MESS 2013: Ses3d-NT Handout

Stefan Mauerberger

March 9, 2013

0 Build Instructions (not part of the practical)

This section is for the sake of completeness and not part of the MESS practical. It provides several information on how to get the Ses3d-NT source-code, which software dependencies need to be met and how to build the executable.

Build prerequisites

Before you try to build Ses3d-NT, make sure you have the correct tools, and have configured these tools correctly. At first, a very recent GCC Fortran compiler is needed. This is gfortran 4.7.x and newer, which comes with almost all Linux distributions released quite recently. Unfortunately, up to now no other Fortran compilers as Intel, NAG or PGI are supporting modern F03 and F08 features exploited by Ses3d-NT. In addition to gfortran, some MPI 2.x implementation is needed. Ses3d-NT was successfully built and tested with Intel-MPI, IBM-MPI, MPICH and OpenMPI.

Optionally, you may want Ses3d-NT to write 3d volume snapshots in the netCDF file format. Activating this feature makes the libraries zlib, hdf5 and netCDF necessary. Please make sure all of those libraries are compiled with *parallel support*. Unless GCC and openMPI no major LINUX-distribution ships netCDF with parallel support enabled. Some information on how to build parallel-netCDF is provided in the Ses3d-NT wiki.[1]

Getting the Source-Code

The development of Ses3d-NT takes place under strict revision control. To get the latest version it is highly recommended downloading the sources from the SVN-repository. To checkout a working copy first make sure Subversion is installed and then type:

```
svn checkout https://svn.geophysik.uni-muenchen.de/svn/ses3d/trunk/ ./ses3d
```

This will copy all the content of the current development branch called *trunk* into a local folder named `./ses3d`. Besides the bare source-code this also includes several examples and a python-toolbox for pre- and post-processing.

Reminder: Ses3d-NT is under heavy development. To benefit from current development, do not forget to update and rebuild regularly. To do so, switch to the `./ses3d` directory and simply enter:

```
svn update
```

Building the Binarie

Ses3d-NT uses GNU make to drive the build-process. GNU make gets its knowledge of how to build a program from the file `./src/Makefile`. That file provides several options, flags and switches. Documentation can be found inside the Makefile. On most systems it should be sufficient just typing

```
cd src
make
```

[1] svn.geophysik.uni-muenchen.de/trac/ses3d/wiki/netcdf

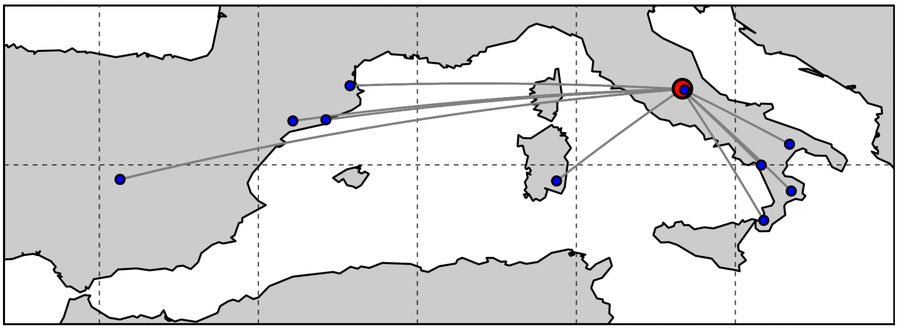
to build SES3D-NT. Up to now, LINUX and UNIX-like platforms are supported only.

Hint: By default the feature writing netCDF files is disabled. For more information how to deliberately enabling that very useful feature please address annotations given in the Makefile and consulate the SES3D-NT wiki.

1 The 2009 L'Aquila Earthquake

This practical is wrapped around the L'Aquila earthquake which occurred in the region of Abruzzo, in central Italy. The main shock occurred at 1:32 UTC on 6 April 2009, and was rated $M_\mathrm{w} = 6.3$ on the moment magnitude scale.

The aim of that practical is to get started with the simulation of seismic wave-propagation on commonly use personal computers. Due to limitations in computational resources, simulated frequencies are limited to an ultra-low frequency band of approximately 0.5 mHz to 20.0 mHz. It is the practical's main intention to present SES3D-NT's flexibility, simplicity and prowess in combination with making the reader a little familiar with common pre- and post-processing steps.



Earth-Model and Computational Domain

The physical representation used for the simulation is a spherical section bound by 35 °N to 45 °N latitude and −8 °W to 20 °E longitude reaching to a depth of 1,000 km, which is a narrow strip of southern Europe depicted in the figure shown above. For the beginning keeping this practical rather simple, the underlying earth model is chosen to be isotropic PREM. In the next paragraph, the creation of custom models will be discussed in detail.

Discretization

For the reason of discretization, as a first step the modeling domain is partitioned into elements. There are a 14-elements in lateral-direction, 28-elements in longitudinal-direction and a 15-elements in depth, which leads to a 5,880 elements in total. With the Lagrange-Polynomial-Degree chosen to be $lpd = 4$, each element is sampled by a 125 collocation points, five into each coordinate direction.

Spatial Sampling & Exciting Frequencies

For solving the elastic wave-equation the Spectral Element Method is used. This numerical approach promises a spatial sampling-rate of approximately two elements per minimal wavelength. (The following is just a rough estimate.)

Element dimension in that discretization are in the order of $\Delta \approx 65\,\mathrm{km}$, which limits the minimal wavelength to $\lambda_{min} \approx 130\,\mathrm{km}$. On the other hand model extents are limiting the maximal wavelength to $\lambda_{max} \approx 1,000\,\mathrm{km}$.

Knowing the seismic velocities $c_{min} \approx 3\,\mathrm{km/s}$ to $c_{max} \approx 11\,\mathrm{km/s}$ in combination with admissible wavelengths, is restricting the frequency content of exciting signals. Taking into account $c = \lambda f$ leads to an exciting frequency range of $f_{max} \approx 23\,\mathrm{mHz}$ to $f_{min} \approx 2\,\mathrm{mHz}$

Time-Stepping

Time-stepping is chosen to achieve a *courant*-number of $c = 0.5$. Considering the spatial discretization, this leads to a time-increment of $dt = 0.5\,\mathrm{s}$. To grantee wave-propagation throughout the whole domain, the number ob time-steps is set to $nt = 1,500$.

Source

The source mechanism is taken from the Global CMT catalog. Source location and moment-tensor components are:

$$\begin{array}{lr} \theta = 42.35\,^\circ & \text{lat} \\ \phi = 13.38\,^\circ & \text{lon} \\ r = 9.46\,\mathrm{km} & \text{depth} \end{array} \qquad M = \begin{pmatrix} 1.43 & -1.77 & -1.43 \\ -1.77 & 1.87 & -0.27 \\ -1.43 & -0.27 & -3.30 \end{pmatrix} \times 10^{18}\,\mathrm{Nm}$$

For reasons of simplicity, the source-time-function is chosen to be a Ricker-pulse whit a dominant frequency of $f_d \approx 17\,\mathrm{mHz}$.

Receivers

Synthetic seismograms are going to be recorded at ten different positions, indicated in the figure above. In addition, corresponding values for each station are listed in the subsequent table. For all stations the recorded channels are displacement velocities in N, E and Z directions.

Network	Station	latitude	longitude
IU	PAB	40.54	-4.349
MN	AQU	42.35	13.40
MN	CEL	38.26	15.89
MN	CUC	39.99	15.81
MN	TIP	39.17	16.75
MN	VSL	39.49	9.378
CA	FBR	41.41	2.125
CA	POBL	41.37	1.084
FR	SJAF	42.48	2.882
GE	MATE	40.64	16.70

2 Running the Simulation

Running the simulation is fairly simple. The is already a pre-built executable available throughout the VirtualBox. A configuration file with all the parameters set as described above is provided, too. The only ingredients needed, are the number of processes the simulation shall be parallelized onto and a MPI execution environment. It depends on the actual MPI implementation but in most cases the commands are either `mpirun` or `mpiexec`.

In our practical the commands running the L'Aquila toy-example are

```
cd Desktop/ses3d
nice -19 mpirun -np 1 ses3d l_aquila.fwd
```

where `ses3d l_aquila.fwd` are the actual executable followed by the configuration file. This is encapsulated by `nice -19` which defines the execution priority and the MPI execution environment `mpirun` with the flag `-np` specifying the number of processors invoked.

Note: The number of processes can not be chosen arbitrarily. It has to divide the number of elements in at least one direction as a whole number. The L'Aquila example is setup pretty flexible allowing parallelization onto 1 to 8 CPUs, depending on someones hardware.

Hint: If your system provides multiple cores or threads you might want to try to run the VirtualBox on more than just one process which promises a speedup in computations.

3 Viewing the Results

There is a huge variety on programs which can be used for visualizing and processing SES3D-NT's output. In the following only two of them are introduced, namely ObsPy and ParaView.

3.1 Plotting Synthetic Seismograms

SES3D-NT writes synthetic seismograms in the SAC file format, trace by trace. SAC files can easily read and processed with ObsPy. To do so, switch to `Desktop/ses3d/` - where the L'Aquila simulation was carried out - and open your preferred Python environment, which might be IPython. By executing the following code

```
import obspy
st = obspy.read('./data/*.sac')
stations = list(set(tr.stats.station for tr in st))
for station in stations:
    st.select( station=station ).plot()
```

all SAC files in the director `./data/` will be read into a stream-object. As a second step all station names are extracted and stored in a python-list. Finally, the code iterates through all stations, selecting all those traces correspond to the current station name and plotting these.

3.2 Volume-snapshots

Probably the most user-friendly way to visualize SES3D-NT's 3d-datasets is the 3d visualization software ParaView, developed by Kitware. Volume snapshots which are written in the netCDF file format can be opened directly, without conversion.

The L'Aquila toy-example produces a whole bunch of output. One file is `rho_mu_lambda_1.nc` with the simulation's model-parameters ρ, λ and μ inside. The other 74 files `vx_vy_vz_[#].nc` are volume-snapshots of displacement-velocities v_θ, v_ϕ and v_r. To visualize:

1. Load ParaView, click 'open', move to `Desktop/ses3d/data/` and select the group `vx_vy_vz_..nc`.
2. In the occurring window, select the netCDF file format.
3. In the left hand panel *Properties* click *Apply* will show an outline of the computational domain.
4. For better data representation, someone should switch from *Outline*-view to the *Surface*-view.
5. To to actually show data included in the netCDF files, a dataset needs to be selected explicitly. Select from the drop-down menu *Solid Color* the dataset you are interested in.

ParaView comes with a huge variety of post-processing-, visualization and filtering-tools. Some features are self-explanatory others are not.