

# **Transdimensional Hierarchical Bayesian 2D (THB2D) Seismic Traveltime Inversion User Guide**

Developed and written by  
Berit Hudson-Rasmussen<sup>1,2</sup>

Scott Burdick<sup>3</sup>

Vedran Lekic<sup>1</sup>

Mong-Han Huang<sup>1</sup>

<sup>1</sup>University of Maryland, College Park, MD

<sup>2</sup>Carleton College, MN

<sup>3</sup>Wayne State University, MI

March, 2021

# Table of Contents

<b>0. Citation and Naming Conventions</b>	<b>2</b>
<b>1. Introduction</b>	<b>3</b>
<b>2. System requirement</b>	<b>5</b>
<b>3. Quick start (running the test example)</b>	<b>6</b>
<b>4. Data preparation</b>	<b>7</b>
<b>5. Model Setup and Model Parameters</b>	<b>8</b>
5.1 Iteration and Burn-in Constraints	8
5.2 Initial Model Setup	8
5.3 Priors	9
5.4 Proposal Sigmas	10
<b>6. Running THB2D MCMC and how to continue</b>	<b>13</b>
6.1 Time Issues for running the program	13
6.2 Continuation code	13
6.3 How to Update the Posterior Distribution with a new Burn-in Value	14
6.4 Vertical Profile	14
<b>7. Explanation of Figures</b>	<b>15</b>
<b>8. Useful tips</b>	<b>27</b>
<b>9. References</b>	<b>28</b>

# 0. Citation and Naming Conventions

Please cite this user guide as:

Huang, M.-H., Hudson-Rasmussen, B., Burdick, S., Lekic, V., Nelson, M.D., Fauria, K.E., and Schmerr, N., (2021), Bayesian seismic refraction inversion for critical zone science and near-surface applications, *submitted to Geochem. Geophys. Geosys.*

This user guide is the supplementary material of *Huang et al.*, *submitted to Geochem. Geophys. Geosystems.*

Note: For this user guide, folder names are indicated by underlined text, file names by *italics*, and variables by **bold** text.

# 1. Introduction

THB inversion utilizes a reversible-jump Markov-chain Monte Carlo (rjMCMC) algorithm to create a set of velocity models that best describe observed seismic traveltimes (Bodin *et al.*, 2012; Bodin *et al.*, 2014; Burdick and Lekic, 2017). The result of THB inversion is a collection of possible solutions, whose characteristics appear in proportion to their likelihood given the data. This makes it straightforward to quantify the range of plausible solutions and the uncertainty associated with the velocity profiles. Using this method, one can analyze the posterior probability distribution on every model parameter, rather than solely relying on one single best-fit model.

As described in the main text, rjMCMC involves the following steps. First, we create a new velocity model by randomly selecting a parameter to vary. The initial velocity model is set up by user-defined parameter distributions called priors. These distributions should be based on background knowledge and not on the analysis of the dataset being inverted. Velocity structures are parameterized from location and velocity of each control point. More specifically, a proposed velocity structure is interpolated linearly using Delaunay triangles of from the location and velocity of all control points (**Figure 5-1**). Once the initial model is generated, one of six functions is selected to vary the parameters and create a new model: (1) change a control point velocity, (2) add a control point, (3) remove a control point, (4) move a control point, (5) exchange velocities between two control points, and (6) change the noise hyperparameter.

For the change steps, the rjMCMC algorithm proposes parameter values by randomly sampling from normal distributions with user-defined standard deviations, called proposal sigmas. Three distinct proposal sigmas are used when the algorithm is trying to: (1) change control point velocity, (2) move a control point, and (3) change the noise hyperparameter. More discussion can be found in **Supplementary Text S3** of Huang *et al.*, submitted to *Geochem. Geophys. Geosys.*

For each proposed model, traveltimes are predicted using the fast-marching method and then computes the probability that the proposed traveltimes can explain the observed traveltimes (Bodin *et al.*, 2012; Bodin *et al.*, 2014; Peyre, 2020). The program decides to accept or reject each proposed model based on its fit to the data and the prior information, according to the acceptance probabilities also defined in the **Supplementary Text S3** of Huang *et al.*, submitted to *Geochem. Geophys. Geosys.*

At the start of the rjMCMC, proposed models are correlated with the starting model and do not, generally, provide good fit to the data. This is called the burn-in period and models sampled during it do not represent the posterior distribution accurately. Therefore models produced before a user-defined burn-in period are discarded. After

many iterations, model misfit will stabilize, at which point the model solutions can be saved to the ensemble. To reduce correlation between successive models saved to the ensemble, we only save every Nth model (default value is 200), where N is a user-defined integer called “thinning”. Readers interested in more details about the inversion method can refer to *Bodin et al. (2012)* and *Burdick and Lekic (2017)*.

This tutorial provides an overview of the THB algorithm, detailed explanation of the model parameters and setup, execution of the program with an example, and figures produced by the program.

## 2. System requirement

This program is written in Matlab and can be run in multiple platforms including Linux, Mac, and Windows systems. We have only tested this program in Matlab 2017b and later versions, but it should work as long as with parallel computation capability. The “fast marching toolbox” is not part of the program but is used for calculating P-wave arrival times. This toolbox can be downloaded here (<https://www.mathworks.com/matlabcentral/fileexchange/6110-toolbox-fast-marching>) (Peyre, 2020). Please make sure you locate the path to the fast marching toolbox in line 30 of the *run\_model.m* script.

Alternatively, you can use the Accurate Fast marching toolbox (<https://www.mathworks.com/matlabcentral/fileexchange/24531-accurate-fast-marching>) (Kroon, 2021). Please make sure you locate the path to the toolbox in *run\_model.m* (e.g. below line 30). You also need to comment out line 416 and uncomment line 419 of *run\_model.m*.

We use the crameri perceptually uniform scientific colormaps in plotting. Please make sure to download this toolbox here (<https://www.mathworks.com/matlabcentral/fileexchange/68546-cramer-perceptually-uniform-scientific-colormaps>) (Cramer, 2019) and locate the path to the Cramer toolbox in line 27 of the *PlotEnsembleRef2D\_verD.m* script.

We use the flow cytometry data reader and visualization toolbox to generate the scatter. Please make sure to download the toolbox here ([https://www.mathworks.com/matlabcentral/fileexchange/8430-flow-cytometry-data-reader-and-visualization?s\\_tid=srchtitle](https://www.mathworks.com/matlabcentral/fileexchange/8430-flow-cytometry-data-reader-and-visualization?s_tid=srchtitle)) (Henson, 2020) and locate the path this toolbox in line 28 of the *PlotEnsembleRef2D\_verD.m* script.

### 3. Quick start (running the test example)

In this section we will run an example test with given station location (elevation and along-profile position) and travel time measurements. The purpose of this section is to make sure the THB code can run on your computer. The data preparation and explanation are described further in Sections 3 and 4, respectively.

1. Download the entire THB file. Do not change the folder names or locations. You should see a matcodes folder that includes the subroutine scripts, a data folder with the example data, and a models folder which will be initially empty.
2. In the data folder, *Input\_test.mat* is the name of the input file, and *elevation\_20191216.txt* is the name of the surface elevation file.
3. Open the *run\_model.m* script and insert the file names above into the script in the **input\_file** and **elevation\_file** spots, lines 41 and 42, respectively.

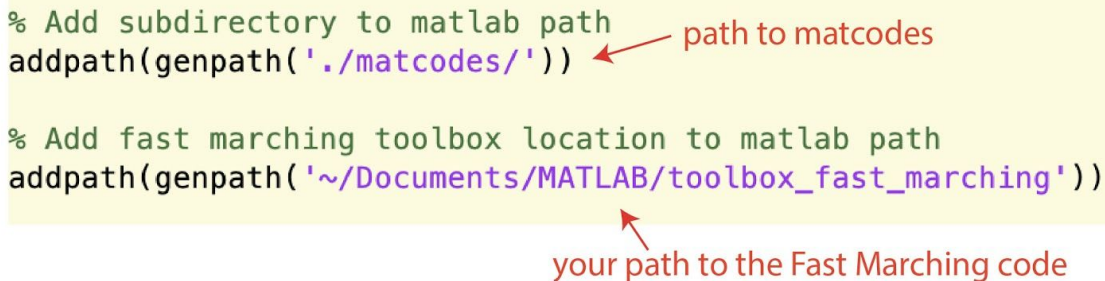
```
%% Model setup and Priors
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Input files (p-wave arrival times and surface topography)
input_file = 'data/Input_test.mat'; % order in format: arrival time (sec), shot loc
elevation_file = 'data/elevation_20191216.txt'; % order in format: horizontal distan
% Load Master data file
```



4. Check that the correct location path is linked for the fast marching toolbox and the functions within the matcodes folder (lines 26 and 29).

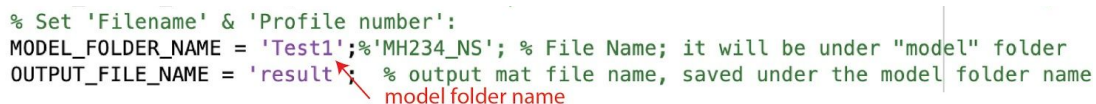
```
% Add subdirectory to matlab path
addpath(genpath('./matcodes/'))

% Add fast marching toolbox location to matlab path
addpath(genpath('~\Documents\MATLAB\toolbox_fast_marching'))
```

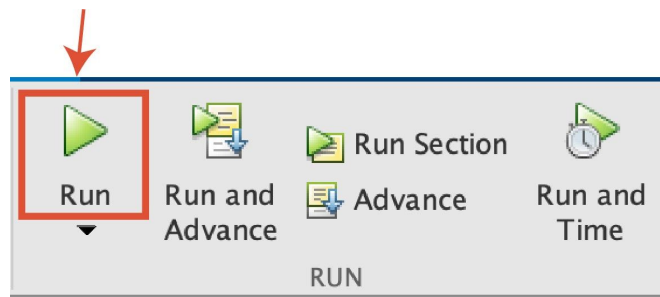


5. In line 32, choose a name for the model folder that will be created to store the result (saved in the models folder; lines 51 and 52).

```
% Set 'Filename' & 'Profile number':
MODEL_FOLDER_NAME = 'Test1'; %MH234_NS'; % File Name; it will be under "model" folder
OUTPUT_FILE_NAME = 'result'; % output mat file name, saved under the model folder name
```



6. Run the example with the parameters listed in column 3 of Table 1 (section 5), with the exception of resolution and iterations. For **delta\_X** and **delta\_Z** use 3 meters and for NumIter use 1000. This will allow the program to run very quickly, producing a rough solution but allowing you to check whether the program is running properly.
- Note: Running a model of this size at 1 m resolution for 200,000 iterations takes about 8 hours in a workstation with a 18-core Intel Xeon Gold 6140 processor, and more than 1 million iterations were required for the chains to converge.



7. Check for the following files under models/<your\_model\_name> after the inversion has finished:
- Figures
  - *Result.mat*

Refer to later sections for more detail, Table 5.1

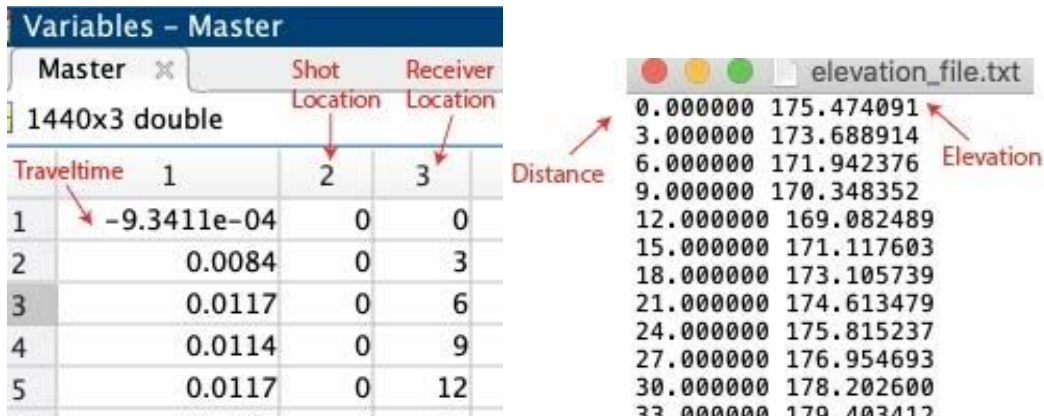


## 4. Data preparation

Two input files must be prepared before running the inversion: A file containing traveltimes data and station locations (assumed to be at the surface), and an elevation file specifying the topography.

**input\_file.** Data from the seismic survey should be formatted into three columns: column one for traveltimes (in seconds), column 2 for shot location (in meters), and column three for the location of each receiver (in meters). Data must then be named as **Master** and saved as a “.mat” file under the data folder.

**elevation\_file.** The elevation file should be formatted as a two-column “.txt” file with the first column for horizontal distance and the second column for the elevation corresponding to each distance point. This should also be saved in the data folder.



The image shows two parts: a MATLAB Variables window on the left and a text file named 'elevation\_file.txt' on the right. The Variables window shows a variable 'Master' of type '1440x3 double'. It has three columns: 'Traveltimes' (column 1), 'Shot Location' (column 2), and 'Receiver Location' (column 3). The first five rows of data are shown. The 'elevation\_file.txt' file contains two columns: 'Distance' and 'Elevation'. The first five rows of data are shown.

Traveltimes	Shot Location	Receiver Location
-9.3411e-04	0	0
0.0084	0	3
0.0117	0	6
0.0114	0	9
0.0117	0	12

Distance	Elevation
0.000000	175.474091
3.000000	173.688914
6.000000	171.942376
9.000000	170.348352
12.000000	169.082489
15.000000	171.117603
18.000000	173.105739
21.000000	174.613479
24.000000	175.815237
27.000000	176.954693
30.000000	178.202600
33.000000	179.403417

Add the name of your travel time file to the path for **input\_file** and the name of your elevation file to the path for **elevation\_file** in the main code.

A folder must also be created to store the final results. This subfolder will be created when you run the program and will be located in the models folder. Name your output folder using the **MODEL\_FOLDER\_NAME** variable.

## 5. Model Setup and Model Parameters

### 5.1 Iteration and Burn-in Constraints

**NumChain. (line 76)** Using multiple chains can allow a faster and more complete mapping of the posterior distributions on the model parameters. However, a multi-core CPU is required to run multiple chains at a practical speed. We recommend setting the number of chains to the number of CPU threads minus one.

**maxiter. (line 77)** We recommend running the inversion so that a sufficient number of models can be saved to the ensemble after the misfit of all chains has converged (i.e. after “burn-in”). Assessing convergence is not always straightforward, but can be done by ensuring that the mean and standard deviation of the misfit is indistinguishable among chains. We refer the user to Appendix B of *Gao and Lekic (2018)* for more detail. The number of iterations necessary therefore depends on the size of your model and your specific data. With the datasets we have analyzed, at least 100,000 iterations are necessary.

**datasav. (line 79)** This value determines how often a model is saved to the final ensemble. A higher value means a model will be added after a larger number of iterations, so that there will be fewer models in the final ensemble. This will take up less space on your computer, but having too few samples can make estimates of the posterior distribution from the ensemble unreliable. In practice, **datasav** should be set no larger than the expected number of steps it takes the rjMCMC to produce uncorrelated models.

**burn\_percent. (line 80)** The burn-in value determines at what point the rjMCMC converges and the program starts saving models to the ensemble. Here, it is defined as a percentage of **maxiter**.

### 5.2 Initial Model Setup

Here you will specify the depth and grid size of your model, assign the number of layers and a velocity and depth for each layer, and determine the amount of lateral variation allowed. Fig. 5.1 shows an example of the setup of the velocity model. Note, this is only for setting the initial model. The final model will be generated by sufficient amounts of iterations after burn-in and is unlikely to be the same as the initial setting.

**maxZ, minZ. (lines 56 and 57)** Z represents the depth range of your model domain. Set **minZ** to zero and determine a value for **maxZ** based on your seismic survey. **maxZ** should be at least as half the length of a survey line. You can make the model as deep as you want, but deeper regions may not be sampled by the seismic rays. Larger domains will also take much longer to run.

**delta\_X, delta\_Z. (lines 60 and 61)** These grid size parameters determine the resolution of the grid on which the FMM traveltimes predictions are computed. **delta\_X** is the horizontal distance between grid points, and **delta\_Z** is the vertical distance, both in meters. Smaller values will produce higher resolution results. The grid size must be smaller than the distance of your geophone spacing, and small enough to capture the relevant scale of heterogeneity. Note that small grid sizes will produce grids with a larger number of points, slowing down the FMM computation of traveltimes.

**\*\* The initial model parameters listed below should not significantly influence the final model, with sufficient iterations after burn-in \*\***

**v0. (line 64)** You will assign velocities to the layers defining your initial model, and format them in a bracketed list from lowest to highest velocity. The number of velocities you use determines the number of layers in the initial model. You must use a minimum of three layers.

**zz0. (line 65)** This depth vector must be exactly the same length as **v0**. Assign a depth to the top of each layer defined above, with the first value as 0.

**Ncol0. (line 66)** The number of hingelines determines how many control points will be generated in the initial model. The structure of the initial model is that 4 control points will occupy the 4 corners of the model, and the rest of the control points are determined by the number of layers and hingelines, as shown in **Figure 5-1**.

## 5.3 Priors

Prior parameters determine a range of values for each term that will be varied in the inversion process.

**prior.v1D. (line 70)** Set a minimum and maximum bound defining the prior uniform distribution on the velocities, based on knowledge of the region, previous studies, and geological information. It should not be informed by any analysis of the data being inverted.

**prior.h1D. (line 71)** The range of depths a layer can exist at should be set to the full extent of your model, so zero to **maxZ**.

**prior.Nuclei. (line 72)** The maximum number of control points allowed in the model. This value must be greater than four and should be set to a sufficiently large number that its choice does not affect the retrieved model. This can be validated after inversion by looking at the histogram of the number of layers in the ensemble (i.e. the posterior distribution on the number of layers; Figure 7-11) and ensuring that the choice of **prior.Nlay** does not strongly truncate the distribution.

**prior.noise. (line 73)** Set a minimum and maximum bound defining the prior uniform distribution on the log of the 1-sigma uncertainties on the traveltimes (i.e. noise hyper-parameter). These bounds should be based on your knowledge of seismology and past experience in P-wave picking uncertainty, but should not be informed by any analysis of the data being inverted.

## 5.4 Proposal Sigmas

Proposal sigma values define the zero-mean normal distributions from which changes to a model are proposed. The proposal sigma values should be tuned so proposed models are accepted sufficiently frequently, which is necessary for efficient convergence and search of the model parameter space. See discussion in *Brooks et al.* (2003).

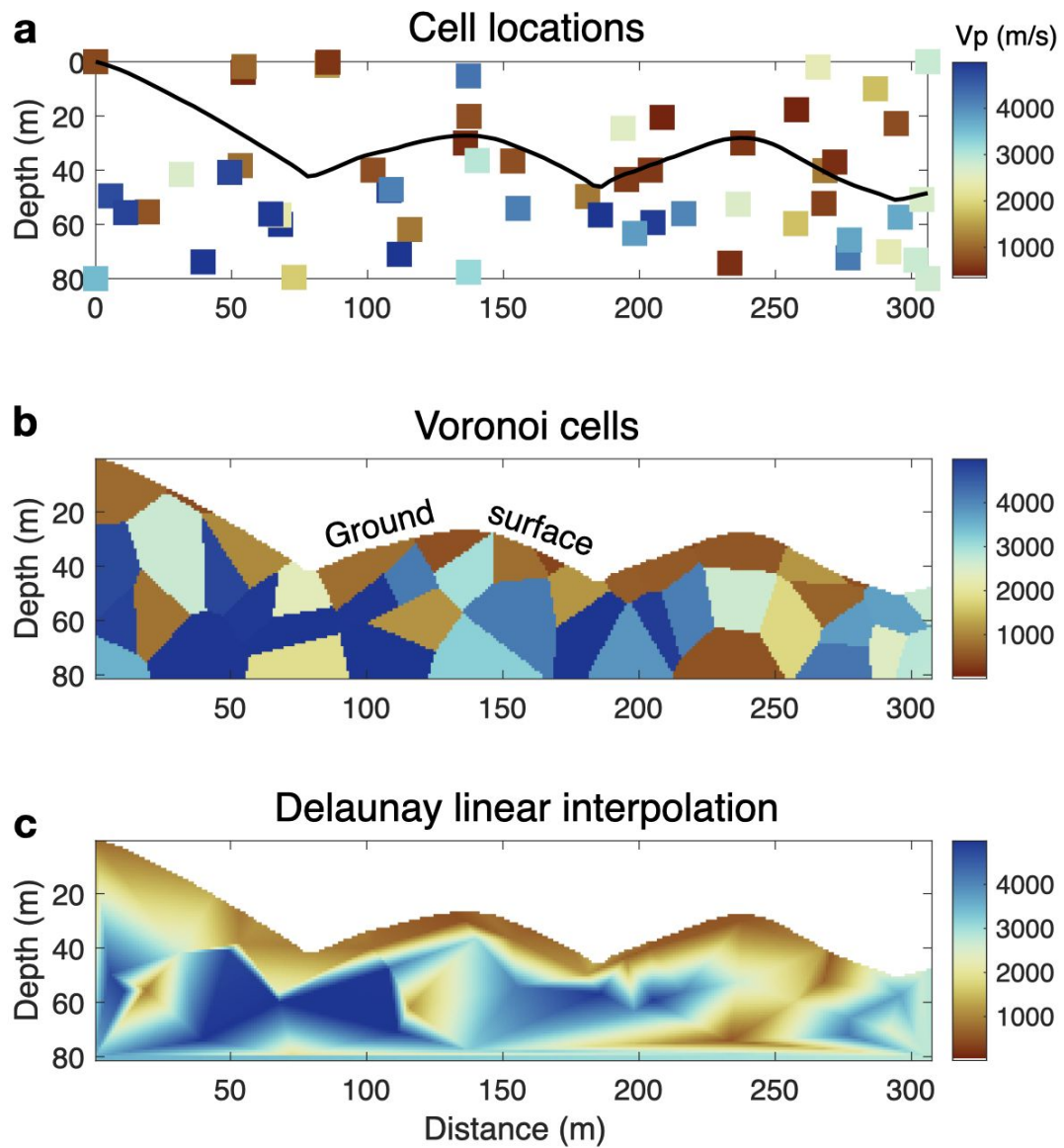
**psig.v1D. (line 85)** Standard deviation of the normal distribution from which changes in velocity (in meters per second) are sampled.

**psig.z1D. (line 86)** Standard deviation of the normal distribution from which vertical changes of a control point are sampled.

**psig.x1D. (line 87)** Standard deviation of the normal distribution from which lateral changes of a control point are sampled.

**psig.noise. (line 88)** Standard deviation of the normal distribution from which changes in the noise parameter (in  $\log_{10}$  scale) are sampled.

No changes should be necessary to the code following the proposal model parameters.



**Figure 5-1 Schematic model geometry and interpolation of THB2D.** (a) Location of cells in the model domain. The color of the cells represents seismic velocity. The black line is the surface topography. (b) Interpolation using Voronoi cells. (c) Delaunay linear interpolation. Both methods are using the *scatteredInterpolant* command in Matlab.

**Table 1.** Important parameters for the THB code with function descriptions and example values used for inversion of a 25 m -relief hill field survey in central California.

Parameter	Function of Parameter	Example Values	Notes
<b>Iterations</b>	Number of times the inversion code varied a parameter and created a probability distribution	1 million	Value depends on how quickly the misfit stabilizes
<b>numChain</b>	Number of parallel iterations of the code later combined	10 chains	
<b>delta_X, delta_Z</b>	Resolution: grid size of model	1 (m)	
<b>maxZ</b>	Depth of model	60 (m)	
<b>v0</b>	Vector of velocities for initial model parameterization	[400 600 1000 1500 2000 2500 3000 4000 4500 5000] (m/s)	
<b>Ncol0</b>	Number of hinge-lines	15 (hinge-lines)	Only for initial structure setup
<b>zz0</b>	Vector of depths for layers in initial model	[0 5 10 15 20 25 30 40 50 <b>maxZ</b> ] (m)	
<b>prior.v1D</b>	Lowest and highest velocities allowed	[300 5000] (m/s)	
<b>prior.h1D</b>	Shallowest and deepest depths allowed for a layer	[0, <b>maxZ</b> ] (m)	Set to the bounds of the model
<b>prior.Nuclei</b>	Maximum number of control points	500 (control points)	Minimum number is always 5
<b>prior.n</b>	Range for prior noise hyper-parameter estimate	[-6 1] ( $\log_{10}$ milliseconds)	
<b>psig.v1D</b>	Maximum value the velocity could be varied by	200 -1000 (m/s)	We started at 1000 m/s for first 500k iterations, then changed to 400 m/s, then 200 m/s

<b>psig.z1D</b>	Standard deviation for moving a layer or hinge point vertically	5 to 20 (m)	
<b>psig.x1D</b>	Standard deviation for moving a layer or hinge line horizontally	9 to 30 (m)	
<b>psig.n</b>	Proposal sigma for changing the noise hyper-parameter	0.1 ( $\log_{10}$ milliseconds)	

## 6. Running THB inversion and how to continue

Once you have your input file, elevation file, and model folder names in the proper locations, you can run the THB code. Make sure that you have the correct path linked for the fast-marching toolbox code (line 29 in *run\_model.m*). Below are some considerations for timing and plotting figures.

### 6.1 Time Issues for running the program

Once all parameters have been decided, you can run the THB2D inversion. The run time for this program is highly variable and can range from hours to days, depending on your model size, grid-resolution, the number of iterations, amount of data, and computer performance. Below are guidelines for running the program smoothly.

If you intend to run high resolution (grid size finer than one meter) across a domain larger than a few dozen meters, do not attempt to run the program on a laptop. We strongly recommend running the program on a multiple-core CPU workstation. When using remote log-in to run the program, we recommend using the Linux “screen” function to run the program. The GNU page about “screen” can be found here (<https://www.gnu.org/software/screen/>).

When the inversion has finished running, you will find figures and a *result.mat* file within models under the model folder name you chose (models/<your model name>). See section 7 for a description of these figures.

### 6.2 Continuation code

A second version of the THB code is available if you wish to continue running your model without having to restart, for instance if your chains have not yet converged. Most parameters cannot be edited when running the continuation code, but proposal sigmas can be edited if you wish to vary the parameters by a larger or smaller amount. This can help you tune the proposal distributions to more effectively sample the model space.

You will have to update a few parameters in the continuation code:

**Last\_Model\_File\_name (line 24).** This input file must be changed to the name of your existing output file. Check what your inversion result is saved as under models/<your model name>. Usually this is saved as *result.mat*, but you can

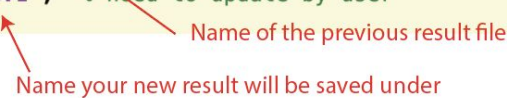


name it as you prefer. Use that name as the **Last\_Model\_File\_name** for the continuation code.

**Output\_name (line 25).** Change the output file to *result.1* or some other name to save it as a new result file and not overwrite your previous result.

```
%%
clear
close all

MODEL_FOLDER_NAME = 'Test1'; % File Name; it will be under "model" folder
Last_Model_File_Name = 'result'; % output mat file name, saved under the model folder
output_name = 'result.1'; % need to update by user
```



All other parameters can stay the same, or you can edit the proposal sigma parameters as you wish. The number of iterations can also be adjusted.

## 6.3 How to Update the Posterior Distribution with a new Burn-in Value

Check the RMSE misfit evolution graph (Figure 7-1). If you think the burn-in number is too small (meaning that models are being sampled prior to convergence) or too large (meaning that more models can safely be added to the ensemble), you can update the value and recalculate/replot figures using the *plot\_update\_burnin.m* script. To use this function, type “plot\_update\_burnin(**MODEL\_OUTPUT\_NAME**, **output\_name**, **new\_burnin**)” into the Matlab Command Window and then press enter. **output\_name** is the name of your result file without the “.mat” extension and **new\_burnin** should be a number representing the percentage of models you want to skip before calculating the mean model. For example, typing in “plot\_update\_burnin('MH234', 'result', 70);” will reset the burn-in percent to 70%. Please note that the iteration in the RMSE plot is in log<sub>10</sub>-scale.

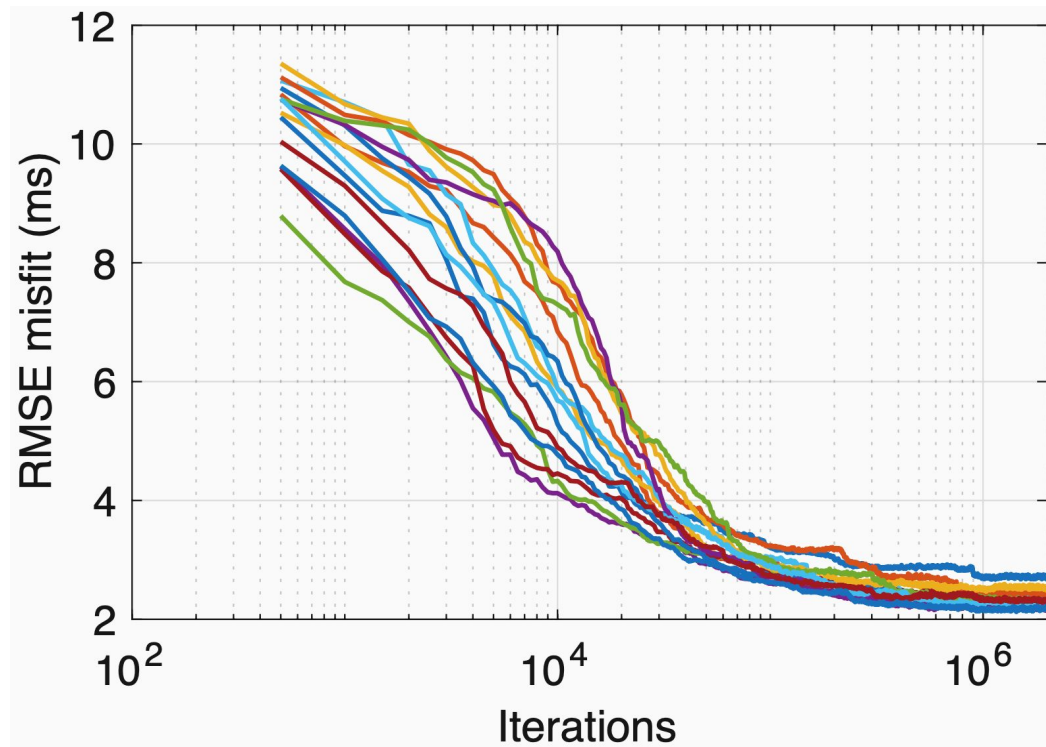
Note: the new figures will overwrite the previously saved figures for that model, but it doesn't overwrite the *result.mat* file, so previous figures can be restored by resetting to the original burn-in number.

## 6.4 Vertical Profile

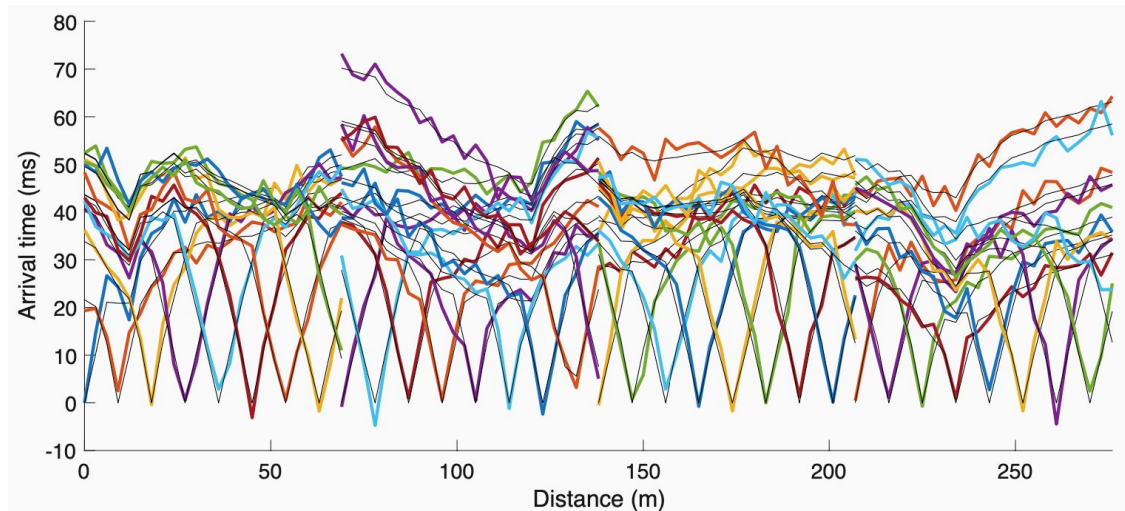
To create a 1D velocity-depth profile, use the *plot\_vertical\_profile.m* script found in the matcodes folder. First, choose the 2D model that you wish to work with and double click on the most recent *result.mat* file for this model, found within models/<your model

name>. Once the result file has loaded, you can run the vertical profile script. A pop-up figure of the arithmetic mean 2D velocity (computed by averaging over the models in the ensemble) will appear. Use your cursor to select a location along the x-axis where you would like to create a 1D profile and click this location. A vertical profile will pop up which you can then manually save.

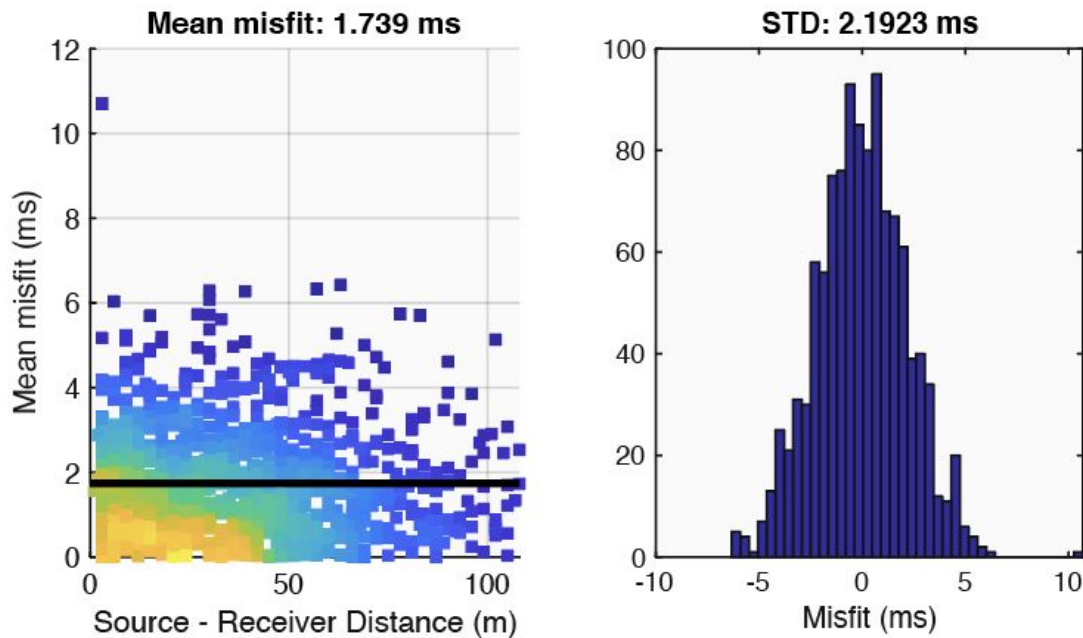
## 7. Explanation of Figures



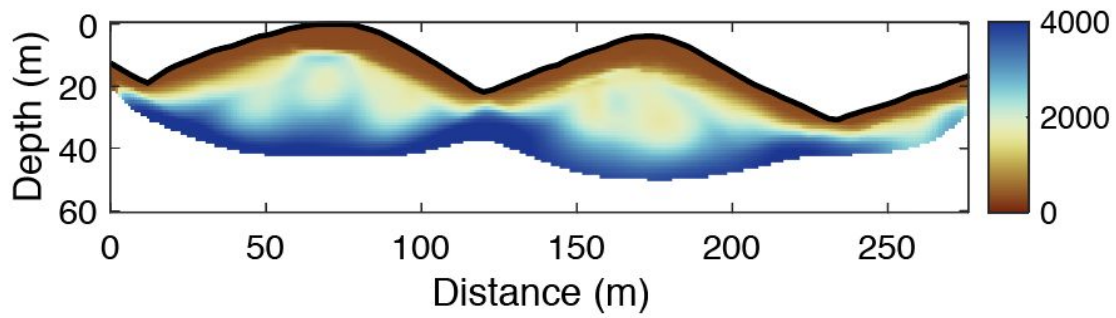
**Figure 7-1. RMSE (root mean square) misfit of individual Markov Chains.** The evolution of RMSE misfit over many iterations is shown in log-scale. Color indicates different Markov chains. Here, the misfit stabilizes around a value of 2.5-3.0 ms after 1 million iterations. Convergence of the Markov Chains and stabilization around one misfit value is a good indicator that the inversion process can be stopped and results can be interpreted. However, assessment of distributions of model parameter values across chains is also advised (see *Gao and Lekic, 2018*).



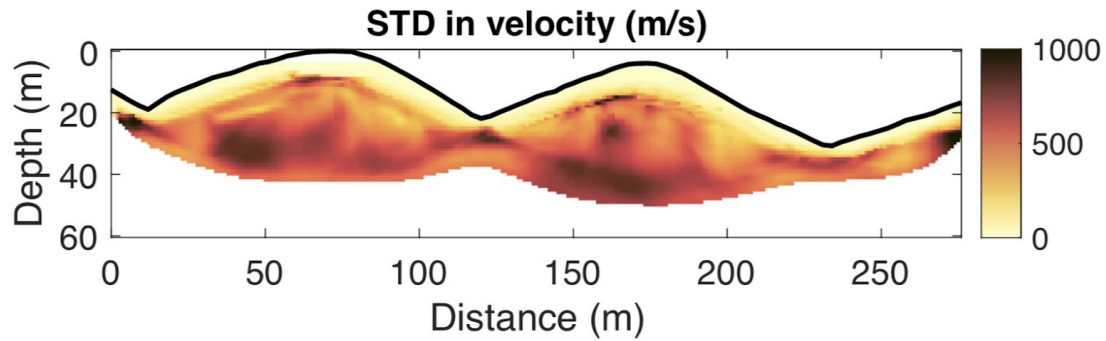
**Figure 7-2. Travel-time curve and fitting.** Each colored line represents the traveltimes to 24 geophones for an individual shot. The thin black lines are the traveltimes predicted by the mean of the ensemble (**Figure 4**). Fits should improve with further iterations and better fits may be provided by individual models in the ensemble.



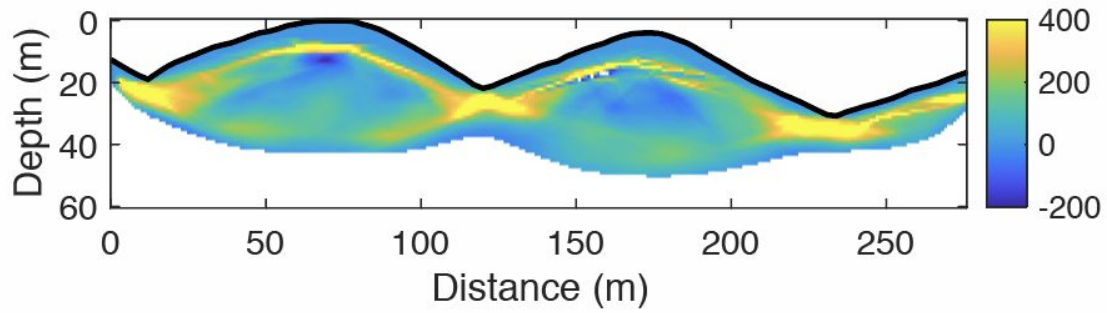
**Figure 7-3. Model misfit comparison.** Mean misfit is calculated by averaging the absolute values of misfit. (a) Misfit as a function of source-receiver distance. The mean misfit is shown by the black line. Here there does not seem to be a relationship between distance from the source and the misfit, which suggests that there is no systematic bias in the inverted velocities. (b) Histogram of the model misfit. The shape indicates we have a Gaussian distribution, a relatively narrow range of misfit, and no bias towards positive or negative misfit. This figure can be useful in determining error or bias in p-wave picking and is normally close to the inferred noise hyper-parameter (Figure 7-9).



**Figure 7-4. Posterior mean velocity.** The mean velocity model is plotted in color over the elevation profile of the field site. The mean model is computed as the arithmetic mean of the velocities across all models in the ensemble. In this particular model, low-velocity material ( $< 500$  m/s) seems to extend to a 10 m depth, while high velocity material ( $> 4000$  m/s) is found at a shallower depth at channels than at ridges. Note the velocity below the deepest raypath (**Figure 7-7**) is removed, since it is unconstrained by data and reflects solely the user-defined prior distribution.

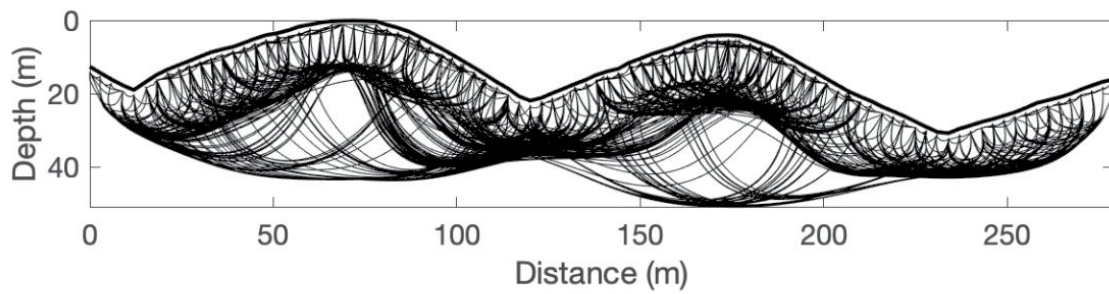


**Figure 7-5. Velocity standard deviation of the posterior distribution.** In addition to computing the mean velocity model across the ensemble of accepted velocity models, we can calculate the standard deviation of velocities across these models. Plotting this standard deviation at each point in the model illustrates areas where there is low uncertainty on the velocity estimate and also areas where uncertainty of the velocity is large. Large uncertainty can indicate either poorly constrained regions of the model, or areas in the vicinity of an abrupt change in velocity, where the precise location of change varies among models in the ensemble (see, for example, Figure 10 of *Burdick and Lekic, 2017* and Figure 4 of *Olugboji et al., 2017*). Note the velocity below the deepest raypath (**Figure 7-7**) is removed as done in **Figure 7-4**.

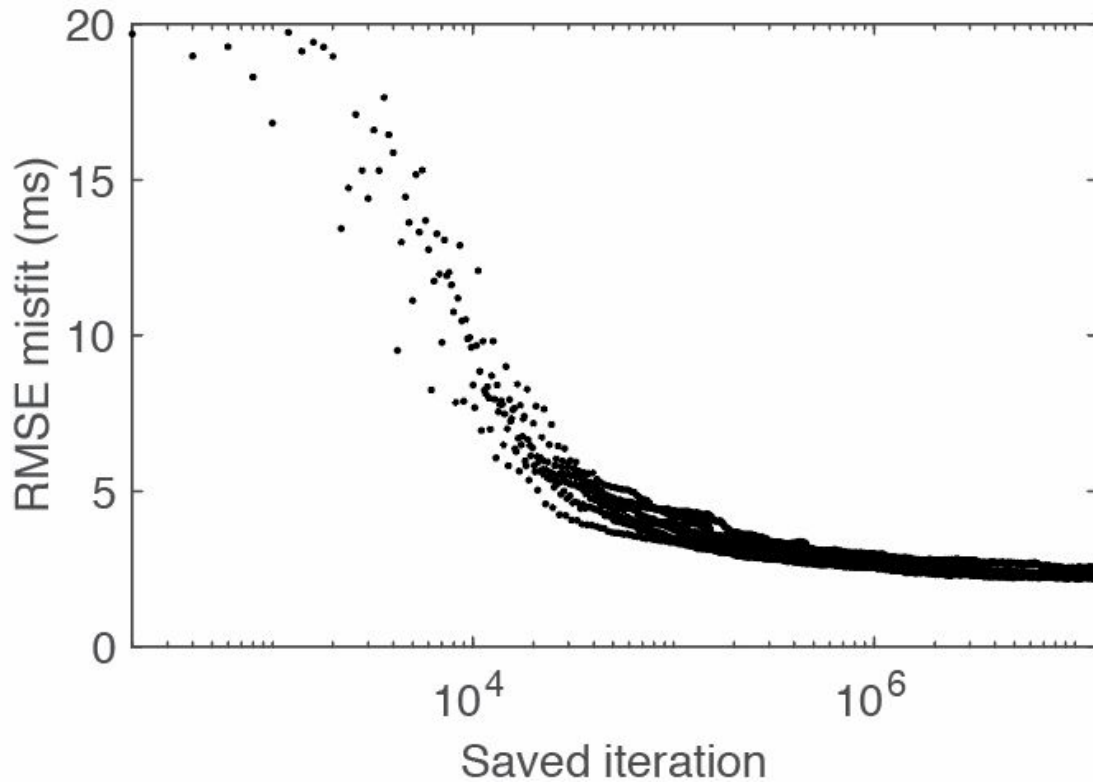


**Figure 7-6. Posterior mean vertical velocity gradient from models of the ensemble.** Higher gradient in m/s/m indicates more rapid velocity increase with depth. This figure clearly illustrates a boundary at ~10 m depth across which velocity increases substantially, and a deeper boundary at ~35 m depth beneath ridges, across which the velocity increase is somewhat smaller. Note the velocity below the deepest raypath (**Figure 7-7**) is removed.

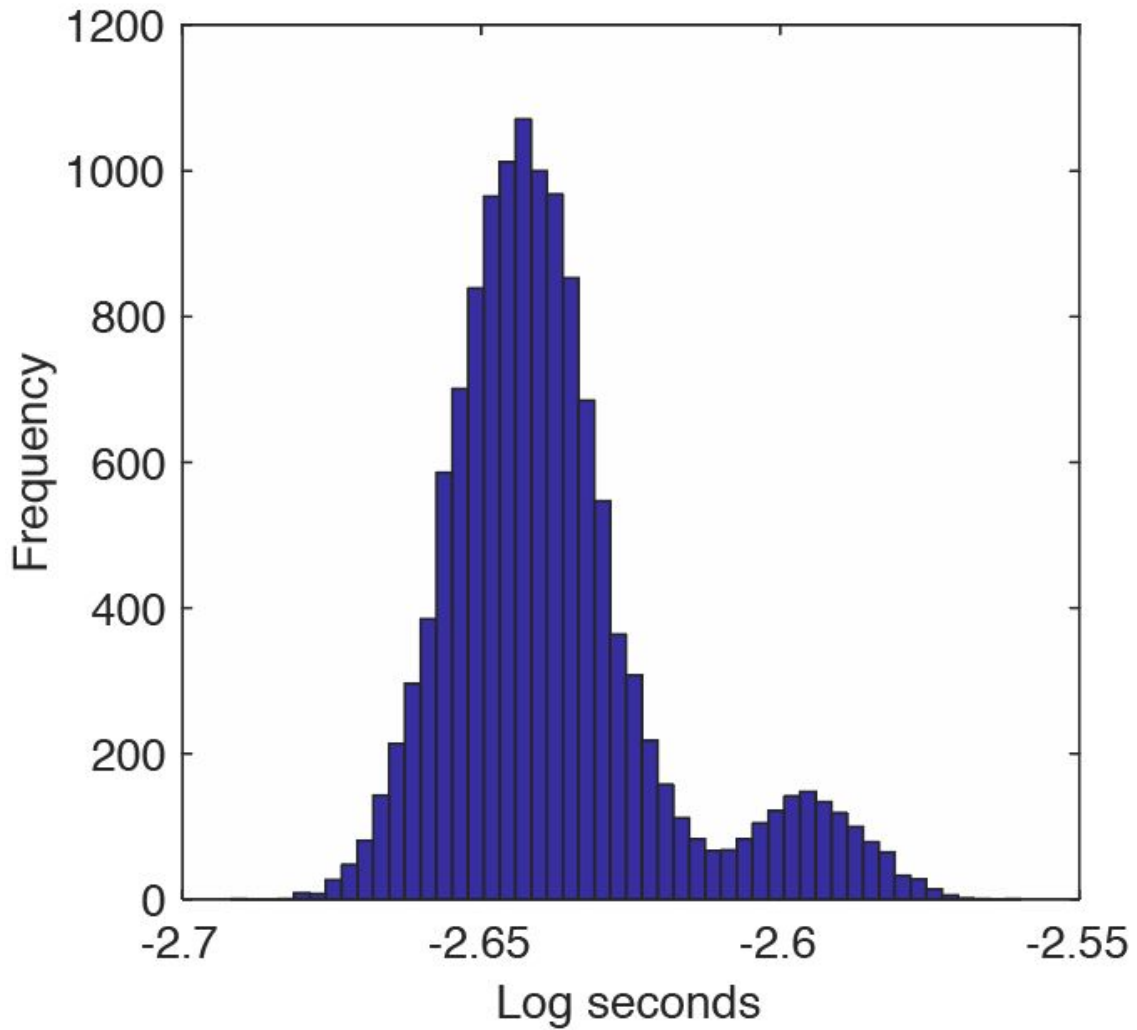




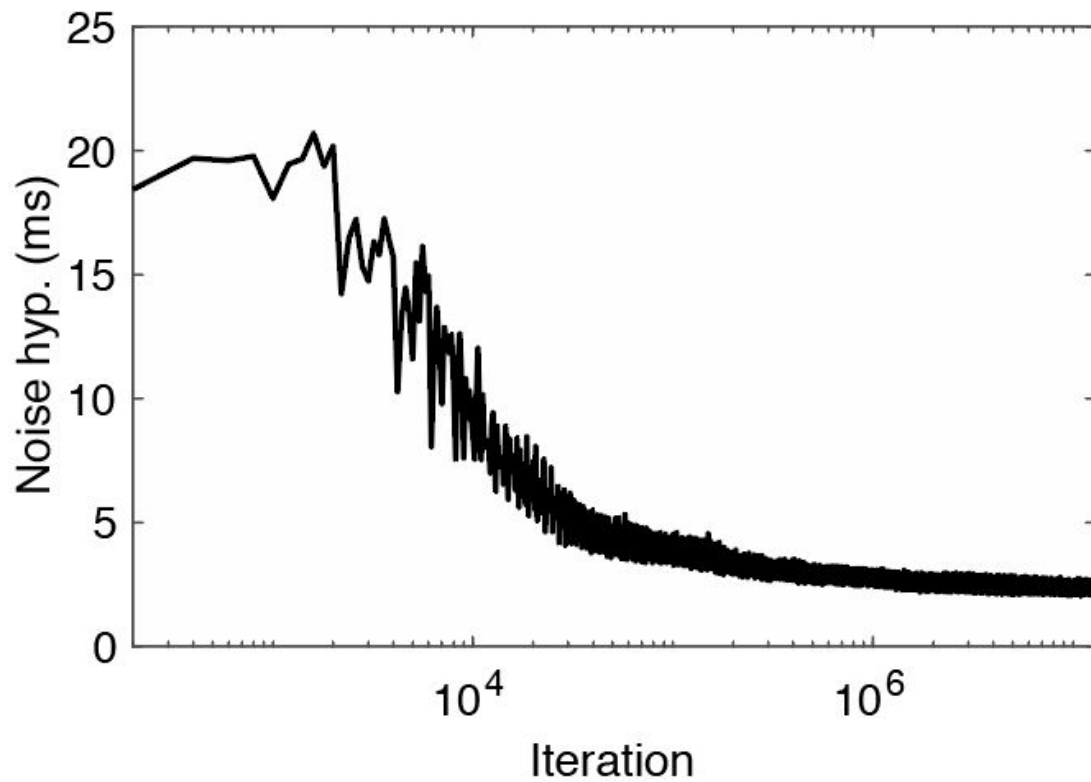
**Figure 7-7. Raypath of the posterior mean velocity model.** The thick black line represents the ground surface topography. The black curves show raypaths calculated for the mean velocity model. This figure can be used to get a sense of which portions of the model domain are well sampled by data.



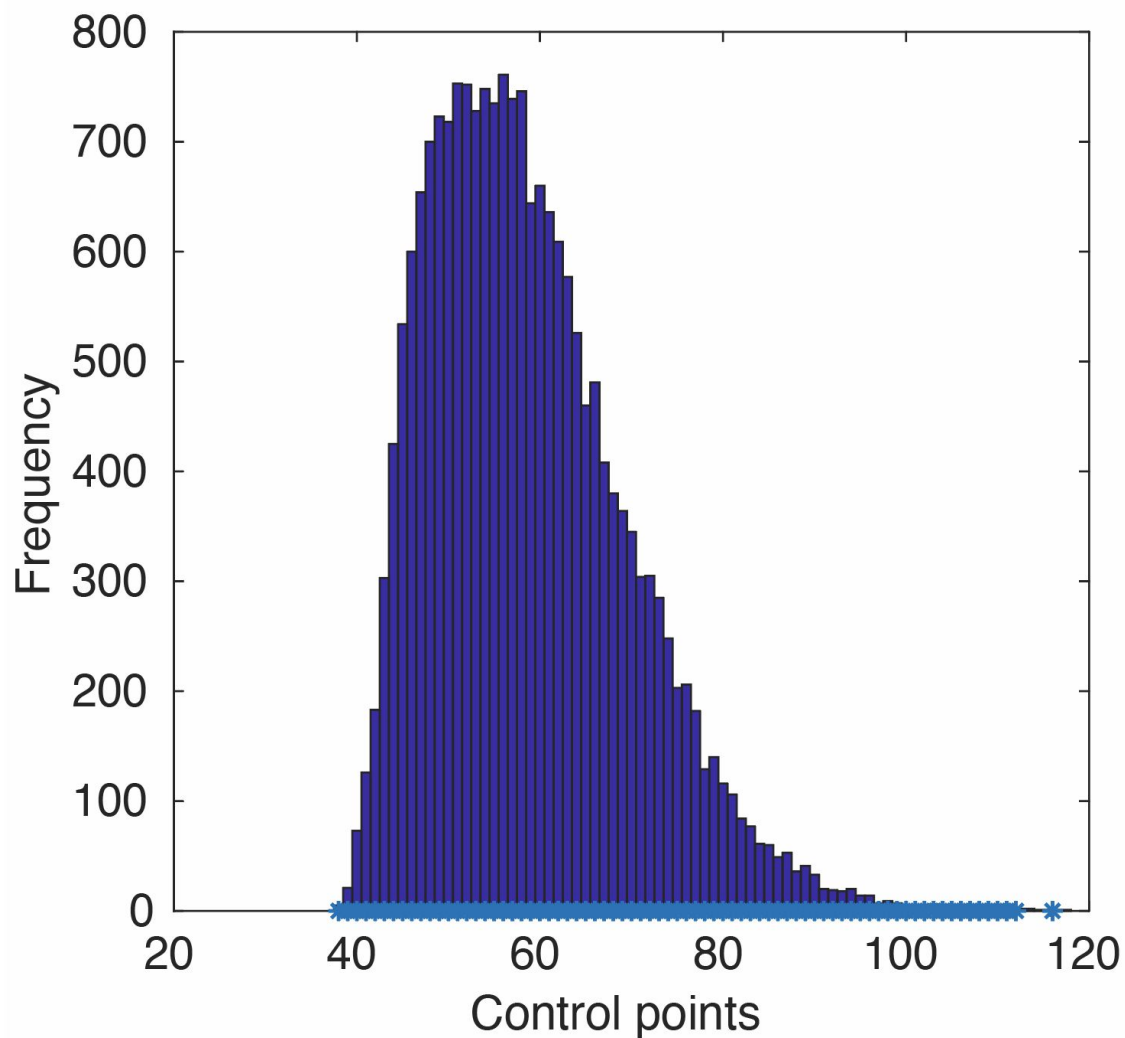
**Figure 7-8. Raw RMSE misfit with all chains combined.** Evolution of RMSE misfit with many iterations, shown for all chains together. This plot is similar to Figure 7-1, but Markov chains with the same iteration index are plotted next to each other. In other words, the total saved iteration is equal to the number of Markov chains multiplied by the total iteration of one Markov chain.



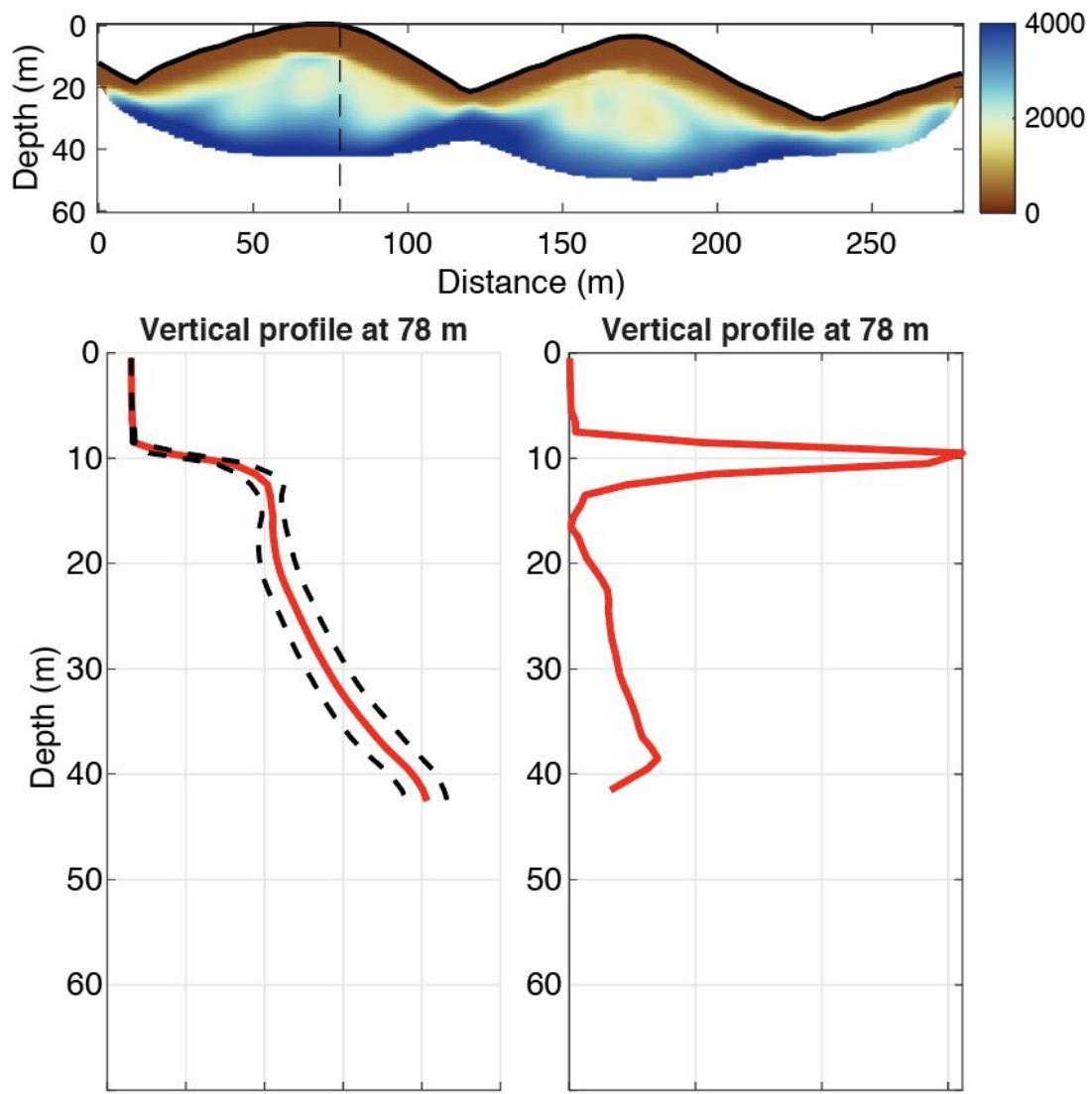
**Figure 7-9. Hyper-noise parameter for all chains.** The hyper-noise parameter is the value of estimated 1-sigma uncertainty of the traveltime data assigned to each model. This figure shows the probability distribution of predicted uncertainty, with -2.64 log seconds as the most frequently used data noise. The inferred value should also reflect the 1-sigma value of the model misfit (Figure 7-3).



**Figure 7-10. Evolution of noise hyper-parameter.** The evolution of noise hyperparameter values over time for all chains. After a sufficiently large number of iterations, the noise hyper-parameter should stabilize around a value that reflects the combined effect of the noise of the data (i.e. standard deviation of error on traveltime measurements) and errors inherent in the modeling assumptions (in our case, validity of the Eikonal equation).



**Figure 7-11. Number of control points.** Histogram of the number of control points of the models in the ensemble solution. The histogram reflects the posterior probability distribution on the number of control points estimated by the THB2D. Here, models with around 60 control points are most likely.



**Figure 7-12. Vertical Profile.** The vertical profile plot shows velocity plotted against depth on the left, and velocity gradient plotted with depth on the right. The black dashed curves on the left show 1-sigma model uncertainty. These plots are useful for examining where sharp increases in velocity occur beneath a user-selected location, and are well-suited for comparing subsurface structure at different locations within a field site and for calculating thicknesses of interpreted layers.

## 8. Useful tips

- **How do I know if the inversion is done?**
  - A good standard to go off of is to set the burn-in value at when the Markov chains have converged and the iterations afterwards are close to the same value. Ideally, you still want to continue the iterations with numbers a few times greater than the numbers needed to converge. For the example model given, the chains converged to be within 0.5-1 ms misfit of each other.
- **How do I make it faster?**
  - The most effective way to speed up the inversion process is to make the grid-resolution size (**delta\_X** and **delta\_Z**) coarser. However, this will come at a cost in that finer-scale variations in velocity will not be accurately represented in the FMM traveltimes predictions.
  - You can adjust the values of **psig.v1D**, **psig.x1D**, **psig.z1D**, which represent the amount by which a parameter is allowed to vary. Making these values larger allows for wider variation which can help the chains converge faster. But, values that are too large will lead to proposed models being rejected too often. Ideally, you are aiming to adjust these parameters to achieve an acceptance rate of ~40%.
  - If you are running the program on a computer with 4 or fewer CPUs, decreasing the number of Markov Chains (**numChain**) is recommended to make the process faster.
- **How do I interpret the results?**
  - It is important to note that we have no resolution power below the deepest raypath, as there is no data constraining these results.
  - In general, higher velocities represent more competent materials. Velocities greater than 4,000 m/s likely represent fresh bedrock, but velocities lower than this may also represent bedrock depending on the rock type and degree of weathering. Velocities less than 500 m/s likely represent soil.
  - To assess the robustness of a particular feature, consult the standard deviation maps. Additionally, you may want to plot covariance maps, analogous to Figure 9 in *Burdick and Lekic (2017)*.

## 9. References

- Bodin, T., Sambridge, M., Tkalcic, H., Arroucau, P., Gallagher, K., and Rawlinson, N. (2012). Transdimensional inversion of receiver functions and surface wave dispersion, *J. Geophys. Res.*, 117, B02301, doi:10.1029/2011JB008560.
- Bodin, T., Yuan, H., Romanowicz, B. (2014). Inversion of receiver functions without deconvolution—application to the Indian craton, *Geophysical Journal International*, 196, 1025–1033, <https://doi.org/10.1093/gji/ggt431>
- Brooks, S. P., Giudici, P., & Roberts, G. O. (2003). Efficient construction of reversible jump Markov chain Monte Carlo proposal distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(1), 3-39.
- Burdick, S., and Lekić, V. (2017). Velocity variations and uncertainty from transdimensional *P*-wave tomography of North America, *Geophysical Journal International*, 209, 1337–1351, <https://doi.org/10.1093/gji/ggx091>
- Cramer, F. (2019). Scientific Colour Maps. Zenodo, doi:10.5281/ZENODO.1243862.
- Gao, C., & Lekić, V. (2018). Consequences of parametrization choices in surface wave inversion: insights from transdimensional Bayesian methods. *Geophys. J. Int.*, 215(2), 1037-1063.
- Henson, R. (2020). Flow Cytometry Data Reader and Visualization (<https://www.mathworks.com/matlabcentral/fileexchange/8430-flow-cytometry-data-reader-and-visualization>), MATLAB Central File Exchange. Retrieved October 26, 2020.
- Dirk-Jan Kroon (2021). Accurate Fast Marching (<https://www.mathworks.com/matlabcentral/fileexchange/24531-accurate-fast-marching>), MATLAB Central File Exchange. Retrieved March 9, 2021.
- Olugboji, T. M., Lekic, V., and McDonough, W. (2017), A statistical assessment of seismic models of the U.S. continental crust using Bayesian inversion of ambient noise surface wave dispersion data, *Tectonics*, 36, 1232– 1253, doi:10.1002/2017TC004468.
- Peyre, G. (2020). Toolbox Fast Marching (<https://www.mathworks.com/matlabcentral/fileexchange/6110-toolbox-fast-marching>), MATLAB Central File Exchange. Retrieved March 31.