

1/peak/data/direct/trace/rift/users/oliver/bio
1/peak/data/direct/RMF/mittles

GENERATING SEISMIC TOMOGRAPHIC MODELS WITH A SLIDING MODEL SPACE AND TSVD TRUNCATED SINGULAR VALUE DECOMPOSITION

The following is a description on the use of Matlab and C programs that produce a tomographic model derived from a sliding model space and the use of truncated singular value decomposition. These programs are presented in the order used under the following categories:

Produce G matrices for the linear inverse problem $d = Gm$

MakeG.csh
dWG.csh
tstarmatrix

Calculate singular values and vectors for G 's produced above.

GI.m
ReadG.m
AddEv2G.m
AddSt2G.m

Calculate the model vectors, m , for each G .

FindModel.m
ReadG.m
SolveInvSVD.m

Sequentially combine the model vectors to generate rough tomographic model.

CombineMods.m

Smooth tomographic model

SmoothMod.m

The basic input to these models are:

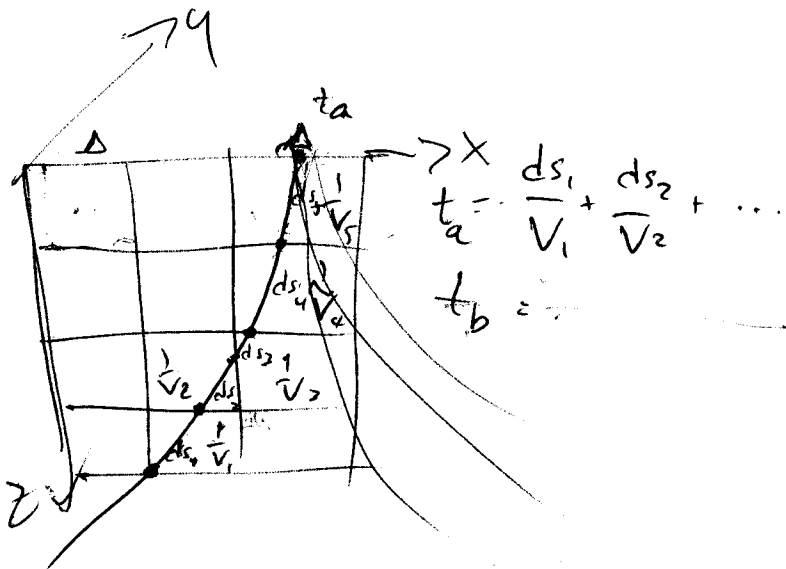
1. ray paths for each data point
2. the 1-D velocity model used to produce these ray paths
3. the values for the data points
4. the weights for the data points

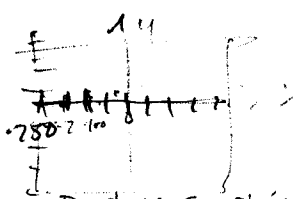
$$d = \begin{pmatrix} t_a \\ t_b \\ \vdots \\ t_z \end{pmatrix}$$

$$d = Gm$$

$$t = \int \frac{ds}{V}$$

$$m = \left[\frac{1}{V_1} \dots \frac{1}{V_n} \right]$$





Produce G matrices for the linear inverse problem $d = Gm$

MakeG.csh

dWG.csh

tstarmatrix

These programs create the G matrix, that is a matrix that contains travel times in each model bin for each data point or seismic ray. The columns correspond to the model bins and the rows correspond to the data points. A data point could be t^* for attenuation tomography or delay time for velocity tomography (in which case G is a matrix of distances traversed in each bin by the seismic ray rather than travel times). But since the structure (model bins, m) being solved for likely do not coincide with the actual structure, it is preferable to do one of two things. I can either slide the model bins and combine and average the multiple realizations of the model space or make the model bins correlated (with the application of a smoothing matrix) and much smaller than the structures to be found. The latter could require significant computer resources and may lead to significant smearing along regions of poor ray coverage. The former method is employed here.

The program **MakeG.csh** utilizes **dWG.csh** and **tstarmatrix** to generate G matrices for the inverse problem to be solved. Please read the following information for each program.

MakeG.csh - program to generate sliding G matrices

MakeG.csh -help

use this command to get this message.

Loops over the x , y , and z offsets. In each iteration the **BinParam.dat** file is created which is used in the call to **dWG.csh**.

TO USE:

Edit this file to specify the x , y , and z offsets as well as the starting x , y , and z positions. The units are the same as the ray path files sent to **tstarmatrix** from **dWG.csh**. Also change the output directory with the variable **curFile**.

dWG.csh - simple shell script to call **tstarmatrix**

Each row in this file is a call to **tstarmatrix** and represents a single data point. See **tstarmatrix** below on the format for the call to **tstarmatrix**

tstarmatrix - c program to convert a ray path into travel times within model bins assuming a given 1D velocity profile. Output is in binary format in a file named **dWG.dat**.

TO USE:

/perl/data/dverb/temp/rift/users/dverb/srdpays

tstarmatrix RayFile data weight VFile BPFFile

RayFile contains the ray path and must be of the form (for each line in the file):

\dot{x} y z

data is the value of the data point for this particular ray path. Since this value may change often for a given ray path, it is usually ignored in subsequent programs.

weight is the value of the weight for the particular data point. For the same reasons as for the data point, it is usually ignored.

VFile - 1D velocity structure of the form:

z velocity

BPFFile - Bin parameter file of the form:

InitialX XIncrement NumberXBins

InitialY YIncrement NumberYBins

InitialZ ZIncrement NumberZBins

Creating rays:
CreateRay.m

→ you'll need slowness for each datapoint
↓
Arrival.m

Calculate singular values and vectors for G 's produced above.

GI.m

ReadG.m
AddEv2G.m
AddSt2G.m

After the G matrices have been created using the above procedure, The singular values and vectors are created from those G matrices. The matlab program **GI.m** is used which includes calls to **ReadG.m**, **AddEv2G.m**, and **AddSt2G.m**. I will not discuss the use of the latter three programs as an understanding of their operation is not necessary to understand the operation of **GI.m**. Type 'help MATLABPROGRAM' at the matlab command prompt to get helpful information about these programs.

GI.m - program to calculate singular values and vectors for G matrices in a format as produced by **tstarmatrix**.

TO USE:

GI(W, tr, st, xoff, yoff, zoff, flBASE1, flBASE2)

tr - number of traces for each event. Used to modify G matrix to include solution for event static.

st - vector containing station number for each event. Used to modify G matrix to include solution for station static. *Set to zero for now*

Substitute 0 for the above parameters to signify no use of those parameters, e.g. tr set to 0 will not include a solution for event static.

W - weights given to each data point. *$\frac{1}{\sigma}$ for each data point*
xoff - how the x bins are offset
yoff - how the y bins are offset
zoff - how the z bins are offset

Ex:

xoff = 0:5:45;
yoff = 0:5:45;
zoff = 0:5:45;

flBASE1 - name of the file base containing G matrices. Remaining file name will be generated based on xoff, yoff, zoff, and the convention adopted by **MakeG.csh**.

flBASE2 - name of the file base containing resulting singular values and vectors. Remaining file name will be generated as for flBASE1.

Ex: flBASE1 = '/peak/data/dWG_';

z =

SVG_';

Calculate the model vectors, m , for each G .

FindModel.m

ReadG.m

SolveInvSVD.m

We are now ready to perform the inversion using the singular values and vectors generated in the previous step and the data values and weights determined separately using whatever technique you desire. A model is found for each G matrix and in the following step, these models are combined and smoothed.

FindModel.m - calculates models for specified G matrices.

st=0 for now

TO USE:

(st)

FindModel(Data, W, tr, StatNum, xoff, yoff, zoff, flBASE1, flBASE2, flBASE3, tp)

Data - vector of data points that are the result of the integrated effect of the model space.

W - vector of weights for each data point.

tr - number of traces for each event. Used to modify G matrix to include solution for event static. *Substitute 0 for tr to signify no use of that parameter, e.g. tr set to 0 will not include a solution for event static.*

xoff - how the x bins are offset

yoff - how the y bins are offset

zoff - how the z bins are offset

Ex:

xoff = 0:5:45;

yoff = 0:5:45;

zoff = 0:5:45;

flBASE1 - name of the file base containing G matrices. Remaining file name will be generated based on xoff, yoff, zoff, and the convention adopted by MakeG.csh.

flBASE2 - name of the file base containing singular values and vectors. Remaining file name will be generated as for flBASE1.

flBASE3 - name of the file base containing resulting models. Remaining file name will be generated as for flBASE1.

Ex: flBASE1 = '/peak/data/dWG_';

3

"

MOD_';

Sequentially combine the model vectors to generate rough tomographic model.

CombineMods.m

Once the models have been created, they must be combined. CombineMods sequentially combines the models to create a single high resolution model.

CombineMods.m - combines models into single high resolution model.

TO USE:

```
[Model, Res, Fold, Statics, X, Y, Z] =  
CombineMods(flBASE3)
```

CombineMod returns the following variables:

Model - the combined model

Res - the combined resolution matrix.

Fold - the combined fold.

Statics - the event and station means and standard deviations.

X, Y, and Z - vectors for the X, Y, and Z directions of Model

CombineMod uses the following input:

flBASE3 - name of the file base containing resulting models. Remaining file name will be generated as for flBASE1.

Ex: flBASE3 = '/peak/data/~~MOD~~_';

Smooth tomographic model

SmoothMod.m

The model resulting from CombineMods.m is typically very rough. Since only features on the order of the bin size can be resolved, the high frequency roughness is not a part of the model and should be smoothed. SmoothMod.m convolves a three dimensional object with the model using the matlab function convn.m. The object should have unit magnitude and could have an equal distribution over a cube or rectangle or have a three dimensional cosine distribution.

SmoothMod - Smooths the input 3D model, *Mod*, by creating and convolving with a filter of size *smx* by *smy* by *smz*.

TO USE:

smMod = SmoothMod(*Mod*, *smx*, *smy*, *smz*, *tp*, *f*)

CombineMod returns the following variables:

smMod - resulting smoothed model.

CombineMod uses the following input:

Mod - input, rough, model

Define filter

smx - number of bins in the X direction

smy - number of bins in the X direction

smz - number of bins in the X direction

tp - type of filter

1 - unit ~~rectangle~~ *Cube*

2 - 3D unit cosine

f - specify if a filter has been created outside of the program, *otherwise zero*.