Nonlinear reversible-jump Markov chain Monte Carlo tomography

Erica Galetti · erica.galetti@ed.ac.uk

April 2014

This document describes how the Fortran 90 codes in *rjmcmc_erica_distr.zip* can be used to perform rj-McMC tomography, process the results and visualize the outputs.

The tomography codes *rj_tomo_mksamples.f90* and *rj_tomo_procsamples.f90* are non-linearized versions of Thomas Bodin's (TB) code *rj_tomo.f90* (which can be downloaded from http://rses.anu.edu.au/~thomas/Codes/RJ_MCMC_TOMO/). Non-linearization is achieved by updating the raypath geometry for each velocity model proposed along the Markov chain using a subroutine based on the eikonal ray tracer *fm2dss.f90* by Nick Rawlinson (NR - http://rses.anu.edu.au/~nick/surftomo.html).

Samples are generated using *rj_tomo_mksamples.f90* (executable `mksamples`), are saved to disk, and can then be processed using *rj_tomo_procsamples.f90* (executable `procsamples`). The format of input and output files is similar to that of the Fast-Marching Surface Wave Tomography (FMST) package by NR. For instructions to the FMST package, click here.

Code *rj_tomo_mksamples.f90* is parallelized using MPI, so that a number of Markov chains can be run in parallel at the same time. The chains start from different initial conditions which are determined randomly (the random seed is equal to the processor number + 1).

makefile

Script used to compile the code. To run the code on the ECDF cluster ('eddie'), you need to:

```
> module load intel
> module load openmpi-intel
```

In the *makefile*, change line 34 to the path where the directory containing this document is located.

To compile all the codes, type

```
> make all
```

To compile only the rj-McMC tomography code, type

```
> make driver
```

rj_tomo_mksamples.f90 · mksamples

This code creates samples using the rj-McMC algorithm. All input parameters are given in file *mksamples.in*. This file and all other input files have to be located in the same directory as the executable `mksamples`.

The **input files** are the following:

- ***sources.dat*** · File of source coordinates in the following format:

  ```
  61
  56.633500 -3.918900
  58.285900 -4.412590
  52.300700 1.478170
  58.031000 -4.883400
  58.481900 -4.201320
  …
  ```

  where the first line specifies the total number of sources and the remaining lines contain the coordinates of each source in [latitude longitude]. Southern hemisphere latitudes and western hemisphere longitudes are negative.

- ***receivers.dat*** · File of receiver coordinates in the same format as *sources.dat*.

- ***otimes.dat*** · File of traveltimes in the following format:

  ```
  0 100.0 0.1
  0 100.0 0.1
  0 100.0 0.1
  1 48.198 0.785399
  0 100.0 0.1
  0 100.0 0.1
  1 39.0534 0.53121
  0 100.0 0.1
  1 41.4224 0.492536
  0 100.0 0.1
  1 40.301800 0.203018
  1 45.4852 0.407051
  0 100.0 0.1
  0 100.0 0.1
  …
  ```

  where the first column is a switch indicating that the traveltime is to be included (=1) or excluded (=0) from the inversion, the second column specifies the traveltime in seconds, and the third column specifies the error in seconds (N.B. cannot be zero!). For n_s sources and n_r receivers, the file is $n_s \times n_r$ row-long. The order of the entries is

```
DO i=1,n_s
     DO j=1,n_r
          switch(j,i),ttime(j,i),terr(j,i)
     END DO
END DO
```

- ***raypath.dat*** · File of raypath coordinates – only used when raypaths are *not* updated at each proposed model (see line 69 of *mksamples.in*). It is a binary file that can be generated using code `fm2dssRJ` (in directory `/EGcodes`), which is a modified version of NR's `fm2dss`.

- ***datasets.txt*** · File containing datasets associations for each traveltime in *otimes.dat*, in the following format:

  ```
  1
  1
  0
  2
  2
  0
  1
  0
  2
  1
  1
  1
  1
  0
  …
  ```

  where each line relates to the corresponding line in *otimes.dat*, and each entry denotes which dataset that traveltime belongs to (values equal to 0 are ignored – but every active raypath in *otimes.dat* needs to have a dataset value >0 in *datasets.txt*). The total number of datasets is given at line 31 of *mksamples.in*. This information is used when, for instance, two different datasets are combined in the inversion, each with a different noise parametrization that we want to invert for.

The code is run for a number of samples that are given at line 14 of *mksamples.in*. Results are then saved to disk as output files ending in *.tmp** (last models of the current run) and *.out** (all models in the chain), where * indicates the processor number. To begin the inversion, set line 13 of *mksamples.in* to 1. To re-start the inversion, set this line to 2, 3, 4… making sure that the directory also contains all the *.tmp** and *.out** files from the previous runs.

N.B. While on eddie the first processor is given rank number 0, within `mksamples` and `procsamples` the numbering of processors (i.e. of Markov chains) starts from 1.

The **output files** are the following:

- ***samples.out**** · Binary file that contains information on the accepted samples along the Markov chain.
- ***ncells.out**** · Text file that contains the number of cells of the accepted samples along the Markov chain.
- ***sigmas.out**** · Binary file that contains the values of noise parameters along the Markov chain. If noise from data is used (i.e. set line 21 of *mksamples.in* to 4 and use the 3rd column of *otimes.dat* as error), only one *.out* file is produced and the *.tmp** files are empty.
- ***misfit.out**** · Text file that contains the values of misfit along the Markov chain.
- ***aratios.out**** · Text file that contains the acceptance ratios along the Markov chain for: all samples, birth, death, move, velocity, sigma change.
- ***parameters.dat*** · Binary file that contains information that is passed between one run of `mksamples` and the next, and then used to process the samples with `procsamples`.
- ***last.tmp**** · Binary file that contains the last velocity model of the current run of `mksamples`, which is used as the starting point for the next run.
- ***sigma.tmp**** · Binary file that contains the last noise parameters of the current run of `mksamples`, which are used as the starting point for the next run.
- ***arats.tmp**** · Binary file that contains the acceptance ratios at the last model of the current run of `mksamples`, which are fed into the next run to calculate global acceptance ratios over all runs.

The *.out** files can be processed using `procsamples` (the executable of *rj_tomo_procsamples.f90*) to get information on average model, posterior on number of cells, noise etc.

rj_tomo_procsamples.f90 · procsamples

This code processes the samples (*.out** files) created using `mksamples`. All input parameters are given in file *procsamples.in*. This file and all other input files have to be located in the same directory as the executable `procsamples`.

The **input files** are the outputs of *mksamples*: *samples.out**, *ncells.out**, *sigmas.out**, *misfit.out**, *aratios.out**, *parameters.dat*.

The **output files** are the following:

- ***Average.out*** · Text file of average velocity at a regular grid of points across the whole ensemble of accepted models. The file has the following format (which is the same as that generated by NR's `grid2dss` code):

```
       209          193
61.00000000   -9.00000000
 0.06250000    0.06250000

3.04283270255864
3.04283270255864
3.04241831032819
3.04249730373439
…
```

  The first line specifies the number of velocity nodes, at which the average is calculated, in [latitude longitude] – this corresponds to the input given at line 20 of *procsamples.in*. The second line gives the coordinates of the NW corner of the area as [latitude longitude] – this corresponds to the input given at lines 26-27 of *mksamples.in*. The third line gives the spacing between the velocity nodes in degrees in the NS and WE direction. The following lines give the average velocity at each velocity node, starting from the grid origin and looping in the order latitude, longitude:

```
DO i=1,nlong
     DO j=1,nlat
          WRITE(*,*)vel(j,i)
     END DO
END DO
```

- ***Stdev.out*** · Text file of velocity standard deviations (model uncertainty) at each point of the regular grid at which the average field is calculated. The file has the same format as *Average.dat*.

- ***Median.out*** · Text file of the median velocity field at each point of the regular grid at which the average field is calculated. The file has the same format as *Average.dat*.

- ***Maximum.out*** · Text file of the maximum-probability velocity field at each point of the regular grid at which the average field is calculated. The file has the same format as *Average.dat*.

- ***Ncellsi.out*** · Text file of window-averaged number of cells across all Markov chains (width of window defined at line 19 of *procsamples.in*).

- ***Misfit.out*** · Text file of window-averaged misfit across all Markov chains (width of window defined at line 19 of *procsamples.in*).

- ***Ncell.out*** · Text file of posterior probability distribution on number of cells (uses the bounds given at line 18 of *mksamples.in*).

- ***Noise.out*** · Text file of posterior probability distribution on noise parameters (uses the bounds given at lines 23-25 of *mksamples.in* and the pixelization given in the 2nd column of line 22 of *procsamples.in*). The number of columns of the file depends on the parametrization and number of datasets.

- ***Vels.out*** · Binary file of posterior probability distribution on velocity at each point of the regular grid defined above (uses the bounds given at lines 19-20 of *mksamples.in* and the pixelization given in the 1st column of line 22 of *procsamples.in*).

- ***Voro.ini*** and ***Voro.fin*** · Text file containing the number and locations of Voronoi nodes for the first and last model of all chains in the following format:

```
          4
57.3642183629247   2.96621770038414   4.40523569515719
49.6656178214274   -8.99862742703376   2.42544752854955
49.9078266036016   2.98848618752986   2.21734991020841
53.1555494410348   -7.89193685327003   2.06081782785587

          3
48.3370409629627   -6.77901502181287   4.40204246955811
60.3204979438615   -2.40405028718618   2.84322176630924
53.4647814215777   -3.18301084409072   2.48263316116518
…
```

  where the integer entries represent the number of cells and each subsequent row gives latitude, longitude and velocity of the Voronoi cell.

- ***Node.ini***, ***Node.fin*** and ***Node.ens*** · Text files of initial, final and ensemble Voronoi-cell density, calculated using pixels of width given at line 21 of *procsamples.in* and centred at each point of the regular grid at which the average field is calculated. The files have the same format as *Average.out*.

- ***First*** and ***Last*** · Text files of initial and final Voronoi cell coordinates, which get renamed with extension *.vor* or *.vtx* and the processor number (e.g. *First_p1.vor*, *First_p1.vtx*). The first three lines of both files have the same format as those of *Average.out*, except the number of velocity nodes (line 1) and the node spacing (line 3) are those defined at line 78 of *mksamples.in* for the calculation of raypaths. The subsequent lines follow the format of *Average.out* in the *.vtx* file, and of *Voro.ini*/*fin* in the *.vor* file.

At line 43 of *procsamples.in* it is possible to define the total number of Markov chains that should be excluded from processing (if, for instance, they haven't converged). The chain numbers are specified in the subsequent lines, one below the other.

Directory MT

This directory contains the random number generator that is used by `mksamples`. The subroutine uses the Mersenne-Twister algorithm and can be downloaded from http://jblevins.org/mirror/amiller/mt19937.f90.

Directory fm2d

This directory contains the subroutines that are used to model raypaths within `mksamples`. These codes are an adapted version of the eikonal ray tracer `fm2dss` by NR, which is part of the FMST code package.

Directories nn and qhull

These directories contain the subroutines that are used by `mksamples` to compute the Delaunay triangulation (Voronoi cells) of the 2D model space. N.B.: a small change has been made to subroutine `delaun` in *nn*/*delaun.f* compared to the original version!

Directory EGcodes

This directory contains a number of additional codes that can be used for further processing, data visualization etc.:

- **bin2txt** · Simple program which converts the binary output raypath file from `fm2dssRJ` into a text file.

- **fm2dssRJ** · Eikonal ray tracer which is a modified version of NR's `fm2dss` and which produces a binary output file of raypaths that is compatible with `mksamples`. Its input file is *fm2dss.in* – see the FMST instructions for a detailed description of the various parameters.

- **gauserr** · This program produces a file of Gaussian errors that can be added as noise to synthetic traveltimes. More than one dataset can be combined – the file of datasets associations is given at line 1 of *gauserr.in*. The output file consists of two columns: the 1st column contains the actual error (as a random deviate from the Gaussian distribution), the 2nd column contains the standard deviation of the distribution (which is given as input at lines 6 and below in *gauserr.in*).

- **gauserrdist** · This program uses the traveltime data in file *otimes.dat* and the noise parameters given in *gauserrdist.in* to add distance-dependent Gaussian noise σ to a traveltime dataset: $\sigma = a \times d + b$, where a and b are defined at line 9 of *gauserrdist.in* and d is the source-receiver distance in degrees. There are two output

files: file *otimes.dat* acts as both input and output (i.e. the traveltimes in the 2nd column are modified by adding the noise, and the 3rd column is substituted with the new values of standard deviation); file *gauserr.txt* has the same format as the output from `gauserr`.

- **mkstraightrays** · Simple program that creates a text and a binary file of 2-point (source-to-receiver) raypaths.

- **raylength** · This program reads the binary raypath file produced by `fm2dss` and produces a text output with raypath lengths in degrees. N.B.: all raypaths must be calculated with `fm2dss`, so the number of rays in the `fm2dss` output must match the number of records in *otimes.dat*. Its input file is *raylength.in*.

- **velpost** · This program reads the binary file of velocity posteriors produced by `procsamples` (e.g. *Vels.out*) and creates text files of posteriors for various points. The points at which the posteriors are calculated must be part of the grid of points that is used when processing the samples. The number of points is specified at line 6 of the input file *velpost.in*, and their coordinates are specified as [latitude longitude] in the subsequent lines.

- **vorogrid** · This program converts a file containing coordinates and velocities of Voronoi cells (such as *First_p1.vor*) into a file that is compatible with the ray tracer `fm2dss`. Its input file is *vorogrid.in*.

- **vororay** · This program models raypaths in a Voronoi-tessellated 2D space and outputs information on the path taken by the different rays through the medium. Its parameter input file is *vororay.in*. Input files include the file of Voronoi cells (such as *First_p1.vor*) and the traveltime file *otimes.dat*. Only the active rays in *otimes.dat* are considered in the output, and raypath and cell numbers are given in the order that they have in the input files. Output files are *rayjour.dat* (in which each line corresponds to a different active raypath and each entry corresponds to the cells traversed by the ray, traced from the receiver to the source), *raylengths.dat* (in which each line corresponds to a different active ray and each column corresponds to a Voronoi cell – each entry gives the length of each ray in each cell), *raycell.dat* (with the same format as the previous, it gives 1 if a cell is traversed by a certain ray and 0 otherwise), and *rayneigh.dat*, which lists the rays that pass through a certain Voronoi cell (given at line 30 of *vororay.in*) and its neighbours.

- **voroslice** · This program converts a file containing coordinates and velocities of Voronoi cells (such as *First_p1.vor*) into a file that is suitable for input into GMT (similarly to `tslicess` in FMST). Its input file is *voroslice.in*.

Directory fmst

This directory contains the source codes, input files and instructions of NR's Fast-Marching Surface Tomography (FMST) package. It is included because a lot of the FMST codes are very useful for visualizing the data by converting it to a GMT-friendly format (e.g. `tslicess`). It also contains the eikonal ray tracer `fm2dss` that has been adapted and integrated into `mksamples` to calculate the raypath geometry. To run an FMST inversion, set the path to the `/bin` directory at line 8 of `ttomoss`.

Script tomosub.sh

The `tomosub.sh` script can be used to submit one `mksamples` job to the ECDF cluster.

Script run_mksamples.exe

In order to automate the job submission, script `run_mksamples.exe` can be run from `/workdir` to automatically start a new run of `mksamples` when the previous run has finished and the results have been saved to disk. Splitting jobs into multiple runs is necessary because of the 48-hour runtime limit and the absence of checkpointing for MPI jobs on eddie.

Plotting scripts

The `plot_rjmcmc.exe` script in `/workdir` can be used to plot the outputs of rj-McMC tomography. It uses the Generic Mapping Tools (GMT). Alternatively, maps can be plotted in Matlab using function `plotvtx.m` in the `/source/matlab_scripts` directory.

Running an inversion and plotting the results: a few steps

Follow these steps to run an automated rj-McMC inversion on eddie with `run_mksamples.exe`, process the results with `procsamples`, and plot maps and histograms using the `plot_rjmcmc.exe` script:

1. Ensure that `run_mksamples.exe` and `plot_rjmcmc.exe` are in the `/workdir` directory and that they are executable (if not, then execute > `chmod +x "scriptname"`).

2. Adjust the various parameters in *mksamples.in* (N.B. at line 69, choose to either update the raypaths at each iteration (=2) or to keep them fixed (=0) – option =1 has not been tested!!). If line 69 =2, make sure that line 71 =1, otherwise the wrong raypath geometries will be used.

3. Set a number of parameters in `run_mksamples.exe`:

    - line 12: the project directory
    - line 19: the number of samples per run (remember that eddie has a 48-hour runtime limit!)
    - line 20: the number of runs
    - line 21: the number of processors to use
    - line 22: the run time
    - line 23: the time interval at which the script checks if a `mksamples` job has completed

4. Move files *sources.dat*, *receivers.dat*, *otimes.dat* to the `/bin` directory. If more than one dataset is used (line 31 >1 in *mksamples.in*), then make sure the `/bin` directory contains the *datasets.txt* file (or whichever file is specified at line 32). If raypaths are kept fixed during inversion, make sure the `/bin` directory also contains the raypath file specified at line 9 of *mksamples.in* – a raypath file can be created using the `fm2dssRJ` code.

5. Run the automated tomography script:

    ```
    > ./run_mksamples.exe
    ```

    The script will progressively submit jobs to eddie, save partial results to disk, and submit a new run when the previous one has finished (up to the number specified at line 20). Partial results will be saved in .zip files in the `/bin/${tomotime}` directory.

6. Once `run_mksamples.exe` has finished running, the results can be processed using `procsamples`. In *procsamples.in*, set the burn-in period at line 14 and the thinning interval at line 15. If some of the Markov chains are chosen to be ignored in processing, set line 43 to 1 and enter the chain numbers (i.e. processor numbers) from line 44 downwards.

7. Use an interactive session to run `procsamples` and plot the results!!

    - `> qlogin`
    - move to `/bin` directory
    - `> ./procsamples`

8. Once `procsamples` has finished running, the results can be plotted using `plot_rjmcmc.exe` (in `/workdir`). Make sure `./plot_rjmcmc.exe` is run in a `qlogin` session!!! Set a number of parameters in `plot_rjmcmc.exe`:

    - line 17: the project directory
    - line 32: the number of processors used in the inversion (same as line 21 of `run_mksamples.exe`)

- line 36: the iteration number in case the raypaths were kept fixed (line 69 of *mksamples.in* =0)
- line 40: choose if the probability histograms should be normalised to 1 or display the number of samples
- lines 43-46: define the coordinates of 4 points for which posterior distributions on velocity should be plotted
- lines 49-52: grid dicing (the larger this number, the less 'pixelated' the maps look – 10 is a good compromise between how the maps look and how slow the code is)
- lines 54-146: file names and parameters are read directly from *mksamples.in* and *procsamples.in*, so these line should not need to be changed

9. Plots will appear in the directory specified at line 22, with the same name as the original text files but as .png image files.

To test the code, an example dataset is provided in the `/example_dataset` directory.