Craig Allen

8-2-2024

Stites Lab

MCMC Job Guide

*Running MCMC using main.py*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

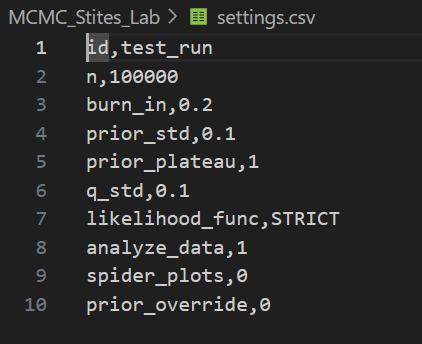
Included in the MCMC\_Stites\_Lab folder is a main.py file and a settings.csv file. With these two files, one can use the command line to run MCMC RAS runs and automatically save the results to a folder within MCMC\_Stites\_Lab/runs. Each run folder will contain a copy of the settings used to run it, run\_settings.csv, an /analysis/ folder with .SVG graphs, and a /data/ folder with csvs with all the MCMC data.

Running the MCMC using the main.py is very straightforward:

1. Open the command line (preferably within an HPC environment or in a Job with access to multiple CPUs that are fast).
2. Enter >>> cd path/to/MCMC\_Stites\_LAB
3. Enter >>> python3 main.py settings.csv

This should run the MCMC RAS algo using the settings within the settings.csv file. Relevant information will be output to the console including common errors. The data is saved *before* the graphs are made and saved. Therefore if the MCMC runs but fails in making or saving the graphs, you can still access the data.

The settings file looks as follows:



Each setting is as follows:

* id: the string name to be used in naming the folder for results and for files within the folder.
* n: the number of accepted, post burn-in iterations to run until.
* burn\_in: the percent (in decimal form) of iterations to burn. For example 0.2 is 20% of the first iterations.
* prior\_std: the standard deviation to be used for all priors. E.g. 0.1 will be a std of 10^0.1. **Note: in future with better priors this could be removed, as in main.py, Parameter priors are defined.**
* prior\_plateau: the KDE density to plateau the prior.
* q\_std: the standard deviation to be used in the proposal step. E.g. 0.1 will be a std of 10^0.1.
* likelihood\_func: sets the likelihood function. Currently, STRICT, SIGMOID, and EXP accepted, will throw error otherwise.
* analyze\_data: 1 has all analysis graphs automatically generated and saved, 0 turns this off.
* spider\_plots: 1 generates and saves all spider plots, 0 turns this off.
* prior\_override: 1 overrides prior so that all prior\_p’s are set at 1 uniformly. 0 turns this off.

If specific Parameters (e.g. only run MCMC on kcat,Km, and GAP), or changes to an individual Parameters mu and/or std is to be made, one can go inside main.py and change these directly.

*Running MCMC using main.py – As a Job*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

As of writing this, with all kinetic parameters and 6 rules with parallel processing, 10,000 steps takes 1-2 hours, 100,000 steps takes 10-15 hours. This is unreasonable for a Code Server session, and longer jobs should be run as a Job on the HPC. Specific usage of the HPC to run jobs is outlined in the HPC Python Usage Guide document.

1. Create new Job and add:
   1. cd path/to/MCMC\_Stites\_Lab
   2. python3 main.py settings.csv
2. Modify job script to run for at least 24 hours, modify partition to ycga, and CPUs per task to at least 6 or more than the number of rule functions.
3. Modify settings.csv to the current settings for your run
   1. **Important to change id as if you don’t, will override existing data.**
4. Modify main.py if necessary for changes in Parameters to be tested.
5. Submit Job 😊