543363247797
-15.588462991933238       -8.999990086057197       3.0000000000000004
-15.09748632198884        -9.801321684209732       4.6464080418264695
...
4.819485695100072         7.213004229437904        -21
6.132247818763501         6.1312437087137726       -21
7.205903105711457         4.812787301349522        -20.999999999999996
7.997695961021116         3.3104000054066693       -20.999999999999993
8.490582217284308         1.6872575040783466       -20.999999999999996
% Elements (triangles)
1          2          3
4          1          5
3          5          1
6          7          2
7          3          2
...
3687       2340       3680
3648       3686       3687
3648       3687       3650
3637       3685       3686
3637       3686       3648
% Data (Velocity field, z-component)
0
0
0
0
0
...
0
0
0
0
0
0.05012347150285308
0.026587676543436013
0.02710377813820205
0.04472625630620988
0.02713934919161217
...
```

The abbreviated example below, based on a MEMS model, shows voltage values for the edge elements of a 3D line plot.

```
% Model:              mems_pressure_sensor.mph
% Version:            COMSOL 6.2.0.266
% Date:               Oct 13 2023, 10:11
% Dimension:          3
% Nodes:              1756
% Elements:           1764
% Expressions:        1
% Description:        Line
% Coordinates
-104.99999999999999       -50.714285714285715      0
-105                      -48.80952380952381       0
-105                      -46.904761904761905      0
-104.99999999999999       -44.99999999999999       0
-104.99999999999999       -56.42857142857143       0
...
104.99999999999999        49.285714285714285       0
105                       47.38095238095238        0
```

```
105                     45.476190476190474      0
105                     53.09523809523809       0
105                     51.19047619047619       0
% Elements (lines)
287       599
599       600
600       601
601       602
602       603
...
1173      1174
1174      1175
1175      1190
1190      1191
1191      1059
% Data (Electric potential)
1.4874145983222065
1.487414598322207
1.4874145983222067
1.4874145983222067
1.4874145983222065
...
```

Example files showcasing the sectionwise data format can be downloaded from:

https://www.comsol.com/model/72351

The filenames are:

```
busbar_box_T_data_sectionwise.txt
non_newtonian_flow_normal_sectionwise_w.txt
```

EVALUATION OPTIONS

When exporting to the sectionwise format as described in the section Exporting Data from the Results Node, the entered expressions are evaluated at a number of points in each mesh element. For example, evaluating in Lagrange points of order 1 means that the expressions are evaluated at the vertices of each mesh element. When more than one mesh element share a vertex (as is typically the case), the expressions are evaluated several times at that coordinate using the shape functions in the different mesh elements. The values of these evaluations at the same point might not be equal, depending on the expression being evaluated. In particular, derivatives are commonly discontinuous across mesh element boundaries and usually have different values. Once all the evaluations have been made, the data is checked for duplicate values (that is, evaluations with the same coordinates and the same values of the expressions). Such duplicates are removed before the data is exported to file. With smoothing, a smoothed variant of the derivative is evaluated, which is continuous across mesh element boundaries; in such cases, there are many duplicates. When evaluating at Gauss points, the evaluation points are always in the interior of mesh elements, so there are never any duplicates.

Mesh Data Formats

Volumetric mesh data can be imported and exported using the COMSOL native text format (`.mphtxt`), COMSOL native binary format (`.mphbin`), COMSOL sectionwise format (`.txt`), and NASTRAN format (`.nas`). The COMSOL native file text format contains a section with mesh vertex coordinates, followed by sections with mesh element information and divided into separate subsections for each mesh element type. Surface mesh data can be imported and exported using the formats mentioned above, but also the 3MF format (`.3mf`), STL format (`.stl`), PLY format (`.ply`), and VRML format (`.vrml`). A simulation in COMSOL Multiphysics normally needs a volumetric mesh, so an imported surface mesh often needs to be filled with a volume mesh using the *Free Tetrahedral*, *Swept*, and *Boundary Layers* operations.

This section gives an overview with examples of the COMSOL native file formats for mesh data. For a formal specification, see the section Specification of COMSOL Formats for Mesh Data. See the section The Sectionwise Interpolation Format for more information about the sectionwise format.

The COMSOL Mesh

A COMSOL Multiphysics mesh contains elements for all space dimension levels: vertices (0D), edge elements (1D), face elements (2D), and domain elements (3D), together with groupings of elements into geometric domains, boundaries, edges, and points. For example, a tetrahedral mesh consists of domain (tetrahedra, or tet), boundary (triangles), edge, and vertex elements, as well as mesh vertices. Furthermore, each element has an index to the geometric entity it belongs to. For example, consider the mesh shown in Figure 12. The visualization is a mesh plot with shrunk elements.

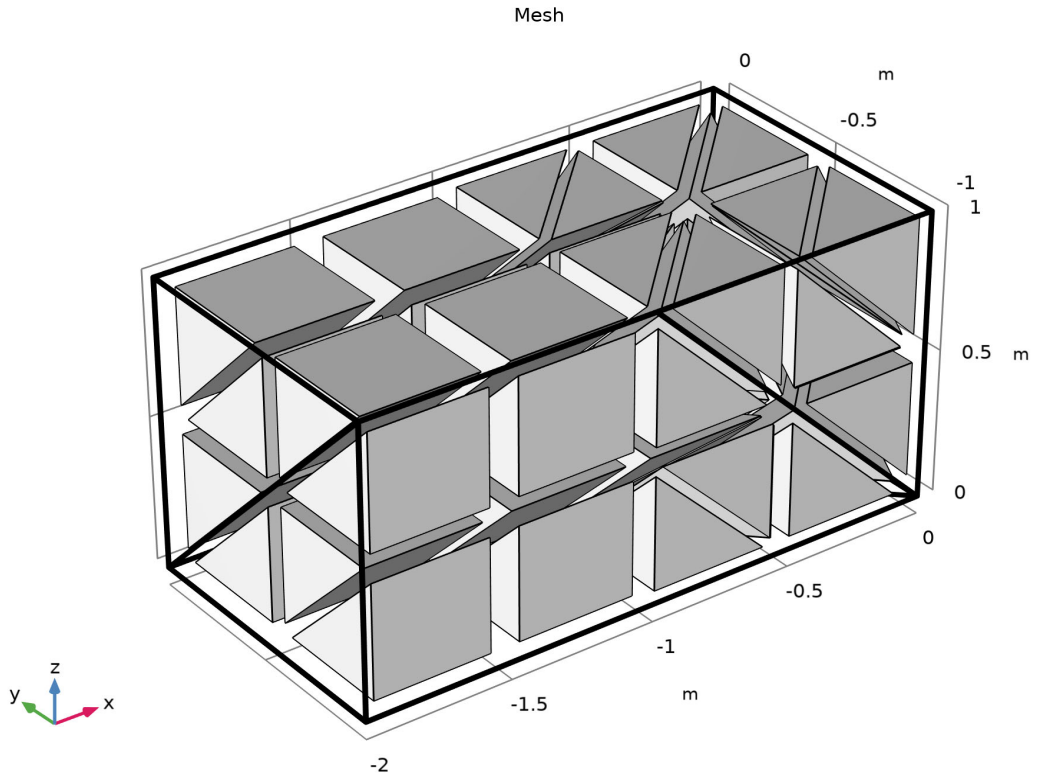


Figure 12: Mesh containing prism and tetrahedral elements, used to show how groups of mesh elements belong to different geometric entities.

This mesh consists of 44 mesh vertices, 40 tetrahedral domain elements, 16 prism domain elements, 52 triangle boundary elements, 20 quad boundary elements, 38 edge elements, and 10 vertex elements. The elements are further grouped into geometric entities; for example, 2 domains, 10 faces, 17 edges, and 10 points. This grouping into geometric entities can be changed by the user and is used in modeling for assigning materials as well as domain, boundary, edge, and point conditions. Figure 13 show the grouping of the mesh in this example into geometry domains, boundaries, edges, and points, where the geometric entity indices are represented by colors.

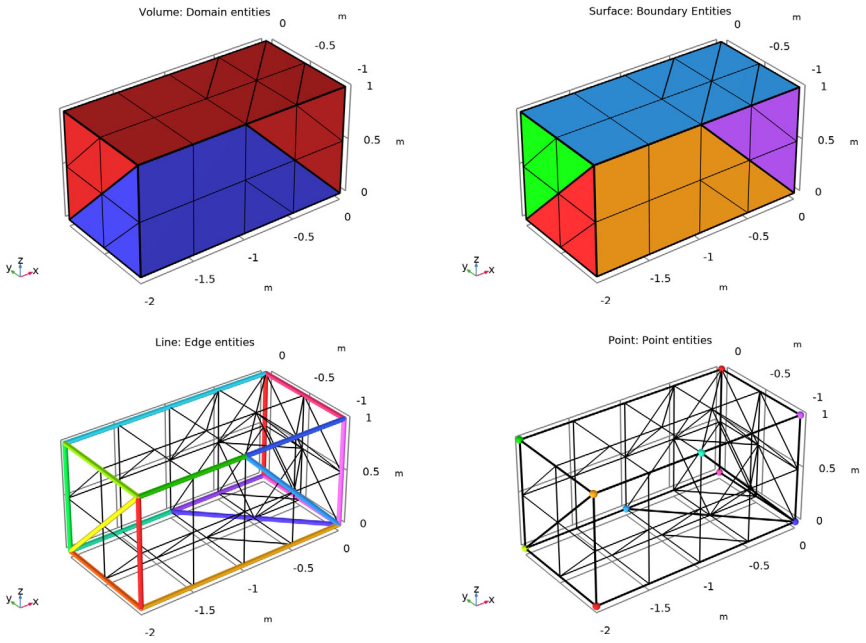


Figure 13: Distinguishing geometric entities by color using he example mesh in Figure 12.

Example files corresponding to this example can be downloaded from:

https://www.comsol.com/model/72351

The filenames are:

```
mesh_example_intro.mphtxt
mesh_example_intro.mph
```

The file `mesh_example_intro.mphtxt` contains fewer mesh elements, but the entities corresponds to the example shown in the images above.

Mesh Elements for 1D, 2D, and 3D Meshes

A geometric model consists of domains, boundaries, edges (only 3D), and vertices (only 2D and 3D), that is referred to as *geometric entities*. Each geometric entity is represented in the mesh by at least one element, but typically consists of a connected set of mesh elements.

In 1D meshes, the domains (intervals) are represented by *edge elements*. The endpoints of the mesh elements are called *mesh vertices*. The boundaries (or vertices) defined in the geometry are represented in the mesh by boundary elements (*vertex elements*).

In 2D meshes, the domains are represented by triangular or quadrilateral mesh elements. The sides of the triangle and quadrilateral elements are called *mesh edges*, and their corners are mesh vertices. A mesh edge must not contain mesh vertices in its interior. The boundaries defined are discretized into *boundary elements* (edge elements), which

must conform with the mesh elements of the adjacent domains. The geometric vertices (or points) are represented by vertex elements.

In 3D meshes, the domains are represented by tetrahedral, hexahedral, prism, or pyramid elements whose faces, edges, and corners are called *mesh faces*, mesh edges, and mesh vertices, respectively. The boundaries are discretized into triangular or quadrilateral boundary elements. The geometric edges are discretized into edge elements. Similar to 2D, the geometric vertices are represented by vertex elements.

The following table summarizes the terminology used for geometric and mesh entities in 3D:

TABLE 2: GEOMETRIC AND MESH ENTITIES (3D)

ENTITY DIMENSION	GEOMETRIC ENTITY	ALTERNATIVE GEOMETRIC ENTITY NAME	MESH ENTITY	ALTERNATIVE MESH ENTITY NAME	COMSOL NATIVE FORMAT ENTRIES
0			Mesh vertex	Mesh point, mesh node, node point	
0	Vertex	Point	Vertex element		vtx
1			Edge element		edg
1	Edge		Chain of edge elements		
2			Boundary element	Triangular or quad element, face element	tri, quad
2	Face	Boundary	Connected set of boundary elements	Boundary	
3			Domain element	Tetrahedron, pyramid, prism, or hexahedron element	tet, pyr, prism, hex
3	Domain		Connected set of domain elements		

The term *node* is used when the mesh data contains information about higher-order elements. The mesh data formats used for import and export support, at most, second-order element information and is limited to certain element types; see Element Types and Numbering Conventions. However, using higher-order elements for simulations in COMSOL Multiphysics is not restricted to the order specified in the imported mesh file. See Conversions on Import and Second-Order Element Data for more information.

A complete set of mesh data on the COMSOL native text or binary format contains information at all space dimension levels for both mesh elements and geometric entities. If the contents of a file are incomplete — for example, if it only contains information about mesh vertex coordinates and connectivity of tetrahedral elements — the mesh import automatically generates the missing element data including partitioning into domains, boundaries, and edges. This means that the structure of a COMSOL native text or binary file that is used for import can be different depending on the level of completeness of the mesh data. For more information, see Importing Incomplete Mesh Data in the COMSOL Native Format.

Geometric domain entity numbering typically starts at 1 (0 is reserved for empty domains or voids), whereas geometric face, edge, and vertex entity numbering typically starts at 0. Mesh vertex numbering typically starts at either 0 or 1. The lowest indices can be specified in the mesh data file.

Conforming, Nonconforming, and Nonmatching Meshes

Meshes generated in the COMSOL Multiphysics software are *conforming* with a *geometric model*, where a geometric model is a collection of geometric entities, their geometric shape, and their connection to each other. In a *conforming mesh*, the intersection between any two mesh elements in the mesh is defined by subelements (mesh

face, mesh edge, or mesh vertex). Each mesh element belongs to exactly one geometric entity. For a mesh conforming with a geometry, each geometric entity is either unmeshed or fully meshed.

A mesh of assembly type can either be conforming with a geometry where the *Form an Assembly* finalization action has been used for the model tree *Geometry* node, or it defines its own geometric model with two or more *Mesh>Import* nodes. It can also be one *Mesh>Import* node if the mesh file contains a mesh of assembly type. Regardless of its origin, an assembly type of mesh defines several disconnected components with duplicated boundaries, edges, and points where the components are touching. The boundary meshes at touching surfaces between two parts do not need to define geometrically matching mesh vertices and elements. The image to the left in Figure 14 shows a *nonmatching* assembly mesh with two disconnected components. The exploded view to the right shows that the two components are indeed disconnected. The mesh contains 21 mesh vertices and 20 triangle elements, 2 domains (indicated by yellow and green colors), 8 geometry edges, and 8 geometry points.

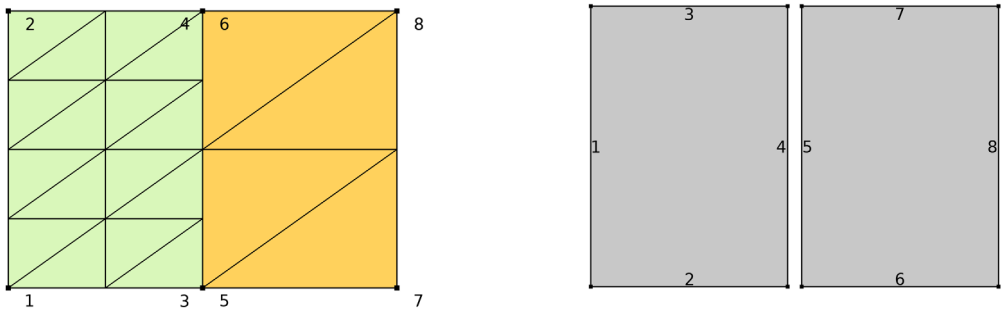


Figure 14: The left image shows a nonmatching assembly mesh. The right image shows the geometric model of the mesh in an exploded view where the disconnected components and the duplicated boundaries (edges) are seen more clearly.

To connect the physics in disconnected components, add *Identity Boundary Pairs* nodes and add pair boundary conditions in the physics interfaces. Alternatively, if you want to connect the meshes (and avoid setting up pairs), use a *Merge Entities* operation (only supported in 3D) and merge the boundaries of the disconnected components.

When importing *nonconforming* mesh data, the *Import* operation will typically create edge and boundary elements of the mesh edges and boundaries corresponding to each nonconformity in the mesh, since these mesh edges and boundaries are typically only adjacent to one element each. In Figure 15, the image to the left shows the result after importing nonconforming mesh data, with so-called *hanging nodes*. The mesh contains 18 mesh vertices and 20 triangle elements, compared to the assembly mesh in Figure 14 which contains 21 mesh vertices. The resulting domains are connected in points 3, 4, and 6 with slit-like holes between the domains, as seen in the exploded view in the right image. Note that the mesh gets 2 domains (indicated by color), 10 geometry edges, and 7 geometry points.

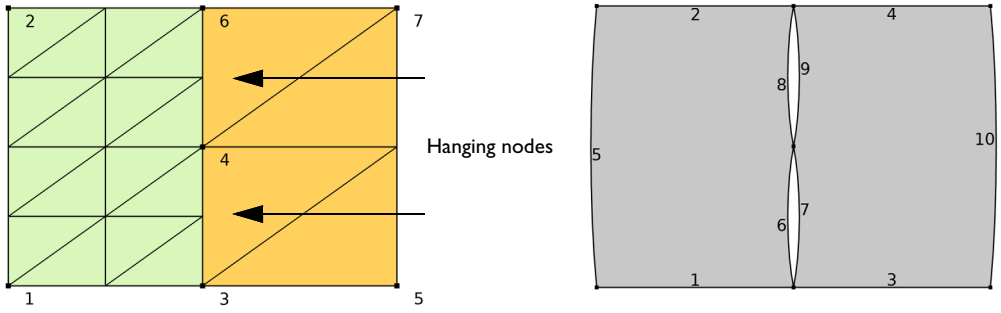


Figure 15: Two "hanging nodes" in an imported mesh (left). The image to the right shows the geometric model of the mesh in an exploded view where the slit-like holes and the duplicated boundaries (edges) are seen more clearly.

Similarly, importing nonconforming mesh data containing a 3D volume mesh normally results in many duplicated boundaries (faces) with slit-like pocket holes. Such a mesh is not well suited for simulations in COMSOL Multiphysics. Use the mesh operation *Merge Entities* to merge the nearby faces defining the interface between two parts and to create a mesh that is connected across the interface, as seen in Figure 16.

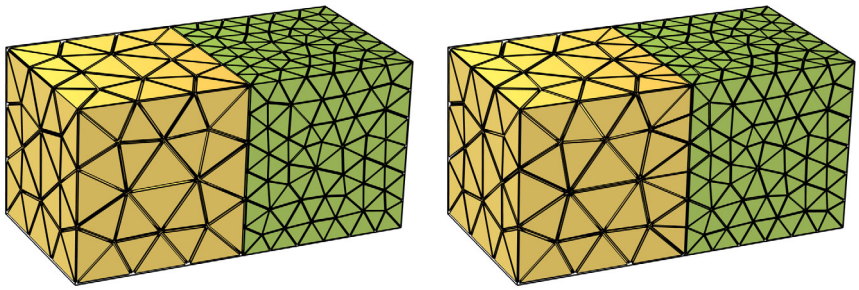


Figure 16: Imported nonconforming mesh data in 3D (left) and the mesh where the Merge Entities operation has been used to merge the adjacent faces.

Import of nonconfoming mesh data with so called *hanging edges* results in an error. A hanging edge appears when two triangles (yellow mesh faces to the right in Figure 17) coincide with a quad (blue mesh face) with which they share mesh vertices.

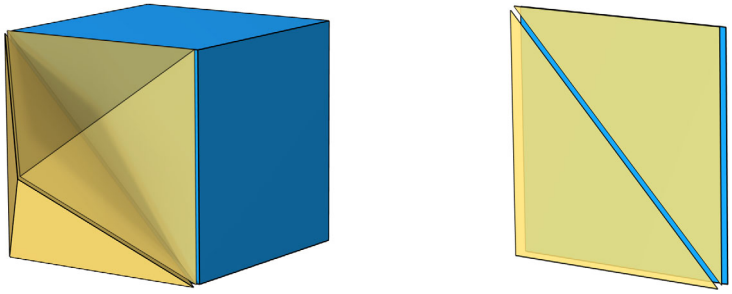


Figure 17: A hanging edge appears when the mesh faces of two tetrahedral elements (yellow) coincide with the mesh face of a hexahedral element (blue). The triangle and quad mesh faces (right image) share mesh vertices.

Element Types and Numbering Conventions

The COMSOL Multiphysics software supports a variety of finite element types of varying geometric shapes and associated shape functions. This includes what amounts to traditional nodal-based isoparametric elements, also known as Lagrange elements, of order up to 7, curved vector elements of order up to 3, as well as other more specialized elements. In 2D, triangular and quad element shapes are supported. In 3D, tetrahedral, pyramid, prism, and hexahedral element shapes are supported. For more information, see the *COMSOL Multiphysics Reference Manual*.

THE COMSOL NATIVE FORMAT

The COMSOL native mesh data format supports all element types but is limited to first- and second-order Lagrange elements (shape functions). The local numbering of the vertices (or nodes) of a mesh element is defined in Figure 18 to Figure 24.

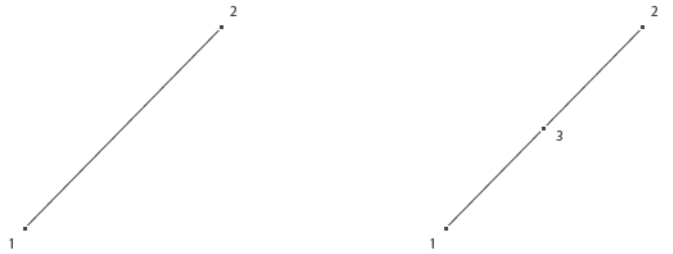


Figure 18: Edge element (`edg`*) as a first-order element (left) and a second-order element (right).*

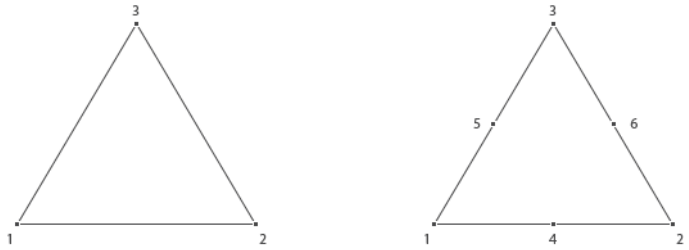


Figure 19: Triangular element (`tri`*) as a first-order element (left) and a second-order element (right).*

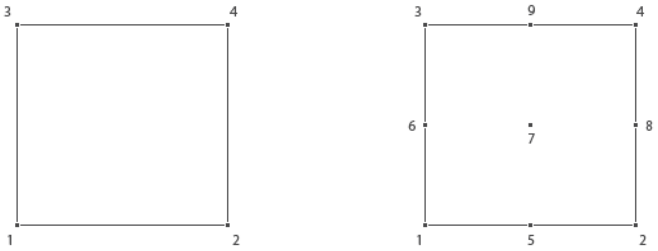


Figure 20: Quadrilateral element (`quad`*) as a first-order element (left) and a second-order element (right).*

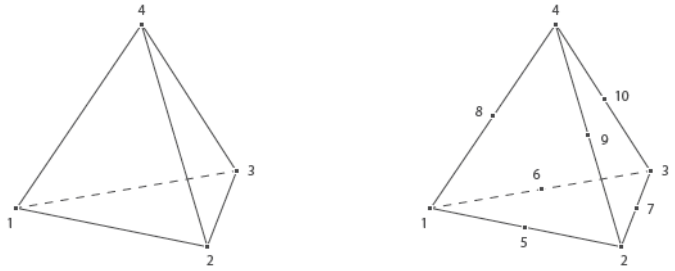


Figure 21: Tetrahedral element (`tet`*) as a first-order element (left) and a second-order element (right).*

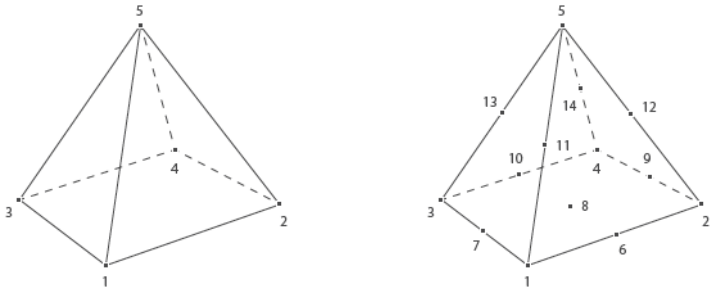


Figure 22: Pyramid element (`pyr`*) as a first-order element (left) and a second-order element (right).*

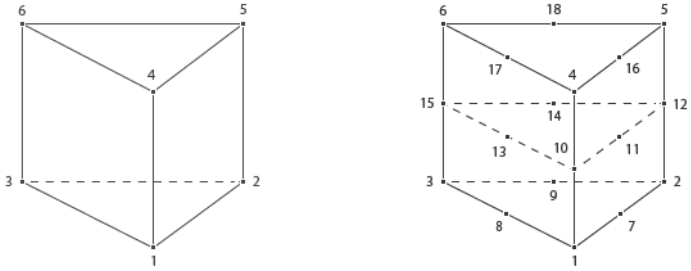


Figure 23: Prism element (`prism`*) as a first-order element (left) and a second-order element (right).*

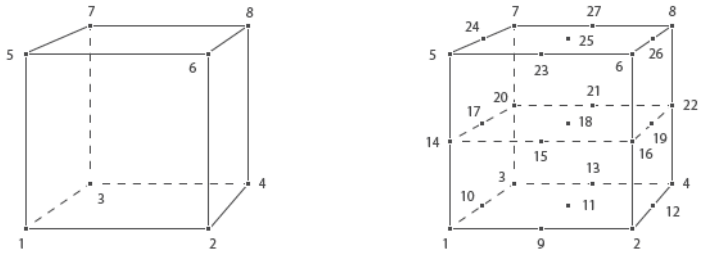


*Figure 24: Hexahedral element (*hex*) as a first-order element (left) and a second-order element (right).*

THE NASTRAN FORMAT

COMSOL Multiphysics can import from and export to the NASTRAN format with support for first- and second-order elements. The following NASTRAN format bulk data entries are supported:

TABLE 3: NASTRAN BULK DATA ENTRIES FOR IMPORT FROM AND EXPORT TO COMSOL MULTIPHYSICS

NASTRAN ENTRY	COMSOL ELEMENT	ELEMENT ORDER	IMPORT TO COMSOL	EXPORT FROM COMSOL	COMMENTS
CBAR	edg	1 and 2	Yes	No	
CBEAM	edg	1 and 2	Yes	No	
CHEXA	hex	1 and 2 (serendipity)	Yes	Yes	
CORD1C	N/A	N/A	Yes	No	Coordinate transform
CORD1R	N/A	N/A	Yes	No	Coordinate transform
CORD1S	N/A	N/A	Yes	No	Coordinate transform
CORD2C	N/A	N/A	Yes	No	Coordinate transform
CORD2R	N/A	N/A	Yes	No	Coordinate transform
CORD2S	N/A	N/A	Yes	No	Coordinate transform
CPENTA	prism	1 and 2 (serendipity)	Yes	Yes	
CPYRAM	pyr	1 and 2 (serendipity)	Yes	Yes	
CQUAD4	quad	1	Yes	Yes	
CQUAD8	quad	2 (serendipity)	Yes	Yes	
CTETRA	tet	1 and 2	Yes	Yes	
CTRIA3	tri	1	Yes	Yes	
CTRIA6	tri	2	Yes	Yes	
GRID	N/A	N/A	Yes	Yes	Mesh vertex coordinate
MAT1	N/A	N/A	Yes	No	Material data
MAT9	N/A	N/A	Yes	No	Material data
MAT10	N/A	N/A	Yes	No	Material data
PSHELL	N/A	N/A	Yes	No	Used for selections
PSOLID	N/A	N/A	Yes	No	Used for selections
RBE2	vtx	1	Yes	No	Used for selections
RBE3	vtx	1	Yes	No	Used for selections

When importing a file containing PSHELL entries, the boundary partitioning is normally determined by these entries. In addition, an associated *Variable* node may be created under the model tree *Definitions* node for each PSHELL entry. The PSHELL entries define the shell thickness on the selections they determine.

An entry of the type PSOLID is often the link between an element, such as CHEXA, and the corresponding material, such as MAT1. In addition, entries of the type PSOLID are often used to define domain selections.

RBE2 and RBE3 elements are typically specifications of rigid body data. Selections will be created for the fixed point in the RBE entry. If this point is not adjacent to any other mesh element, a vertex element will also be created in this point. The rest of the RBE2 entry is used to set up selections that can, together with the selection for the fixed point, be used to automatically set up *Rigid Connectors* for simulations using any of the Solid Mechanics, Shell, and Beam physics interfaces in the software.

The NASTRAN bulk data format uses reduced second-order elements, so-called serendipity elements; that is, the center node on quadrilateral mesh faces (`quadNode`) and the center node of hexahedral elements (`hexNode`) are not included. When importing a NASTRAN mesh with second-order elements, the coordinates of these missing node points are interpolated from the surrounding node points using the following formulas: `quadNode = 0.5*quadEdgeNodes - 0.25*quadCornerNodes`, where `quadEdgeNodes` is the sum of the coordinates of the surrounding 4 edge nodes and `quadCornerNodes` is the sum of the coordinates of the surrounding 4 corner nodes, and `hexNode = 0.25*hexEdgeNodes-0.25*hexCornerNodes`, where `hexEdgeNodes` is the sum of the coordinates of the surrounding 12 edge nodes and `hexCornerNodes` is the sum of the coordinates of the surrounding 8 corner nodes.

The vertex (node) numbering used in the NASTRAN format is different from that of the COMSOL native format. The following table describes the relationship between the two formats by providing ordered lists of node numbers. The table lists the NASTRAN format node numbering expressed in terms of the COMSOL native vertex (node) numbering (see the images in section The COMSOL Native Format):

TABLE 4: VERTEX MAPPING BETWEEN COMSOL NATIVE FORMAT AND NASTRAN FORMAT

COMSOL ELEMENT	COMSOL ORDER	COMSOL NATIVE FORMAT	NASTRAN ORDER	NASTRAN FORMAT
Triangle	1	1,2,3	1	1,2,3
Triangle	2	1,2,3,4,5,6	2	1,2,3,4,6,5
Quad	1	1,2,3,4	1	1,2,4,3
Quad	2	1,2,3,4,5,6,7,8,9	2 (serendipity)	1,2,4,3,5,8,-,6,7
Tet	1	1,2,3,4	1	1,2,3,4
Tet	2	1,2,3,4,...,10	2	1,2,3,4,…,10
Pyramid	1	1,2,3,4,5	1	1,2,4,3,5
Pyramid	2	1,2,3,...,14	2 (serendipity)	1,2,4,3,5,6,9,-,7,8,10,11,13,12
Prism	1	1,2,3,4,5,6	1	1,2,4,3,5,6
Prism	2	1,2,3,...,18	2 (serendipity)	1,2,3,4,5,6,7,9,8,10,-,11,-,-,12,13,15,14
Hex	1	1,2,3,4,5,6,7,8	1	1,2,4,3,5,6,8,7
Hex	2	1,2,3,…,27	2 (serendipity)	1,2,4,3,5,6,8,7,9,12,-,10,11,13,-,14,-,-,-,16,-,15,17,20,-,18,19

The NASTRAN format does not support Lagrange (nonserendipity) elements, such as a 27-node hex element. For Lagrange elements, the corresponding serendipity element vertices (or nodes) are exported. This is indicated in the table by a hyphen (-) for the “missing” vertex. To use serendipity elements in COMSOL Multiphysics, the setting for the element order used by the physics needs to be changed as described in the next section.

SECOND-ORDER ELEMENT DATA

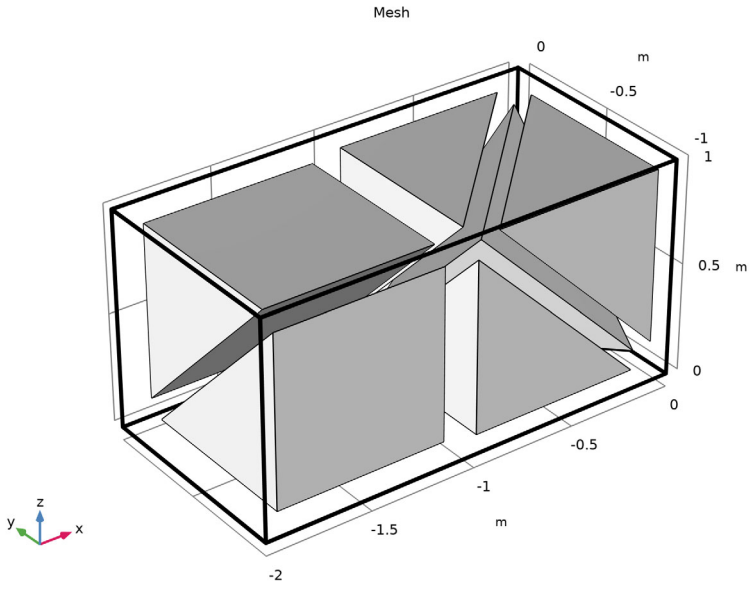
You can import and export mesh data on the COMSOL native or NASTRAN formats containing first- and second-order element information.

After importing mesh data, you can change the element order used for the COMSOL Multiphysics simulation regardless of whether the imported data contains first- or second-order element information. Note that element order is sometimes also referred to as geometry shape function. See Conversions on Import for more information. You change the element in order to curve mesh elements on boundaries. The second-order element information in an imported mesh is used for representing curved boundaries and placing higher-order nodes. However, if second-order information is not available in the imported mesh data file, this information will be estimated from the linear mesh, curving the elements to adjust to the curvature of the face. This estimation is done for all orders higher than two.

If you import a mesh containing second-order element information and then lower the geometric shape function to linear, then the curvature of the boundary is no longer respected. If you import a mesh containing second-order element information and increase the geometry shape function to cubic or higher, then the higher-order nodes are inserted on the boundary based on interpolation using the second-order element data.

Importing Incomplete Mesh Data in the COMSOL Native Format

The examples below illustrate importing a simple mesh with 5 tetrahedral elements and 2 prism elements with various levels of information. The figure below shows the mesh after import. The visualization is created by using a mesh plot with shrunk elements.



IMPORTING MESH DATA WITH DOMAIN ELEMENTS ONLY

The following example illustrates mesh data that contains:

- A header (explained in Specification of COMSOL Formats for Mesh Data)
- Mesh vertices (coordinate values)
- Tetrahedral elements (connectivity data referencing mesh vertices)
- Prism elements (connectivity data referencing mesh vertices)

The file does not contain any geometric entity information for the domain elements. For this type of mesh data, the import will assign the geometric domain number 1 to all elements, if they form one connected component. If there is more than one connected component, the others will be assigned domain numbers 2, 3, 4, etc. Furthermore, a user can manually partition or intersect the mesh by the methods listed in Operations for Editing Imported Meshes.

The example file `mesh_example_without_domain_geom_info.mphtxt` is available from:

https://www.comsol.com/model/72351

File Listing

Domain elements only.

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh2
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
12 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 0 0
0 0 1
0 -1 1
-1 0 1
0 -1 0
-1 0 0
-1 -1 1
-2 0 1
-1 -1 0
-2 0 0
-2 -1 1
-2 -1 0

2 # number of element types

# Type #0

3 tet # type name


4 # number of vertices per element
5 # number of elements
# Elements
0 5 4 1
2 4 6 1
5 4 1 6
3 5 1 6
8 5 6 4

0 # number of geometric entity indices

# Type #1

5 prism # type name
```

```
6 # number of vertices per element
2 # number of elements
# Elements
3 5 6 7 9 10
8 6 5 11 10 9

0 # number of geometric entity indices
```

The mesh data is an object belonging to the mesh class. For more information, see Mesh Class.

IMPORTING MESH DATA WITH DOMAIN ELEMENTS AND DOMAIN INDICES

The following examples illustrate mesh data that contains:

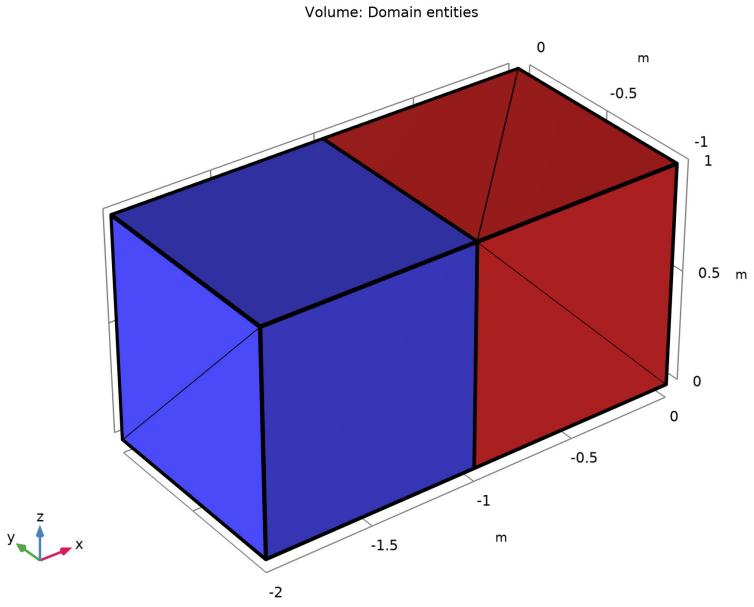
- A header (explained in Specification of COMSOL Formats for Mesh Data)
- Mesh vertices (coordinate values)
- Tetrahedral elements (connectivity data referencing mesh vertices)
- Domain entity indices associated with each tetrahedral element
- Prism elements (connectivity data referencing mesh vertices)
- Domain entity indices associated with 1) the element types, 2) each domain element, and 3) mixed element types

The domain entity information in the file divides the mesh into two domains, numbered 1 for the tetrahedral elements and 2 for the prism elements. Note that any grouping of elements, including elements of different types, can be associated with a domain number or geometric entity. For example, each of the 7 elements can have their respective domain number (from 1 to 7), or all of them have the domain number 1.

The example file `mesh_example_with_domain_geom_info.mphtxt` is available from:

https://www.comsol.com/model/72351

The figure below shows a visualization of the domain information, where blue corresponds to domain 1 and red corresponds to domain 2.



File Listing

Domain elements with geometric entity information.

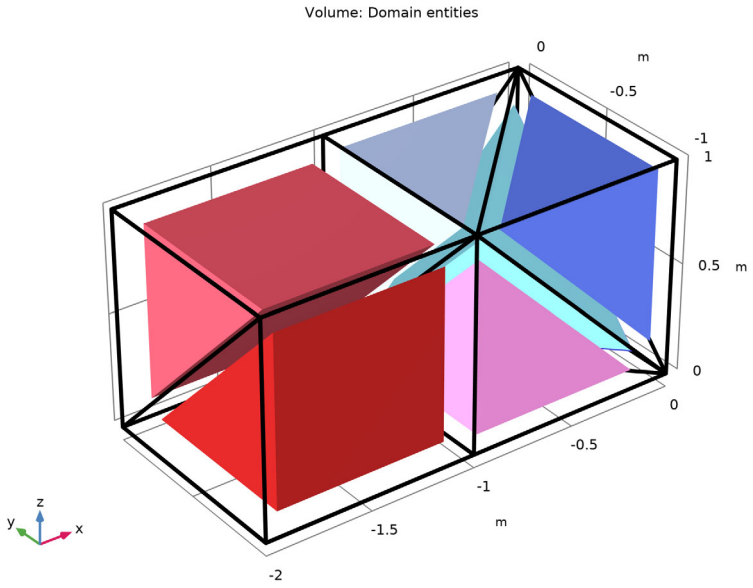
```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh2
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
12 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 0 0
0 0 1
0 -1 1
-1 0 1
0 -1 0
-1 0 0
-1 -1 1
-2 0 1
-1 -1 0
-2 0 0
-2 -1 1
-2 -1 0

2 # number of element types

# Type #0

3 tet # type name


4 # number of vertices per element
5 # number of elements
# Elements
0 5 4 1
2 4 6 1
5 4 1 6
3 5 1 6
8 5 6 4

5 # number of geometric entity indices
# Geometric entity indices
2
2
2
2
2

# Type #1

5 prism # type name


6 # number of vertices per element
2 # number of elements
# Elements
3 5 6 7 9 10
8 6 5 11 10 9

2 # number of geometric entity indices
# Geometric entity indices
1
```

1

Replacing the element and domain entity information of the example above with the following data gives a domain numbering from 1 to 7 for each element. The corresponding example file `mesh_example_with_domain_geom_info_1_7.mphtxt` is available from:

https://www.comsol.com/model/72351

The figure below shows the corresponding domain numbering by color (using shrunk elements):



File Listing

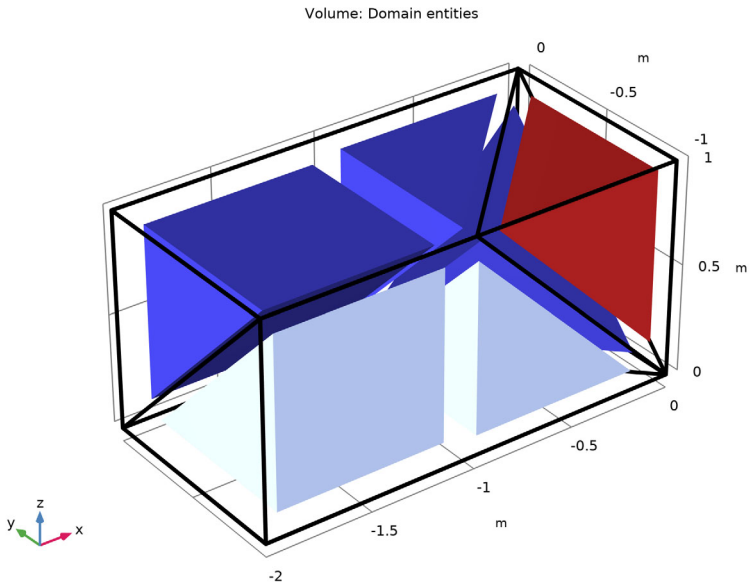
```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh2
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
12 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 0 0
0 0 1
0 -1 1
-1 0 1
0 -1 0
-1 0 0
-1 -1 1
-2 0 1
-1 -1 0
```

```
-2 0 0
-2 -1 1
-2 -1 0

2 # number of element types

# Type #0

3 tet # type name


4 # number of vertices per element
5 # number of elements
# Elements
0 5 4 1
2 4 6 1
5 4 1 6
3 5 1 6
8 5 6 4

5 # number of geometric entity indices
# Geometric entity indices
1
2
3
4
5

# Type #1

5 prism # type name


6 # number of vertices per element
2 # number of elements
# Elements
3 5 6 7 9 10
8 6 5 11 10 9

2 # number of geometric entity indices
# Geometric entity indices
6
7
```

Replacing the element and geometric entity information with the following data gives a domain numbering from 1 to 3. The corresponding example file `mesh_example_with_domain_geom_info_1_3.mphtxt` is available from:

https://www.comsol.com/model/72351

The figure below shows the corresponding domain numbering by color (showing shrunk mesh elements):



File Listing

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh2
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
12 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 0 0
0 0 1
0 -1 1
-1 0 1
0 -1 0
-1 0 0
-1 -1 1
-2 0 1
-1 -1 0
-2 0 0
-2 -1 1
-2 -1 0

2 # number of element types

# Type #0

3 tet # type name
```

```
4 # number of vertices per element
5 # number of elements
# Elements
0 5 4 1
2 4 6 1
5 4 1 6
3 5 1 6
8 5 6 4

5 # number of geometric entity indices
# Geometric entity indices
1
3
1
1
2

# Type #1

5 prism # type name


6 # number of vertices per element
2 # number of elements
# Elements
3 5 6 7 9 10
8 6 5 11 10 9

2 # number of geometric entity indices
# Geometric entity indices
1
2
```

If the import algorithm detects that a domain is not a connected component, it will split the domain until each resulting domain is connected; similarly for faces and edges.

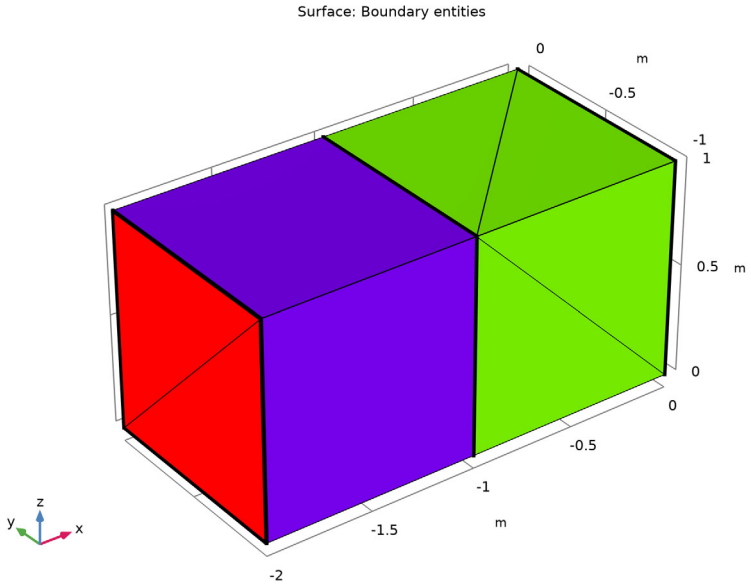
IMPORTING MESH DATA WITH ADDITIONAL BOUNDARY ELEMENTS AND FACE INDICES

The following example illustrates mesh data that contains:

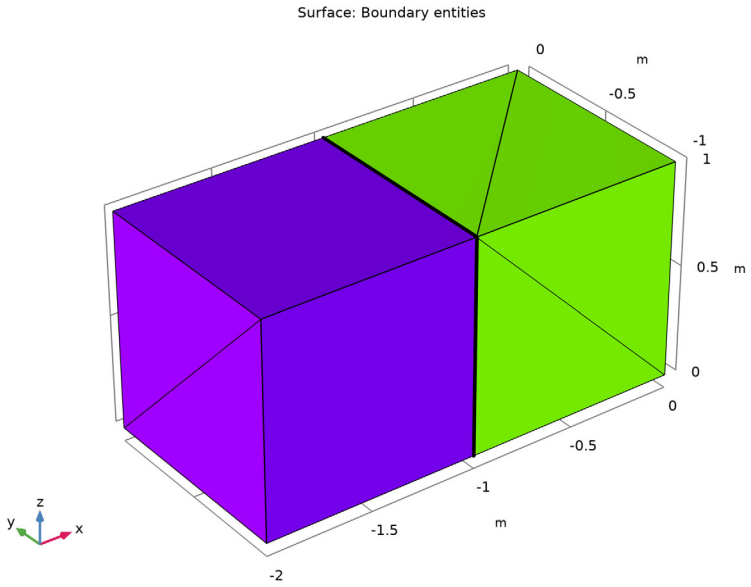
- A header (explained in Specification of COMSOL Formats for Mesh Data)
- Mesh vertices (coordinate values)
- Tetrahedral elements (connectivity data referencing mesh vertices)
- Domain entity indices associated with each tetrahedral element
- Prism elements (connectivity data referencing mesh vertices)
- Domain entity indices associated with each prism element
- Triangular elements (connectivity data referencing mesh vertices)
- Face entity indices associated with each triangular element
- Quad elements (connectivity data referencing mesh vertices)
- Face entity indices associated with each quad element

To transfer boundary partitioning information, you need to include boundary elements with the corresponding geometric entity information in the `.mphtxt` file. To illustrate this, import the file `mesh_example_with_domain_and_boundary_geom_info.mphtxt` with minimal boundary partitioning. This file contains the domain and boundary elements with geometric entity information defining 5 boundaries, as shown in

the figure below. Some of the faces cannot be seen in this figure; the one on the far side of the mesh and the face between the two domains.



If you would import the file `mesh_example_with_domain_geom_info.mphtxt`, which has no boundary information, using the same import settings, the resulting boundary partitioning is shown in the figure below.



The imported mesh now has just 3 faces, because the minimal boundary partitioning generates the minimal possible partitioning that is required by the topological criteria used by the import. Note that the interior face entity (index 2) between domains 1 and 2 is not visible in the figure.

The example file `mesh_example_with_domain_and_boundary_geom_info.mphtxt` is available from:

https://www.comsol.com/model/72351

File Listing

Domain and face elements with geometric entity information.

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh2
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
12 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 -1 1
0 0 1
0 0 0
-1 0 1
0 -1 0
-1 -1 1
-1 0 0
-2 0 1
-1 -1 0
-2 -1 1
-2 0 0
-2 -1 0

4 # number of element types

# Type #0

3 tet # type name


4 # number of vertices per element
5 # number of elements
# Elements
2 4 1 6
4 0 1 5
1 6 4 5
1 3 6 5
6 8 4 5

5 # number of geometric entity indices
# Geometric entity indices
1
1
1
1
1

# Type #1

5 prism # type name


6 # number of vertices per element
2 # number of elements
# Elements
5 3 6 9 7 10
6 8 5 10 11 9
```

```
2 # number of geometric entity indices
# Geometric entity indices
2
2

# Type #2

3 tri # type name


3 # number of vertices per element
14 # number of elements
# Elements
2 1 4
2 6 1
2 4 6
4 1 0
4 0 5
0 1 5
1 6 3
1 3 5
6 4 8
8 4 5
5 6 3
9 7 10
6 5 8
10 11 9

14 # number of geometric entity indices
# Geometric entity indices
0
1
1
0
1
1
1
1
1
1
2
4
2
4

# Type #3

4 quad # type name


4 # number of vertices per element
4 # number of elements
# Elements
5 3 9 7
3 6 7 10
6 8 10 11
8 5 11 9

4 # number of geometric entity indices
# Geometric entity indices
3
3
3
3
```

IMPORTING SURFACE MESH DATA WITH BOUNDARY ELEMENTS AND FACE INDICES

The following example illustrates mesh data that contains:

- A header (explained in Specification of COMSOL Formats for Mesh Data)
- Mesh vertices (coordinate values)
- Triangular elements (connectivity data referencing mesh vertices)
- Face entity indices associated with each triangular element
- Quad elements (connectivity data referencing mesh vertices)
- Face entity indices associated with each quad element

The example file `mesh_example_with_boundary_geom_info.mphtxt` is available from:

https://www.comsol.com/model/72351

File Listing

Face elements with geometric entity information.

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh2
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
12 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 -1 1
0 0 1
0 0 0
-1 0 1
0 -1 0
-1 -1 1
-1 0 0
-2 0 1
-1 -1 0
-2 -1 1
-2 0 0
-2 -1 0

2 # number of element types

# Type #0

3 tri # type name


3 # number of vertices per element
14 # number of elements
# Elements
2 1 4
2 6 1
2 4 6
4 1 0
4 0 5
0 1 5
```

```
1 6 3
1 3 5
6 4 8
8 4 5
5 6 3
9 7 10
6 5 8
10 11 9

14 # number of geometric entity indices
# Geometric entity indices
0
1
1
0
1
1
1
1
1
1
2
4
2
4

# Type #1

4 quad # type name


4 # number of vertices per element
4 # number of elements
# Elements
5 3 9 7
3 6 7 10
6 8 10 11
8 5 11 9

4 # number of geometric entity indices
# Geometric entity indices
3
3
3
3
```

The boundary subdivision will be taken “as is” from the file unless a boundary does not form a connected component. In that case, a further subdivision will be made during the import.

IMPORTING MESH DATA WITH EDGE AND VERTEX ELEMENTS AND INDICES

The following example illustrates mesh data that contains:

- A header (explained in Specification of COMSOL Formats for Mesh Data)
- Mesh vertices (coordinate values)
- Vertex elements (data referencing mesh vertices)
- Vertex entity indices associated with each vertex element
- Edge elements (connectivity data referencing mesh vertices)
- Edge entity indices associated with each edge element

The example file `mesh_example_with_edge_and_vertex_geom_info.mphtxt` is available from:

https://www.comsol.com/model/72351

File Listing

Edge and vertex elements with geometric entity information.

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh2
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
12 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 -1 1
0 0 1
0 0 0
-1 0 1
0 -1 0
-1 -1 1
-1 0 0
-2 0 1
-1 -1 0
-2 -1 1
-2 0 0
-2 -1 0

2 # number of element types

# Type #0

3 vtx # type name


1 # number of vertices per element
12 # number of elements
# Elements
0
1
2
3
4
5
6
7
8
9
10
11

12 # number of geometric entity indices
# Geometric entity indices
3
1
0
5
2
7
4
9
6
```

```
11
8
10

# Type #1

3 edg # type name


2 # number of vertices per element
20 # number of elements
# Elements
10 7
7 9
11 9
10 11
6 3
3 5
8 5
6 8
4 0
0 5
4 8
2 4
2 6
1 0
1 3
2 1
5 9
8 11
6 10
3 7

20 # number of geometric entity indices
# Geometric entity indices
16
18
19
17
8
11
13
9
5
7
6
1
2
3
4
0
15
14
10
12
```

If a file is missing the vertex elements information, then the import will reconstruct vertex elements from the edge intersection points.

Note that not all possible variants are illustrated in the examples above. It is also possible to import a file with other combinations of element and index data.

Complete Mesh Data on the COMSOL Native Format

When exporting mesh data on the COMSOL native formats, the default settings ensure that complete mesh data is always provided with geometric entity information for all spatial dimensions.

MESH DATA WITH COMPLETE ELEMENT AND GEOMETRIC ENTITY INFORMATION

- A header (explained in Specification of COMSOL Formats for Mesh Data)
- Mesh vertices (coordinates)
- Tetrahedral elements (connectivity data referencing mesh vertices)
- Domain entity indices associated with each tetrahedral element
- Prism elements (connectivity data referencing mesh vertices)
- Domain entity indices associated with each prism element
- Vertex elements (data referencing mesh vertices)
- Vertex entity indices associated with each vertex element
- Edge elements (connectivity data referencing mesh vertices)
- Edge entity indices associated with each edge element
- Triangular elements (connectivity data referencing mesh vertices)
- Face entity indices associated with each triangular element
- Quad elements (connectivity data referencing mesh vertices)
- Face entity indices associated with each quad element

The example file `mesh_example_with_complete_geom_info.mphtxt` is available from:

https://www.comsol.com/model/72351

Note that you can get a file containing complete mesh data when exporting by first importing any of the incomplete example data files above and then exporting using the default settings.

File Listing

Complete mesh and geometric entity information.

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh2
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
12 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 -1 1
0 0 1
0 0 0
-1 0 1
0 -1 0
-1 -1 1
-1 0 0
-2 0 1
-1 -1 0
-2 -1 1
-2 0 0
-2 -1 0
```

```
6 # number of element types

# Type #0

3 tet # type name


4 # number of vertices per element
5 # number of elements
# Elements
2 4 1 6
4 0 1 5
1 6 4 5
1 3 6 5
6 8 4 5

5 # number of geometric entity indices
# Geometric entity indices
1
1
1
1
1

# Type #1

5 prism # type name


6 # number of vertices per element
2 # number of elements
# Elements
5 3 6 9 7 10
6 8 5 10 11 9

2 # number of geometric entity indices
# Geometric entity indices
2
2

# Type #2

3 vtx # type name


1 # number of vertices per element
12 # number of elements
# Elements
0
1
2
3
4
5
6
7
8
9
10
11

12 # number of geometric entity indices
# Geometric entity indices
3
1
0
5
2
7
4
9
6
```

```
11
8
10

# Type #3

3 edg # type name


2 # number of vertices per element
20 # number of elements
# Elements
10 7
7 9
11 9
10 11
6 3
3 5
8 5
6 8
4 0
0 5
4 8
2 4
2 6
1 0
1 3
2 1
5 9
8 11
6 10
3 7

20 # number of geometric entity indices
# Geometric entity indices
16
18
19
17
8
11
13
9
5
7
6
1
2
3
4
0
15
14
10
12

# Type #4

3 tri # type name


3 # number of vertices per element
14 # number of elements
# Elements
2 1 4
2 6 1
2 4 6
4 1 0
4 0 5
0 1 5
1 6 3
1 3 5
```

```
6 4 8
8 4 5
5 6 3
9 7 10
6 5 8
10 11 9

14 # number of geometric entity indices
# Geometric entity indices
0
1
2
0
4
3
1
3
2
4
5
10
5
10

# Type #5

4 quad # type name


4 # number of vertices per element
4 # number of elements
# Elements
5 3 9 7
3 6 7 10
6 8 10 11
8 5 11 9

4 # number of geometric entity indices
# Geometric entity indices
8
6
7
9
```

Second-Order Element Data

You can also export second-order elements. See Exporting to the COMSOL Native Formats for more information.

The abbreviated example below shows a mesh with second-order tetrahedral elements exported to the COMSOL native (`.mphtxt`) format.

The example file `mesh_example_with_second_order_info.mphtxt` is available from:

https://www.comsol.com/model/72351

Partial File Listing

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh8
1 # number of types
# Types
3 obj

# --------- Object 0 ----------
```

```
0 0 1
4 Mesh # class
4 # version
3 # sdim
113 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
0 0 0
0 0.5 0.5
0.5 0.5 0
0 1 0
0.5 0 0.5
...
0.5 0.25 0.75
0 0.25 0.75
0.25 0 0.75
0.5 0 1
0.25 0.25 1

4 # number of element types

# Type #0

3 vtx # type name


1 # number of vertices per element
12 # number of elements
# Elements
0
3
5
6
9
11
13
15
17
20
21
22

12 # number of geometric entity indices
# Geometric entity indices
0
2
1
4
3
6
5
8
7
10
9
11

# Type #1

4 edg2 # type name


3 # number of vertices per element
20 # number of elements
# Elements
0 6 24
0 5 28
6 13 30
6 11 35
0 3 39
```

```
...
17 22 91
9 17 96
5 9 101
13 17 104
5 13 111

20 # number of geometric entity indices
# Geometric entity indices
2
0
8
9
1
...
15
7
3
11
4

# Type #2

4 tri2 # type name


6 # number of vertices per element
44 # number of elements
# Elements
4 0 6 27 32 24
2 6 0 34 23 24
2 0 3 23 40 39
1 3 0 42 26 39
8 11 3 47 45 46
...
1 5 9 109 98 101
4 5 0 110 27 28
4 13 5 107 110 111
7 9 5 100 112 101
7 5 13 112 105 111

44 # number of geometric entity indices
# Geometric entity indices
1
2
2
0
4
...
0
1
1
3
3

# Type #3

4 tet2 # type name


10 # number of vertices per element
48 # number of elements
# Elements
2 4 0 6 25 23 27 34 32 24
6 2 4 10 34 32 25 33 36 31
0 2 1 4 23 26 38 27 25 29
2 1 3 0 38 40 42 23 26 39
2 3 1 8 40 38 42 41 45 43
...
4 10 7 13 31 108 103 107 106 105
4 1 0 5 29 27 26 110 109 28
4 1 5 7 29 110 109 108 102 112
```

```
1 7 9 5 102 98 100 109 112 101
4 7 5 13 108 110 112 107 105 111

48 # number of geometric entity indices
# Geometric entity indices
1
1
1
1
1
...
1
2
2
1
2
...
1
1
1
```

As can be seen from the contents, the number of vertex elements is 12, whereas the number of mesh vertices is 113.

For version 5.4 and earlier, an exported file may contain additional sections.

Mesh Data with Selections

When exporting to the COMSOL native format, you can include selections. For more information about selections, see Selections.

The example below shows the header and the last part of a file containing three selections. Each selection is an object with an individual tag.

The corresponding example files can be downloaded from:

https://www.comsol.com/model/72351

The filenames are:

```
busbar_with_selections.mph
busbar_with_selections.mphtxt
```

Partial File Listing

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
4 # number of tags
# Tags
5 mesh1
10 mesh1_sel1
10 mesh1_sel2
10 mesh1_sel3
4 # number of types
# Types
3 obj
3 obj
3 obj
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
232 # number of mesh vertices
0 # lowest mesh vertex index
```

```
# Mesh vertex coordinates
...

# --------- Object 1 ----------

0 0 1
9 Selection # class
0 # Version
8 Ti Bolts # Label
5 mesh1 # Geometry/mesh tag
3 # Dimension
6 # Number of entities
# Entities
2
3
4
5
6
7

# --------- Object 2 ----------

0 0 1
9 Selection # class
0 # Version
12 Copper Piece # Label
5 mesh1 # Geometry/mesh tag
3 # Dimension
1 # Number of entities
# Entities
1

# --------- Object 3 ----------

0 0 1
9 Selection # class
0 # Version
11 Single Bolt # Label
5 mesh1 # Geometry/mesh tag
3 # Dimension
2 # Number of entities
# Entities
2
3
```

The selections are objects belonging to the selection class. For more information, see Selection Class.

Specification of COMSOL Formats for Mesh Data

A COMSOL native text or binary data file, with extensions `.mphtxt` and `.mphbin`, respectively, is used to store COMSOL-specific data related to the mesh and geometry. Both the text and binary file formats contain the same data in the same order. The text files can also contain comments.

The section Mesh Data Formats includes specific examples of COMSOL native text files for mesh data. This section gives a formal specification of the format, again for storing mesh data. For a more general description, including information on how geometry data is stored, see the *COMSOL Multiphysics Programming Reference Manual.*

File Structure

VERSION NUMBER

The COMSOL native data format has a global version number so that it is possible to revise the whole structure in the future if needed. The first entry in each file is the file format, indicated by two integers. The first integer is the major file version and the second is referred to as the minor file version. For the current version, supported by COMSOL Multiphysics version 6.2, the first two entries in a file are `0 1`.

The example below shows the first few lines of a mesh data file.

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
...
```

Comments, starting with the character `#`, are ignored. (These can all be removed and the file will still be imported.)

TAGS

The next data fields are the number of tags and the tags, as shown in the example below.

```
...
# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh1
...
```

Just as in this example, the number of tags is typically 1.

The *tags* are strings that function as identifiers so that objects can refer to each other. Tags are listed in the subsequent lines. In this example, there is one tag, `mesh1`. The number `5` preceding the tag is the number of characters in the string `mesh1`.

In the example below, there are 3 tags.

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
3 # number of tags
# Tags
5 mesh1
10 mesh1_sel1
10 mesh1_sel2
```

TYPES

The next data fields are the number of types and the types, as shown in the example below.

```
...
3 # number of types
# Types
3 obj
3 obj
3 obj
...
```

In the current version, 0 1, of the COMSOL native data format, only Object types (`obj`) are supported.

Note that references to `type` in the bulk of a COMSOL native data file, such as in the excerpt below:

```
1 # number of element types

# Type #0

3 tri # type name
```

are referring to element type and are not related to the `type` of the header.

OBJECTS

The objects represent the serialization (conversion to a file format) of the COMSOL data structures. Each object starts with the header 0 0 1, followed by a string that defines which type of object follows. For mesh data, objects are instances of the serializable classes Mesh Class or Selection Class.

The example below shows file export using the API command `model.mesh(<tag>).export(<filename>)` or *Mesh>Export to File* in the user interface to save a COMSOL mesh. The tag equals the variable name `mesh1`; the type is set to `obj`; and the file contains the serialization of the mesh object, including point coordinates and element data of the mesh. In this example, the file also contains a selection object.

```
# Created by COMSOL Multiphysics.

# Major & minor version
0 1
2 # number of tags
# Tags
5 mesh1
10 mesh1_sel1
2 # number of types
# Types
3 obj
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
3 # sdim
782 # number of mesh vertices
0 # lowest mesh vertex index

# Mesh vertex coordinates
-0.70710678118654791 -0.70710678118654768 0
-0.83146885388289216 -0.55556976368981836 0
…
…
# --------- Object 1 ----------

0 0 1
9 Selection # class
0 # Version
5 Fluid # Label
5 mesh1 # Geometry/mesh tag
3 # Dimension
1 # Number of entities
# Entities
1
```

Data Types

The following data types are used in the COMSOL mesh format:

- *Boolean* refers to an 8-bit signed character that must be 0 or 1.
- *Character* refers to an 8-bit signed character.
- *Integer* refers to a 32-bit signed integer.
- *Double* refers to a 64-bit double.

Matrices are stored in row-major order. In this documentation, brackets are used to indicate a matrix. Hence, `integer[3][4]` means that 12 integers representing a matrix are stored in the file. The first three entries correspond to the integers in the first row of the matrix, and so on.

Text and Binary File Formats

Both the text and binary file formats contain the same data in the same order. The file size of the binary file format is generally smaller than that of the text file format. The text format is human-readable in any text editor, whereas the binary format is not.

TEXT FILE FORMAT

In a COMSOL native text data file, with the file extension `.mphtxt`, values are stored as text separated by whitespace characters.

Lexical conventions:

- Strings are serialized as the length of the string followed by a space and then the characters of the string; for example, "`6 COMSOL`".
- The software ignores everything following a `#` on a line except when reading a string. This makes it possible to store comments in the file.

BINARY FILE FORMAT

In the COMSOL native binary data file, with the extension `.mphbin`, integers and doubles are stored in little-endian byte order. Strings are stored as the length of the string (integer) followed by the characters of the string (integers).

Serializable Classes for Mesh Data

The `Serializable Class` is the base class, and all other classes are subclasses of `Serializable Class` or of its subclasses.

In the **Field** sections for each class below, the fields appear in the table in the exact order they must appear in the data files.

The **Variable** column lists the internal variables used in the descriptions of data types and in the definitions of what the class implements.

For an example of the serialization format — specifically, of a file containing a 3D mesh with tetrahedral and prism elements — see `mesh_example_4.mphtxt` in the installation folder `\applications\COMSOL_Multiphysics\Meshing_Tutorials`.

The classes listed below are only for mesh data. For a listing of all classes, see the *COMSOL Multiphysics Programming Reference Manual*. Note that there are additional undocumented classes that are only for internal purposes and that may in some cases be exported.

Mesh Class

SUPPORTED VERSIONS

Up to 4. Version 4 is documented below.

SUBCLASS OF

`Serializable Class`

FIELDS

The class is defined by the following fields:

DATA TYPE	VARIABLE	DESCRIPTION
integer		Version.
integer	d	Space dimension (if equal to 0 no more fields).
integer	np	Number of mesh points.
integer		Lowest mesh point index.
double[d][np]	p	Mesh vertices.
integer	nt	Number of element types (equals the number of repeats of the following fields).
string		Element type.
integer	nep	Number of nodes per element.
integer	ne	Number of elements.
integer[ne][nep]	elem	Matrix of point indices for each element.
integer	ndom	Number of geometric entity values.
integer[ndom]	dom	Vector of geometric entity labels for each element.

DESCRIPTION

The geometric entity numbering for points, edges, and boundaries typically starts from 0 and the geometric entity numbering for domains normally starts from 1 when defining a mesh through a COMSOL Multiphysics mesh file.

EXAMPLE

The following displays a mesh with triangular elements on a unit square. Neither vertex nor edge elements are present.

```
# Major & minor version
0 1
1 # number of tags
# Tags
5 mesh1
1 # number of types
# Types
3 obj

# --------- Object 0 ----------

0 0 1
4 Mesh # class
4 # version
2 # sdim
4 # number of mesh vertices
0 # lowest mesh vertex index
```

```
# Mesh vertex coordinates
0 1
0 0
1 1
1 0

1 # number of element types

# Type #0

3 tri # type name

3 # number of vertices per element
2 # number of elements
# Elements
0 1 2
3 2 1

2 # number of geometric entity indices
# Geometric entity indices
1
1
```

Selection Class

SUPPORTED VERSIONS

0

SUBCLASS OF

Serializable Class

FIELDS

The class is defined by the following fields:

DATA TYPE	DESCRIPTION
integer	Version.
string	Selection label. The string is encoded in UTF-8.
string	Tag of corresponding object (mesh) in file.
integer	Dimension of selection (0: vertex; 1: edge; 2: face; 3: domain in 3D).
integer	Number of entities.
integer[]	The indices of the entities for the selection. The integers specify the 0-based indices of the entities (1-based for domains).

DESCRIPTION

Selections can appear in files containing a mesh. Each selection refers to a set of entities that needs to be defined by the mesh in the file.

EXAMPLE

The following example displays a domain selection in 3D named Fluid, specifying domains 1 and 3.

```
0 # Version
5 Fluid # Label
5 mesh1 # Geometry/mesh tag
3 # Dimension
2 # Number of entities
# Entities
1
3
```

SUPPORTED VERSIONS

0

FIELDS

This is the abstract base class of all other classes. It has no fields.

DESCRIPTION

Serializable is the abstract base class. It is used to indicate that a field can contain all supported types.

Index