SphGLLTools

User Manual - Version: 1.0

Caio Ciardelli

October 26, 2020

Northwestern
University

Contents

1 Author

This package was developed by Caio Ciardelli at the Colorado School of Mines, under the supervision of Prof. Ebru Bozdağ, the Northwestern University, under the supervision of Prof. Suzan van der Lee and the University of São Paulo, under the supervision of Prof. Marcelo Assumpção, from 2018 to 2020. The post-processing tools of SPECFEM3D_GLOBE written by Prof. Daniel Peter served as the basis for many routines of SphGLLTools.

2 Citation

If you use SphGLLTools, please, cite the following paper:

Ciardelli, C., Bozdağ, E. and Peter, D., 2020. SphGLLTools: A set of routines for visualization, processing, sharing, and spherical harmonics analysis of tomographic models defined on GLL meshes. *Computer & Geosciences*, submitted.

3 Introduction

SphGLLTools is a set of C, Python, Shell Script, and GMT6 (Wessel et al., 2019) routines for visualization, processing, conversion, spherical harmonics analysis, and sharing of tomographic models defined on Gauss-Lobatto-Legendre (GLL) meshes. It was designed to work with SPECFEM3D_GLOBE (Komatitsch et al., 2016) but, in theory, should work with other solvers with little modification.

4 Getting started

4.1 Dependencies

SphGLLTools requires:

- GCC 5.5.0 or greater
- OpenMPI 1.10.2 or greater
- Python 2.7.12 or greater
- GMT 6.0.0 or greater

4.2 Download

The easiest way of downloading the last stable version of SphGLLTools is using Git:

```
1 $ git clone https://github.com/caiociardelli/sphglltools.git
```

4.3 Installation

SphGLLTools requires no installation. The code is designed to run on clusters alongside SPECFEM3D_GLOBE. But it can also run on a personal computer, provided that you have enough computational power or are using low-resolution meshes. Just like SPECFEM3D_GLOBE, the routines allocate

memory on the stack, for performance reasons. However, the default stack size is too small (usually, around 65 MB).

On Linux systems, you can remove that limit by adding **ulimit -S -s unlimited** to your *.bash_profile* file or **limit stacksize unlimited** to your *.cshrc* file, to suppress any potential limit to the size of the stack.

Some Mac OS have hard stack limits that cannot be removed. In those cases, the code can crash due to a *stack overflow*. In general, wherever you can run SPECFEM3D_GLOBE, you should also be able to run SphGLLTools.

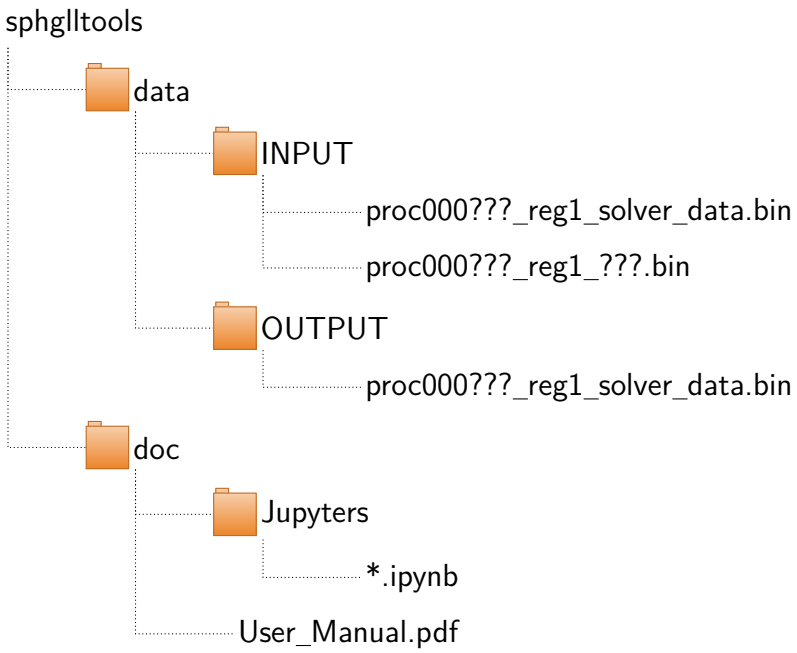
4.4 Compiling

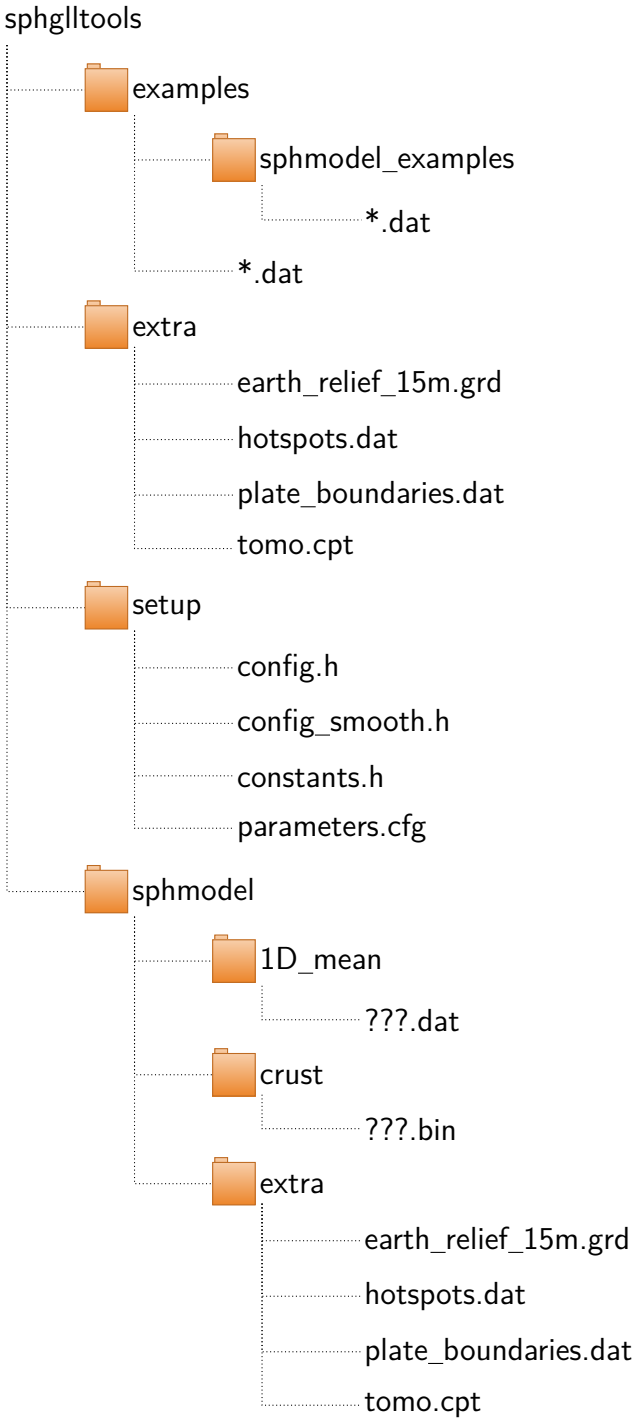
All you need to compile the routines is to run the Makefile inside the parent directory:

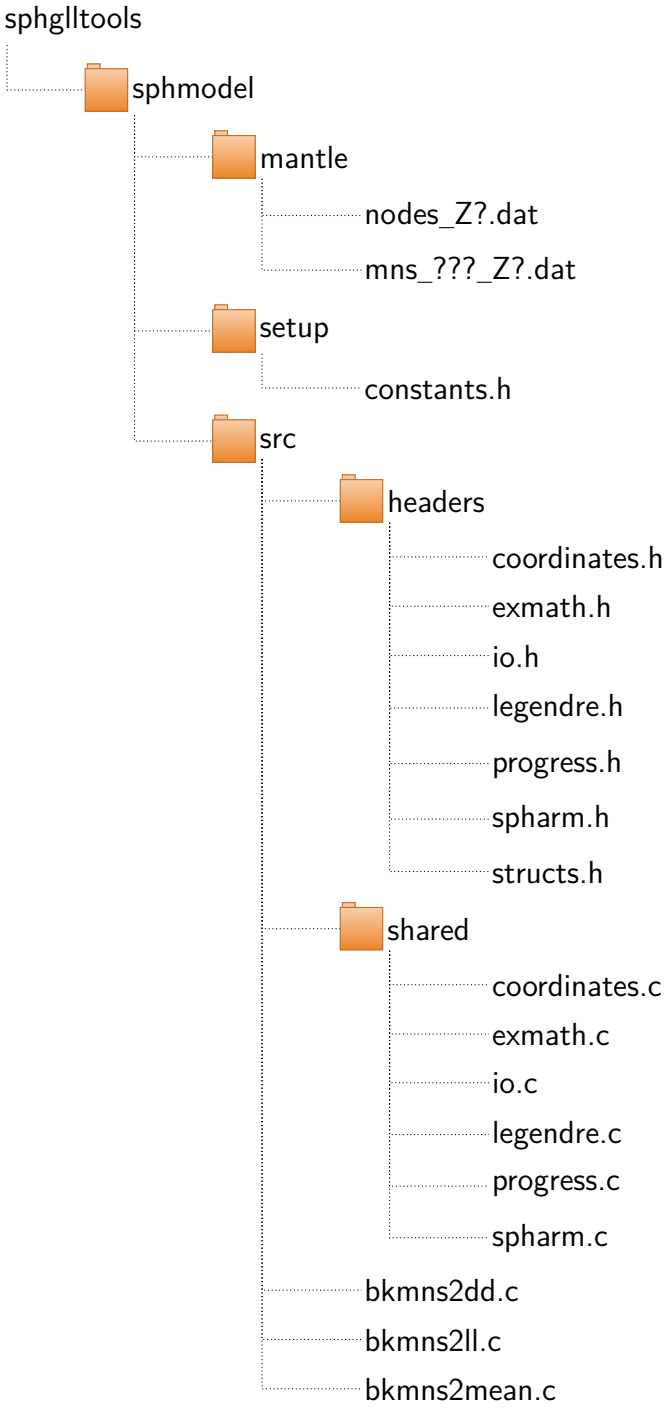
```
1 $ make -j4 all
```

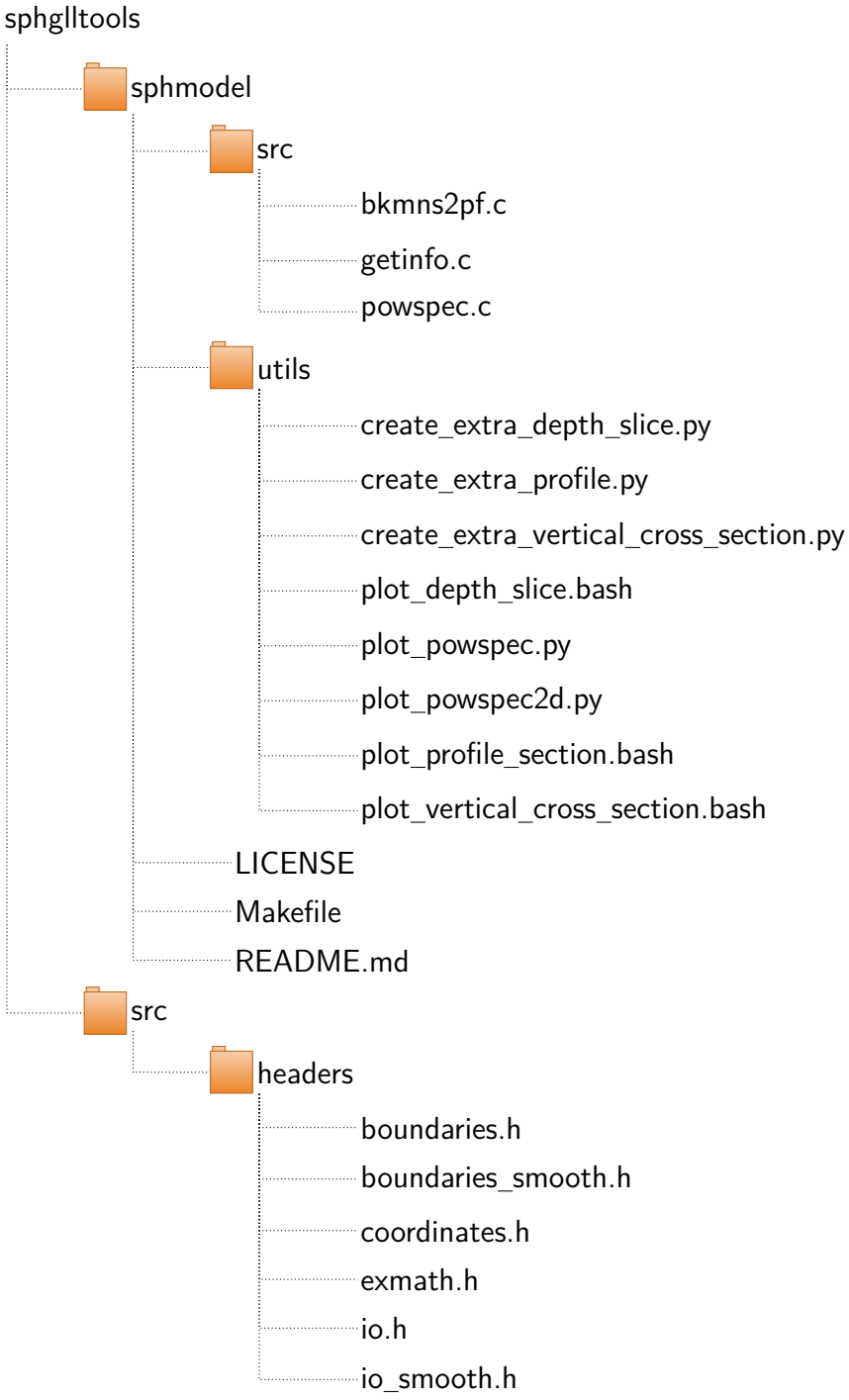
5 Overview

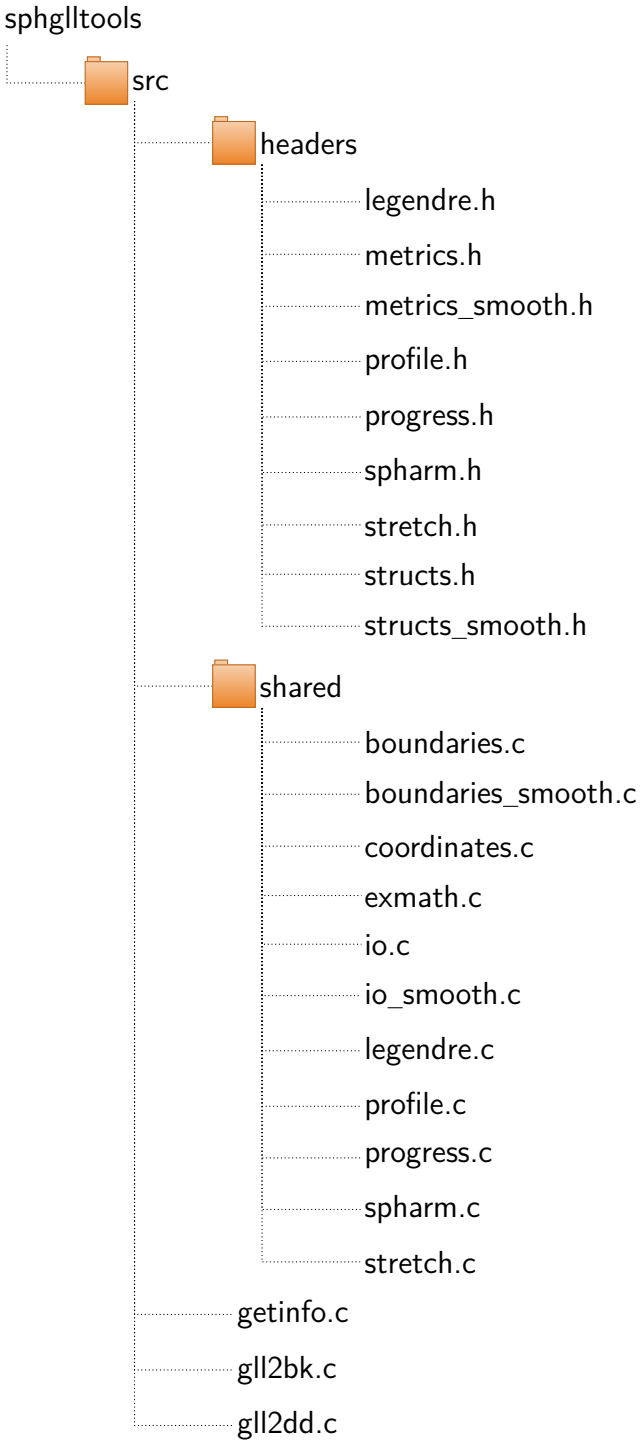
The structure of the package looks like this:

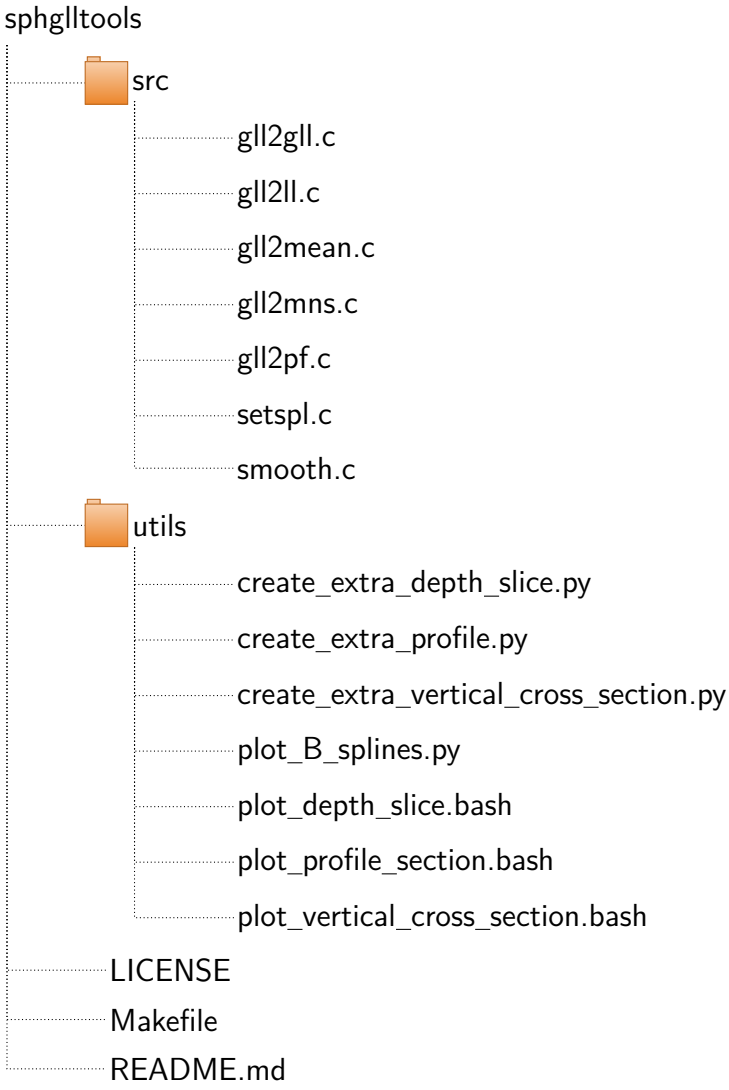



sphglltools
examples
sphmodel_examples
*.dat
*.dat
extra
earth_relief_15m.grd
hotspots.dat
plate_boundaries.dat
tomo.cpt
setup
config.h
config_smooth.h
constants.h
parameters.cfg
sphmodel
1D_mean
???.dat
crust
???.bin
extra
earth_relief_15m.grd
hotspots.dat
plate_boundaries.dat
tomo.cpt


sphglltools
sphmodel
mantle
nodes_Z?.dat
mns_???_Z?.dat
setup
constants.h
src
headers
coordinates.h
exmath.h
io.h
legendre.h
progress.h
spharm.h
structs.h
shared
coordinates.c
exmath.c
io.c
legendre.c
progress.c
spharm.c
bkmns2dd.c
bkmns2ll.c
bkmns2mean.c


sphglltools
sphmodel
src
bkmns2pf.c
getinfo.c
powspec.c
utils
create_extra_depth_slice.py
create_extra_profile.py
create_extra_vertical_cross_section.py
plot_depth_slice.bash
plot_powspec.py
plot_powspec2d.py
plot_profile_section.bash
plot_vertical_cross_section.bash
LICENSE
Makefile
README.md
src
headers
boundaries.h
boundaries_smooth.h
coordinates.h
exmath.h
io.h
io_smooth.h


sphglltools
src
headers
legendre.h
metrics.h
metrics_smooth.h
profile.h
progress.h
spharm.h
stretch.h
structs.h
structs_smooth.h
shared
boundaries.c
boundaries_smooth.c
coordinates.c
exmath.c
io.c
io_smooth.c
legendre.c
profile.c
progress.c
spharm.c
stretch.c
getinfo.c
gll2bk.c
gll2dd.c



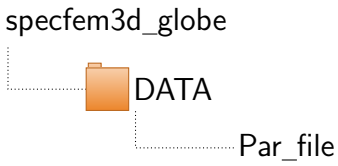
As pictured above, inside the main package, there is a subpackage called *sphmodel*. That is a stand-alone set of routines designed to share a model represented onto spherical harmonics with other researchers. SphGLLTools provide all the programs needed to carry out the expansion.

6 Basic visualization with SphGLLTools

SphGLLTools is composed of two main sets of routines: those who work using interpolation and those related to the spherical harmonics expansions. The first ones can handle both surface and internal topographies, but the second ones need a perfectly spherical mesh. None of them, however, can deal with ellipticity. For educational purposes, let's describe a workflow in which we use all routines in conjunction.

Since the package works in conjunction with SPECFEM3D_GLOBE, the first step is creating the mesh and the model files using the mesher.

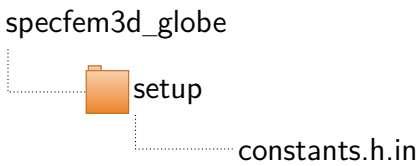
Because we don't want ellipticity, let's turn it off. You can do that under:



In section **# parameters describing the Earth model**, change the **ELLIPTICITY** flag to **false**.

The interpolation routines can handle any number of chunks, but spherical harmonics only make sense in a global mesh. So, set **NCHUNKS** = 6. For performance reasons, use a low-resolution mesh by setting **NEX_XI** = **NEX_ETA** = 64 and **NPROC_XI** = **NPROC_ETA** = 2. For the model, choose S362ANI (Kustowski et al., 2008) + CRUST2.0 (Bassin, 2000) by setting **MODEL** to s362ani_crust2.0. Lastly, you need to set **SAVE_MESH_FILES** as **true** and **ADIOS_ENABLED** as **false** (this last one is important because SphGLLTools cannot handle the ADIOS format yet).

The next step is disabling the topography of the seismic-velocity discontinuities at depths of 410 and 650 km. That is mandatory for the spherical harmonics expansion and advisable for the interpolation. You can do that under:



Set the **SUPPRESS_INTERNAL_TOPOGRAPHY** flag to **false**. Afterward, reconfigure the package to update constants.h:

```
$ ./configure FC=mpif90 CC=mpicc MPIFC=mpif90 MPICC=mpicc FCFLAGS
  ='-g -O3' CFLAGS='-g -O3'
```

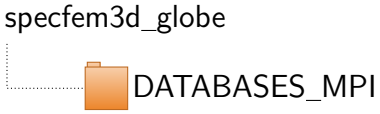
Now, you can compile everything with:

```
$ make -j4 all
```

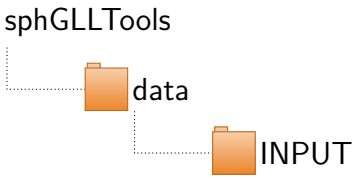
Run the mesher:

```
$ mpiexec -n 24 ./bin/xmeshfem3D
```

The mesher will save both the topological and the model files to the "DATABASES_MPI" directory:

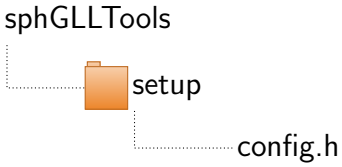


Copy the topological files "proc0000??_reg1_solver_data.bin" and the model files "proc0000??_reg1_[vp,vs,rho].bin" to "data/INPUT" directory:



Within "specfem3d globe/OUTPUT FILES/" there is a header file called "values from mesher.h". The flag **NSPEC_CRUST_MANTLE** (on line 78) is the number of spectral elements in each mesh slice, and the flag **NGLOB_CRUST_MANTLE** (on line 82) is the total number of points on each mesh slice. These values are necessary to set up SphGLLTools.

Under:



Set **NC** (number of cores) to 24, **NEL** (number of spectral elements) to 8896 and **NG** (total number of points) to 592913. Also, enable the parameters V_P, V_S and ρ. Then, just run the Makefile to recompile all the routines.

All the ASCII and binary files created in the following examples that are not too big are within the directory "examples". They are useful to check your results.

6.1 Creating depth slices

To create depth slices directly from the model parameters, use the routine GLL2LL. All routines have a help menu that shows up wherever you run them with no or with a wrong number of command-line parameters. The same menu is also at the beginning of each source code. For example, if we run the routine GLL2LL with no parameters, a message similar to the one in Fig. 1 will show up:

```
1 $ ./bin/gll2ll
```

For performance reasons, all the interpolating routines should be run in parallel, using MPIRUN or MPIEXEC. The ideal number of cores for many routines, including GLL2LL, is only constrained by your hardware and should be set according to the size of your problem. In this example, we picked 12. The larger the mesh resolution and/or the resolution of the output files, the more memory and cores you should request. Some routines, though, must be run with the same number of cores used to create the mesh. Those are GLL2GLL, SETSPL, GLL2MNS, and SMOOTH.

Following the example suggested in the help menu, you can create a depth slice of the model at 300 km depth with a 0.5° resolution by running:

```
1 $ mpiexec -n 12 bin/gll2ll 300 0.5 data/INPUT/ .
```

The above command should create three output files on your local directory: "vp_300_DS.dat", "vs_300_DS.dat" and "rho_300_DS.dat".

```
Error: wrong number of parameters on the comand line...

 GLL2LL

 USAGE
    mpiexec -n 12 bin/gll2ll DEPTH HORIZONTAL_RESOLUTION INPUT_DIRECTORY OUTPUT_DIRECTORY

 EXAMPLE
    mpiexec -n 12 bin/gll2ll 300 0.5 data/INPUT/ .

 COMMAND LINE ARGUMENTS
    DEPTH                  - depth in which the depth slice will be created
    RESOLUTION             - spatial distance (in degrees) between both latitude and longitude
                             grid points of the depth slice
    INPUT_DIRECTORY        - directory containing the input files
    OUTPUT_DIRECTORY       - directory where the routine will write the output files

 DESCRIPTION
    Reads the depth, the horizontal grid spacing, the input, and output directory names from
    the command line and creates a block model for the desired parameters (defined in 'config.h').
    The routine writes the output to files called PARAMETER_DEPTH_DS.dat.

--------------------------------------------------------------------------
MPI_ABORT was invoked on rank 0 in communicator MPI_COMM_WORLD
with errorcode 1.

NOTE: invoking MPI_ABORT causes Open MPI to kill all MPI processes.
You may or may not see output from other processes, depending on
exactly when Open MPI kills them.
--------------------------------------------------------------------------
```

Figure 1: Help menu for GLL2LL, printed when the user calls the routine with a wrong number of command-line parameters.

6.2 Plotting depth slices

Once you created the desired depth slices, you can easily plot any of them using the provided GMT6 scripts (see the result in Fig. 2):

```
1 $ ./utils/plot_depth_slice.bash vs 300
```

GLL2LL can only create depth slices for the available model parameters. However, you can use those parameters to derive others, such as the V_P/V_S ratio, T_i (transverse isotropy, requires V_{SV} and V_{SH}), and the bulk sound speed (C_{Bulk}) using the Python script CREATE_EXTRA_DEPTH_SLICE. For example, since you have the depths slices for V_P and V_S already, to create a depth slice for the bulk sound speed, just run:

```
1 $ ./utils/create_extra_depth_slice.py cb 300
```

To plot the result:

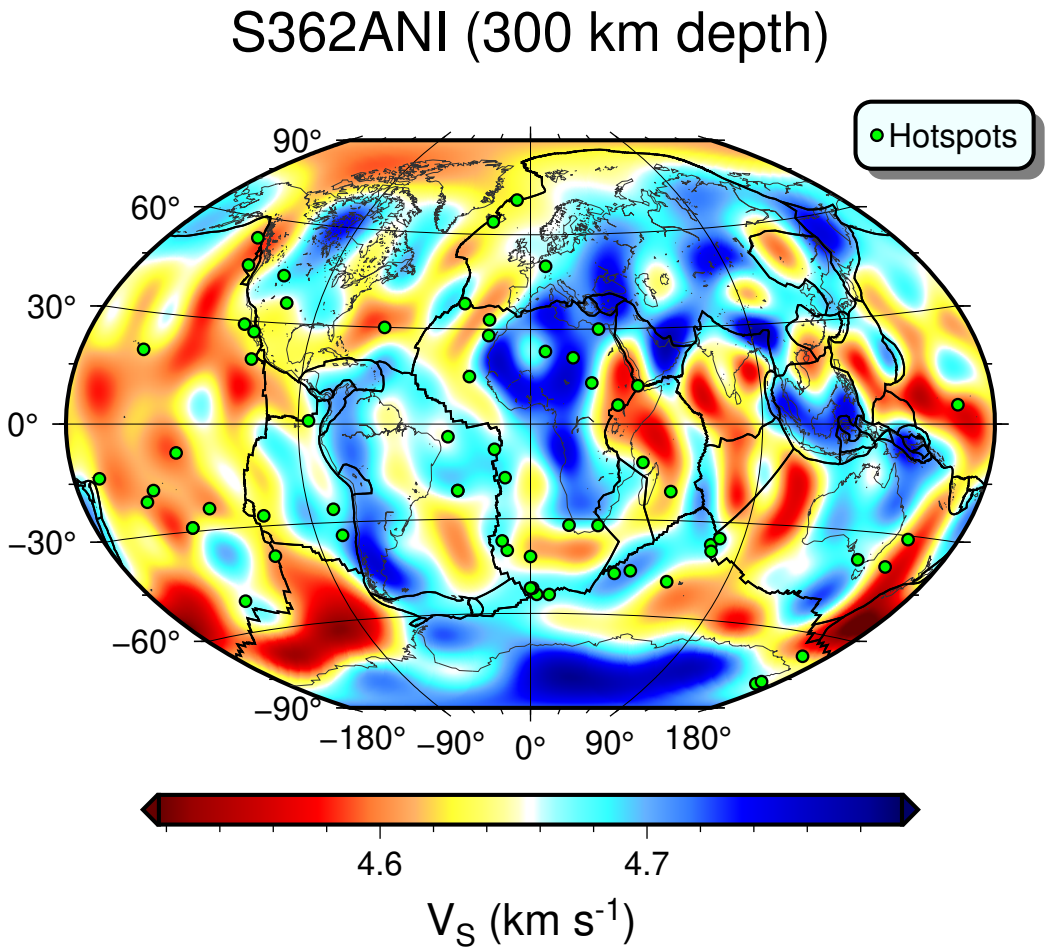


Figure 2: Depth slice at 300 km for V_S. The plate boundaries were extracted from DeMets et al. (1990).

```
$ ./utils/plot_depth_slice.bash cb 300
```

Fig. 3 shows the result.

6.3 Creating vertical cross-sections

The routine GLL2DD is analogous to GLL2LL, but instead of depth slices, it can create vertical cross-sections. For example, to write a vertical cross-section from 80 to 2891 km depth, connecting the points with coordinates

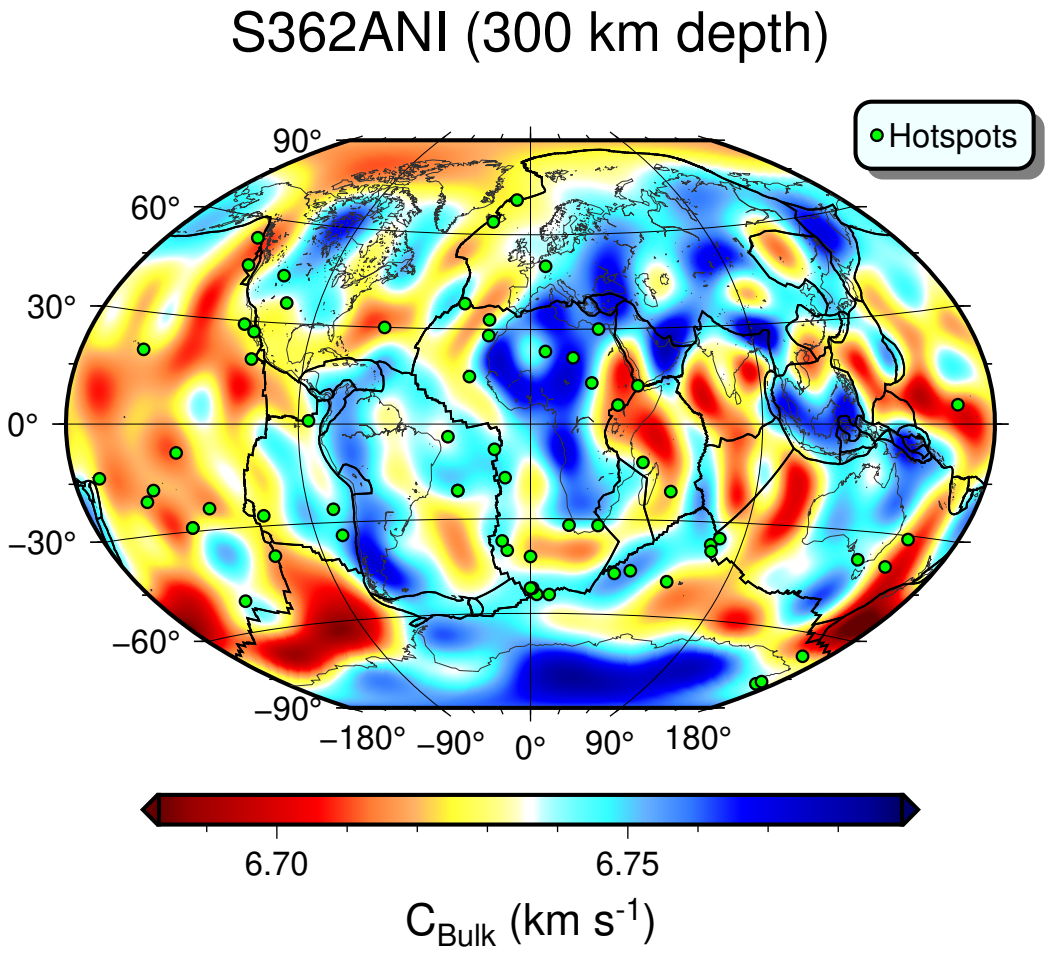


Figure 3: Depth slice at 300 km for the bulk sound speed.

(1.4S, 25.1W) and (23.7N 51.3E) along a geodesic (great circle) with 0.1° of horizontal grid spacing and 1 km of vertical grid spacing, you should run:

```
1 $ mpiexec -n 12 bin/gll2dd 80 2891 -1.4 -25.1 23.7 51.3 0.1 5 data/INPUT/ .
```

6.4 Plotting vertical cross-sections

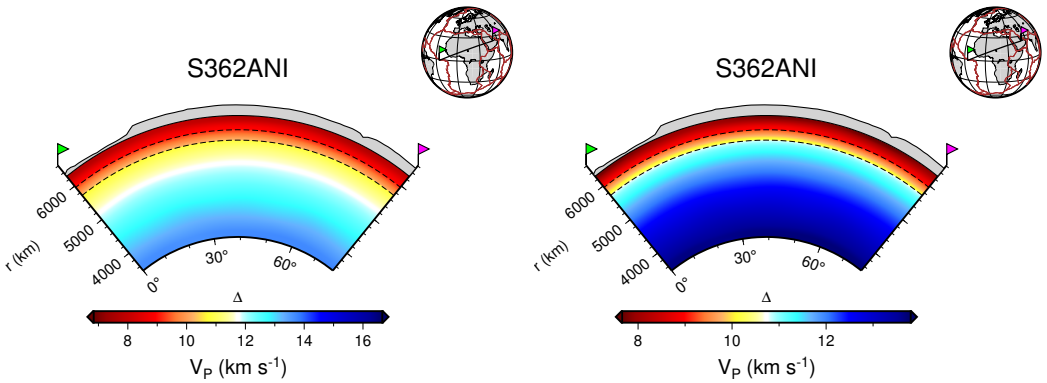
Like the depth slices, there is also a GMT6 routine for plotting vertical cross-sections:

```
$ ./utils/plot_vertical_cross_section.bash vp
```

If no limits are given, the routine tries to set them automatically by using the mean and standard deviation. However, if the result is not satisfying, you can manually set them. From the output of the first plot, we know that the V_P values range between 7.66 and 13.8 km/s. So, you can recreate the figure with:

```
$ ./utils/plot_vertical_cross_section.bash vp 7.66 13.8
```

See the results in Fig. 4:



(a) (b)

Figure 4: a) Vertical cross-section for V_P along the great circle illustrated on the inset map (upper-right corner) using the automatic color bar limits. b) Same figure but with limits manually set.

You can also compute the V_P/V_S ratio (Fig. 5):

```
$ ./utils/create_extra_vertical_cross_section.py vpvs
$ ./utils/plot_vertical_cross_section.bash vpvs
```

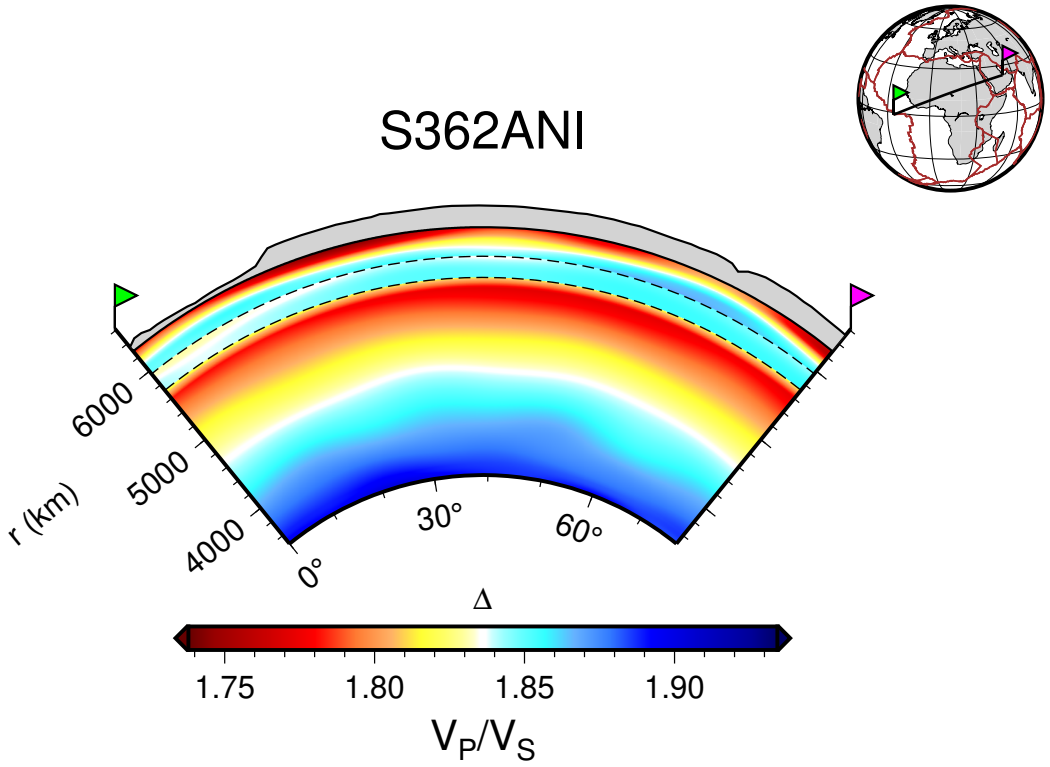


Figure 5: Vertical cross-section for the V_P/V_S ratio.

6.5 Creating one-dimensional profiles

The third visualization routine is for creating one-dimensional profiles at any point on Earth. That routine is serial, so it always uses a single core, no matter how many are requested. However, for compatibility reasons, it should also be run using MPI. The following example creates the profiles for a location in Australia, from the sea level down to the CMB, using a grid spacing of 1 km:

```
1 $ mpiexec -n 1 bin/gll2pf 0 2891 -28 140 1.0 data/INPUT/ .
```

6.6 Plotting one-dimensional profiles

There is also a third GMT6 script for plotting these profiles (see Fig. 6):

```
$ ./utils/plot_profile.bash rho
```

For all of those routines, it is possible to plot the perturbations instead of absolute values. For that, you can run the Python scripts with a letter "d" preceding the code of the parameter (e.g. ds to compute the V_S perturbations) and then plot the resulting files in the same way. The following commands will create and make a plot of the vertical cross-section for the V_S perturbations:

```
$ ./utils/create_extra_vertical_cross_section dvs
$ ./utils/plot_vertical_cross_section dvs
```

However, to compute the perturbations, you first need to calculate the mean model using the GLL2MEAN routine.

7 Computing mean models

Creating a one-dimensional model from a three-dimensional one requires averaging its values at different depths using a reasonable sampling rate. That is a computationally intensive task and requires a long time unless you can afford to use a relatively large number of cores. Even using that low-resolution mesh, to create a 1D profile every 1 km by sampling each depth at 0.5° resolution may take more than one hour, even using 36 cores. So, if you are running these examples on a cluster, you can choose a larger number of cores to speed up the calculations.

In the example below, −5 denotes a starting depth at 5 km above the sea level:

```
$ mpiexec -n 36 bin/gll2mean -5 2891 0.5 1.0 data/INPUT/ .
```

8 Remeshing

When carrying out an adjoint tomography with SPECFEM3D_GLOBE, it is usual that, eventually, we need to increase or decrease the mesh resolution or

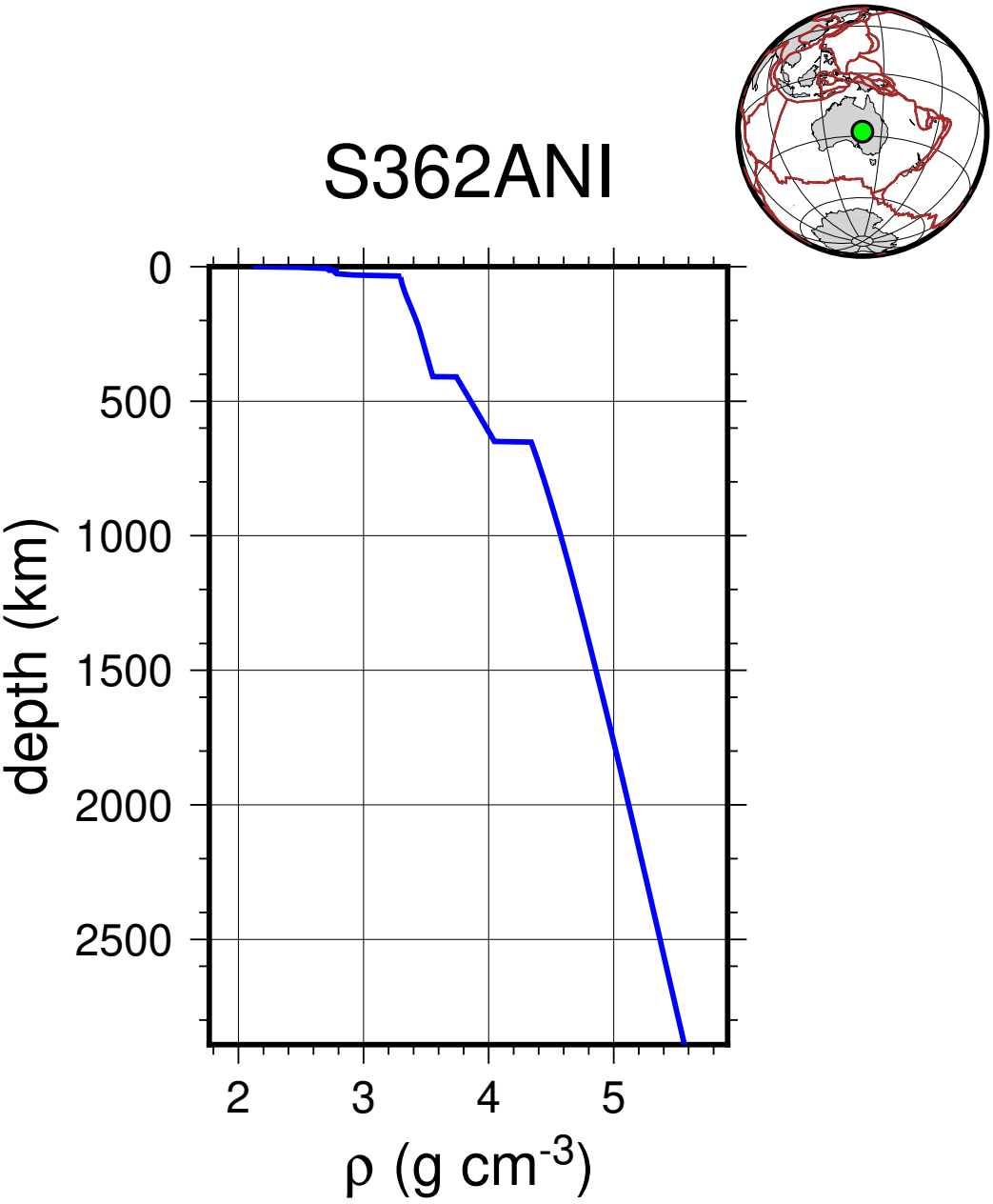
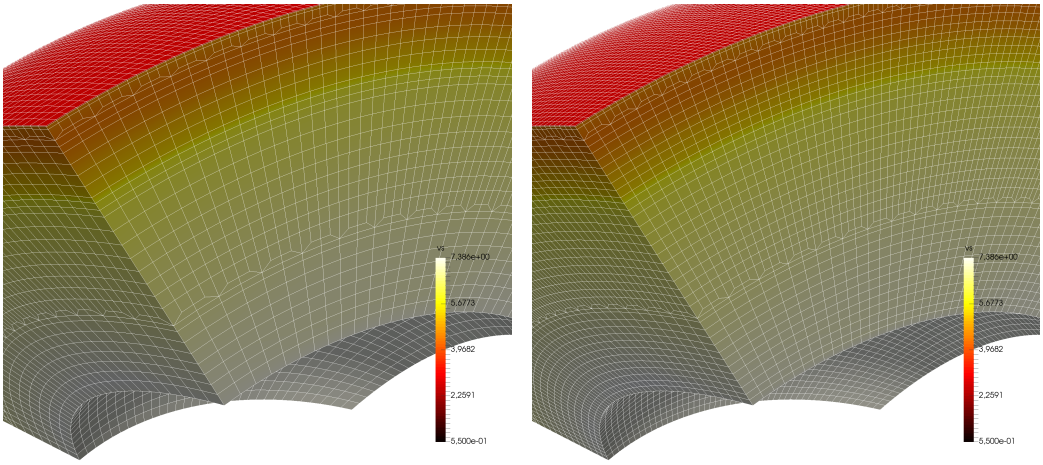


Figure 6: One-dimensional density profile in Australia.

the number of cores for various reasons. That task is way more complicated than it may seem at first glance, as accurately interpolating spectral elements

is far from trivial. Nevertheless, using routine GLL2GLL, you can easily accomplish that. Fig. 7 shows two meshes with different resolutions.



(a) NXI = NETA = 128 (b) NXI = NETA = 192

Figure 7: a) Low-resolution mesh with 128 spectral elements on each side. b) Intermediate-resolution mesh with 192 elements on each side. In both figures, the color palette denotes S-wave velocity (V_S). Extracted from Ciardelli et al., 2020.

To interpolate from one mesh to another, you must set the "config.h" file again. All you need to do is to specify the number of spectral elements a total number of points of both the input and output meshes. Set the **NEL1** flag to 8896 and the **NG1** flag to 592913.

The input topological and model files are under "data/INPUT" already. Put the topological files of the new mesh under "data/OUTPUT". The GLL2GLL routine will create the interpolated model files in "data/OUTPUT" too.

The output mesh may have any number of slices and resolution. Its number os spectral elements and points, as expected, should be provided on **NEL2** and **NG2**. The only requirement to run the routine, like for any other in the package, is that you have enough memory on your machine. In case your memory is limited, you can interpolate one model parameter at a time. Just V_S, for instance.

To run the routine, just do:

```
$ mpiexec -n 24 bin/gll2gll data/INPUT/ data/OUTPUT/
```

Remeshing is useful for changing the mesh resolution, but it is also a critical step for expanding 3D Earth models onto spherical harmonics, as you shall see in section 10.1.

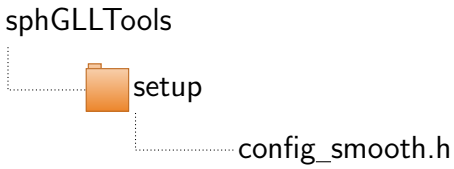
9 Smoothing

Smoothing is equivalent to applying a low-pass spatial filter to the mesh. The SMOOTH routine uses a 3D-Gaussian filter to low-pass any desired model parameter. If you have enough memory, you can smooth all the desired parameters at once. That dramatically speeds up the computations.

Usually, it is essential to smooth sensitivity *kernels*. As explained in the SPECFEM3D_GLOBE manual, even filtering the adjoint sources, the *kernels* often present numerical noise in the form of high-frequency artifacts. Low-pass filtering them is critical before computing the gradients for the model update.

However, in case your model has numerical noise already (perhaps because you didn't filter the *kernels* of the last updates enough), SMOOTH can also be used to get rid of these numerical artifacts. Smoothing your model may also be useful to prevent aliasing when interpolating it to a lower-resolution mesh.

The smoothing routine has a separated configuration file:

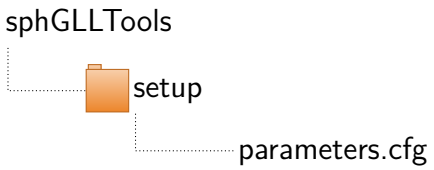


As in the previous cases, just set **NC** to the number of cores (24, in our case), **NEL** to the number of spectral elements (8896), and **NG1** to the number of points (592913).

If you set **SEPARATE_ZONES** as **true**, the routine will smooth the upper mantle, the transition zone, and the lower mantle separately. That is useful if you are filtering a model parameter and want to preserve the 410

and 650 km discontinuities. For kernels, however, you should set that flag as **false**.

The SMOOTH routine can apply different filters in the horizontal direction and the vertical direction. And those filters can also vary with depth. These parameters should be set under:



For example:

```
 1 #Number of layers in the profile
 2 #Smoothing radius for the profile (km)
 3 #Depth of the top of the layer (km)  sigma_h (km)  sigma_v (km)
 4 9
 5 100
 6 -10   100   5
 7 100   100   10
 8 200   100   20
 9 300   100   30
10 410   110   40
11 520   120   50
12 650   130   60
13 1000  140   80
14 2000  150  100
```

Figure 8: Example of smoothing parameters.

In the fourth line, we write the number of layers in the smoothing profile, which is 9, in our example.

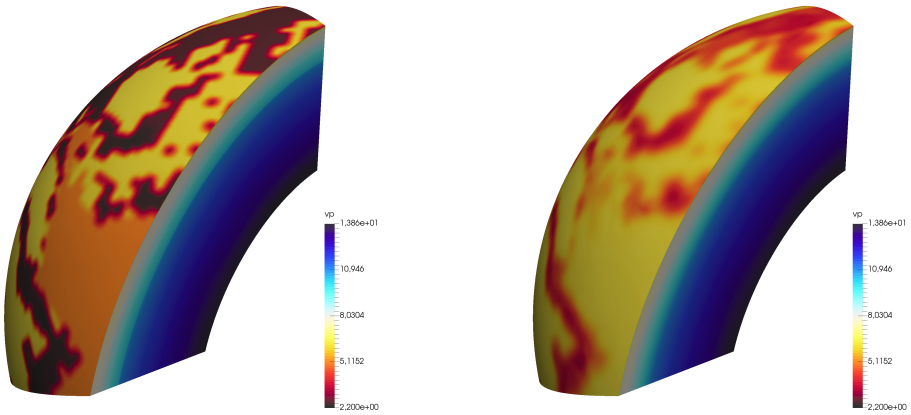
Sharp changes in smoothing with depth can create artificial discontinuities in the model. To prevent this from happening, the smoothing profile itself is convolved with a Gaussian kernel. We specify the width of this filter in the tenth line (100).

From lines 6 to 14, we write the layers. The first layer begins 10 km above the sea level and goes down to 100 km depth. The horizontal smoothing σ_h for this layer is 100 km, and the vertical smoothing σ_v is just 5 km. The last layer begins at 2000 km depth and goes all the way down to the CMB. Notice that if set the starting depth of the first layer at 20 km depth, for instance, the routine will ignore any spectral element above that depth.

After setting all the parameters, run the routine with:

```
$ mkdir smooth
$ mpiexec -n 24 bin/smooth data/INPUT/ smooth/
```

Fig. 9 shows the upper chunk for V_P before and after smoothing:



(a) (b)

Figure 9: a) Model S362ANI + CRUST2.0 with no smoothing. b) The Same model after smoothing with the parameters of Fig. 8.

10 Converting a GLL model to spherical harmonics

Expanding a model onto spherical harmonics facilitates visualization and sharing a lot, but also allows for further analysis, like computing power spectra.

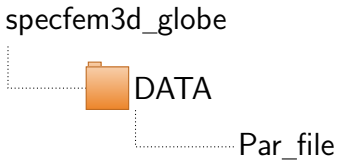
Converting the model from the GLL basis to a spherical harmonics basis is very challenging. However, SphGLLTools has many tools that allow you to do that.

10.1 Remeshing before the expansion

To expand the model onto spherical harmonics, we need a spherical mesh. Disabling the ellipticity and the internal discontinuities are enough to make the lower mantle and the transition zone spherical, but the crust and the upper mantle remain distorted due to the surface topography. Switching off the surface topography is possible but creates other problems as it moves the spectral elements along the radial direction, creating artifacts in the upper mantle. Besides, three-dimensional models have a doubling layer right beneath the crust that extends down to 120 km depth.

The proper way of dealing with the upper mantle is interpolating it to a perfectly spherical mesh. The right choice is STW105 (Kustowski et al., 2008), which is the 1D reference model for S362ANI. Choosing the correct reference model is crucial, as different models may have internal discontinuities at different depths.

To create the new mesh, go back to:

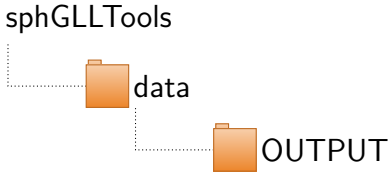


Change **MODEL** to 1D_ref and **TOPOGRAPHY** to **false**. You can also set **SAVE_MESH_FILES** as **false**, as the new model files will be created by the interpolation routine.

Then, run the following commands to recompile and create the new mesh:

```
$ make clean
$ make -j4 all
$ mpiexec -n 24 bin/xmeshfem3D
```

Copy the new topological files (proc000???_reg1_solver_data.bin) to:



In the "config.h" file, set **NEL2** to 8896 and **NG2** to 592913. As you can see, the STW105 mesh has the same number of spectral elements and points as the S362ANI mesh. Also, make sure of enabling the interpolation of V_P, V_S, and ρ.

Run the interpolation routine:

```
$ mpiexec -n 24 bin/gll2gll data/INPUT/ data/OUTPUT/
```

Taking some depth slices from the original model and the interpolated one to compare is always a good practice to make sure that everything looks fine.

10.2 Setting up the B-splines

The radial basis is made of cubic B-splines, whose knots must be carefully set to achieve good results. SphGLLTools provide examples of knots files that were fine-tunned for S362ANI, called "knots_Z?.dat". Each region has its own set of splines because B-splines cannot handle discontinuities properly. They strongly oscillate wherever a zero-order discontinuity is present (Fornberg and Flyer, 2007). That's why we must expand each region separately. Besides, the upper mantle requires more spherical harmonics coefficients to be properly represented than the transition zone and the lower mantle.

Each "knots_Z?.dat" file is composed of the number of knots, the degree of the B-splines, approximate the minimum, and the maximum radius of the region, and the knots.

All values are normalized by the Earth's radius. Therefore, 0.900 corresponds to a radius of 3440 km (2930 km depth). The code will use these minimum and maximum values to select the spectral elements of each region. After it selects all of them, it will calculate the exact radial boundaries of the zones.

The routine responsible for fine-tuning the B-splines is SETSPL. It requires the "knots_Z?.dat" files and the mean model of the parameter used for the fine-tuning. You may use "vs.dat", created by GLL2MEAN in section 7.

To check the knots for the three zones, use:

```
$ mpiexec -n 24 bin/setspl vs 2 data/OUTPUT/ .
$ mpiexec -n 24 bin/setspl vs 3 data/OUTPUT/ .
$ mpiexec -n 24 bin/setspl vs 4 data/OUTPUT/ .
```

For each region, SETSPL calculates a misfit between the mean model provided and the B-splines radial expansion. You can move the position of the knots or add/remove more splines to try to decrease the misfit. It's also useful to visualize the quality of the fitting. For that, SETSPL writes out the "vs_Z?_KC.dat" files.

To plot the B-splines, run:

```
$ ./utils/plot_B_splines.py vs
```

Fig. 10 shows the result:

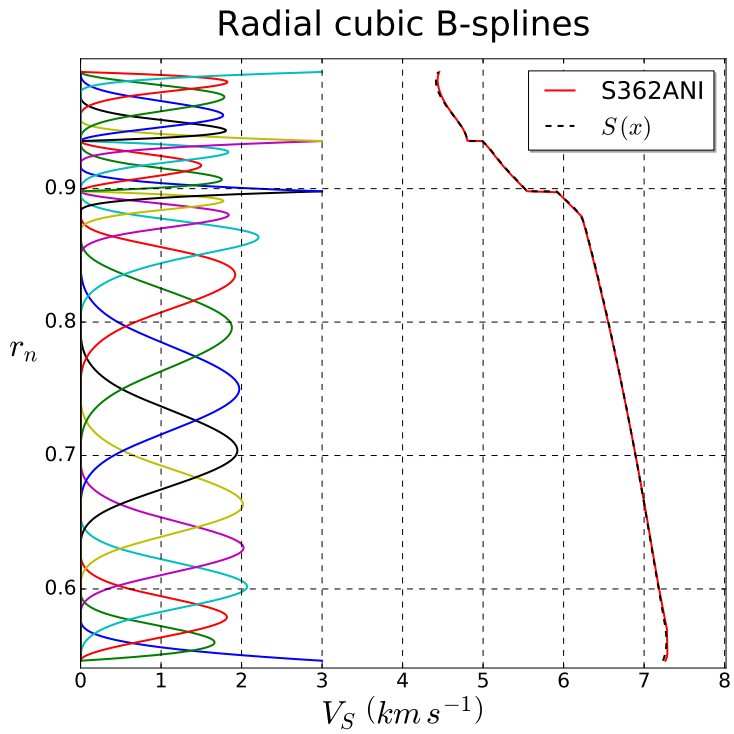


Figure 10: Twenty-five cubic *B-splines* used in the radial direction in the S362ANI expansion. The mean V_S as a function of depth are shown in red. The black dashed line shows $S\,(x)$, the reconstruction of the model from the *B-splines* and the coefficients.

In case you observe a reasonable agreement like the one in Fig. 10, you can proceed to the expansion of the coefficients.

10.3 Expanding onto spherical harmonics

The routine to carry out the spherical harmonics expansion is GLL2MNS. Its input files are also the "knots_Z?.dat".

To expand V_P, V_S and ρ for the three regions, run:

```
$ mpiexec -n 24 bin/gll2mns vp 25 2 data/OUTPUT/ .
$ mpiexec -n 24 bin/gll2mns vp 23 3 data/OUTPUT/ .
$ mpiexec -n 24 bin/gll2mns vp 20 4 data/OUTPUT/ .
$ mpiexec -n 24 bin/gll2mns vs 25 2 data/OUTPUT/ .
$ mpiexec -n 24 bin/gll2mns vs 23 3 data/OUTPUT/ .
$ mpiexec -n 24 bin/gll2mns vs 20 4 data/OUTPUT/ .
$ mpiexec -n 24 bin/gll2mns rho 25 2 data/OUTPUT/ .
$ mpiexec -n 24 bin/gll2mns rho 23 3 data/OUTPUT/ .
$ mpiexec -n 24 bin/gll2mns rho 20 4 data/OUTPUT/ .
```

After the expansion, the routine uses the coefficients to recreate the model and calculates the RMSD between the original model and the expanded version. By checking various degrees and comparing their RMSDs, you can have an idea of the optimal degree for the expansion. Be careful with the lower mantle as it is big and the intermediate part (1500 to 2000 km depth) requires a lower degree than its top and its bottom. Sometimes, the degree that gives you the minimum misfit overall will overshoot the intermediate lower mantle.

Using a degree higher than required create numerical artifacts in the expansion. The ideal degree can also vary from one model parameter to another. That's why the best way of checking your results is, again, comparing some depth slices of the expansion with the original model.

The output files are called "mns_Z?_???.dat".

In section 13, you will learn how to recreate the model from the coefficients.

11 Creating the crustal block model

The crust cannot be expanded onto spherical harmonics and B-splines, as it has many zero-order discontinuities that cannot be avoided. So, one way to represent the crust is by using a block model.

To create a block model from 5 km above the sea level down to 80 km depth, sampling the mesh every 0.5° in the horizontal and 1 km in the vertical, you need to run:

```
$ mpiexec -n 12 bin/gll2bk -5 80 0.5 1.0 data/INPUT/ .
```

Notice that you should create the model from the original mesh to represent the topography properly. The block model is the first region of the model. That's why the upper mantle is called Zone 2.

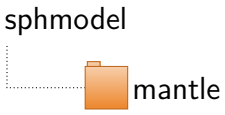
12 Setting up SphModel

The next step requires that you create the files with the accurate positions of the knots. These files are called "knots_Z?.dat" (notice the replacement of Zone? by Z? in the names).

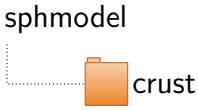
For that, you can use SET_KNOTS:

```
$ ./utils/set_knots.py vs
```

Now, copy the files "knots_Z?.dat" and "mns_Z?_???.dat" to the directory:



Copy the binary files of the block model "???.bin" to:



Enter into the "sphmodel" directory and compile the routines:

```
$ make -j4 all
```

Because SphModel is intended for sharing your spherical harmonics model, it does not use parallelization, as not all users are familiar with OpenMPI. Besides, all the large arrays use dynamic memory allocation, which prevents stack overflow in systems with a small stack size limit.

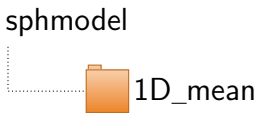
It is advisable to compute the mean models for SphModel using the expansion itself, avoiding the creation of artifacts in the radial direction when plotting the perturbations.

For that, use the BKMNS2MEAN:

```
$ ./bin/bkmns2mean vp -5 2891 0.5 1.0 20
$ ./bin/bkmns2mean vs -5 2891 0.5 1.0 20
$ ./bin/bkmns2mean rho -5 2891 0.5 1.0 20
```

Because that routine is serial, it can take several hours to compute each parameter. One way of speeding up the process is opening multiple tabs in your terminal and running one parameter on each one of them. That is a primitive kind of parallelization, but works.

Then, copy the files "vp.dat", "vs.dat" and "rho.dat" to:



That completes the expansion. Just write the relevant information to your model inside the "README.md" file, and you are ready to share it with other people. You can reduce the final size of your model package by compressing it:

```
$ tar -cvJf your_happy_model.tar.xz sphmodel
```

13 Advanced visualization with SphModel

The following sections will show how you can use SphModel to do a more sophisticated visualization of the model.

13.1 Creating and plotting depth slices

To create a depth slice of the expanded model up to degree 10 run:

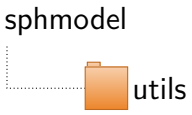
```
$ ./bkmns2ll vs 100 0.5 10
```

The default behavior, if you don't provide the last parameter, is to expand all the coefficients. Notice that limiting the degree of the expansion only makes sense for depth slices below 80 km.

Besides the portability, the possibility of choosing the maximum degree, and the fact that these routines don't require OpenMPI, another convenience is that you can directly compute the perturbations just by adding a "d" before the parameter code. Such as:

```
$ ./bkmns2ll dvs 100 0.5 10
```

To compute V_P/V_S ratios, transverse isotropy and the bulk sound speed, use the create_extra_*.py routines under:



To plot the depth slices, use the GMT scripts in the same directory. They are identical to the GMT scripts used in section 6.

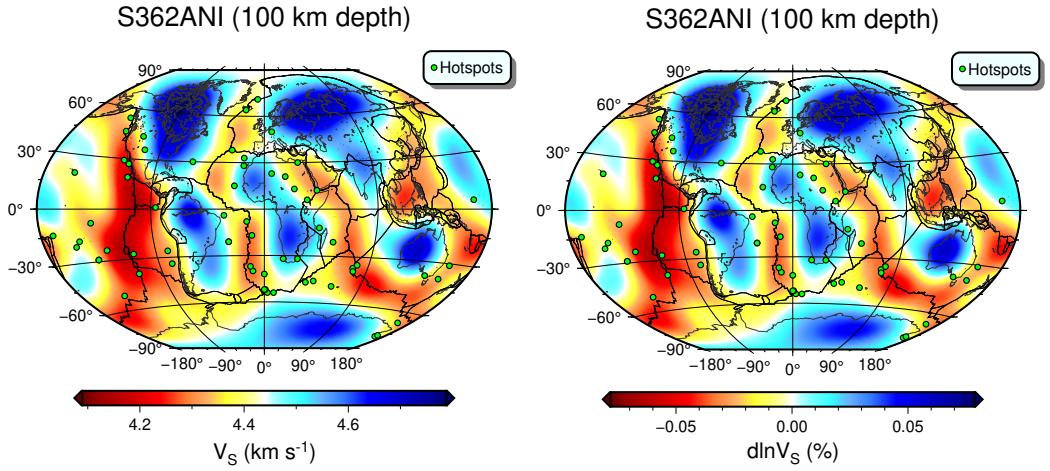
```
$ ./utils/plot_depth_slice.bash vs 100
$ ./utils/plot_depth_slice.bash dvs 100
```

See the results in Fig. 11:

13.2 Creating and plotting vertical cross-sections

To create a vertical cross-section expanding all the coefficients:

```
$ ./bin/bkmns2dd dvs 80 2891 −1.4 −25.1 23.7 51.3 0.1 2.0
```



(a) (b)

Figure 11: a) Depth slice for V_S at 100 km depth. b) Same depth slice, now showing the perturbations.

To plot it (see Fig. 12):

```
$ ./utils/plot_vertical_cross_section.bash dvs
```

13.3 Creating and plotting one-dimensional profiles

Two examples of one-dimensional profiles expanded up to degree 10 are:

```
$ ./bin/bkmns2pf vp 10 2891 0.8 −24.2 1.0 10
$ ./bin/bkmns2pf dvs 10 2891 0.8 −24.2 1.0 10
```

You can plot them with (13):

```
$ ./utils/plot_profile.bash vp
$ ./utils/plot_profile.bash dvs
```

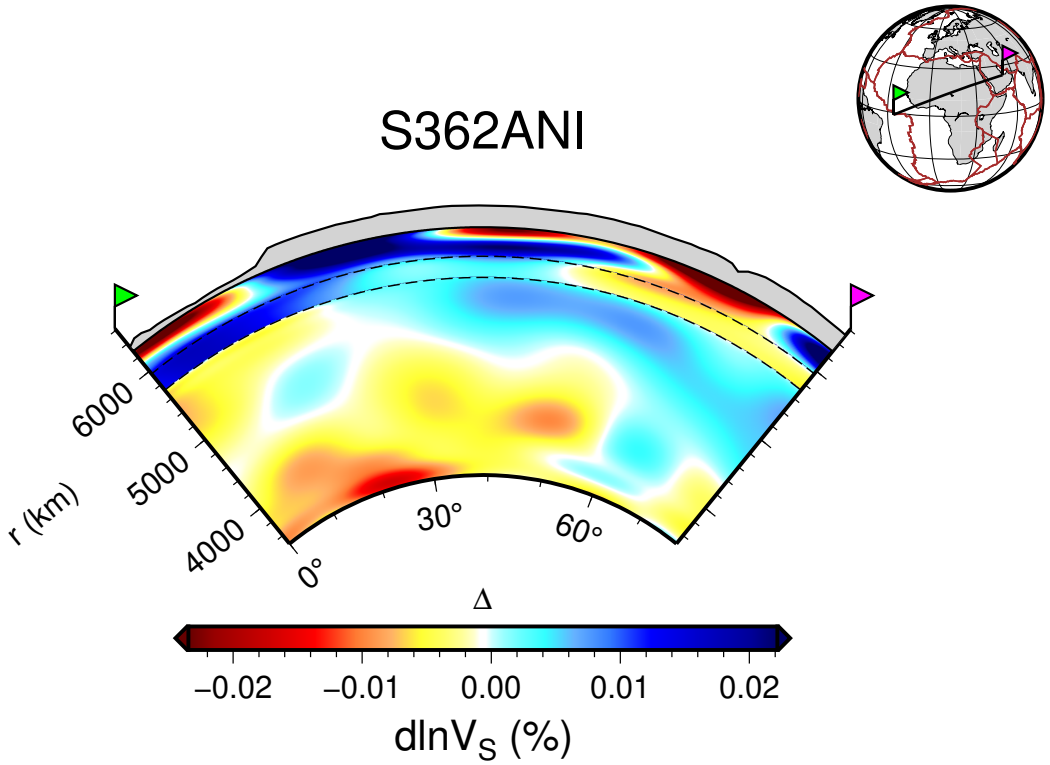


Figure 12: Vertical cross-section showing the V_S perturbations for a great circle crossing North Africa from 80 km depth to the CMB.

13.4 Power spectra

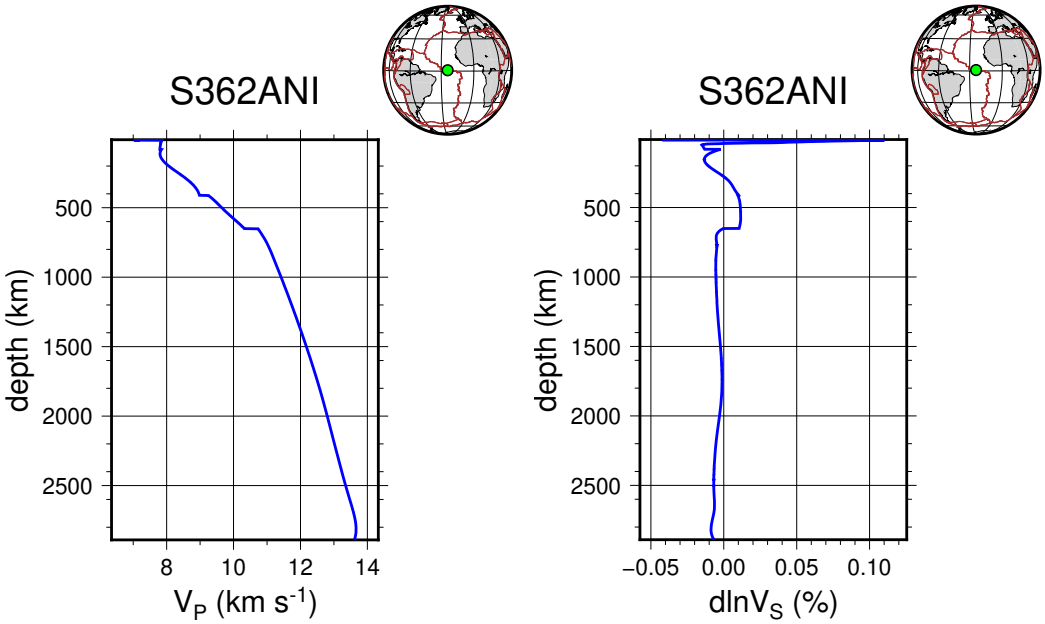
One more analysis that can be carried out using spherical harmonics is power spectra. They are useful for assessing the relative importance of each frequency in the model.

The routine POWSPEC simultaneously creates the one-dimensional power spectrum (power per degree) and the two-dimensional version (power per depth and degree).

Run:

```
1 $ ./bin/powspec vs 20
```

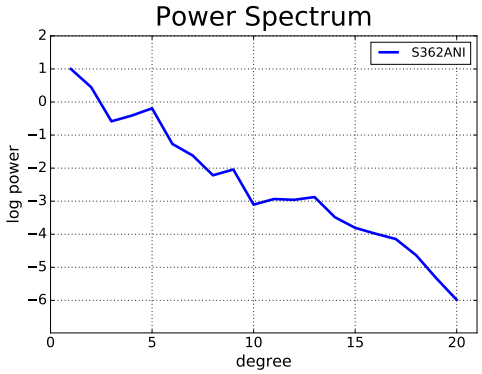
And plot the results using (see Fig. 14):



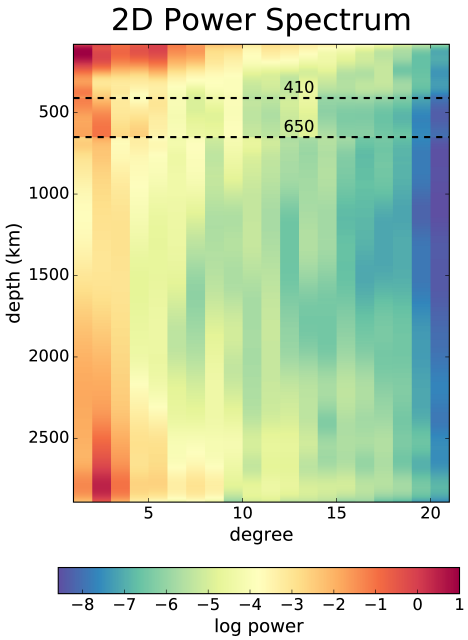
(a) (b)

Figure 13: a) One-dimensional profile for V_P from 10 km depth to the CMB in the Mid-Atlantic Ridge. b) The same profile, but now showing the V_S perturbations.

```
1 $ ./bin/powspec vs 20
```



(a)



(b)

Figure 14: a) One-dimensional power spectrum of V_S for S362ANI. b) Two-dimensional power spectrum of V_S for S362ANI.

List of Figures

References

Bassin, C., 2000. The current limits of resolution for surface wave tomography in north america. EOS Trans. AGU. 81: Fall Meet. Suppl., Abstract .

DeMets, C., Gordon, R.G., Argus, D., Stein, S., 1990. Current plate motions. Geophysical journal international 101, 425–478. URL: https://doi.org/10.1111/j.1365-246X.1990.tb06579.x.

Fornberg, B., Flyer, N., 2007. The gibbs phenomenon for radial basis functions. The Gibbs Phenomenon in Various Representations and Applications , 201–224URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.70.1832.

Komatitsch, D., Xie, Z., Bozdağ, E., Sales de Andrade, E., Peter, D., Liu, Q., Tromp, J., 2016. Anelastic sensitivity kernels with parsimonious storage for adjoint tomography and full waveform inversion. Geophysical Journal International 206, 1467–1478. URL: https://doi.org/10.1093/gji/ggw224.

Kustowski, B., Ekström, G., Dziewoński, A., 2008. Anisotropic shear-wave velocity structure of the earth's mantle: A global model. Journal of Geophysical Research: Solid Earth 113. URL: https://doi.org/10.1029/2007JB005169.

Wessel, P., Luis, J., Uieda, L., Scharroo, R., Wobbe, F., Smith, W., Tian, D., 2019. The generic mapping tools version 6. Geochemistry, Geophysics, Geosystems 20, 5556–5564. URL: https://doi.org/10.1029/2019GC008515.

14 License

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

   A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if

any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

   All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

   You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

   No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

   When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

   (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This

requirement modifies the requirement in section 4 to "keep intact all notices".

(c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

(d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

(a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

(b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as

you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

(c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

(d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

(e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a

consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking,

reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

(a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

(b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

(c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

(d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

(e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

(f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any

liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

    Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

    An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

    You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrange-

ment, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

    If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

    If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

    Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

    THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED

OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

    If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

End of Terms and Conditions

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it
does.>

Copyright (C) <textyear>  <name of author>

This program 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.

This program 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 this program. If not, see <https://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program>  Copyright (C) <year>  <name of author>

This program comes with ABSOLUTELY NO WARRANTY; for details type
‘show w’.
This is free software, and you are welcome to redistribute it
under certain conditions; type ‘show c’ for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if

necessary. For more information on this, and how to apply and follow the GNU GPL, see `https://www.gnu.org/licenses/`.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read `https://www.gnu.org/licenses/why-not-lgpl.html`.