Supplementary Information

S1. THB2D MCMC User Guide

Table of contents

Table of figures

Table

1. Introduction

THB inversion utilizes a reversible-jump Markov-chain Monte Carlo (MCMC) algorithm to create a set of velocity models that best describe the observed data (Bodin et al., 2012; Burdick and Lekic, 2017). Using this method, one analyzes the posterior probability assigned to every given velocity model, rather than producing one single best-fit model (Burdick and Lekic, 2017). The result of this process is a collection of possible solutions, in which solutions with a higher likelihood of describing the data are represented at a higher likelihood (Burdick and Lekic, 2017). Because the results consist of multiple possible velocity structures, it is easier to understand both the range of plausible solutions and the uncertainty associated with the velocity profiles (Burdick and Lekic, 2017).

As mentioned in the main text, MCMC involves the following steps. First, create a new velocity model by randomly selecting a parameter to vary. The initial velocity model is set up by user-defined parameter limits called prior sigmas. These parameters construct a velocity structure in terms of horizontal layers and hinge points, which allow for lateral heterogeneity within a layer (Burdick and Lekic, 2017). Once the initial model is generated, one of five functions is selected to vary the parameters and create a new model: change the velocity, move a cell, create a new cell, delete a cell, or change the noise parameter (Burdick and Lekic, 2017; Bodin et al., 2012). The MCMC algorithm selects parameter values from within a normal distribution of user-defined values, representing the amount the program is allowed to vary a parameter by, called proposal sigmas.

Second, the method calculates the posterior probability given estimated travel times for the new velocity profile. Travel times for the proposed model are generated using the fast-marching method, described by Sethian (1995). Posterior probability is calculated to reflect the difference between modeled and observed travel times (Bodin et al., 2012). Finally, the program decides to accept or reject the model based on its effect on the posterior probability, according to the Metropolis-Hastings algorithm (Burdick and Lekic, 2017). The new model is either accepted and added to the current model or rejected and removed. After one thousand iterations, the current model is saved to an ensemble. Earlier models are predicted to have the highest uncertainty, therefore models produced before a user-defined burn-in period are discarded. After many iterations, model misfit will stabilize, at which point standard deviation of ensemble velocity can be calculated to indicate areas where the velocity profile has greater uncertainty. 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 MCMC algorithm, detailed explanation of the model parameters and setup, execution of the program with an example, and figures produced by the program.

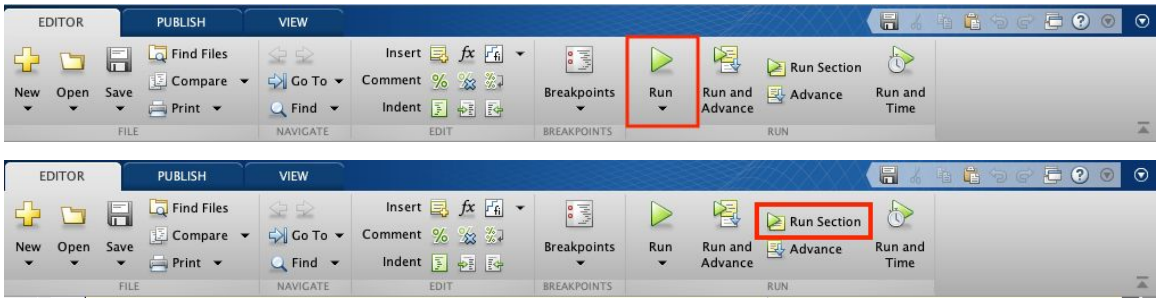
2. System requirement

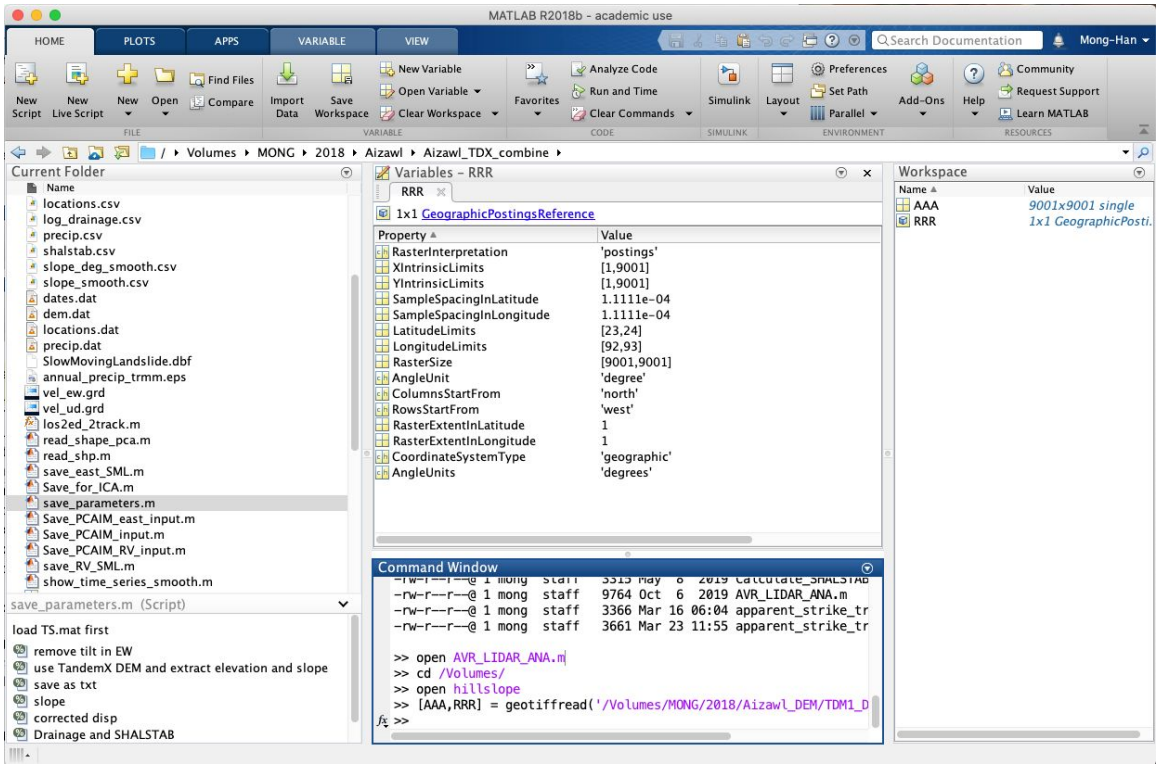
This program is written in Matlab and can be run in multiple platforms including Linux, Mac, and Windows systems. We have only tested this program in Matlab 2017b and later versions, but it should work as long as with parallel computation capability. The "fast marching toolbox" is not part of the program 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 correctly in the "*run_model.m*" script (see Section ??? for more detail).

3. Quick start (running the test example)

In this section we will run an example test with given elevation and travel time measurements. The purpose of this section is to make sure the THB MCMC 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 elevation file.
3. Open the run_model_par.m script and insert the file names above into the script in the “input_file” and “elevation_file” spots, respectively.
4. Check that the correct location path is linked for the fast marching toolbox and the functions within the “matcodes” folder.
5. Choose a name for the model folder that will be created to store the result (saved in the “models” folder).
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 0.5 m resolution for 200,000 iterations takes about 2 days in a workstation with a 16-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 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 inverion: a file containing traveltime data, and an elevation file.

Input_file. Data from the seismic survey should be formatted into three columns: column one for traveltime (in seconds), column 2 for shot location (in meters), and column three for the location of each receiver (in meters). Data must then be 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.

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 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. Using multiple chains will allow a faster and more accurate understanding of model misfit, however a high-CPU computer is required to run multiple chains at a practical speed. If you are using a laptop, we recommend setting the number of chains no higher than 4. If you have a high-CPU device, you can set up to 15 chains.

maxiter. We recommend running the inversion until the misfit of all chains have converged. The number of iterations necessary therefore depends on the size of

your model and your specific data. Typically, at least 100,000 iterations are necessary.

Hier. Leave this value as 1.

Datsav. This value determines how often a model is saved to the final ensemble. A higher value means a model will be added after a higher number of iterations, therefore there will be fewer models in the final ensemble. This will take up less space on your computer.

Burn_percent. The burn-in value determines at what point the program starts calculating the mean model and standard deviation. These calculations should start after the point where the markov chains have converged, so setting to a higher percentage is typically better.

5.2 Initial Model Setup

Here you will determine 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.

MaxZ, MinZ. Z represents the depth range of your model. Set MinZ to zero and determine a value for MaxZ based on your seismic survey. MaxZ should be at least the same value as the length of your survey line. You can make the model as deep as you want, but deeper will also take much longer to run.

Delta_X, delta_Z. These grid size parameters determine the resolution of your model. 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 values must be smaller than the distance of your geophone spacing, but a resolution finer than 1m takes a long time to run.

** The initial model parameters listed below should not significantly influence the final model, with sufficient iterations**

V0. You will assign velocities to your desired number of layers and format them in a bracketed list from lowest to highest velocity. The number of velocities you use determines the number of layers. You must use a minimum of three layers.

Zz0. This depth vector must be exactly the same length as **v0**. Assign a depth to the start of each layer defined above, with the first value as 0.

Ncol0. The number of hingelines determines how much lateral variation can be accommodated by the model. Setting too many hingelines may result in the model overfitting and creating wavy interfaces, however too few hingelines will not fit the data as well.

5.3 Priors

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

Prior.v1D. Set a minimum and maximum velocity based on knowledge of your data.

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

Prior.Nlay. The maximum number of layers allowed in the model. This value must be greater than four.

Prior.smoothing_X. Minimum hinge-line distance. This value is smoothing_X * delta_X.

Prior.n. The proposed data noise range in log scale

5.4 Proposal Sigmas

Proposal sigma values are a standard deviation for proposing changes to a model. Larger values will allow the model misfit to stabilize sooner, but will also result in a rougher finished product.

Psig.v1D. Change in velocity in meter per second.

Psig.h1D. Distance in meters for moving a layer vertically.

psig.d1D. Change in the velocity of an added layer in meters per second.

Psig.n. Change in the noise parameter in log scale

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

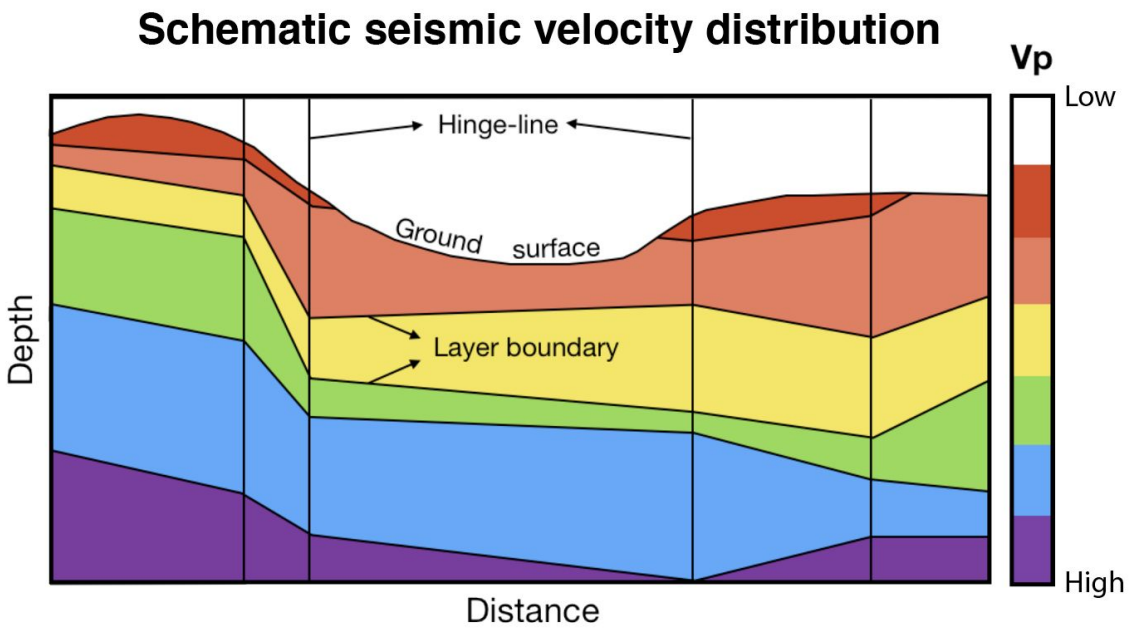


Figure 5-1 Schematic model geometry described the setup of THB MCMC. There are 6 hinge-lines including the model boundaries and 6 layers (5 layer boundaries). Color represents P-wave seismic velocity, Vp. The velocity above ground is set to airwave speed (white color).

Parameter	Function of Parameter	Example Values	Notes
Iterations	Number of times the inversion code varied a parameter and created a probability distribution	1 million - 1.5 million	Value depended on how quickly the misfit stabilized
NumChain	Number of parallel iterations of the code later combined	20 chains	
Delta_X/ delta_Z	Resolution: grid size of model	0.5 m	
MaxZ	Depth of model	100 or 150 m	150 used for NE-SW profile of MH7
V0	Vector of velocities for initial model parameterization	[300 500 1000 3000 4000 5000] m/s	
Ncol0	Initial number of hinge lines	30 hinge lines	
Zz0	Vector of depths for layers in initial model	[0 10 20 50 80 maxZ] m	
Prior.v1D	Lowest and highest velocities allowed	[300, 5000] m/s	
Prior.h1D	Shallowest and deepest depths allowed for a layer	[0, maxZ] m	Set to the bounds of the model, which varied
Prior.Nlay	Maximum number of layers	70 layers	
Prior.Ncol	Maximum number of hinge lines, allows for lateral variability in velocity structure	300 hinge lines	
Prior.n	Range for prior noise estimate	[-12 0] ms	In log scale
Psig.v1D	Maximum value the velocity could be varied by	200-400 m/s	
Psig.h1D	Standard deviation for moving a layer or hinge line	5-20 m	
Psig.d1D	Adding or removing a layer	100-200	
Psig.n	Proposal sigma for changing the noise parameter	0.1 ms	

6. Running THB2D MCMC 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. 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 inversion program. The run time for this program is highly variable and can range from hours to days, depending on your model size, resolution, and the number of iterations. Below are guidelines for running the program smoothly.

If you intend to run high resolution (grid size finer than one meter), do not attempt to run the program from a laptop. We strongly recommend running the program from a multiple CPU computer in a Command window, or running on a laptop through a remote login to a high CPU computer. In the Command window, create a new screen by typing "screen" and then open matlab from this screen with the "-nodesktop" qualifier. Navigate via terminal to the "Refract2D" folder where your inversion code is stored and then type the name of your inversion code into terminal and hit enter. You should see a message pop up that says the parallel pool is starting. Hold *control+a+d* to exit this 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. See section 7 for a description of these figures. Next, section 6 describes how to run further iterations of your model and how to create additional plots.

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.

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

Input_file. The input file name must be changed to the name of your existing output file. Check what your initial inversion result is saved as under

MODEL_FOLDER_NAME and use that name plus ".result" as your new input name for the continuation code.

OUTPUT_FILE_NAME. Change the output file to "result.2" or some other name to save it as a new result file and not overwrite your previous result.

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

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

Check the RMSE misfit evolution graph. If you think the burn-in number is too small or large, you can update the value and recalculate/replot figures using the Plot_update_burnin.m script. To use this function, type "Plot_update_burnin(**folder_name, output_name, new_burnin**)" into your Matlab Command Window and then press enter. **Folder_name** is equivalent to **MODEL_OUTPUT_NAME** for the desired model, and **new_burnin** should be a number representing the percentage of models to skip before calculating the mean model.

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. 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" under the specific model folder of your choice. Once the result file has loaded, you can run the vertical profile script. A pop-up figure of the mean 2D velocity 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

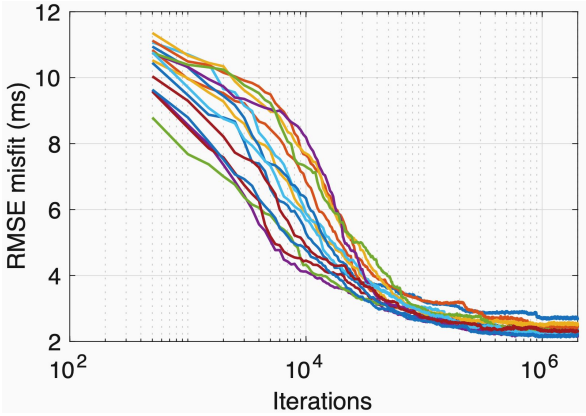


Figure 7-1. RMSE (root mean square) misfit of individual Markov Chain. Color indicates different Markov chains. Note this is plotted in log-scale.

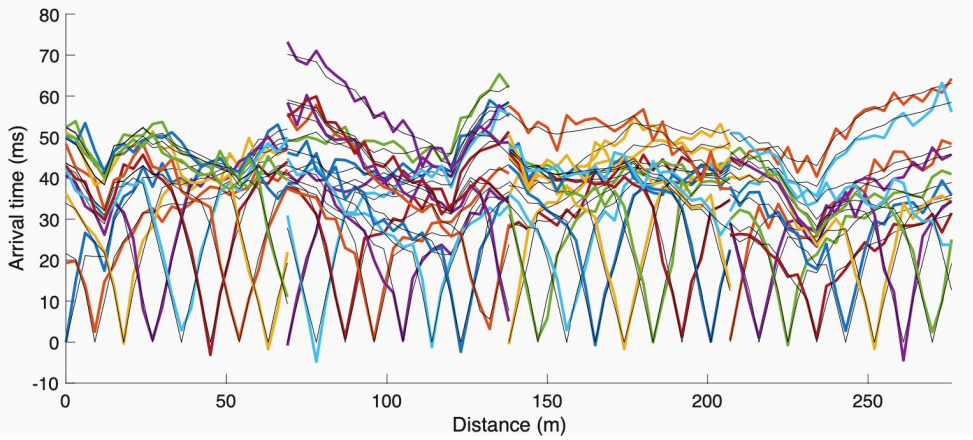


Figure 7-2. Travel-time curve and fitting. Each colored line represents the arrival time to 24 geophones for an individual shot. The black lines are the model fitting of the mean velocity (**Figure 4**). Model to data fitting should improve with further iterations.

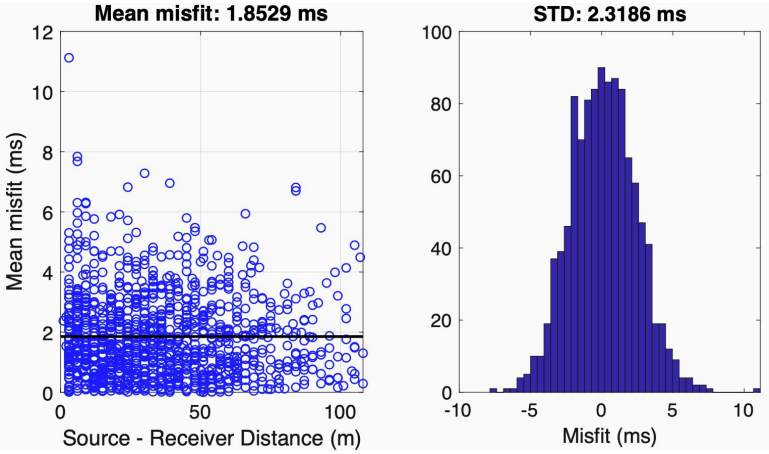


Figure 7-3. Model misfit comparison. Mean misfit is calculated by averaging the absolute values for misfit. (**a**) Misfit vs different source-receiver distance. The mean misfit is shown in the black line. Here there does not seem to be a relationship between distance from the source and the misfit. (**b**) Histogram of the model misfit. Shows we have a Gaussian distribution, a relatively narrow range of misfit, and no bias towards positive or negative misfit. This can be useful in determining error/bias in p-wave picking?

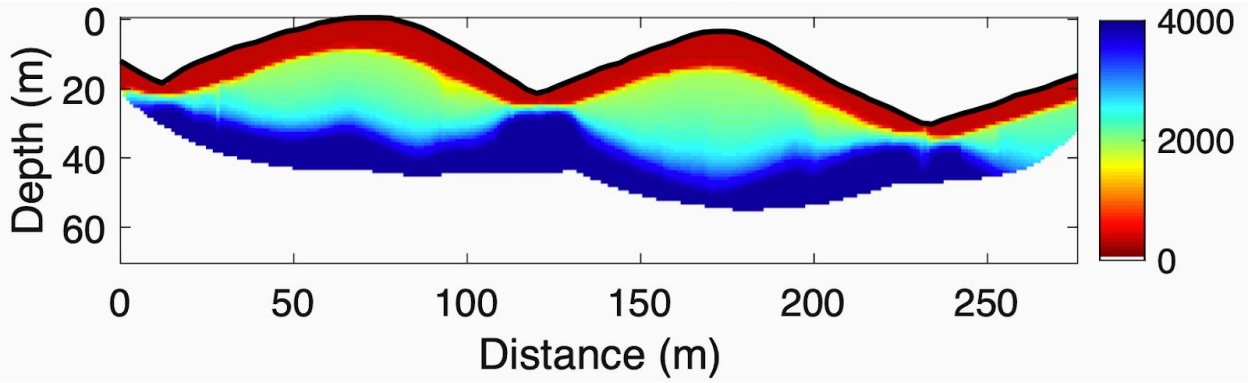


Figure 7-4. Posterior mean velocity. Note the velocity below the deepest raypath (**Figure 7-7**) is removed. The mean velocity is plotted in color over the elevation profile of the field site. Mean velocity represents a combination of many models that had low enough misfit to be accepted into the ensemble. In this particular image, 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.

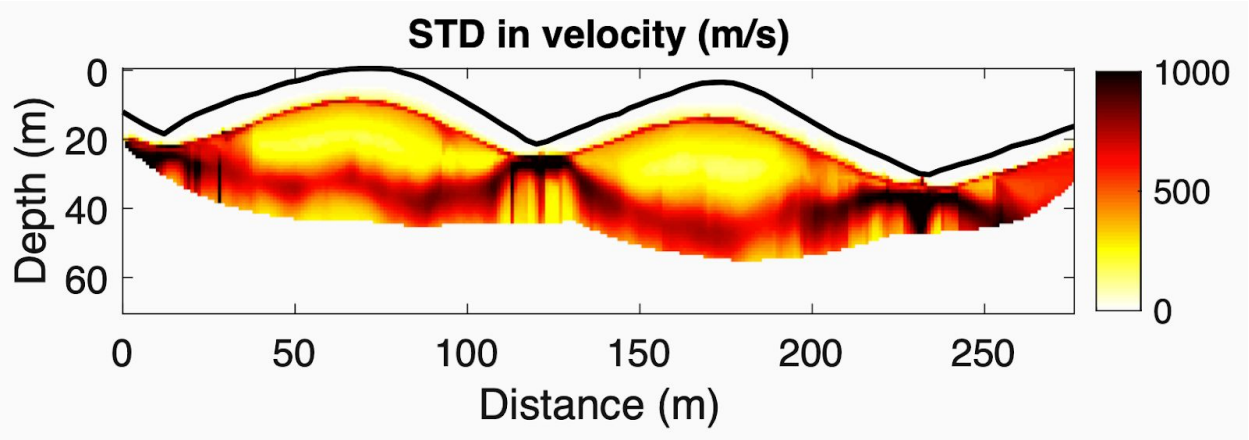


Figure 7-5. Velocity standard deviation of the posterior distribution. Note the velocity below the deepest raypath (**Figure 7-7**) is removed. Since the mean velocity represents an ensemble of possible velocity models, we can calculate the standard deviation of velocity between these models. This figure illustrates areas where there is the least agreement between models, and also areas where there is a sudden change in velocity. Notice that there are several vertical spikes highlighted with a high standard deviation that likely represent artifacts from the inversion process.

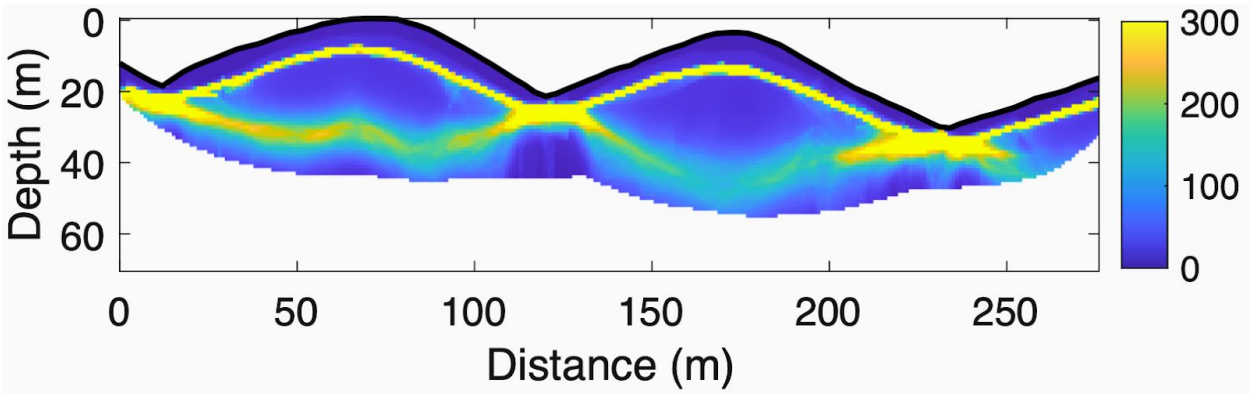


Figure 7-6. Posterior vertical velocity gradient. Note the velocity below the deepest raypath (**Figure 7-7**) is removed. Higher gradient in m/s indicates more rapid velocity increase along depth. This figure clearly illustrates a boundary of rapid velocity increase at a 10 m depth and another boundary of slightly less increase at a 40 meter depth, beneath ridges.

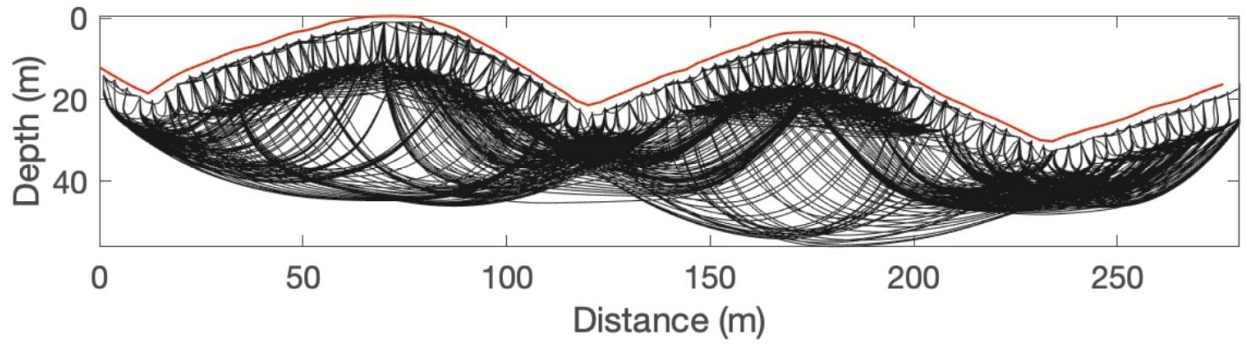


Figure 7-7. Raypath of the posterior mean velocity model. The red line represents the ground surface topography. The raypaths simulate where each seismic wave travelled below ground, giving us a sense of where we have data and where we do not.

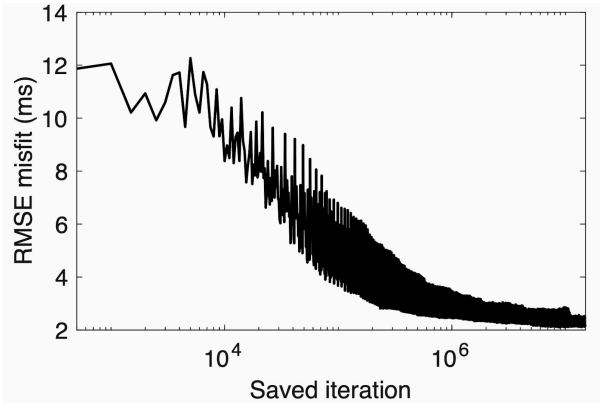


Figure 7-8. Raw RMSE misfit (with combining all chains)

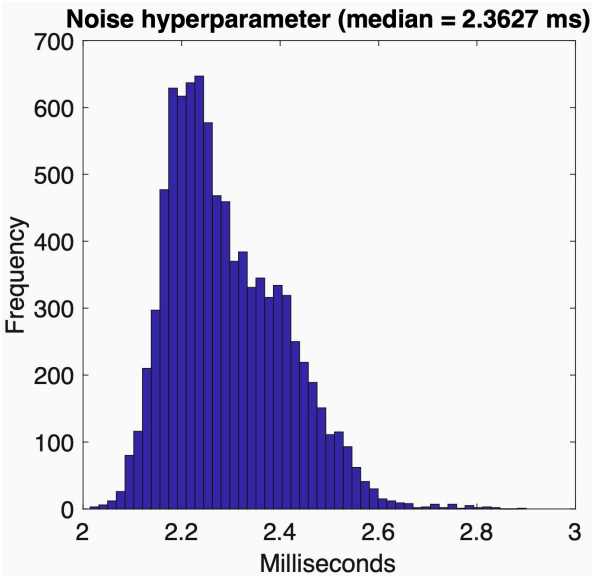


Figure 7-9. Hyper-noise parameter (with combining all chains)

The hyper-noise parameter is the value of estimated data noise assigned to each model. This figure shows the probability distribution of predicted uncertainty, with ~2.2ms as the most frequently used data noise.

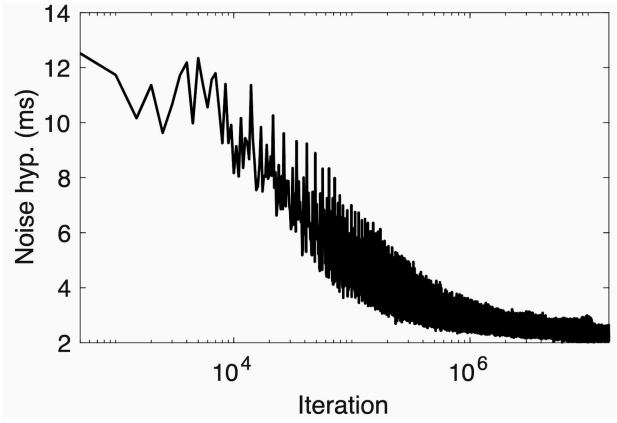


Figure 7-10. Evolution of hyper-noise (with combining all chains)

The evolution of noise hyperparameter values over time. As further iterations run, the noise parameter should become smaller and more closely reflect the noise of the data.

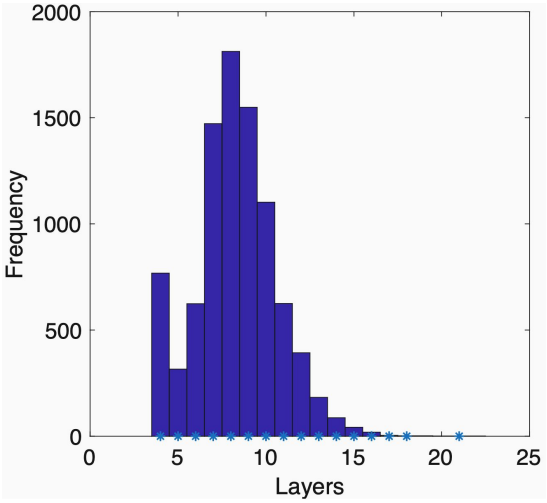


Figure 7-11. Number of layers (combining all chains). The number of layers histogram includes results from all chains, and shows the number of layers that most frequently were accepted into the ensemble of possible model solutions. Here, models with 7-9 layers had the highest frequency within the final ensemble.

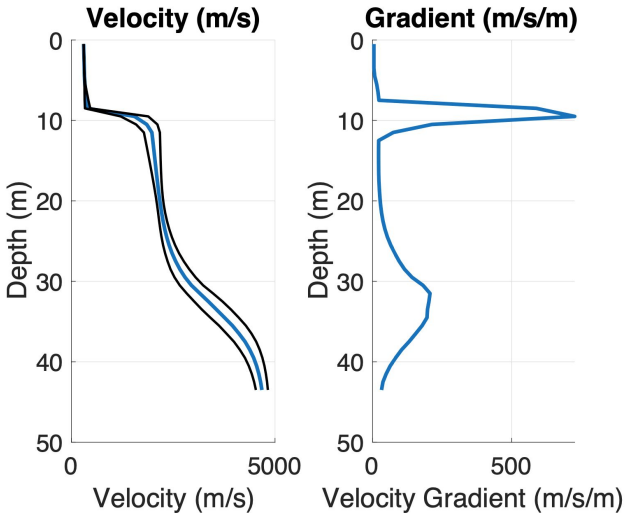


FIgure 7-12. Vertical Profile. The vertical profile plot shows velocity plotted against depth on the left, and velocity gradient plotted with depth on the right. The dark blue bars on the left are error bars. These plots are useful for examining where sharp increases in velocity occur at one particular location, and are particularly good for comparing subsurface structure at different locations within a field site and for calculating thicknesses of interpreted layers.

7. Useful tips

- **How do I know it is done?**
    - A good standard to go off of is when the Markov chains have converged so that they are close to the same value. 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 resolution size (delta_X and delta_Z) coarser. However, this will come at a cost in that the output will be less precise and unable to resolve subtler changes in velocity structure.
    - You can also decrease the maximum number of hinge lines (**prior.Ncol**) or the number of layers (**prior.Nlay**) which allows for fewer elements within the model that have to be varied. Having few hinge lines and layers may also make the output coarser. On the other hand, having too many can cause the model to over fit and produce spikes and artifacts that do not actually exist.
    - You can increase the values of **psig.v1D**, **psig.h1D**, **psig.d1D**, which represent the amount by which a parameter is allowed to vary. Making these values larger allows for wider variation which will help the chains converge faster.
    - If you are running the program on a laptop, or with 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 4000 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.
    - Spikes that appear as vertical lines are likely artifacts of the inversion process and should not be interpreted.

References

THB manual
Bodin
Burdick and Lekic
Gabriel Peyre (2020). Toolbox Fast Marching (https://www.mathworks.com/matlabcentral/fileexchange/6110-toolbox-fast-marching), MATLAB Central File Exchange. Retrieved April 4, 2020.