This file contains step by step instructions about how to carry out sequencing data analysis for a cysteine scanning mutagenesis library. The parameters that need to be modified by the user are indicated by comments preceded by 'INPUT' in all the Perl scripts. So, users can ctrl+F for 'INPUT' and change values of those fields as per their requirement. Whenever users are required to enter the path for the input files, they can append the path for the 'seq\_analysis' folder to the already indicated downstream path. It is recommended that the user does not change the folder structure, file names and extensions mentioned in the workflow. Users can download 'seq\_analysis.zip' file from GitHub (<https://github.com/skshrutikhare/cys_library_analysis>). This zipped file contains the ‘seq\_analysis’ folder with all the scripts required for analysis. Users can copy it to their location of choice and run all the scripts by following instructions in this document. Sample input (\*.fastq) and output files are also available on the GitHub repository.

All the commands in this document are preceded by a '$' sign. This workflow requires that the Water program is installed on your system for alignment of reads with the wild-type (WT) sequence (http://manpages.ubuntu.com/manpages/trusty/man1/water.1e.html). All scripts generate a LOG file indicating the time required to run the script. Some scripts generate summary\_counts\* files which provide some additional information about the data.

**Dividing sequencing output files into smaller parts**

Sequencing was performed using Illumina MiSeq platform. Sequencing output is in the form of two fastq files called \*R1.fastq and \*R2.fastq.

On a Linux/Unix operating system let 'seq\_analysis' be the main folder where all the analysis scripts are executed. All the commands mentioned below can be executed within this folder.

Copy R1.fastq file to seq\_analysis/seq1\_split.

Copy R2.fastq file to seq\_analysis/seq2\_split.

Split the files into smaller parts in order to make the processing faster. We have calculated the no. of lines such that we get 4 parts containing approximately equal number of lines. We suggest users to do the same so that they don't have to alter the filenames in the scripts. Please note that each read is represented by 4 lines so, all the smaller parts should have line numbers divisible by 4.

For example,

$ cd seq\_analysis/seq1\_split

$ split -d -a 3 -l X R1.fastq initial1a

$ cd seq\_analysis/seq2\_split

$ split -d -a 3 -l X R2.fastq initial2b

X: number of lines in the split parts of the file

Use command 'man split' to get information about all the options.

**Initial filtering**

Before sending the data for sequencing, 3 random nucleotides were added at the start of the reads to ensure proper sequencing of reads with identical Multiplex IDentifier (MID, for details refer to Najar et al., 2017 (doi:10.1016/j.str.2016.12.016)) tags. These 3 N's are removed from the reads at this step. Even if a user’s data does not have such nucleotides added, they should execute these scripts so as to get the output files with the correct naming required by scripts in the next step.

$ cd seq\_analysis/seq1\_split/

$ perl initial\_filter\_fwd.pl

Input files: Initial1a\* - they are already present in this folder.

Output files: 1a\* files, noclass\_seq – containing sequences not containing the expected MID or primer sequence

Ignore the warning messages generated by this script.

$ cd seq\_analysis/seq2\_split/

$ perl initial\_filter\_rev.pl

Input files: Initial2b\* - they are already present in this folder.

Output files: 2b\* files, noclass\_seq – containing sequences without the expected MID or primer sequence

Ignore the warning messages generated by this script.

**Separation into bins**

Data is separated into different bins depending on the MID tag and forward (fwd) or reverse (rev) primer. After this step all reads in a given bin will have the same MID.

$ cd seq\_analysis/cys\_library/seq1\_split

$ perl bin\_all\_fwd.pl

Input files: 1a\* files – provide the path for their location in the script.

Output files: bin\_1a, bin\_1b, bin\_2a etc., undef\_bin.seq1 – containing sequences without the expected MID or primer sequence

Ignore the warning messages generated by this script.

$ cd seq\_analysis/cys\_library/seq2\_split

$ perl bin\_all\_rev.pl

Input files: 2b\* files – provide the path for their location in the script.

Output files: bin\_1a, bin\_1b, bin\_2a etc., , undef\_bin.seq2 – containing sequences without the expected MID or primer sequence

Ignore the warning messages generated by this script.

**Combining the bins**

Both \*R1.fastq and \*R2.fastq files have fwd and rev reads. This step combines all the forward reads from seq1\_split and seq2\_split folders having the same MID tag into one file. This is done for all the reverse reads as well.

$ cd seq\_analysis/cys\_library/BINS\_ALL

$ perl import\_bins.pl

Input files: None required but make sure to edit all the parameters indicated by 'INPUT' in this Perl script.

Output file: import\_bins.sh

$ sh import\_bins.sh

Input files: bin\_1a, bin\_1b, bin\_2a etc. from seq1\_split and seq2\_split folders. This script reads these files from their respective locations.

Output files: bin\_1a.txt, bin\_1b.txt, bin\_2a.txt etc.

**Quality filtering**

Reads are filtered according to their quality scores. Minimum read length cutoff was taken as 75 and quality score cutoff was taken as Q20. If a low quality base was observed at a position, the read was trimmed at that position. The script needs to be modified if user wishes to change the quality score cutoff. Information about the required modifications is mentioned as comments in the script.

$ cd seq\_analysis/cys\_library/BINS\_ALL

$ perl filter\_scores\_Q20.pl

Input files: bin\_1a.txt, bin\_1b.txt, bin\_2a.txt etc. - they are already present in this folder.

Output files: bin\_1a.Q20, bin\_1b.Q20, bin\_2a.Q20 etc.

The LOG\* file created here lists some details about accepted and rejected reads.

Move \*.Q20 files to the 'Qfiltered' folder

**Conversion to FASTA format**

Reads are converted to FASTA format as they need to be aligned with the WT sequence.

$ cd seq\_analysis/cys\_library/BINS\_ALL/Qfiltered

$ perl Convert2Fasta.pl

Input files: \*a.Q20 files - they are already present in this folder.

Output files: \*.fasta files

Move these files to the 'fasta\_files\_fwd' folder.

This is a necessary step as revComp\_Convert2Fasta.pl will otherwise act on these .fasta files.

$ perl revComp\_Convert2Fasta.pl

Input files: \*b.Q20 files - they are already present in this folder.

Output files: \*.fasta files

Move these files to the 'fasta\_files\_rev' folder.

**Alignment with WT sequence**

$ cd seq\_analysis/cys\_library/BINS\_ALL/Qfiltered/Align\_input

Move \*.fasta files from the 'fasta\_files\_fwd' folder to Align\_input/fwd\_reads folder.

Move \*.fasta files from the 'fasta\_files\_rev' folder to Align\_input/rev\_reads folder.

Also copy the WT sequence in fasta format to the 'fwd\_reads' and 'rev\_reads' folders. Sample WT sequence is present in these folders.

$ cd seq\_analysis/cys\_library/BINS\_ALL/Qfiltered/Align\_input/fwd\_reads/

$ perl run\_water\_fwd.pl

Input files: fwd\_reads/\*.fasta files – provide the path for their location in the script.

Output files: \*.out files

$ cd seq\_analysis/cys\_library/BINS\_ALL/Qfiltered/Align\_input/rev\_reads/

$ perl run\_water\_rev.pl

Input files: rev\_reads/\*.fasta files – provide the path for their location in the script.

Output files: \*.out files

All parameters were kept at default values except the Gap Opening Penalty, which was increased to 20 while running the Water program. Users can modify the command in the run\_water\_\*.pl script to make changes to the parameters. We do not recommend changing these parameters.

run\_water\_\*.pl scripts create a wt\_seq.out file. As the script reads all .fasta files, the wt seq file is also read and is aligned with itself. This file can be deleted as it is not required for analysis.

**Reformatting the alignment**

Move the alignment output files for fwd reads (Align\_input/fwd\_reads/\*.out) to the folder 'fwd\_reads/fwd\_watout\_Q20'.

Move the alignment output files for rev reads (Align\_input/rev\_reads/\*.out) to the folder 'rev\_reads/rev\_watout\_Q20'.

$ cd seq\_analysis/cys\_library/BINS\_ALL/Qfiltered/Align\_input/fwd\_reads/fwd\_watout\_Q20

$ perl reformat\_watout\_fwd.pl

Input files: \*.out - provide the path for their location in the script.

Output files: \*.Q20 files

$ cd seq\_analysis/cys\_library/BINS\_ALL/Qfiltered/Align\_input/rev\_reads/rev\_watout\_Q20

$ perl reformat\_watout\_rev.pl

Input files: \*.out - provide the path for their location in the script.

Output files: \*.Q20 files

**Combining the alignments**

In this step, alignments for overlapping reads will be combined. Reads which are not identical in the overlapping region are omitted. For non-overlapping reads, the gap between fwd and rev reads is filled by the WT sequence. Reads without a corresponding fwd or rev partner are used without any modifications, assuming that the missing sequence is WT.

$ cd seq\_analysis/cys\_library/BINS\_ALL/Qfiltered/Align\_input/combine\_all

$ perl combine\_all.pl

Input files: \*.Q20 files from 'fwd\_watout\_Q20' and 'rev\_watout\_Q20' folders - provide the path for their location in the script.

Output files: bin\*.combined\_all files,

summary\_counts\_combine\_all.txt – counts of different types of reads,

check\_these\_reads.txt – this file should be empty.,

subset.txt – reads where rev read ends before the fwd read.,

diffolap.txt – reads differing in overlapping region,

short\_reads.txt – reads which don't span the entire sequence length,

reads\_indels.txt – reads containing insertions and deletions

**Obtaining substitution information**

This step generates a table listing substitutions observed in each read. Single mutants were identified from this table using 'classify\_substitutions.pl' script.

$ perl get\_substitution\_table.pl

Input files: bin\*.combined\_all files - provide the path for their location in the script.

Output files: Table\_sub.txt, summary\_counts\_get\_substi.txt – counts of different types mutations

Ignore the 'uninitialized value in concatenation' warnings generated by this script. They are due to presence of incomplete codons which will be removed at later stages.

$ perl classify\_substitutions.pl

Input file: Table\_sub.txt – this file is already present in the folder.

Output files: single\_muts.txt – reads with single mutation,

multi\_muts.txt – reads containing more than 1 mutation

wt.txt – number of WT reads at each MID,

summary\_counts\_classifyMuts.txt – counts of reads with WT, single, double, triple and multiple (>3) mutants

**Calculation of cysteine mutant frequencies**

$ perl get\_CYS\_freq.pl

Input files: single\_muts.txt - this file is already present in the folder.

Output files: Freq\_CYS\_mutants.txt

summary\_counts\_Freq.txt – counts of complete and incomplete codons

$ perl get\_CYS\_freq\_reslevel.pl

Input files: Freq\_CYS\_mutants.txt - this file is already present in the folder.

Output files: Freq\_CYS\_mutants\_res-level.txt

Freq\_CYS\_mutants\_res-level.txt file lists the frequency of a cysteine mutation at each position in the protein of interest at each experimental condition (corresponding to each MID tag). The counts from this file can be normalized with respect to the total no. of reads at each MID and then used for comparison across MIDs. For our CcdB work, we did an additional normalization with respect to the expression MID. Details are explained in 'Deep Sequencing of Sorted Population' section of Najar et al., 2017 (doi:10.1016/j.str.2016.12.016).