**TMCMC methods code instructions**

**Outline**

* **Overall instructions**
* **Parameters**
  + **TMCMC**
  + **GTMCMC**
  + **T-RJMCMC**
* **How to generate data**

This directory contains codes and tests for all TMCMC methods, including the basic TMCMC, the GTMCMC, and the T-RJMCMC. It contains a library folder (‘lib’) and an application folder (‘app’). Two scripts (‘compile\_libs.sh’ and ‘compile\_apps.sh’) are provided to compile the libraries and the applications. The script ‘run\_test.sh’ runs all examples in the ‘apps’ folder.

**Your system must install an MPI library to compile and run these codes.**

**Overall instructions**

1. Run command

bash compile\_libs.sh

1. Run command

bash compile\_apps.sh

1. Run command

bash run\_test.sh

**(Ignore below description if you don’t have SLURM:**

If you have the SLURM job manager in your system, run ‘run\_test\_slurm.sh’ instead, but you have to modify each sbatch script in the apps folder. )

**Parameters**

**TMCMC**

* algo\_config.dat: hyper-parameters of the algorithm, including:

problem dimension, CoV, number of samples per core.

* prior\_types.dat: Type of prior. Contains N rows, N is the problem dimension. Each row can only be either ‘G’ or ‘U’, representing Gaussian and Uniform.
* prior\_params.dat: prior parameters, containing N rows x 2 columns. N is the problem dimension. For ‘G’ priors, the 2 values are the mean and the std. For ‘U’ priors, the 2 values are the min and max values.
* boundary.dat: boundary of parameters.

**GTMCMC**

CASE 1: Specify the importance distribution as uniform or Gaussian as we did for the prior. This case runs the executable ‘gtmcmc\_lwd’.

Requires all input files of TMCMC. Additional files:

* proposal\_params.dat: same definition as prior\_params.dat
* proposal\_types.dat: same definition as prior\_types.dat

CASE 2: Specify the mean, covariance matrix, inversion of the covariance matrix of the importance distribution. These parameters are usually computed from a set of samples, this is how we run the sequential Bayesian inference experiences.

Additional files:

* proposal\_mean.dat: mean of importance distribution, computed from samples.
* proposal\_covariance.dat: covariance of importance distribution.
* proposal\_cov\_inv.dat: inverse matrix of the covariance.
* proposal\_cov\_sqrt.dat: square root of covariance matrix.

The Python script ‘get\_matrices.py’ in the test folder is used to compute these matrices from samples. The overall work flow is:

GTMCMC for point 1 (importance distribution = prior) -> get samples -> Python script compute required matrices from samples -> GTMCMC for point 2 (use those matrices for importance distribution) -> get samples -> …

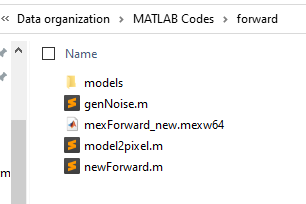
**T-RJMCMC**

* generalPrior.dat: the min and max values of each type of parameter (noise sigmas, resistivity (log-scale), Zbed 1, thickness)
* Zrange.dat: the range for generating new boundary.
* rjSettings.dat: hyper-parameters of the algorithm
  + resistivity sigma
  + Perturb probability
  + Birth probability
  + Min number of layers
  + Max number of layers
* algo\_config.dat: hyper-parameters of the algorithm

number of parameters (only used if you specify generate initial parameters with specified dimension), TMCMC CV, number of samples, acceptance threshold, fix initial dimension (0 for no, 1 for yes), lambda of exponential distribution, I forgot what is the last one, just leave it as is.

**How to generate data**

I usually manually design earth models in MATLAB and use MATLAB to compute their tool responses. The tool responses are saved as ‘\*.dat’ files which are used as the input data for inversion. The MATLAB codes are saved in ‘Data organization/MATLAB codes/forward’. Use the function ‘newForward’ to compute tool responses. The function is explained as commented.



If you save the tool responses as an matrix where is the number of models and is the output dimension, and you name the output as ‘data\_clean’, you run ‘genNoise’ to add **Gaussian noise** to the measurements. Some example data are also saved in the folder.