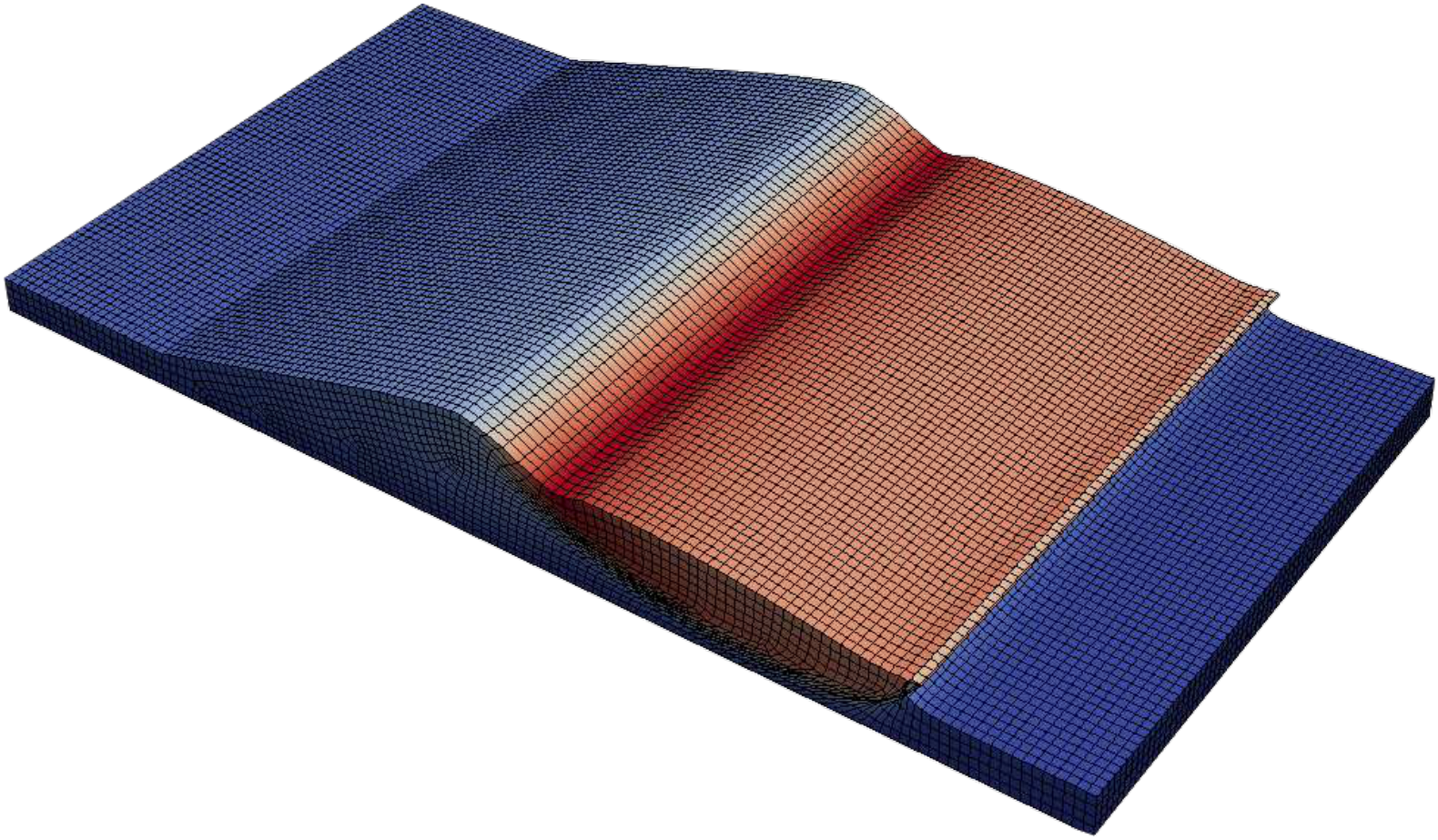
SPECFEM 3D Geotech

v1.2



An open-source, parallel and multi-platform geotechnical engineering application

SPECFEM3D_GEOTECH 1.2
User Manual

Hom Nath Gharti[1], Princeton University, USA
Dimitri Komatitsch, CNRS/University of Aix-Marseille, France
Leah Langer, Princeton University, USA
Roland Martin, University of Toulouse, France
Volker Oye, NORSAR, Norway
Jeroen Tromp, Princeton University, USA
Uno Vaaland, Princeton University, USA
Zhenzhen Yan, Institute of Remote Sensing and Digital Earth, CAS, China

June 19, 2017

[1]Previously at: NORSAR, Norway

Licensing

SPECFEM3D_GEOTECH 1.2 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

SPECFEM3D_GEOTECH 1.2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with SPECFEM3D_GEOTECH 1.2. If not, see <`http://www.gnu.org/licenses/`>.

Acknowledgments

This work was funded in part by the Research Council of Norway, and supported by industry partners BP, Statoil, and Total. Some of the routines were imported and modified from the "Programming the finite element method" (Smith and Griffiths, 2004) and the original "SPECFEM3D" package (e.g., Komatitsch and Vilotte, 1998; Komatitsch and Tromp, 1999; Peter et al., 2011).

Contents

Chapter 1

Introduction

1.1 Background

SPECFEM3D_GEOTECH is a free and open-source command-driven software for 3D slope stability analysis (for more details see Gharti et al., 2012) and simulation of 3D multistage excavation (for more details see Gharti et al., 2012) based on the spectral-element method (e.g., Patera, 1984; Canuto et al., 1988; Seriani, 1994; Faccioli et al., 1997; Komatitsch and Vilotte, 1998; Komatitsch and Tromp, 1999; Peter et al., 2011). The slope stability and the excavation routines were originally started from the routines found in the book "Programming the finite element method" (Smith and Griffiths, 2004). The software can run on a single processor as well as multi-core machines or large clusters. It is written mainly in FORTRAN 90, and parallelized using MPI (Gropp et al., 1994; Pacheco, 1997) based on domain decomposition. For the domain decomposition, the open-source graph partitioning library SCOTCH (Pellegrini and Roman, 1996) is used. The element-by-element preconditioned conjugate-gradient method (e.g., Hughes et al., 1983; Law, 1986; King and Sonnad, 1987; Barragy and Carey, 1988) is implemented to solve the linear equations. For elastoplastic failure, a Mohr-coulomb failure criterion is used with a viscoplastic strain method (Zienkiewicz and Cormeau, 1974).

This program does not automatically determine the factor of safety of slope stability. Simulations can be performed for a series of safety factors. After plotting the safety factor verses maximum displacement curve, one can determine the factor of safety of the given slope. Although the software is optimized for slope stability analysis and multistage excavation, other relevant simulations of quasistatic problems in solid (geo)mechanics can also be performed with this software.

The software currently does not include an inbuilt mesher. Existing tools, such as Gmsh (Geuzaine and Remacle, 2009), CUBIT/Trelis (CUBIT, 2011), TrueGrid (Rainsberger, 2006), etc., can be used for hexahedral meshing, and the resulting mesh file can be converted to the input files required by SPECFEM3D_GEOTECH. Output data can be visualized and processed using the open-source visualization application ParaView (`www.paraview.org`).

1.2 Status summary

Slope stability analysis	: Yes
Multistage excavation	: Yes
Gravity loading	: Yes
Surface loading	: Yes (point load, uniformly distributed load, linearly distributed load) [Experimental]
Water table	: Yes [Experimental]
Pseudo-static earthquake loading	: Yes [Experimental]
Automatic factor of safety	: No

Last revision

June 19, 2017

Chapter 2

Getting started

2.1 Package structure

The original SPECFEM3D_GEOTECH package comes in a single compressed file `SPECFEM3D_GEOTECH.tar.gz`, which can be extracted using `tar` command:

```
tar -zxvf SPECFEM3D_GEOTECH.tar.gz
```

or using, for example, `7-zip (www.7-zip.org)` under WINDOWS. The package has the following structure:

`SPECFEM3D_GEOTECH/`

`COPYING`	: License.
`README`	: brief description of the package.
`CMakeLists.txt`	: CMake configuration file.
`bin/`	: all object files and executables are stored in this folder.
`doc/`	: documentation files for the SPECFEM3D_GEOTECH package. If built this file is created.
`input/`	: contains input files.
`partition/`	: contains partition files for parallel processing.
`output/`	: default output folder. All output files are stored in this folder unless the different output path is defined in the main input file.
`src/`	: contains all source files.
`util/`	: contains several utilities source files.

2.2 Prerequisites

- CMake build system. The CMake version $\geq$ 2.8.4 is necessary to configure the software. It is free and open-source, and can be downloaded from `www.cmake.org`. In order to check if the CMake is already installed, type
  `cmake --version`

- Make utility. The make utility is necessary to build the software using Makefile. This utility is usually installed by default in most LINUX systems. Under WINDOWS, one can use Cygwin (`www.cygwin.com`) or MinGW (`www.mingw.org`) to install the make utility. In order to check if the CMake is already installed, type
  `make --version`

- A recent FORTRAN compiler. The software is written mainly in FORTRAN 90, but it also uses a few FORTRAN 2003 features (e.g., streaming IO). These features are already available in most of the FORTRAN compilers, e.g., gfortran version $\geq$ 4.2 (`gcc.gnu.org/wiki/GFortran`) and g95 (`www.g95.org`).

Following libraries are necessary for parallel processing.

- A recent MPI library. It should be built with the same FORTRAN compiler used to compile the software. Please see `www.open-mpi.org` or `www.mcs.anl.gov/research/projects/mpich2` for details on how to install MPI library and how to run MPI programs.

- SCOTCH graph partitioning library. This library should be compiled with the same FORTRAN compiler used to compile the software. Please see `www.labri.fr/perso/pelegrin/scotch` for details on how to install SCOTCH.

Finally, the following compiler is necessary to build the documentation (this file):

- LaTeX compiler. This is necessary to compile the documentation files.

2.3 Configure

Software package SPECFEM3D_GEOTECH is configured using CMake, and the package uses an out-of-source build. Hence, DO NOT build in the same source directory. Let's say the full path to the package (source directory) is `$HOME/source/SPECFEM3D_GEOTECH`.

- Create a separate build directory, e.g.,
  `mkdir $HOME/build/SPECFEM3D_GEOTECH`

- Go to the build directory
  `cd $HOME/build/SPECFEM3D_GEOTECH`

- CMake configuration
  You can set the necessary compilers explicitly, and configure the package with the default options by typing, for example,
  `CC=gcc CXX=g++ FC=gfortran MPIFC=mpif90 ccmake $HOME/source/SPECFEM3D_GEOTECH`
  for GNU compilers.
  OR

`CC=icc CXX=icpc FC=ifort MPIFC=mpif90 ccmake $HOME/source/SPECFEM3D_GEOTECH` for Intel compilers.

For more options and flexibility, we recommend using CMake GUI. Type: `ccmake $HOME/source/SPECFEM3D_GEOTECH`

This will open the CMake GUI (Figure 2.1), in which you can select several options as described below.

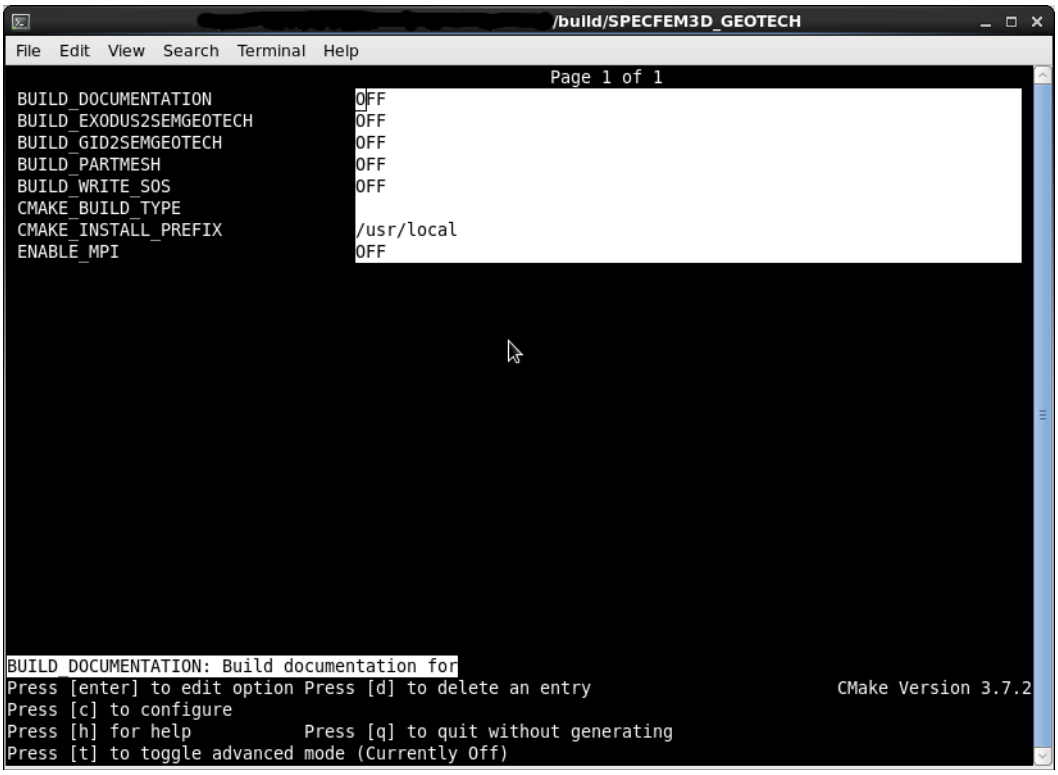


Figure 2.1: CMake configuration of SPECFEM3D_GEOTECH .

CMake configuration is an iterative process (See Figure 2.1):

- Configure ('c' key or 'Configure' button)
- Change variables' values if necessary
- Generate ('g' key or 'Generate' button. This key or button appears once the configurtion is successful.)

If WARNINGS or ERRORS occur, press the 'e' key (or the 'OK' button) to return to configuration. These steps have to be repeated until successful configuration. Then, press the 'g' key (or the 'Generate' button) to generate build files. Check carefully that all necessary variables are set properly. Unless configuration is successful, 'g' key or 'Generate' button is not enabled. Sometimes, the 'c' key or 'Configure' button has to be pressed repeatedly until 'g' key or 'Generate' is enabled. Initially, all variables may not be visible. To see all variables, toggle advanced mode by pressing the 't' key (or the Advanced button). To set or change a variable, move the cursor to the variable and press 'Enter' key. If the variable is a boolean (ON/OFF), it will flip the value on pressing the 'Enter'

key. If the variable is a string or a file, it can be edited. For more details, please see the CMake documentation (`www.cmake.org`).

Following are the main CMake variables for the SPECFEM3D_GEOTECH (See Figure 2.1)

`BUILD_DOCUMENTATION`	: If `ON`, the user manual (this file) is created. The default is `OFF`.
`BUILD_PARTMESH`	: If `ON`, the `partmesh` program is built. The default is `OFF`. The `partmesh` program is necessary to partition the mesh for parallel processing.
`BUILD_EXODUS2SEMGEOTECH`	: If `ON`, the `exodus2semgeotech` program is built. The default is `OFF`. The `exodus2semgeotech` program convert exodus mesh file to input files required by the SPECFEM3D_GEOTECH package (see also Chapter 5).
`BUILD_GID2SEMGEOTECH`	: If `ON`, the `gid2semgeotech` program is built. The default is `OFF`. The `gid2semgeotech` program convert GiD mesh file to input files required by the SPECFEM3D_GEOTECH package (see also Chapter 5).
`BUILD_WRITE_SOS`	: If `ON`, the `write_sos` program is built. The default is `OFF`. The `write_sos` program writes a EnSight SOS file necessary for the parallel visualization (see also Chapter 5).
`ENABLE_MPI`	: If `ON`, the main parallel program `psemgeotech` is built otherwise main serial program `semgeotech` is built. The default is `OFF`.
`SCOTCH_LIBRARY_PATH`	: This is required if `BUILD_PARTMESH` is `ON`. If not found automatically, it can be set manually.
`CMAKE_Fortran_COMPILER`	: This defines the Fortran compiler. If not found automatically or the automatically found compiler is not correct, it can be set manually.
`MPI_Fortran_COMPILER`	: This defines the MPI Fortran compiler. This is required if `ENABLE_MPI` is `ON`. If not found automatically or the automatically found compiler is not correct, it can be set manually.

Note 1: If `CMAKE_Fortran_COMPILER` *has to be changed, first change this and configure, and then change other variables if necessary and configure.*
Note 2: Even if some of the above variables are set `ON`*, if appropriate working compilers are not found, corresponding variables are internally set* `OFF` *with WARNING messages.*

2.4 Compile

Once configuration and generation are successful, the necessary build files are created. Now to build the main program, type:
`make`

On multi-processor systems (let's say eight processors), type:
`make -j 8`

To clean, type
`make clean`

Note: If reconfiguration is necessary, it is better to delete all Cache files of the build directory.

2.5 Run

Serial run

- To run the serial program, type
  `./bin/semgeotech` *input_file_name*

  Example:

  `./bin/semgeotech ./input/validation1.sem`

Parallel run

- To partition the mesh, type
  `./bin/partmesh` *input_file_name*

  Example:

  `./bin/partmesh ./input/validation1.psem`

- To run the parallel program, type
  `mpirun -n` *number_of_nodes* `./bin/psemgeotech` *input_file_name*

  OR

  `mpirun -n` *number_of_nodes* `--hostfile` *host_file* `./bin/psemgeotech` *input_file_name*

  Example:

  `mpirun -n 8 ./bin/psemgeotech ./input/validation1.psem`

Note: see Chapter 3 for details on input and input files. Try to run one or more examples included in `input/`*. By default, example files included in the package are not copied to build directory during build process. If necessary, copy files within* `input/` *folder of source directory to the* `input/` *folder of build directory.*

Chapter 3

Input

3.1 Main input file

The main input file structure is motivated by the "E3D" (Larsen and Schultz, 1995) software package. The main input file consists of legitimate input lines defined in the specified formats. Any number of blank lines or comment lines can be placed for user friendly input structure. The blank lines contain no or only white-space characters, and the comment lines contain "#" as the first character.

Each legitimate input line consists of a line type, and list of arguments and corresponding values. All argument-value pair are separated by comma (,). If necessary, any legitimate input line can be continued to next line using FORTRAN 90 continuation character "&" as an absolute last character of a line to be continued. Repetition of same line type is not allowed.

Legitimate input lines have the format
line_type $arg_1 = val_1$, $arg_2 = val_2$, ..., $arg_n = val_n$

Example:
```
preinfo:  nproc=8, ngllx=3, nglly=3, ngllz=3, nenod=8, ngnod=8, &
inp_path='./input', part_path='./partition', out_path='./output/'
```

All legitimate input lines should be written in lower case. Line type and argument-value pairs must be separated by a space. Each argument-value pair must be separated by a comma(,) and a space/s. No space/s are recommended before line type and in between argument name and "=" or "=" and argument value. If argument value is a string, the FORTRAN 90 string (i.e., enclosed within single quotes) should be used, for example, `inp_path='./input'`. If the argument value is a vector (i.e., multi-valued), a list of values separated by space (no comma!) shoud be used, e.g, `srf=1.0 1.2 1.3 1.4`.

3.1.1 Line types

Only the following line types are permitted.

`preinfo:` preliminary information of the simulation

`mesh:` mesh information

`bc:` boundary conditions information

`traction:` traction information [optional]

`stress0:` initial stress information [optional]. It is generally necessary for multistage excavation.

`material:` material properties

`eqload:` pseudo-static earthquake loading [optional]

`water:` water table information [optional]

`control:` control of the simulation

`save:` options to save data

3.1.2 Arguments

Only the following arguments under the specified line types are permitted.

`preinfo:`

`nproc` : number of processors to be used for the parallel processing [integer > 1]. Only required for parallel processing.

`ngllx` : number of Gauss-Lobatto-Legendre (GLL) points along x-axis [integer > 1].

`nglly` : number of GLL points along y-axis [integer > 1].

`ngllz` : number of GLL points along z-axis [integer > 1].

Note: Although the program can use different values of `ngllx`, `nglly`, *and* `ngllz`, *it is recommended to use same number of GLL points along all axes.*

`inp_path` : input path where the input data are located [string, optional, default $\Rightarrow$ `'./input'`].

`part_path` : partition path where the partitioned data will be or are located [string, optional, default $\Rightarrow$ `'./partition'`]. Only required for parallel processing.

`out_path` : output path where the output data will be stored [string, optional, default $\Rightarrow$ `'./output'`].

`mesh:`

`xfile` : file name of x-coordinates [string].

`yfile` : file name of y-coordinates [string].

`zfile` : file name of z-coordinates [string].

`confile` : file name of mesh connectivity [string].

`idfile` : file name of element IDs [string].

`gfile` : file name of ghost interfaces, i.e., partition interfaces [string]. Only required for parallel processing.

`bc:`

`uxfile` : file name of displacement boundary conditions along x-axis [string].

`uyfile` : file name of displacement boundary conditions along y-axis [string].

`uzfile` : file name of displacement boundary conditions along z-axis [string].

`traction:`

`trfile` : file name of traction specification [string].

`stress0:`

`type` : type of initial stress [integer, optional, 0 = compute using SEM itself, 1 = compute using simple vertical lithostatic relation, default $\Rightarrow$ 0].

`z0` : datum (free surface) coordinate [real, m]. Only required if `type`=1.

`s0` : datum (free surface) vertical stress [real, kN/m^2]. Only required if `type`=1.

`k0` : lateral earth pressure coefficient [real].

`material:`

`matfile` : file name of material list [string].

`ispart` : flag to indicate whether the material file is partitioned [integer, optional, 0 = No, 1 = Yes, default $\Rightarrow$ 1]. Only required for parallel processing.

`matpath` : path to material file [string, optional, default $\Rightarrow$ `'./input'` for serial or unpartitioned material file in parallel and `'./partition'` for partitioned material file in parallel].

`allelastic` : assume all entire domain as elastic [integer, optional, 0 = No, 1 = Yes, default $\Rightarrow$ 0].

`eqload:`

`eqkx` : pseudo-static earthquake loading coefficient along x-axis [real, 0 <= `eqkx` <= 1.0, default $\Rightarrow$ 0.0].

`eqky` : pseudo-static earthquake loading coefficient along y-axis [real, 0 <= `eqky` <= 1.0, default $\Rightarrow$ 0.0].

`eqkz` : pseudo-static earthquake loading coefficient along z-axis [real, 0 <= `eqkz` <= 1.0, default $\Rightarrow$ 0.0].

Note: For the stability analysis purpose, these coefficients should be chosen carefully. For example, if the slope face is pointing towards the negative x-axis, value of `eqkx` *is taken negative.*

`water:`

`wsfile` : file name of water surface file.

`control:`

`cg_tol` : tolerance for conjugate gradient method [real].

`cg_maxiter` : maximum iterations for conjugate gradient method [integer $>$ 0].

`nl_tol` : tolerance for nonlinear iterations [real].

`nl_maxiter` : maximum iterations for nonlinear iterations [integer $>$ 0].

`ninc` : number of load increments for the plastic iterations [integer$>$0 default $\Rightarrow$ 1].This is currently not used for slope stability analysis.

`Arguments specific to slope stability analysis:`

`nsrf` : number of strength reduction factors to try [integer $>$ 0, optional, default $\Rightarrow$ 1].

`srf` : values of strength reduction factors [real vector, optional, default $\Rightarrow$ 1.0]. Number of `srf`s must be equal to `nsrf`.

`phinu` : force $\phi - \nu$ (Friction angle - Poisson's ratio) inequality: $\sin\phi \geq 1 - 2\,\nu$ (see Zheng et al., 2005) [integer, 0 = No, 1 = Yes, default $\Rightarrow$ 0]. Only for TESTING purpose.

`Arguments specific to multistage excavation:`

`nexcav` : number of excavation stages [integer $>$ 0, optional, default $\Rightarrow$ 1].

`nexcavid` : number of excavation IDs in each excavation stage [integer vector, default $\Rightarrow$ 1].

`excavid` : IDs of blocks/regions in the mesh to be excavated in each stage [integer vector, default $\Rightarrow$ 1].

Note: Do not mix arguments for slope stability and excavation.

`save:`

`disp` : displacement field [integer, optional, 0 = No, 1 = Yes, default $\Rightarrow$ 0].

`porep` : pore water pressure [integer, optional, 0 = No, 1 = Yes, default $\Rightarrow$ 0].

3.1.3 Examples of main input file

Input file for a simple elastic simulation

```
#---------------------------------
#input file elastic.sem
#pre information
preinfo:  ngllx=3, nglly=3, ngllz=3, nenod=8, ngnod=8, &
inp_path='./input', out_path='./output/'

#mesh information
mesh:  xfile='validation1_coord_x', yfile='validation1_coord_y', &
zfile='validation1_coord_z', confile='validation1_connectivity', &
idfile='validation1_material_id'

#boundary conditions
bc:  uxfile='validation1_ssbcux', uyfile='validation1_ssbcuy', &
uzfile='validation1_ssbcuz'

#material list
material:  matfile='validation1_material_list', allelastic=1

#control parameters
control:  cg_tol=1e-8, cg_maxiter=5000
#---------------------------------
```

Serial input file for slope stability

```
#--------------------------------
#input file validation1.sem
#pre information
preinfo:  ngllx=3, nglly=3, ngllz=3, nenod=8, ngnod=8, &
inp_path='./input', out_path='./output/'

#mesh information
mesh:  xfile='validation1_coord_x', yfile='validation1_coord_y', &
zfile='validation1_coord_z', confile='validation1_connectivity', &
idfile='validation1_material_id'

#boundary conditions
bc:  uxfile='validation1_ssbcux', uyfile='validation1_ssbcuy', &
uzfile='validation1_ssbcuz'

#material list
material:  matfile='validation1_material_list'

#control parameters
control:  cg_tol=1e-8, cg_maxiter=5000, nl_tol=0.0005, nl_maxiter=3000, &
nsrf=9, srf=1.0 1.5 2.0 2.15 2.16 2.17 2.18 2.19 2.20
#--------------------------------
```

Parallel input file for slope stability

```
#--------------------------------
#input file validation1.psem
#pre information
preinfo:  nproc=8, ngllx=3, nglly=3, ngllz=3, nenod=8, &
ngnod=8, inp_path='./input', out_path='./output/'

#mesh information
mesh:  xfile='validation1_coord_x', yfile='validation1_coord_y', &
zfile='validation1_coord_z', confile='validation1_connectivity', &
idfile='validation1_material_id', gfile='validation1_ghost'

#boundary conditions
bc:  uxfile='validation1_ssbcux', uyfile='validation1_ssbcuy', &
uzfile='validation1_ssbcuz'

#material list
material:  matfile='validation1_material_list'

#control parameters
control:  cg_tol=1e-8, cg_maxiter=5000, nl_tol=0.0005, nl_maxiter=3000, &
nsrf=9, srf=1.0 1.5 2.0 2.15 2.16 2.17 2.18 2.19 2.20
#--------------------------------
```

Serial input file for excavation

```
#--------------------------------
#input file excavation_3d.sem
#pre information
preinfo:  ngllx=3, nglly=3, ngllz=3, nenod=8, ngnod=8, &
inp_path='./input', out_path='./output/'

#mesh information
mesh:  xfile='excavation_3d_coord_x', yfile='excavation_3d_coord_y', &
zfile='excavation_3d_coord_z', confile='excavation_3d_connectivity', &
idfile='excavation_3d_material_id'

#boundary conditions
bc:  uxfile='excavation_3d_ssbcux', uyfile='excavation_3d_ssbcuy', &
uzfile='excavation_3d_ssbcuz'

#initial stress stress0:  type=0, z0=0, s0=0, k0=0.5, usek0=1

#material list
material:  matfile='excavation_3d_material_list'

#control parameters
control:  cg_tol=1e-8, cg_maxiter=5000, nl_tol=0.0005, nl_maxiter=3000, &
nexcav=3, excavid=2 3 4, ninc=10
#--------------------------------
```

Parallel input file for excavation

```
#--------------------------------
#input file excavation_3d.psem
#pre information
preinfo:  nproc=8, ngllx=3, nglly=3, ngllz=3, nenod=8, &
ngnod=8, inp_path='./input', out_path='./output/'

#mesh information
mesh:  xfile='excavation_3d_coord_x', yfile='excavation_3d_coord_y', &
zfile='excavation_3d_coord_z', confile='excavation_3d_connectivity', &
idfile='excavation_3d_material_id', gfile='excavation_3d_ghost'

#boundary conditions
bc:  uxfile='excavation_3d_ssbcux', uyfile='excavation_3d_ssbcuy', &
uzfile='excavation_3d_ssbcuz'

#initial stress stress0:  type=0, z0=0, s0=0, k0=0.5, usek0=1

#material list
material:  matfile='excavation_3d_material_list'

#control parameters
control:  cg_tol=1e-8, cg_maxiter=5000, nl_tol=0.0005, nl_maxiter=3000, &
nexcav=3, excavid=2 3 4, ninc=10
#--------------------------------
```

There are only two additional pieces of information, i.e., number of processors `'nproc'` in line `'preinfo'` and file name for ghost partition interfaces `'gfile'` in line `'mesh'` in parallel input file.

3.2 Input files detail

All local element/face/edge/node numbering follows the EXODUS II convention.

3.2.1 Coordinates files: `xfile, yfile, zfile`

Each of the coordinates files contains a list of corresponding coordinates in the following format:

number of points
coordinate of point 1
coordinate of point 2
coordinate of point 3
..
..

..

Example:

```
2354
40.230394465164999
40.759090909090901
42.700000000000003
40.957142857142898
40.230394465164999
40.759090909090901
42.700000000000003
40.957142857142898
...
...
```

3.2.2 Connectivity file: `confile`

The connectivity file contains the connectivity lists of elements in the following format:

number of elements
n_1 n_2 n_3 n_4 n_5 n_6 n_7 n_8 *of element 1*
n_1 n_2 n_3 n_4 n_5 n_6 n_7 n_8 *of element 2*
n_1 n_2 n_3 n_4 n_5 n_6 n_7 n_8 *of element 3*
n_1 n_2 n_3 n_4 n_5 n_6 n_7 n_8 *of element 4*
..
..

Example:

1800
```
1 2 3 4 5 6 7 8
9 10 2 1 11 12 6 5
9 1 4 13 11 5 8 14
15 16 10 9 17 18 12 11
15 9 13 19 17 11 14 20
21 22 16 15 23 24 18 17
21 15 19 25 23 17 20 26
27 28 22 21 29 30 24 23
27 21 25 31 29 23 26 32
33 34 28 27 35 36 30 29
33 27 31 37 35 29 32 38
34 33 39 40 36 35 41 42
33 37 43 39 35 38 44 41
...
...
```

3.2.3 Element IDs (or Material IDs) file: `idfile`

This file contains the IDs of elements. This ID will be used in the program mainly to identify the material regions. This file has the following format:

number of elements
ID of element 1
ID of element 2
ID of element 3
ID of element 4
...
...

Example:

```
1800
1
1
1
1
1
1
1
1
1
1
...
...
```

3.2.4 Ghost partition interfaces file: `gfile`

This file will be generated automatically by a program `partmesh`.

3.2.5 Displacement boundary conditions files: `uxfile`, `uyfile`, `uzfile`

This file contains information on the displacement boundary conditions (currently only the zero-displacement is implemented), and has the following format:

BCtype BCvalue
number of element faces
elementID faceID
elementID faceID
elementID faceID
...
...

The *BCtype* represents BC geometry (0: Point, 1: Line, 2: Surface). Currently, only the surface BC is supported; therefore, always use 2. The *BCvalue* represent the BC value. Use only the 0.0. Non-zero BC values will be supported in next version.
Example:

```
2 0.0
849
2 2
3 4
5 1
6 1
7 1
8 1
9 1
...
...
```

3.2.6 Traction file: `trfile`

This file contains the traction information on the model in the following format:

traction type (integer, 0 = point, 1 = uniformly distributed, 2 = linearly distributed)
if *traction type* = 0
q_x q_y q_z (load vector in kN)
if *traction type* = 1
q_x q_y q_z (load vector in kN/m^2)
if *traction type* = 2
relevant-axis x_1 x_2 q_{x1} q_{y1} q_{z1} q_{x2} q_{y2} q_{z2}
number of entities (points for point load or faces for distributed load)
elementID entityID
elementID entityID
elementID entityID
...
...

This can be repeated as many times as there are tractions.

The *relevant-axis* denotes the axis along which the load is varying, and is represented by an integer as 1 = x-axis, 2 = y-axis, and 3 = z-axis. The variables x_1 and x_2 denote the coordinates (only the *relevant-axis*) of two points between which the linearly distributed load is applied. Similarly, q_{x1}, q_{y1} and q_{z1}, and q_{x2}, q_{y2} and q_{z2} denote the load vectors in kN/m^2 at the point 1 and 2, respectively.

Example:
The following data specify the two tractions: a uniformly distributed traction and a linearly distributed traction.

```
1
0.0 0.0 -167.751
363
56 1
57 1
58 1
59 1
60 1
61 1
62 1
...
...
2
3 7.3 24.4 51.8379 0.0 -159.5407 0.0 0.0 0.0
594
38 1
39 1
40 1
41 1
42 1
43 1
44 1
45 1
46 1
...
...
```

3.2.7 Material list file: `matfile`

This file contains material properties of each material regions. Material properties must be listed in a sequential order of the unique material IDs. In addition, this data file optionally contains the information on the water condition of material regions. Material regions or material IDs must be consistent with the Material IDs (Element IDs) defined in `idfile`. The `matfile` has the following format:

comment line
number of material regions (unique material IDs)
materialID, domainID, blockType, γ*,* E*,* ν*,* ϕ*,* c*,* ψ
materialID, domainID, blockType, γ*,* E*,* ν*,* ϕ*,* c*,* ψ
materialID, domainID, blockType, γ*,* E*,* ν*,* ϕ*,* c*,* ψ
...
...
number of submerged material regions
submerged materialID

submerged materialID
...
...

The *materilID* must be in a sequential order starting from 1. The *doaminID* represents the material domain (e.g., elastic = 1 or viscoelastic = 11), and currently only the elastic domain is supported, therefore, always use 1. The *blockType* represents the type of material properties defined for a particular block/region (0: properties defined as a single block, 1: properties defined on the structured grid). Currently, only the single block definition is supported; therefore always use 0. Similarly, γ represents the unit weight in kN/m^3, E the Young's modulus of elasticity in kN/m^2, ϕ the angle of internal friction in degrees, c the cohesion in kN/m^2, and ψ the angle of dilation in degrees.

Example:

The following data defines four material regions. No region is submerged in water.

```
# material properties (id, domain, type, gamma, ym, nu, phi, coh, psi)
4
1 1 0 18.8 1e5 0.3 20.0 29.0 0.0
2 1 0 19.0 1e5 0.3 20.0 27.0 0.0
3 1 0 18.1 1e5 0.3 20.0 20.0 0.0
4 1 0 18.5 1e5 0.3 20.0 29.0 0.0
```

The following data defines four material regions with two of them submerged.

```
# material properties (id, domain, type, gamma, ym, nu, phi, coh, psi)
4
1 1 0 18.8 1e5 0.3 20.0 0.0 0.0
2 1 0 19.0 1e5 0.3 20.0 27.0 0.0
3 1 0 18.1 1e5 0.3 20.0 0.0 0.0
4 1 0 18.5 1e5 0.3 20.0 29.0 0.0
2
1
3
```

3.2.8 Water surface file: `wsfile`

This file contains the water table information on the model in the format as
number of water surfaces
water surface type (integer, 0 = horizontal surface, 1 = inclined surface, 2 = meshed surface)
if *wstype*=0 (can be reconstructed by sweeping a horizontal line)
relevant-axis x_1 x_2 z
if *wstype*=1 (can be reconstructed by sweeping a inclined line)

relevant-axis x_1 x_2 z_1 z_2
if *wstype*=2 (meshed surface attached to the model)
number of faces
elelemetID, faceID
elelemetID, faceID
elelemetID, faceID
...
...

The *relevant-axis* denotes the axis along which the line is defined, and it is taken as 1 = x-axis, 2 = y-axis, and 3 = z-axis. The variables x_1 and x_2 denote the coordinates (only *relevant-axis*) of point 1 and 2 that define the line. Similarly, z denotes a z-coordinate of a horizontal water surface, and z_1 and z_2 denote the z-coordinates of the two points (that define the line) on the water surface.

Example:
Following data specify the two water surfaces: a horizontal surface and an inclined surface.

```
2
0
1 42.7 50.0 6.1
1
1 0.0 42.7 12.2 6.1
```

Chapter 4

Output and Visualization

4.1 Output files

4.1.1 Summary file

This file is self explanatory and it contains a summary of the results including control parameters, maximum displacement at each step, and elapsed time. The file is written in ASCII format and its name follows the convention *input_file_name_header*`_summary` for serial run and *input_file_name_header*`_summary_proc`*processor_ID* for parallel run.

4.1.2 Mesh files

This file contains the mesh information of the model including coordinates, connectivity, element types, etc., in EnSight Gold binary format (see EnSight, 2008). The file name follows the format *input_file_name_header*`_summary` for serial run and *input_file_name_header*`_summary_proc`*processor_ID* for parallel run.

4.1.3 Displacement field file

This file contains the nodal displacement field in the model written in EnSight Gold binary format. The file name follows the format *input_file_name_header*`_step`*step*`.dis` for serial run and *input_file_name_header*`_step`*step*`_proc`*processor_ID*`.dis` for parallel runs.

4.1.4 Pore pressure file

This file contains the hydrostatic pore pressure field in the model written in EnSight Gold binary format. The file name follows the format *input_file_name_header*`_step`*step*`.por` for serial run and *input_file_name_header*`_step`*step*`_proc`*processor_ID*`.por` for parallel run.

4.1.5 CASE file

This is an EnSight Gold CASE file written in ASCII format. This file contain the information on the mesh files, other files, time steps etc. The file name follows the format *input_file_name_header*`.case` for serial run and *input_file_name_header*`_proc`*processor_ID*`.case` for parallel run.

4.1.6 SOS file

This is an EnSight Gold server-of-server file for parallel visualization. The `write_sos.f90` program provided in the `/util/` may be used to generate this file. See Chapter 5, Section 5.3 for more detail.
All above EnSight Gold files correspond to the model with spectral-element mesh. Additionally, the CASE file/s and mesh file/s are written for the original model. These file names follow the similar conventions and they have the tag `'original'` in the file name headers.

4.2 Visualization

4.2.1 Serial visualization

Requirement: ParaView version later than 3.7. Precompiled binaries available from ParaView web (`www.paraview.org`) may be installed directly or it can be build from the source.

- open a session
- open paraview client
  `paraview`
- In ParaView client: ⇒ File ⇒ Open
  select appropriate serial CASE file (.case file)
  see ParaView wiki `paraview.org/Wiki/ParaView` for more detail.

4.2.2 Parallel visualization

Requirement: ParaView version later than 3.7. It should be built enabling MPI. An appropriate MPI library is necessary.

- open a session
- open paraview client
  `paraview`
- start ParaView server
  mpirun -np 8 pvserver -display :0
- In ParaView client: ⇒ File ⇒ Connect and connect to the appropriate server
- In ParaView client: ⇒ Open
  select appropriate SOS file (.sos file)
  see ParaView wiki (`paraview.org/Wiki/ParaView` for more detail.

Note: Each CASE file obtained from the parallel processing can also be visualized in a serial.

Chapter 5

Utilities

5.1 Convert EXODUS mesh into SEM files

The program `exodus2semgeotech.c` contained in the utilities directory can be used to convert the mesh file in EXODUS II format to input files required by the SPECFEM3D_GEOTECH .

Compile

```
gcc -o exodus2semgeotech exodus2semgeotech.c
```

Run

`exodus2semgeotech` *EXODUS_mesh_file OPTIONS*
For more details, see `/util/README_exodus2semgeotech`. It can also be compiled automatically during the build process of main package SPECFEM3D_GEOTECH (see Section 2.3).

5.2 Convert GiD mesh into SEM files

The program `gid2semgeotech.c` contained in the utilities directory can be used to convert the mesh file in GiD format to input files required by the SPECFEM3D_GEOTECH .

Compile

```
gcc -o gid2semgeotech gid2semgeotech.c
```

Run

`gid2semgeotech` *GiD_mesh_file OPTIONS*
For more details, see `/util/README_gid2semgeotech`. It can also be compiled automatically during the build process of main package SPECFEM3D_GEOTECH (see Section 2.3).

5.3 Generate SOS file

The program `write_sos.f90` contained in the utilities directory can be used to write EnSight Gold server-of-server file (.sos file, see (EnSight, 2008)) to visualize the multi-processors data in parallel. This file does not contain the actual data, but only information on the data location and parallel processing.

Compile

```
gfortran -o write_sos write_sos.f90
```

Run

`exodus2sem` *input_file*

For more details, see `/util/README_write_sos`. It can also be compiled automatically during the build process of main package SPECFEM3D_GEOTECH (see Section 2.3).

Chapter 6

Tutorials

6.1 Building a slope model from scratch

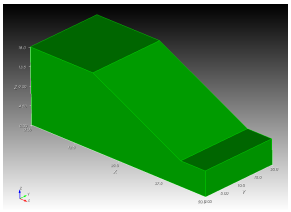


Figure 6.1: A slope model.

In this tutorial we build a slope model as shown in Figure 6.1 using CUBIT/Trelis and simulate slope failure using SPECFEM3D_GEOTECH.

Step 1: Create a mesh in CUBIT/Trelis

Create a journal file "cubit_example.jou" with following content:

```
#-------BEGIN "cubit_example.jou"-----------------------
# create model
create vertex 0 0 0
create vertex 50 0 0
create vertex 50 0 6
create vertex 43 0 6
create vertex 18 0 18
create vertex 0 0 18
create curve vertex 1 2
create curve vertex 2 3
create curve vertex 3 4
create curve vertex 4 5
create curve vertex 5 6
create curve vertex 6 1
create surface curve 1 2 3 4 5 6
sweep surface 1 vector 0 1 0 distance 20
compress all
# mesh surface
surface 8 size 2
surface 8 scheme pave
mesh surface 8
# mesh volume sweeping surface mesh
volume 1 size 2
volume 1 redistribute nodes off
volume 1 scheme Sweep source surface 8 target surface 1 sweep_smooth Auto
sweep_transform least_squares autosmooth_target off
mesh volume 1
```

```
# define block
set duplicate block elements off
block 1 volume 1
# define boundary conditions
Sideset 1 surface 2
sideset 1 name 'bottom_ssbcux_ssbcuy_ssbcuz'
Sideset 2 surface 8
sideset 2 name 'front_ssbcuy'
Sideset 3 surface 1
sideset 3 name 'back_ssbcuy'
Sideset 4 surface 7
sideset 4 name 'left_ssbcux'
Sideset 5 surface 3
sideset 5 name 'right_ssbcux'
compress all
# save and export mesh
save as "cubit_example.cub" overwrite
set large exodus file off
export mesh "cubit_example.e" overwrite
#---------END "cubit_example.jou"--------------------
```

Then open "cubit_example.jou" file in CUBIT/Trelis and run.

Note: Above procedure can also be performed manually using CUBIT's GUI.

Step 2: Convert "Binary" EXODUS file to "ASCII" format

```
ncdump cubit_example.e > cubit_example.txt
```

Note: You must have installed NetCDF libraries. NetCDF is a free software which can be downloaded from `http://www.unidata.ucar.edu/downloads/netcdf/index.jsp`

Step 3: Compile exodus2semgeotech tool

If not already compiled, compile exodus2semgeotech.c located at util/ folder

```
gcc exodus2semgeotech.c -o exodus2semgeotech
```

Note: For more information, please check the header in exodus2semgeotech.c file.

Step 4: Convert exodus file to SPECFEM3D_GEOTECH files

```
./exodus2sem cubit_example.txt
```

Copy all generated files to input/ folder.

Step 5: Prepare BC files

Open BC files "input/cubit_example_ssbcux", "input/cubit_example_ssbcuy", "input/cubit_example_ssbcuz". Add a line containing

```
2 0.0
```

on the top of each file and save.

Step 6: Create a material properties file

Create a file "cubit_example_material_list" in input/ folder with the following content:

```
# material properties (id,domain,type,gamma,ym,nu,phi,coh,psi)
1
1, 1, 0, 18.8, 1e5, 0.3, 20.0, 29.0, 0.0
```

Step 7: Create a main input file

Create a file "cubit_example.sem" in input/ folder with the following content:

```
# ---------BEGIN "cubit_example.sem" --------------------------
#pre information
preinfo:  method='sem', ngllx=3, nglly=3, ngllz=3, nenod=8, ngnod=8, &
inp_path='./input', out_path='./output/'
#mesh information
mesh:  xfile='cubit_example_coord_x', yfile='cubit_example_coord_y', &
zfile='cubit_example_coord_z',confile='cubit_example_connectivity', &
idfile='cubit_example_material_id'
#boundary conditions
bc:  uxfile='cubit_example_ssbcux', uyfile='cubit_example_ssbcuy', &
uzfile='cubit_example_ssbcuz'
#material list
material:  matfile='cubit_example_material_list'
#control parameters
control:  cg_tol=1e-8, cg_maxiter=5000, nl_tol=0.0005, nl_maxiter=3000, &
nsrf=9, srf=1.0 1.5 2.0 2.15 2.16 2.17 2.18 2.19 2.20
#save data options
save:  disp=1
#---------END "cubit_example.sem" --------------------------
```

Step 8: Run prgroam

go to SPECFEM3D_GEOTECH folder in terminal and type

```
./bin/semgeotech ./input/cubit_example.sem
```

Note: You must have compiled SPECFEM3D_GEOTECH. For more detail please see Chapter 2.

Step 9: Visualize result

Start ParaView and Open "output/cubit_example.case" file

Note: You must have installed ParaView. ParaView is a free opensource visualization software which can be downloaded from `http://www.paraview.org/`.

Bibliography

Barragy, E. and G. F. Carey (1988). A parallel element-by-element solution scheme. *International Journal for Numerical Methods in Engineering 26*, 2367–2382.

Canuto, C., M. Y. Hussaini, A. Quarteroni, and T. A. Zang (1988). *Spectral methods in fluid dynamics*. Springer.

CUBIT (2011). *CUBIT 13.0 User Documentation*. Sandia National Laboratories. [Online; accessed 27-May-2011].

EnSight (2008). *EnSight User Manual* (Version 9.0 ed.). Salem Street, Suite 101, Apex, NC 27523 USA: Computational Engineering International, Inc. [Online; accessed 11-July-2011].

Faccioli, E., F. Maggio, R. Paolucci, and A. Quarteroni (1997). 2D and 3D elastic wave propagation by a pseudo-spectral domain decomposition method. *Journal of Seismology 1*, 237–251.

Geuzaine, C. and J. F. Remacle (2009). Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering 79*(11), 1309–1331.

Gharti, H. N., D. Komatitsch, V. Oye, R. Martin, and J. Tromp (2012). Application of an elastoplastic spectral-element method to 3D slope stability analysis. *International Journal for Numerical Methods in Engineering 91*, 1–26.

Gharti, H. N., V. Oye, D. Komatitsch, and J. Tromp (2012). Simulation of multistage excavation based on a 3D spectral-element method. *Computers & Structures 100–101*, 54–69.

Gropp, W., E. Lusk, and A. Skjellum (1994). *Using MPI, portable parallel programming with the Message-Passing Interface*. Cambridge, USA: MIT Press.

Hughes, T. J. R., I. Levit, and J. Winget (1983). An element-by-element solution algorithm for problems of structural and solid mechanics. *Computer Methods in Applied Mechanics and Engineering 36*(2), 241–254.

King, R. B. and V. Sonnad (1987). Implementation of an element-by-element solution algorithm for the finite element method on a coarse-grained parallel computer. *Computer Methods in Applied Mechanics and Engineering 65*(1), 47–59.

Komatitsch, D. and J. Tromp (1999). Introduction to the spectral element method for three-dimensional seismic wave propagation. *Geophysical Journal International 139*, 806–822.

Komatitsch, D. and J. P. Vilotte (1998). The spectral element method: An efficient tool to simulate the seismic response of 2D and 3D geological structures. *Bulletin of the Seismological Society of America 88*(2), 368–392.

Larsen, S. and C. A. Schultz (1995). ELAS3D: 2D/3D elastic finite difference wave propagation code: Technical Report No. UCRL-MA-121792. Technical report.

Law, K. H. (1986). A parallel finite element solution method. *Computers & Structures 23*(6), 845–858.

Pacheco, P. (1997). *Parallel Programming with MPI*. Morgan Kaufmann.

Patera, A. T. (1984). A spectral element method for fluid dynamics: laminar flow in a channel expansion. *Journal of Computational Physics 54*, 468–488.

Pellegrini, F. and J. Roman (1996). SCOTCH: A software package for static mapping by dual recursive bipartitioning of process and architecture graphs. *Lecture Notes in Computer Science 1067*, 493–498.

Peter, D., D. Komatitsch, Y. Luo, R. Martin, N. Le Goff, E. Casarotti, P. Le Loher, F. Magnoni, Q. Liu, C. Blitz, T. Nissen-Meyer, P. Basini, and J. Tromp (2011). Forward and adjoint simulations of seismic wave propagation on fully unstructured hexahedral meshes. *Geophysical Journal International 186*(2), 721–739.

Rainsberger, R. (2006). *TrueGrid User's Manual* (version 2.3.0 ed.). Livermore, CA: XYZ Scientific Applications, Inc.

Seriani, G. (1994). 3-D large-scale wave propagation modeling by spectral element method on Cray T3E multiprocessor. *Computer Methods in Applied Mechanics and Engineering 164*, 235–247.

Smith, I. M. and D. V. Griffiths (2004). *Programming the finite element method*. John Wiley & Sons.

Zheng, H., D. F. Liu, and C. G. Li (2005). Slope stability analysis based on elasto-plastic finite element method. *International Journal for Numerical Methods in Engineering 64*, 1871–1888.

Zienkiewicz, O. and I. Cormeau (1974). Visco-plasticity–plasticity and creep in elastic solids — a unified numerical solution approach. *International Journal for Numerical Methods in Engineering 8*(4), 821–845.