Manual of JDSurfG

Nanqiao Du and Tingwei Yang
Version 4.0

April 29, 2024

Contents

1 Introduction

This document describes how to use `JDSurfG` package to perform 3-D joint inversion of shear wave velocity in crust and upper mantle by surface wave dispersion and gravity anomaly data. This package is consisted of three independent modules: generating gravity matrix in spherical coordinates, conducting direct surface wave tomographic inversion, and performing tomographic joint inversion of dispersion and gravity anomaly data. In joint inversion, we utilize direct surface wave tomography method [Fang et al., 2015] to compute 3-D (pseudo) sensitivity kernel of surface waves traveltime, and the adaptive Gauss-Legendre integration method [Li et al., 2011] is adopted to calculate the gravity forward matrix. Based on empirical relations between seismic velocity and density of rocks [Brocher, 2005], this package would be a useful tool to investigate 3-D shear wave structure in the crust and upper mantle. For more details, please refer to the paper [Du et al., 2021]

This package is written by `C++` and `Fortran`. `C++` is responsible for the main logic such as handling I/O, providing useful data structures and interfaces. And the `Fortran` part, derived from package DSurfTomo with several important modifications (parallel mode, group velocity ray-tracing and analytical derivatives), is used to compute surface wave traveltimes and corresponding 1D/3D sensitivity kernels.

I should note that this package was only tested by several people on similar computational conditions, so it might contain some bugs in it. I'd appreciate to receive bug reporting and modify some part of it when obtaining good suggestions. In the long term, I would add some new functions to this package,

2 Preliminaries

This package utilizes `C++` library Eigen to handle multi-dimensional arrays, So you need to install it in your own computer before compiling this package. Also, you should make sure that your `C++` compiler support `C++11` standards (`GCC` $\geq$ 4.8).

The package can be installed by using the CMake building system. You should make sure your `cmake` $\geq$ 3.1.0.

3 Installation

After you have downloaded the code from Github, you can compile it by using

```
cd JDSurfG
mkdir build; cd build;
cmake .. -DCXX=g++ -DFC=gfortran -DEIGEN_INC=/path/to/eigen
make -j4
```

Then you will find four executable files in the directory `bin/`:

```
mkmat DSurfTomo JointSG syngrav
```

That means you have finished installation.

4 Direct Surface Wave Tomography Module

This module need 3 (4 if checkerboard or other test is required) input files:

- `DSurfTomo.in`: contains information of input model, dispersion data and inversion parameters.
- `surfdataSC.dat`: dispersion data for each station pair
- `MOD`: Initial model
- `MOD.true`: True model (if additional tests are required)

There are some points to note before we precede to the format of each file:

1. Input model is homogeneous in latitudinal and longitudinal direction, but the grid size could vary in depth direction.
2. The input model should be slightly larger than the target one in order to perform `B-Spline` smoothing for forward computation. To be specific, if our target model is in region ($lat_0 \sim lat_1, lon_0 \sim lon_1, 0 \sim dep_1$), you should edit your input model at least in region ($lat_0 - dlat \sim lat_1 + dlat, lon_0 - dlon \sim lon_1 + dlon, 0 \sim dep_1 + dz$).
3. Stations should be far away enough from target model boundaries, or some warnings will be given when you running this package. This is to avoid that surface wave train travels along the models' boundaries.

Now there are the formats of every file below:

4.1 Parameter File

The parameter file is called `DSurfTomo.in` in this instruction. It is a self-explanatory file. Here is a template of this file below:

```
# DSurfTomo inversion Parameters

# # of iterations
NITERS = 16
ITER_CURRENT = 0 # current iteration


```

```
# minimum velocity, maximum velocity in km/s
MIN_VELOC = 2.0
MAX_VELOC = 5.0

# synthetic test
SYN_TEST = 1                # synthetic flag (0: real data,1: synthetic)
NOISE_LEVEL = 1.5           # noise level, std value of gaussian noise

# inverse method: LSMR = 0 CG = 1 LBFGS = 2
INV_METHOD = 2

# for LSMR-based inversion
SMOOTH = 15.0 # 2-nd Tikhonov regularization parameter
DAMP = 0.01  # damping parameter for LSMR
NTHREADS = 2 # # of threads used in LSMR solver

# for CG/LBFGS parameters
SMOOTH_IN_KM = 0 # smooth parameters are in km/rescale,
SIGMA_H = 0.25  # smoothing parameters in horizonal, KM or [0-1]
SIGMA_V = 0.25 # smoothing parameters in vertical
ITER_START = 0 # use information after this flag

# line search
MAX_REL_STEP = 0.04 # max relative variation for next model
```

When you run this program, it will automatically skip all the empty lines and comments (denoted by '#'). So you could modify this file (by adding your own comments) Now let's see the meaning of each parameter in the template file.

`NITERS` Number of iterations it need to run.

`ITER_CURRENT` Current iteration of the initial model. If set to 10, the initial model will be indexed as `mod_iter10` in the result directory. This option can be useful if you want more iterations.

`MIN/MAX_VELOC` Minimum and maximum velocity permitted. This option provide the prior constraints on the result shear wave velocity model.

`SYN_TEST` Do synthetic test. The `MOD.true` file should be provided in this case.

`NOISE_LEVEL` The Gaussian noise level added to synthetic data as observations. It is defined as:

$$d^i_{obs} = d^i_{syn} + noiselevel * N(0,1)$$

`INV_METHOD` Choose the inversion method you want. I strongly recommend LSMR (for small scale problems) and L-BFGS (large scale).

`SMOOTH` and `Damp` The second and first Tikhonov regularization coefficients used in inversion. The two parameters only work when `INV_METHOD` = 0 (LSMR mode).

`NTRHEADS` Number of threads used in LSMR solver. Usually 2-4 is enough.

`SMOOTH_IN_KM` Choose the smoothing operation in NLCG/L-BFGS inversion. Note that in NLCG/L-BFGS iterations, we don't apply Tikhonov regularization. Instead, we just convolve the search direction with a Gaussian smoothing function.

`SIGMA_H,SIGMA_V` Gaussian smoothing parameters in horizontal and vertical direction. The search direction wiil be smoothed as:

$$\bar{g}(\boldsymbol{x}) = \frac{1}{W(\boldsymbol{x})} \int_V g(\boldsymbol{x}) e^{-\frac{1}{2} x^T \Sigma^{-1} x} \mathrm{dV}$$

where $W(\boldsymbol{x})$ is the normalization factor:

$$W(\boldsymbol{x}) = \int_V e^{-\frac{1}{2} x^T \Sigma^{-1} x} \mathrm{dV}$$

and $\Sigma = Diag\left[\sigma_h^2, \sigma_h^2, \sigma_v^2\right]$. If the `SMOOTH_IN_KM` is set to 1, the $\sigma_{h,v}$ will be in km. And the $\boldsymbol{x}$ is the true coordinates. If not, $\boldsymbol{x}$ are the integer indices.

`ITER_START` Only use previous models after this iteration in NLCG/L-BFGS inversion. This value MUST be updated to the current iteration number if you're trying to update your model by trying a new set of smoothing parameters.

`MAX_REL_STEP` In NLCG/L-BFGS inversion, we usually do line search to find the best model in each iteration. Although we've implemented some methods to estimate the line search step size, we use this parameter to set the maximum relative variation of the new model to the previous one. Usually 0.03 or 0.04 is enough.

4.2 Initial Model and True Model File

`Please Note that the format of MOD.true in current version is different from previous versions!`

The initial and true model are called `MOD` and `MOD.true` respectively in this instruction. They are used for initial input model, and for synthetic test dataset (like in checkerboard test). The format of these two files are listed below:
The first line contains three integers which denote the number of grid points in latitude,longitude,and in depth respectively. The second line denotes coordinates of the origin point (NorthWest point), in degree. The third line denotes the grid interval in latitudinal and longitudinal direction, both in degree. Then

the next line are depths (in km) of each grid points, i.e. there would be `nz` numbers in this line. The subsequent lines of `MOD` are shear wave velocities. If we store the velocity with the format V(nz,nlon,nlat) (Row-major,where $V(0,0,0)$ is the shear-wave velocity of the `top NorthWest` point at `dep(0)` km, and V(nz-1,nlon-1,nlat-1) is the velocity of `bottom SouthEast` point at dep(nz-1) km), then it should be print to MOD as following:

```
//python
f = open("MOD","W")
for i in range(nz):
    for j in range(nlon):
        for k in range(nlat):
            f.write("%f "%(V[i,j,k]))
        f.write("\n")
```

4.3 Dispersion Data File

This dispersion data in `example/` is surfdataSC.dat, which contains dispersion (distance/traveltime) data. Here we list the first several lines:

```
1 19 # Rayleigh Phase
4 6 8 10 12 14 16 18 20 22 24. 26 28 30 32 34 36 38 40
1 19 # Rayleigh Group
4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
0 # Love Phase
0 # Love group
# 31.962600 108.646000 0 1 2 0
33.732800 105.764100 3.0840
34.342500 106.020600 3.1154
33.357400 104.991700 3.0484
33.356800 106.139500 3.0820
34.128300 107.817000 3.1250
33.229300 106.800200 3.0575
# 29.905000 107.232500 0 1 2 0
34.020000 102.060100 2.7532
```

This file contains three kinds of information. The data header, source or master station (lines begin with `#`) and receiver stations (without `#` in each line).

The data header contains the dispersion data description of wave type and period information. The first line contains the number of modes n (the first number) will be used for Rayleigh wave phase velocity. Followed this number are n numbers which indicate the number of period points this mode will use. Then there will be n lines contain the period information. For example, if you want to use the fundamental and first mode dispersion data with period [4,7.5,10]s and [3,4]s respectively, the header should be like:

2 3 2 # rayleigh phase
4 7.5 10
3 4

Then you can add information for other wave/velocity types. If no data of this wave type will be used, the line will contain only a number 0.

For a source or master station, the format of this line is:

```
# latitude longitude mode period-index wavetype velotype
```

- `mode` mode index. 0 for fundamental and 1 for the first order ...
- `wavetype`: the type of surface wave, for Rayleigh it is 2, and it is 1 for Love wave.
- `velotype`: velocity type, 0 for phase velocity and 1 for group velocity.
- `period-index`: The period index number, start from 1.

For example, if in our dataset, the dispersion periods for fundamental mode Rayleigh phase velocity is at periods 4s,5s,6s,7s, and this line contains information only at 5s, then the last 4 numbers are `0,2,2,0`.

Then the following lines contain all the receiver stations that you have successfully extracted dispersion curves at this period, this wave type and this velocity type. And the format of each line is:

```
latitude longitude dispersion
```

4.4 Run This Module

After you have prepared all the required files in a folder, then you could enter into this folder,and run this command in your shell:

```
1 ./DSurfTomo -h
```

Then you could type in all required files according to the order on the screen.

4.5 Output Files

This module will output one folder: `results/`. It contains all the intermediate models (`mod_iter*`) and data files (`res*.dat`) for each iteration. There may be some temporary binary files (`*.bin`) that contains the optimization history of NLCG/L-BFGS optimization. The format of model file is:

```
longitude latitude depth Vs,
```

And the format of traveltime file is (9 columns):

```
distance observation synthetic lat1 lon1 lat2 lon2 period wavetype.
```

5 Joint Inversion Module

This module needs 4-6 files in total:

- `JointSG.in`: contains information of input model, dispersion data and inversion parameters.
- `surfdataSC.dat`: surface wave dispersion data.
- `obsgrav.dat`: Gravity anomaly dataset.
- `gravmat.dat`: sensitivity kernel of gravity to density.
- `MOD`: Initial model.
- `MOD.true`: checkerboard model.
- `MOD.ref`: gravity reference model, used for compute gravity anomaly.

Here are the formats of each file.

5.1 Parameter File

There are only subtle differences between `JointSG.in` and `DSurfTomo.in` :

```
# synthetic test
NOISE_LEVEL_SWD = 1.5 # swd noise level
NOISE_LEVEL_GRAV = 5. # gravity noise level
WEIGHT_SWD   = 1.5
WEIGHT_GRAV  = 5.
RELATIVE_P   = 0.5   # relative factor

# remove averge value of gravity
REMOVE_AVG = 1
```

To know the meaning of each parameter, here I list the weighted cost function of joint inverse problem [Julià et al., 2000]:

$$L = \frac{p}{N_1\sigma_1^2}\sum_{i=1}^{N_1}(d_{1,i} - d_{1,i}^o)^2 + \frac{1-p}{N_2\sigma_2^2}\sum_{i=1}^{N_2}(d_{2,i} - d_{2,i}^o)^2 \tag{1}$$

where $\mathbf{d}_{1,2}$ is the data vector for each dataset. $N_{1,2}$ is the size of each dataset and $\sigma_{1,2}$ (The `WEIGTH_SWD` and `WEIGHT_GRAV` in `JointSG.in`) is the corresponding errors. p (`RELATIVE_P`) is a parameter in the range [0,1] to control the contribution of each dataset manually.

The additional parameter is `REMOVE_AVG`. When this integer is equal to 1, the program will automatically remove the average value of observed and synthetic gravity data. This option is applied to remove large-scale density anomalies in the deep mantle. For more details please refer to 5.4.

5.2 Gravity Data File

The format of this file is quite simple, just print all the gravity data to this file. Each line of it contains the longitude,latitude and gravity anomaly at this point.

Here I list the first 9 lines of this file to give you an example:

```
100.000000 35.000000 -224.206921
100.000000 34.950000 -224.110699
100.000000 34.900000 -241.774731
100.000000 34.850000 -240.245968
100.000000 34.800000 -234.187542
100.000000 34.750000 -246.670605
100.000000 34.700000 -238.575939
100.000000 34.650000 -241.357250
100.000000 34.600000 -260.059429
```

5.3 Gravity Matrix File

This file is generated by gravity module, please refer to chapter 6.2.

5.4 Gravity Reference Model File

Based on the definition of gravity anomaly, it reflects the density perturbs according to a normal ellipsoid. Thus we need to convert the absolute anomalies to local anomalies in our research area. Before inversion, we will remove the average value of observed anomalies. So in the real data inversion, there are 3 steps to compute relative anomalies:

- Compute density distribution of current S-wave model and reference model (through empirical relations).
- Compute density anomaly, then utilize gravity matrix to get gravity anomalies.
- Remove average value of the synthetic anomaly data.

This model could be a user self-defined model, and you could also use the default one (the average of initial model in every depth). Please remember, when you are tackling real data, to set the `REMOVE_AVG` to 1 in `JointSG.in` file.

5.5 Run This Module

After you have prepared all these required files in a directory, then you can enter into this folder, and run this command in your shell:

```
1 ./JointSG -h
```

Then you could type in all required files according to the order on the screen.

5.6 Output Files

This module will print some files in the directory: `resultsJ/`.

The format of `kernel/` is the same as the files in 4.5. In the folder `results/`, it contains models (`joint_mod_iter*`), surface wave travel time (`res_surf*.dat`) and gravity anomaly (`res_grav*.dat`) of each iteration. The format of gravity anomaly is:

```
lon lat observed-anomalies synthetic-anomalies
```

6 Gravity Module

This module requires two input files:

- `MOD` : only the header and depth information are needed.
- `obsgrav.dat`: contains coordinates of each obeservation points.

These formats have been illustrated in previous chapters.

6.1 Run This Module

After you prepare all these required files in a folder, then you could enter into this folder,and print in your shell:

```
1 ./mkmat -h
```

Then you could type in all required files according to the order on the screen.

6.2 Output Files

This module output only one file: `gravmat.bin`. This is the derivative matrix for gravity anomaly. It is a a Compressed Sparse Row Matrix (CSR format). If you want to know how to read this file, please refer to the code in `src/shared/csr_matrix.cpp`.

7 Advanced Options

7.1 OpenMP Mode

All the 4 programs can be executed on a single node with OpenMP support. To choose the number of threads used, you can set this environment variable before running any programs:

```
1 export OMP_NUM_THREADS=8
```

The maximum number of threads depends on your own platform. You can use the command :

```
cat /proc/cpuinfo |grep cores |wc -l
```

The maximum number of threads is `half` of the output number from the above command.

7.2 Empirical Relations

Empirical relations are utilized to connect density and velocity of rocks in our program. If you have better relations in your study area, you can change this relation in `src/SWD/exmpirical.cpp`

```
void empirical_relation (float vsz, float &vpz,float &rhoz)
{
    vpz = 0.9409 + 2.0947*vsz - 0.8206*pow(vsz,2)+
          0.2683*pow(vsz,3) - 0.0251*pow(vsz,4);
    rhoz = 1.6612 * vpz - 0.4721 * pow(vpz,2) +
          0.0671 * pow(vpz,3) - 0.0043 * pow(vpz,4) +
          0.000106 * pow(vpz,5);
}
```

And if you want to conduct joint inversion, please remember to change the derivative function in the same file.

7.3 Random Seed

In our program, to make sure the generated random number the same in each independent numerical experiments, we fix the random seed in `src/utils/gaussian.cpp`:

```
static default_random_engine e(11);
```

You could change 11 to other integer if required.

8 Tips and Tools

8.1 Warnings

In fact, `DSurfTomo` and `JointSG` are all dependent on two modules. First one is the forward computation of dispersion curves from 1-D model, and another one is the ray-tracing on 2-D spherical surface. There would be some possible errors in these two modules due to incompatible input model format, terrible initial model, and even the inappropriate location of your stations. To tackle possible problems, the two modules will print some information on the screen or in a output file to help you debug.

The first possible warning is `"WARNING:improper initial value in disper-no zero found"` on the screen. Then a `fort.66` file will be generated in the running directory. This error is due to a strange 1-D model that the dispersion module cannot find dispersion curves for this model. There are some possible reasons for

this warning. If you find this problem for the first iteration, this problem may from that the format of your initial model is different from the required one. You could check the model in file `fort.66` and compare that with your model. Another possible reason is that you take inappropriate Tikhonov regularization parameters (i.e. damping and smooth factors) which lead to abrupt change of previous model.

The second warning is `"Source lies outside bounds of model (lat,long)="`. This reason is caused by that the receivers are too close to the boundary of the current model that ray paths are along this boundary. If this warning happens, please enlarge your model to avoid this problem.

8.2 L-curve Analysis

Usually L-curve analysis is applied to determine the best suitable smoothing and damping factor for a inverse problem. In our numerical experiment, we find that the damping factor affects the final result smaller than smoothing factor. So we `recommend` you to keep the damping factor to a small value (such as 0.001) during your inversion, and only tune the smoothing factor.

In the folder `utils/`, we have prepared one python script `utils/lcurve.py` to help you tune the smoothing factor. The script will output the roughness of the inverted model $\|L(m - m_0)\|_2$ and $\|m - m_0\|_2$

8.3 Checkpoint Restart

Sometimes you want to start from one of the results (like mod_iter5.dat) instead of the initial model. We prepared a python script `utils/out2init.py`, you could run this script like this

```
python out2init.py mod_iter5.dat MOD05
```

Then this script will convert the output model to the format of the initial model. This script will be useful in joint inversion. And also, it may help you restart from some checkpoint if some issues happened.

References

T. M. Brocher. Empirical Relations between Elastic Wavespeeds and Density in the Earth's Crust. *Bulletin of the Seismological Society of America*, 95 (6):2081–2092, 12 2005. ISSN 0037-1106. doi: 10.1785/0120050077. URL https://doi.org/10.1785/0120050077.

N. Du, Z. Li, T. Hao, X. Xia, Y. Shi, and Y. Xu. Joint tomographic inversion of crustal structure beneath the eastern Tibetan Plateau with ambient noise and gravity data. *Geophysical Journal International*, 07 2021. ISSN 0956-540X. doi: 10.1093/gji/ggab299. URL https://doi.org/10.1093/gji/ggab299. ggab299.

H. Fang, H. Yao, H. Zhang, Y.-C. Huang, and R. D. van der Hilst. Direct inversion of surface wave dispersion for three-dimensional shallow crustal structure based on ray tracing: methodology and application. *Geophysical Journal International*, 201(3):1251–1263, 03 2015. ISSN 0956-540X. doi: 10.1093/gji/ggv080. URL https://doi.org/10.1093/gji/ggv080.

J. Julià, C. J. Ammon, R. B. Herrmann, and A. M. Correig. Joint inversion of receiver function and surface wave dispersion observations. *Geophysical Journal International*, 143(1):99–112, 2000. doi: https://doi.org/10.1046/j.1365-246x.2000.00217.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1046/j.1365-246x.2000.00217.x.

Z. Li, T. Hao, Y. Xu, and Y. Xu. An efficient and adaptive approach for modeling gravity effects in spherical coordinates. *Journal of Applied Geophysics*, 73(3):221–231, 2011.