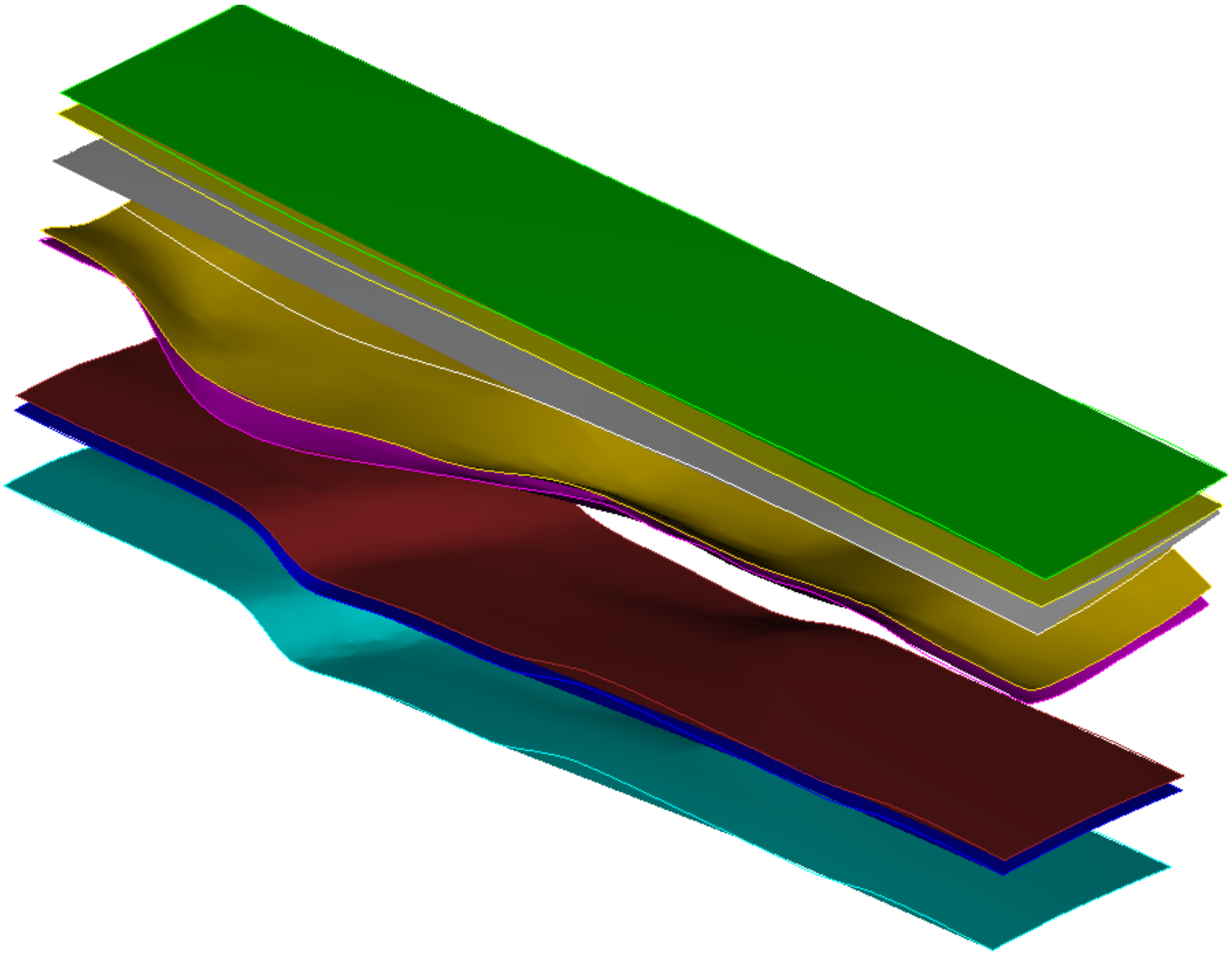
MeshAssist

v1.0.0



An open-source and cross-platform meshing assistant tool

MeshAssist v1.0.0
User Manual

Hom Nath Gharti[1], Princeton University, USA
Leah Langer, Princeton University, USA
Michael Roth, NORSAR, Norway
Jeroen Tromp, Princeton University, USA
Uno Vaaland, Princeton University, USA
Zhenzhen Yan, Institute of Remote Sensing and Digital Earth, CAS, China

November 26, 2017

[1]formerly at: NORSAR, Norway; and Institute of Engineering, Tribhuvan University, Nepal

Licensing

Acknowledgments

Part of this document was prepared by using the documentation generator "Doxygen" (Main developer: Dimitri van Heesch) and the Matlab parser for Doxygen "doxymatlab" (Main developer: Fabrice).

Contents

Chapter 1

Introduction

1.1 Background

MeshAssist is a collection of tools which assists meshing of complex and realistic 2D/3D models for FEM/SPECFEM simulations. As its name suggests, it is NOT a meshing software. It is only a meshing assistant!

1.2 Cite as

Gharti, H. N.; Langer, L.; Roth, M.; Tromp, J.; Vaaland, U.; Yan, Z. (2017). MeshAssist: an open-source and cross-platform meshing assistant tool. Zenodo. http://doi.org/10.5281/zenodo.883448

1.3 Status summary

Digital Elevation Model (DEM)	: Yes
DXF AUTOCAD Model	: Yes
GOCAD Model	: Yes
EXODUS mesh	: Yes
GiD mesh	: Yes
VTK file	: Yes
VTU file	: Yes
XYZ file	: Yes

1.4 Changes made since the last version

- This is the first version.

Chapter 2

Getting started

2.1 Package structure

The MeshAssist package can be obtained using Git. Use the following command in the terminal:

```
git clone recursive https://github.com/homnath/MeshAssist.git
```

The package has the following structure:

`MeshAssist/`

`LICENSE`	: License.
`Makefile`	: brief description of the package.
`bin/`	: all object files and executables are stored in this folder.
`doc/`	: documentation file/s for the MeshAssist package.
`input/`	: contains input files.
`output/`	: default output folder. All output files are stored in this folder unless the different output path is defined from command.
`src/`	: contains all source files.

2.2 Prerequisites

The package requires Make utility, latest C and Fortran compilers. For matlab files, Matlab is necessary.

2.3 Configuration

Open src/Makefile and modify the C and Fortran compilers if necessary.

2.4 Compile

Type the following command in the terminal
make all
Matlab files can be opened in and run from Matlab.

2.5 Run

command input_file [*Options*]
Example:
./bin/xyz2jou ./input/xyz2jou_example.utm
See Chapter "File Documentation" for all available commands.

2.6 Bug Report

hgharti_AT_princeton_DOT_edu

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

File Documentation

4.1 dem2vti.m File Reference

Converts DEM image file to ASCII XYZ file.

4.1.1 Detailed Description

Converts DEM image file to ASCII XYZ file.
This program converts DEM map (TIFF image format) to ASCII XYZ file and optionally ParaView/VTK VTI file format according to the parameters defined in the input file.
Note: Choosing a relatively small sampling interval may freeze the program due to the 'surfl' function.

Usage:

dem2vti(*input_file*, [*output_path*])

Example:
dem2vti('dem2vti_example.in')
OR
dem2vti('dem2vti_example.in','../output')

Input:

input_file: Name of DEM file.

Options:

An optional argument which must be a legitimate path can be provided as the output path. The default path is the current path.

Output:

All output files will be saved in the output_path provided. If no output_path is provided, the current path is used.

4.2 dxf2jou.m File Reference

Converts DXF AUTOCAD file to CUBIT/Trelis journal file.

4.2.1 Detailed Description

Converts DXF AUTOCAD file to CUBIT/Trelis journal file.
This function converts AUTOCAD 2000 (other ?) DXF ASCII file to a CUBIT/Trelis journal file, and optionally to a ParaView/VTK VTU ASCII file — an unstructured mesh file (.vtu). The function extracts only the faces represented by 'AcDbFace' tokens in the DXF file.

Usage:

dxf2jou(*input_file*, [save_vtu])

Example:
dxf2jou('../input/dxf2jou_example.dxf')
OR
dxf2jou('../input/dxf2jou_example.dxf',1)

Input:

input_file: DXF input file name

Options:

An optional argument can be provided to save VTU ASCII file (0: No [DEFAULT] or 1: yes)
.

Output:

VTK .vtu file which can be visualized with ParaView/VTK

4.3 exodus2semgeotech.c File Reference

Converts ASCII exodus file to SPECFEM3D_GEOTECH files.

Functions

- void **removeExtension** (char *, char *)
- int **get_int** (int *, char *, char *)
- int **look_int** (int *, char *, char *)
- int **getfirstquote** (char *, char *)
- int **main** (int argc, char **argv)

4.3.1 Detailed Description

Converts ASCII exodus file to SPECFEM3D_GEOTECH files.
This program converts the Binary (provided that "ncdump" command exists) or ASCII exodus file exported from CUBIT/Trelis to several mesh files required by the SPECFEM3D_GEOTECH package. The "ncdump" command is a part of NetCFD library which is generally installed already in LINUX. If it is not installed, it can be downloaded for free from https://www.unidata.ucar.edu/software/netcdf/

Dependencies:

stringmanip.c: string manipulation routines

Compile:

gcc exodus2semgeotech.c -o exodus2semgeotech

Usage:

exodus2semgeotech *input_file* [*Options*]

Example:
exodus2semgeotech tunnel.e -bin=1
or
exodus2semgeotech tunnel.txt

Options:

- -fac: Use this option to multiply coordinates by some factor. This is important for unit conversion, e.g., to convert m to km use -fac=0.001 [DEFAULT 1]
- -bin: Use this option if you want to convert exodus binary file directly, provided that the command "ncdump" is in the path. The command "ncdump" is a part of netCDF library that can be downloaded for free from http://www.unidata.ucar.edu/downloads/netcdf/index.jsp. Use -bin=1 for binary or -bin=0 for ascii file. [DEFAULT 0]

Issues:

- - This does not work with older versions of CUBIT. For the older versions use exodusold2semgeotech.↩ c.

Basic steps starting from CUBIT/TRELIS:

step 1: prepare mesh in TRELIS/CUBIT

- Define material regions using "Blocks"

  For example:
  block 1 add volume 1
  block 2 add volume 2 3

  will assign material region 1 to volume 1 and material region 2 to volumes 2 and 3. These material regions will be used to define material properties in "∗material_list". This program will NOT generate "∗material_list". The file "material_list" must be created to run SPECFEM3D_GEOTECH!

- Define surface boundary conditions using "Nodesets" or "Sidesets" – nodal boundary conditions must be defined using node set -> each node set name must contain the corresponding BC names as defined in char ∗ns_bcname[] below
  e.g., node set name can be front_nsbcux or front_nsbcux_nsbcuy etc. – surface boundary conditions must be defined using side set -> each side set name must contain the corresponding BC names as defined in char ∗ss_bcname[] below
  e.g., side set name can be front_ssbcux or front_ssbcux_ssbcuy etc.

  For example:
  sideset 1 add surface 1
  sideset 1 name 'bottom_ssbcux_ssbcuy_ssbcuz'

  will define a surface in which all displacement components are prescribed.

  Note: All the above commands can also be executed using TRELIS/CUBIT GUI. "sideset 1 name 'bottom_ssbcux_ssbcuy_ssbcuz'" is equivalent to clicking 'sideset 1' and renaming.

step2: export mesh file as exodus file say "tunnel.e" (use 3D option)

step3: convert "tunnel.e" to SPECFEM3D files exodus2semgeotech tunnel.e -bin=1
There will be several output files:

- ∗_coord_? : coordinates file => total number of nodes followed by nodal coordinate ? (? -> x, y, z)
- ∗_connectivity : element file => total number of elements followed by connectivity list
- ∗_material_id : material file => total number of elements followed by material IDs
- ∗_??bcu? : node IDs which have u? defined as the boundary conditions (?? -> ns or ss, ? -> x, y, z). Total number of entities (nodes or faces) followed by element ID and surface nodes.

4.4 exodus2specfem2d.c File Reference

Convert ASCII exodus file to SPECFEM2D format.

Functions

- void **removeExtension** (char *, char *)
- int **get_int** (int *, char *, char *)
- int **look_int** (int *, char *, char *)
- int **getfirstquote** (char *, char *)
- int **shape** (double, double, double **)
- int **check_normal** (double [3][4], double [3])
- int **isclockwise** (int, double [], double [])
- int **main** (int argc, char **argv)
- int **isclockwise** (int n, double x[n], double z[n])

4.4.1 Detailed Description

Convert ASCII exodus file to SPECFEM2D format.
This program converts the ASCII exodus file exported from CUBIT to several input files required by the SPECFEM2D program. Currently, this program only handles the 2D quadrilateral elements with four nodes. The binary exodus file (e.g., .e file) needs to be converted into an ASCII file, generally using a free console application "ncdump" which is a part of the netCDF library, and can be downloaded from
http://www.unidata.ucar.edu/downloads/netcdf/index.jsp. Please see the detailed steps below.

Dependencies:

stringmanip.c: string manipulation routines

Compile:

gcc exodus2specfem2d.c -o exodus2specfem2d -lm

Usage:

exodus2specfem2d *input_file* [*Options*]

Example:
exodus2specfem2d mesh.e -bin=1
or
exodus2specfem2d mesh.txt

Options:

- -fac: Use this option to multiply coordinates by some factor. This is important for unit conversion, e.g., to convert m to km use -fac=0.001 [DEFAULT 1]
- -bin: Use this option if you want to convert exodus binary directly, provided that the command "ncdump" is in the path. The command "ncdump" is a part of netCDF library that can be downloaded for free from http://www.unidata.ucar.edu/downloads/netcdf/index.jsp. Use -bin=1 for binary or -bin=0 for ascii file. [DEFAULT 0]
- -order: Use this option to check the connectivity order and make sure that the connectivity is in counterclockwise order. Use -order=1 for checking or -order=0 for no checking [DEFAULT 0].
- -head: Use this option to attach head of input file to output file names. Use -head=1 to attach header or -head=0 not to attach [DEFAULT 0]
- -tomo: Use this option for tomography model. Since tomography model uses negative identifiers, this option will write negative block IDs. Use -tomo=1 to make negative block IDs or -tomo=0 not to make [DEFAULT 0]

Basic steps starting from TRELIS:

Step 1: prepare mesh in TRELIS/CUBIT

- Define material regions using "Blocks"

  For example:
  block 1 add surface 1
  block 2 add surface 2 3

  will assign material region 1 to surface 1 and material region 2 to surfaces 2 and 3. These material regions will be used to define material properties in "Par_file". This program will NOT generate "Par_file". The file "Par_file" must be created to run SPECFEM2D!

- Define element type to be QUAD4

  For example:
  block all element type quad4

  NOTE: If the element types are SHELL or SHELL4, "Default" or 3D option should be selected during export. If the element type is QUAD or QUAD4, 3D option should be selected. With default or 2D data, it saves only X and Y coordinates which is not always correct. Make sure that the node ordering is strictly anticlockwise (no longer necessary!) for all the elements in CUBIT.

- Define surface boundary conditions using "Sidesets"

  For example:
  sideset 1 add curve 1
  sideset 1 name 'free_surface_file'

will define a free or absorbing surface boundary condition on surface. Similary,
sideset 2 add curve 3
sideset 2 name 'absorbing_surface_file'

will define absorbing boundary condition on the curve 3. Note: All the above commands can also be executed using TRELIS/CUBIT GUI. "sideset 1 name 'free_surface_file'" is equivalent to clicking sideset 1 and renaming.

Step 2: export mesh file as exodus file say "mesh.e" (always use 3D option!)

Step 3: convert "mesh.e" to SPECFEM2D files exodus2specfem2d mesh.e -bin=1
There will be several output files:

- coordinates : coordinates file => total number of nodes followed by nodal coordinate ? (? -> x, y, z)
- connectivity : element file => total number of elements followed by connectivity list
- materials : material file => total number of elements followed by material IDs
- surface* : sourface boundary condition files => total number of elements followed by element ID and surface nodes

4.5 exodus2specfem3d.c File Reference

Converts ASCII exodus file to SPECFEM3D files.

Functions

- void **removeExtension** (char *, char *)
- int **get_int** (int *, char *, char *)
- int **look_int** (int *, char *, char *)
- int **getfirstquote** (char *, char *)
- int **check_normal** (double [3][4], double [3])
- int **main** (int argc, char **argv)

4.5.1 Detailed Description

Converts ASCII exodus file to SPECFEM3D files.
This program converts the Binary (provided that "ncdump" command exists, type "ncdump" to check whether "ncdump" command exists.) or ASCII exodus file exported from TRELIS/↩ CUBIT to several mesh files required by the SPECFEM3D Cartesian package. The "ncdump" commad is a part of NetCFD library which is generally installed already in LINUX. If this library is not found, it can be downloaded for free from
https://www.unidata.ucar.edu/software/netcdf/

Dependencies:

stringmanip.c: string manipulation routines

Compile:

gcc exodus2specfem3d.c -o exodus2specfem3d

Usage:

exodus2specfem3d *input_file* [*Options*]

Example:
exodus2specfem3d tunnel.e -bin=1
or
exodus2specfem3d tunnel.txt

Options:

- -fac: Use this option to multiply coordinates by some factor. This is important for unit conversion, e.g., to convert m to km use -fac=0.001 [DEFAULT 1]
- -bin: Use this option if you want to convert exodus binary directly, provided that the command "ncdump" is in the path. The command "ncdump" is a part of netCDF library that can be downloaded for free from
http://www.unidata.ucar.edu/downloads/netcdf/index.jsp. Use -bin=1 for binary or -bin=0 for ascii file. [DEFAULT 0]

- -norm: Use this option to check the normal of the faces in order to make sure that the surface nodes are in the right order. Use -norm=1 for checking or -norm=0 for no checking [DEFAULT 0]. Normally this is not necessary.
- -head: Use this option to attach head of input file to output file names. Use -head=1 to attach header or -head=0 not to attach [DEFAULT 0]
- -tomo: Use this option for tomography model. Since tomography model uses negative identifiers, this option will write negative block IDs. Use -tomo=1 to make negative block IDs or -tomo=0 not to make [DEFAULT 0]

Issues:

- - This does not work with older verion of CUBIT. For the older version use exodusold2specfem3d.↩ c.

Basic steps starting from the TRELIS:

step 1: prepare mesh in TRELIS/CUBIT

- Define material regions using "Blocks"

  For example:
  block 1 add volume 1
  block 2 add volume 2 3

  will assign material region 1 to volume 1 and material region 2 to volumes 2 and 3. These material regions will be used to define material properties in "nummaterial_↩ velocity_file". This program will NOT generate "nummaterial_velocity_file". The file "nummaerial_veolicty_file" must be created to run SPECFEM3D!

- Define surface boundary conditions using "Sidesets"

  For example:
  sideset 1 add surface 1
  sideset 1 name 'free_or_absorbing_surface_file_zmax'

  will define a free or absorbing surface boundary condition on surface 1 which lies at the top of the volume (zmax). similary,

  sideset 2 add surface 3
  sideset 2 name 'absorbing_surface_file_bottom'

  will define absorbing boundary condition on the surface 3 which lies at the bottom of the volume (zmin). Note: All the above commands can also be executed using TRELI↩ S/CUBIT GUI. "sideset 1 name 'free_or_absorbing_surface_file_zmax'" is equivalent to clicking sideset 1 and renaming.

step2: export mesh file as exodus file say "tunnel.e" (use 3D option)

step3: convert "tunnel.e" to SPECFEM3D files exodus2specfem3d tunnel.e -bin=1
There will be several output files:

- nodes_coords_file : coordinates file => total number of nodes followed by nodal coordinate ? (? -> x, y, z)
- mesh_file : element file => total number of elements followed by connectivity list
- materials_file : material file => total number of elements followed by material IDs
- surface_file* : sourface boundary condition files => total number of elements followed by element ID and surface nodes

4.6 exodusold2semgeotech.c File Reference

Converts old ASCII exodus file to SPECFEM3D_GEOTECH files.

Functions

- void **removeExtension** (char *, char *)
- int **get_int** (int *, char *, char *)
- int **look_int** (int *, char *, char *)
- int **getfirstquote** (char *, char *)
- int **main** (int argc, char **argv)

4.6.1 Detailed Description

Converts old ASCII exodus file to SPECFEM3D_GEOTECH files.
This program converts the Binary (provided that "ncdump" command exists) or ASCII exodus file exported from the old CUBIT to several mesh files required by the SPECFEM3D_GE↩ OTECH package. The "ncdump" commad is a part of NetCFD library which is generally installed already in LINUX, which can be downloaded for free from
https://www.unidata.ucar.edu/software/netcdf/

Dependencies:

stringmanip.c: string manipulation routines

Compile:

gcc exodusold2semgeotech.c -o exodusold2semgeotech

Usage:

exodusold2semgeotech *input_file* [*Options*]

Example:
exodusold2semgeotech tunnel.e -bin=1
or
exodusold2semgeotech tunnel.txt

Options:

- -fac: Use this option to multiply coordinates by some factor. This is important for unit conversion, e.g., to convert m to km use -fac=0.001 [DEFAULT 1]
- -bin: Use this option if you want to convert exodus binary directly, provided that the command "ncdump" is in the path. The command "ncdump" is a part of netCDF library that can be downloaded for free from
  http://www.unidata.ucar.edu/downloads/netcdf/index.jsp. Use -bin=1 for binary or -bin=0 for ascii file. [DEFAULT 0]

Issues:

- - This does not work with older verion of CUBIT. For the older version use exodusold2semgeotech.↩ c.

Basic steps starting from the CUBIT:

step 1: prepare mesh in CUBIT

- Define material regions using "Blocks"

  For example:
  block 1 add volume 1
  block 2 add volume 2 3

  will assign material region 1 to volume 1 and material region 2 to volumes 2 and 3. These material regions will be used to define material properties in "nummaterial_↩ velocity_file". This program will NOT generate "nummaterial_velocity_file". The file "nummaerial_veolicty_file" must be created to run SPECFEM3D!

- Define surface boundary conditions using "Nodesets" or "Sidesets" – nodal boundary conditions must be defined using node set -> each node set name must contain the corresponding BC names as defined in char ∗ns_bcname[] below
  e.g., node set name can be front_nsbcux or front_nsbcux_nsbcuy etc. – surface boundary conditions must be defined using side set -> each side set name must contain the corresponding BC names as defined in char ∗ss_bcname[] below
  e.g., side set name can be front_ssbcux or front_ssbcux_ssbcuy etc.

  For example:
  sideset 1 add surface 1
  sideset 1 name 'bottom_ssbcux_ssbcuy_ssbcuz'

  will define a surface in which all displacement components are prescribed.

  Note: All the above commands can also be executed using TRELIS/CUBIT GUI. "sideset 1 name 'bottom_ssbcux_ssbcuy_ssbcuz'" is equivalent to clicking 'sideset 1' and renaming.

step2: export mesh file as exodus file say "tunnel.e" (use 3D option)

step3: convert "tunnel.e" to SPECFEM3D files exodusold2semgeotech tunnel.e -bin=1
There will be several output files:

- ∗_coord_? : coordinates file => total number of nodes followed by nodal coordinate ? (? -> x, y, z)
- ∗_connectivity : element file => total number of elements followed by connectivity list
- ∗_material_id : material file => total number of elements followed by material IDs
- ∗_??bcu? : node IDs which have u? defined as the boundary conditions (?? -> ns or ss, ? -> x, y, z). Total number of entities (nodes or faces) followed by element ID and surface nodes.

4.7 exodusold2specfem3d.c File Reference

Converts old ASCII exodus file to SPECFEM3D files.

Functions

- void **removeExtension** (char *, char *)
- int **get_int** (int *, char *, char *)
- int **look_int** (int *, char *, char *)
- int **getfirstquote** (char *, char *)
- int **check_normal** (double [3][4], double [3])
- int **main** (int argc, char **argv)

4.7.1 Detailed Description

Converts old ASCII exodus file to SPECFEM3D files.
This program converts the Binary (provided that "ncdump" command exists) or ASCII exodus file exported from the old CUBIT to several mesh files required by the SPECFEM3D package.

Dependencies:

stringmanip.c: string manipulation routines

Compile:

gcc exodusold2specfem3d.c -o exodusold2specfem3d

Usage:

exodusold2specfem3d *input_file* [*Options*]

Example: exodusold2specfem3d tunnel.txt
or
exodusold2specfem3d tunnel.e -fac=0.001 -bin=1

Options:

- -fac: use this option to multiply coordinates. this is important for unit conversion, e.g., to convert m to km use -fac=0.001
- -bin: use this option if you want to convert exodus binary directly, provided that the command "ncdump" is in the path. The command "ncdump" is a part of netCDF library that can be downloaded for free from http://www.unidata.ucar.edu/downloads/netcdf/index.jsp. use -bin=1 for binary or -bin=0 for ascii file.
- -norm: use this option to check the normal of the faces. use -norm=1 for checking or -norm=0 (default) for no checking

Issues:

- - This does not work with new verion of Trelis/CUBIT. For the new version use exodus2specfem3d.c.

Basic steps starting from the CUBIT:

step 1: prepare mesh in CUBIT

- define material regions using "Blocks"

  For example:
  block 1 add volume 1
  block 2 add volume 2 3

  will assign material region 1 to volume 1 and material region 2 to volumes 2 and 3. These material regions will be used to define material properties in "nummaterial_velocity_file". this program will NOT generate "nummaterial_velocity_file". the file "nummaerial_veolicty_file" must be created to run SPECFEM3D!

define surface boundary conditions using "Sidesets"
For example:
sideset 1 add surface 1
sideset 1 name 'free_or_absorbing_surface_file_zmax'
will define a free or absorbing surface boundary condition on surface 1 which lies at the top of the volume (zmax). similary,
sideset 2 add surface 3
sideset 2 name 'absorbing_surface_file_bottom'
will define absorbing boundary condition on the surface 3 which lies at the bottom of the volume (zmin). Note: All the above commands can also be executed using TRELIS/CUBIT GUI. "sideset 1 name 'free_or_absorbing_surface_file_zmax'" is equivalent to clicking sideset 1 and renaming.

step2: export mesh file as exodus file say "tunnel.e" (use 3D option)

step3: convert "tunnel.e" to SPECFEM3D files exodusold2specfem3d tunnel.e -bin=1
There will be several output files:

- nodes_coords_file : coordinates file => total number of nodes followed by nodal coordinate ? (? -> x, y, z)
- mesh_file : element file => total number of elements followed by connectivity list
- materials_file : material file => total number of elements followed by material IDs
- surface_file* : sourface boundary condition files => total number of elements followed by element ID and surface nodes

4.8 gid2semgeotech.c File Reference

Converts ASCII Gid mesh file to SPECFEM3D_GEOTECH files.

Functions

- void **removeExtension** (char *, char *)
- int **get_int** (int *, char *, char *)
- int **look_int** (int *, char *, char *)
- int **getfirstquote** (char *, char *)
- int **main** (int argc, char **argv)

4.8.1 Detailed Description

Converts ASCII Gid mesh file to SPECFEM3D_GEOTECH files.
This program converts the ASCII GiD mesh file to several mesh files required by the SPECFE↩ M3D_GEOTECH package. GiD (www.gidhome.com) is a commercial pre and post processor for numerical simulations.

Dependencies:

stringmanip.c: string manipulation routines

Compile:

gcc gid2semgeotech.c -o gid2semgeotech

Usage:

gid2semgeotech *input_file* [*Options*]

Example:
gid2semgeotech gid2semgeotech_example.dat
or
gid2semgeotech gid2semgeotech_example.dat -fac=0.001

Options:

- -fac: Use this option to multiply coordinates with a certain factor. This is useful for unit conversion, e.g., to convert m to km use: -fac=0.001

Basic steps starting from GID:

step1: Export mesh file in ASCII format "mesh.dat"

step2: Produce mesh and BC files gid2semgeotech mesh.dat
OR
gid2semgeotech mesh.dat 1000.0
There will be several output files:

- coord_? : Total number of nodes followed by nodal coordinate ? (? -> x,y,z)
- _connectivity : Total number of elements followed by connectivity list
- _material_id : Total number of elements followed by material IDs
- ??bcu? : node IDs which have u? = 0 as the boundary conditions (?? -> ns or ss, ? -> x,y,z)

4.9 gocad2vtu.c File Reference

Converts GOCAD ASCII file to VTU file.

Functions

- int **main** (int argc, char **argv)

4.9.1 Detailed Description

Converts GOCAD ASCII file to VTU file.
This program converts the GOCAD ASCII file (3-noded triangular meshes) to VTK XML .vtu binary file (unstructured mesh file) which can be visualized/processed in ParaView or VTK.

Dependencies:

stringmanip.c

Compile

- in parent folder, type: make OR
- in src/ folder, type gcc gocad2vtu.c -o gocad2vtu

Usage:

./bin/gocad2vtu *input_file* [*Options*]

Example: ./bin/gocad2vtu ./input/gocad2vtu_example.ts

Options:

- -fac: Use this option to multiply the coordinates by a certain factor, this is helpful for unit conversion, e.g. for m to km use 0.001, for km to m use 1000, example: gocad2vtu T2_horizon.ts -fac=0.001

Notes:

- Output .vtu file is binary, therefore endianness of the processor architechture is important.
- This program automatically identify the endianness and write the output accordingly. Hence if you run and process/visualize .vtu file in the architecture with different endianness there may be an error.

4.10 vti2cell.c File Reference

This file converts VTI file to VTU file.

Functions

- int **comp_float** (const void *a, const void *b)
- int **main** (int argc, char **argv)

4.10.1 Detailed Description

This file converts VTI file to VTU file.
This program converts the 2D/3D Binary VTK XML .vti file to unstructured mesh files (.vtu). This program also generates the mesh files required by SPECFEM2D and SPECFEM3D. Note that the file formats in SPECFEM2D and SPECFEM3D are different. This should be made same format as soon as possible. For this, source codes within the decompose folder of SPECFEM3D and cubit2specfem3d.py need to be changed.

Dependencies:

stringmanip.c

Compile:

gcc vti2cell.c -o vti2cell -lm

Usage:

vti2cell *input_file* [*Options*]

Example:
vti2cell py_plane_model.vti

Options:

- -fac=factor (real) Use this option to multiply the coordinates by a certain factor, this is helpful for unit conversion, e.g. for m to km use 0.001, for km to m use 1000 Example: vti2cell2d py_plane_model.vti -fac=1000
- -xmat=exclusion material id/s (integer/s) Use this option to exclude certain region of the model, e.g. exclusion of air. Appropriate id/s should be supplied, id s are number orderd according to the value of corresponding material properties and numbered starting from 1. This way, lowest value will have id 1 and so on. Example: vti2cell py_plane↩ _model.vti -xmat=1,2 This command will exclude the regions with material id 1 and 2.

  Example: vti2cell py_plane_model.vti -fac=1000 -xmat=1 This command multiply the coordinates by 1000 and exclude the region with material id 1

- -step=step size (integer) Use this option to coarsen the mesh. This value represent the number of grids to be used as 1 element, e.g., if you want to make 2 grids as 1 element, use -step=2
- -zup=z axis direction indicator (integer) Use this option to indicate whether the Z axis direction is up

Toto:

- make uniformity for 2D,3D, e.g., writing and reading coordinates

4.11 vtk1d2jou.f90 File Reference

Converts VTK 1D file to CUBIT/Trelis journal file.

Functions/Subroutines

- program **vtk1d2jou**

4.11.1 Detailed Description

Converts VTK 1D file to CUBIT/Trelis journal file.
This program reads ASCII vtk files with unstructured grid of lines and points only, and removes the redundant lines. The redundant nodes can be removed within the paraview itself using the 'Clean to Grid' filter.

Compile:

gfortran vtk1d2jou.f90 -o vtk1d2jou

Usage:

vtk1d2jou *input_file*

vtk1d2jou vtk1d2jou_example.vtk

4.12 vtk2d2jou.c File Reference

Converts VTK file consisting of 2D mesh to CUBIT/Trelis journal file.

Functions

- void **removeExtension** (char *, char *)
- int **main** (int argc, char **argv)

4.12.1 Detailed Description

Converts VTK file consisting of 2D mesh to CUBIT/Trelis journal file.
This program converts an ASCII VTK file consisting of triangular/quadrilateral mesh into a CUBIT/Trelis Journal file.

Dependences:

stringmanip.c: string manipulation routines

Compile:

gcc vtk2d2jou.c -o vtk2d2jou

Usage:

vtk2d2jou *input_file*

Example:
vtk2d2jou vtk2d2jou_example.vtk

4.13 write_vti.m File Reference

Writes 3D gridded data to VTK VTI file.

4.13.1 Detailed Description

Writes 3D gridded data to VTK VTI file.
This function writes the VTI binary file for structured grid data, such as finite difference data and tomography data. The VTI file can be visualized in ParaView (http://www.paraview.↩ org/).

Input:

fname : output file name
ox : origin vector [ox oy oz]
dh : sampling interval vector [dx dy dz]
nx : grid number vector [nx ny nz]
name : output variable name

Usage:

call this functions with appropriate variables.

Notes:

- For a BigEndian architecture, replace "LittleEndian" with "BigEndian" in the line below.

4.14 xyz2jou.f90 File Reference

Coverts UTM/XYZ file to CUBIT/Trelis journal file.

Functions/Subroutines

- program **xyz2jou**

4.14.1 Detailed Description

Coverts UTM/XYZ file to CUBIT/Trelis journal file.
This program converts a UTM or XYZ file to a CUBIT journal file. The UTM or XYZ file contains the three columns of X, Y, and Z coordinates, respectively.

Compile:

- in parent folder, type: make OR
- in src/ folder, type gfortran xyz2jou.f90 -o xyz2jou

Usage:

./bin/xyz2jou *input_file* [*Options*]

Example:
./bin/xyz2jou ./input/xyz2jou_example.utm

Options:

- -nx: Use this option if you know the number of points in a line along X axis . This will speed up the processing. For example, -nx=100. If it is not defined, nx is automatically determined.
- -nskip: Use this option if you want to skip (downsample) certain number of successive points. This will skip along both X and Y axes. For example, -nskip=2. [DEFAULT 0].

Bibliography