**Introduction**

In this document we will present a description of the code we used on the paper. Its purpose is to help the reader to understand how the code is organized, and what are the main files and functions.

The code is organized in a few main scripts that execute the main algorithms we developed in the paper, from data importation to parameter learning and evaluation. Each part is executed by functions defined in other files.

To run each script, one must keep it in the folder it is given in and also put the relevant data files in it too.

Notice that in order for the code to work, a Numpy version that supports Numba must be installed.

**JMCSM and MCSM evaluation on real data**

**Replication strand as mutation-level covariate**

The first main script provided, ***JMCSM\_NEW\_2fold\_CV\_Main*** is organized as follows:

1. Mutation signature data is imported using the function *Import\_Signatures* (see description below).
2. Mutations data is imported using the function *Import\_Mutation\_Data\_LEADLAGG* (see description below).
3. Mutations data is divided into two sets using the function *data\_2fold\_CV* (see description below).
4. Parameters for *JMCSM* and *gLDA* are learned using *SEM\_JMCSM\_NEW* and *SEM\_bLDA\_NEW* respectively (see descriptions below), for both sets.
5. Held-out log-likelihood is calculated for each set of parameters achieved, using *JMCSM\_NEW\_EL* and *bLDA\_NEW\_EL*. To test convergence, we calculate likelihood given the parameters estimation after every iteration of the learning algorithm.

The second main script provided, ***MCSM\_NEW\_2fold\_CV\_Main*** is organized as follows:

1. Mutation signature data is imported using the function *Import\_Signatures* (see description below).
2. Mutations data is imported using the function *Import\_Mutation\_Data\_LEADLAGG* (see description below).
3. Mutations data is divided into two sets using the function *data\_2fold\_CV* (see description below).
4. Parameters for *MCSM* and *LDA* are learned using *SEM\_MCSM\_NEW* and *SEM\_LDA\_NEW* respectively (see descriptions below), for both sets.
5. Held-out log-likelihood is calculated for each set of parameters achieved, using *MCSM\_NEW\_EL* and *LDA\_NEW\_EL*. To test convergence, we calculate likelihood given the parameters estimation after every iteration of the learning algorithm.

Those scripts also use the following functions:

***Import\_Signatures*** – this function imports the cosmic signatures given in the attached file "cosmic-signatures.tsv". In order to operate, the signature file must be in the working directory. To import the signatures relevant to the datasets we work with, three lists are given in the function, where each one of them contain the signatures that are known to be present in each one of the datasets we used (BRCA, MALY and CLLE). The imported signatures are the ones given in the uncommented list (it is possible of course to import other mutations by uncomment the respective list or editing the uncommented list).

***Import\_Mutation\_Data\_LEADLAGG*** – this function imports the mutations data from one of the three databases we used (BRCA, MALY and CLLE). The data is given in JSON format, in three files: BRCA\_JSON.json, MALY\_JSON.json and CLLE\_JSON.json. In order to import a database, one should make sure that the desired file name is specified in line 12. The data is then organized in a list of tumors, where each tumor is a list of two arrays (one for each strand/feature). Each array has a length of 96 and indicates the number of times a specific mutation category occurred.

***data\_2fold\_CV*** – this function gets as an input the mutation data (in a format like the one generated by *Import\_Mutation\_Data\_LEADLAGG*), and divides it into two separate sets, in order to perform a two-fold cross-validation. As specified in the paper, it is done by splitting every tumor to two sets, by alternately picking mutations for every set. The outputs of the function are the two sets.

***SEM\_JMCSM\_NEW*** – this function executes the stochastic EM algorithm described in the paper that learn the parameters of JMCSM. Its inputs are the train set, the signatures, the number of iterations of the Gibbs' sample (C) and the number of EM iterations (D). It starts with setting and initial parameters guess using *SEM\_init\_params\_JMCSM\_NEW* (see documentation in the code) and setting an initial signature assignment using *Initial\_assignment\_guess* (see documentation in the code). It then runs for D iterations, and do as follows in each iteration:

1. Draw an assignment guess given the current parameter estimation. It is done by drawing a random assignment using *Initial\_assignment\_guess*, and then running a Gibbs sampler using *Gibbs\_Sampler\_I\_JMCSM\_NEW* (see documentation in the code). for the first strand and *Gibbs\_Sampler\_J\_JMCSM\_NEW* (see documentation in the code) for the second one.
2. Update the parameter guess given the current signature assignment, using *Update\_a\_and\_b\_JMCSM\_NEW.*

The outputs of the function are an array of the estimated exposures of each tumor (e), the estimated parameters (a and b), and the history of the parameter estimations (a\_history and b\_history) in lists.

***SEM\_bLDA\_NEW*** - this function executes the stochastic EM algorithm described in the paper that learn the parameters of gLDA. It works similarly to JMCSM with small variations.

***SEM\_MCSM\_NEW*** - this function executes the stochastic EM algorithm described in the paper that learn the parameters of MCSM. It works similarly to JMCSM with small variations.

***SEM\_LDA\_NEW*** - this function executes the stochastic EM algorithm described in the paper that learn the parameters of LDA. It works similarly to JMCSM with small variations.

***JMCSM\_NEW\_EL*** – this function calculates the held-out log-likelihood using the empirical likelihood method described in the paper, for the JMCSM model. Its inputs are the signatures (BRCA\_Signatures), the estimated exposures of the tumors (e) estimated parameters a and b, the test set (counted\_test\_set) and the number of iterations used to calculate the empirical likelihood (S). See documentation in the code.

***bLDA\_NEW\_EL*** – this function calculates the held-out log-likelihood using the empirical likelihood method described in the paper, for the gLDA model. It works similarly to *JMCSM\_NEW\_EL*. See documentation in the code.

***MCSM\_NEW\_EL*** – this function calculates the held-out log-likelihood using the empirical likelihood method described in the paper, for the MCSM model. It works similarly to *JMCSM\_NEW\_EL*. See documentation in the code.

***LDA\_NEW\_EL*** – this function calculates the held-out log-likelihood using the empirical likelihood method described in the paper, for the LDA model. It works similarly to *JMCSM\_NEW\_EL*. See documentation in the code.

**Genomic strand as mutation-level covariate**

In addition to the scripts that use replication strand as a mutation-level covariate, we provide two other scripts that use genomic strand instead, which are named *JMCSM\_NEW\_2fold\_CV\_Main\_Genomic* and *MCSM\_NEW\_2fold\_CV\_Main\_Genomic*. Those two scripts work similarly to the replication strand scripts. The only difference is that they use another function to import the data - *Import\_Mutation\_Data\_Genomic\_ALL* instead of *Import\_Mutation\_Data\_LEADLAGG*. The former is almost identical to the latter. However, in the former the mutation data is imported from one other file, in addition to the JSON file. So, for the scripts to work, one must make sure that the second file is also located in the working directory. Each database has its own unique additional file: BRCA – "mutations.tsv", MALY – "MALY.tsv", CLLE – " PCAWG-CLLE-ES-R27.SBS.tsv". Also, in order to import the correct data given the analyzed database, the correct file name should be filled in line 12. Note that the extra file that are needed for the genomic strand data are too large for Github so they are only available upon request.

**Using the code with different data**

It is possible to use the code with different databases. However, one has that make sure that the database is organized similarly to the databases used here.

**JMCSM and MCSM evaluation on generated data**

We also provide scripts that execute our methods on generated data to show their ability to correctly reconstruct the parameters.

Like previous tests, we provide two different scripts named *Generate\_Then\_Relearn\_JMCSM* and *Generate\_Then\_Relearn\_MCSM*.

It is possible to use different parameters to generate the data. For example, one can change the sigma from which the JMCSM parameters are drawn by changing its value in row 55 in the script *Generate\_Then\_Relearn\_JMCSM*.