

## SIO 227B – Practical 4

In this practical, we produce tomographic images using real data. We use phase data for fundamental mode surface waves to construct global phase velocity maps. Such maps are typically made for periods between 250 and 35s. Surface waves at a particular period sense structure over a wide depth range, so we require maps for a whole suite of periods to find out details of the seismic structure at depth (which we won't do in this practical). Even a map at a single period can give you a good idea of typical seismic properties of major geologic features, such as continental shields, old oceans, mid-oceanic ridges, large back-arc basins and, sometimes, even hotspots (though the ability to resolve the latter is quite controversial).

Traditionally, surface spherical harmonics have been chosen as the parameterization of a phase velocity map though spherical harmonics are somewhat impractical if we want to image small-scale features. Hence, more recently, as the data coverage has improved, researchers have been considering alternative parameterizations (e.g. pixels or local splines) to accommodate small-scale features in the maps. Recently, there has also been interest in improving the theoretical basis for the inversions so that sensitivity to off-great-circle path structure can be modelled. Here, we compare phase velocity maps made by using great-circle ray theory versus finite frequency kernels for a parameterization in terms of equal area pixels. All data files and scripts/programs can be found in `/home/guy/class227/newprac4.dir`.

The data we use are minor arc Rayleigh wave phase measurements made at periods of 150s (R150.raw1.sel), 100s (R100.raw1.sel), and 50s (R050.raw1.sel). These measurements were made by Goran Ekstrom's group at Harvard. The first thing that you have to do is to make the matrices which relate the phase measurements to the model (which is  $\delta c/c$  in percent). The script **runphs** makes the matrix for great circle propagation and looks like

```
/home/guy/class227/newprac4.dir/readphs <<!
blksw2
output – matrix
R050.raw1.sel
3.952
!
```

'blksw2' is the file that specifies the parameterization – this one is for 2 degree pixels at the equator – changing the first number in this file changes the coarseness of the parameterization – you can try blocks of 1,2,5 or 10 degrees in size. The programs will complain if you try a block size smaller than 1 degree. The final number in the script is the average phase velocity for this period – you should use 4.280, 4.080, and 3.952, for 150, 100, and 50 seconds respectively.

The script that makes the finite frequency kernel matrix is called **runphsff** and looks like

```
/home/guy/class227/newprac4.dir/phasematff <<!
blksw2
output – matrix
20 .25
3.952
R050.raw1.sel
!
```

Though the order is different, the input is the same as before except for the line `20 .25` which gives the center frequency of the calculation in mHz (20mHz= 50 sec) and the frequency band over which the measurement was made. Here, `.25` means the band is  $0.25 * 20 = 5\text{mHz}$  wide so the measurements were made from 17.5mHz to 22.5mHz. You can use 0.25 for all bands but don't forget to change the first number to reflect the frequency of your data file.

Once you have computed your matrices, you might want to know how well your data sample your model. We do this by computing a 'hit count' where we add 1 each time a pixel is sampled by a matrix element.

Not surprisingly, the hit count for the finite frequency matrix will be much larger than the hit count for the great circle matrix. Use script **runcount**:

```
/home/guy/class227/newprac4.dir/swraycount <<!
blksw2
1
matrix - file
1
!
domaprayct blkcnt.out
```

This script produces the file 'blkcnt.out' which is plotted by **domaprayct**. The only thing you change is the matrix file name. **domaprayct** is set up to read 2 degree models so you will have to change some numbers in it to handle other resolutions. The only critical line is the one that says

```
read 90 180 1
```

which would become

```
read 180 360 1
```

if a 1 degree parameterization were used. Your ability to resolve structure and to get precise models depends mainly on data coverage so take note of what your hit count map looks like.

The next step is to make the actual model. This is done using the script **runlsqr**:

```
/home/guy/class227/newprac4.dir/lsqrphs <<!
blksw2
1
matrix - file
$1
$2
100
!
dolsqmap_gv10 tomo.int3
```

\$1 is the first command line argument which is a multiplier for the errors (which are typically not well-known) – you should use 1 to start with. \$2 is the second command line argument and controls the amount of smoothing – try a number between 1 and 10. The goal is to choose these two numbers such that you end up with a reasonably smooth model which fits the data. This is where the art of inversion comes in! The final number in the script is the maximum number of iterations to be done – 100 is usually ok but you may need to make this larger if you are doing lightly smoothed, high resolution inversions. Some numbers are printed to the screen as the iterations proceed. The program uses convergence of the length of the model vector to decide when to stop since convergence of the fit to the data is usually quite rapid but the effect of smoothing will not be properly captured for several iterations. This script works with both great circle and finite frequency matrices. The final model is in a binary file called 'tomo.int3'. This is plotted by script **dolsqmap\_gv10**. There is a color table in this script which you can adjust if you so desire.

Some of the numbers printed by the previous script reflect the fit to the data but this actually includes how well you have satisfied the smoothing constraint. To see how well you actually fit the data, you should use the script **runchi**:

```
/home/guy/class227/newprac4.dir/chisqphs <<!
tomo.int3
1
matrix - file
$1
!
```

Here, the command line argument \$1 is the multiplier you used for the errors when you made the model. The program outputs  $\chi^2/N$ , the unnormalized variance reduction (data not divided by errors) – multiply

by 100 to get percent, and the normalized variance reduction. You want the variance reduction to be large and  $\chi^2/N$  should be close to 1.

The next step is to evaluate resolution. We do this by doing a checkerboard test though we actually use the shape of a spherical harmonic with a value of 1 when the spherical harmonic is positive and -1 when the spherical harmonic is negative. You are asked to input the  $(l, m)$  of the harmonic you want to recover. Remember that 360 divided by  $l$  roughly gives the wavelength of the harmonic or twice the size of an equivalent block (two blocks per wavelength). You should obviously choose  $l$  so that the implicit block size you are solving for is bigger than your model pixel size. To get something that looks like a checkerboard, you should choose  $m$  to be about half  $l$ . The script is called **runcheck**:

```
/home/guy/class227/newprac4.dir/lqrche2 <<!
blksw2
l m
1
matrix - file
$1
$2
100
!
dolsqmap_gv10 tomo.int3
```

The command line arguments are: \$1 is the multiplier for the errors you used in the actual inversion and \$2 is the smoothing you used in the actual inversion. The output checkerboard goes into 'tomo.int3' so make sure you don't overwrite your actual model.

Finally, we want to make an error map and we do this using the script **runlsqrerr** which does multiple inversions using data sets that differ by having random numbers with the statistical attributes of the data errors added to the true data vector:

```
/home/guy/class227/newprac4.dir/lqrerr <<!
blksw2
1
matrix - file
$1
$2
100
$3
!
```

The command line arguments are: \$1 is the error multiplier you used in the actual inversion, \$2 is the smoothing you used in the actual inversion, and \$3 is the number of realizations you actually want to do (50 to 100 is fine). This program produces a binary file called 'tomo.err3' which contains all your realizations. Next, you run program 'errmod' which asks for the 'tomo.err3' file and the number of realizations it contains. This produces two files: 'meanmodel' which is the mean model, and 'stdmodel' which is the error map. You can plot the mean model using **dolsqmap\_gv10** and the error map using **dolsqmap\_gvrr**.

There are a couple of other programs which allow you to do a comparison of models as a function of wavelength. 'blktsoph' takes a pixel model and produces the equivalent spherical harmonic expansion as an ascii file. The inputs are obvious except for the maximum spherical harmonic degree. Remember, there is no point in computing coefficients for harmonics shorter than allowed by the pixel size. For example, a 2 degree pixel model corresponds to a minimum wavelength of about 4 degrees or a maximum harmonic of about  $360/4 = 90$ . Program 'lcorr' takes two such spherical harmonic files and returns the correlation and amplitudes as a function of harmonic degree.

### c) Comparison of the two theories.

You may need quite different values of the smoothing and error scaling for the two inversions to get a valid comparison. The finite frequency inversion is naturally better constrained since the matrix is fuller so

you may have to use less smoothing to get models which are similar to the ray-theory inversion. There are two different points of comparison. First, if I have two models with similar spectral content, do I fit the data better with the finite frequency kernels or the ray theory kernels? Second, if I have two models with similar fit to the data, which of the two models has better resolved structure – and where is the better-resolved structure?

*c) Geophysical interpretation of the model.*

You might also like to interpret some of the features in your maps. Where do we find anomalously fast and anomalously slow regions? Do the patterns have anything to do with the continent/ocean distribution? Do we see a correlation with other major tectonic features? Which is the most striking feature that is associated with any geologic feature you know?

Rayleigh waves between 250 and 100s typically sense structure down to 500 km depth, while 100 to 50s Rayleigh waves reach to 250–150 km. The thickness of the crust is typically 38km under continents and 12.5 km under oceans (including an average of 4km of water). The seismic velocities and density for the crust are quite low compared to the mantle underneath. What does this mean for the features in your maps and the structure in the upper mantle?