**General notes:**

These scripts have been tested on Python 3.7.4 and run from the MacOS 12.4 terminal. Most of the scripts require BioPython and/or NumPy.

**Scripts for processing X5-Y-X5 library screening data**

translateUnique.py

This script will translate the reads in a FASTQ file using a standard genetic code, assuming a reading frame starting from the first base. For sequences with a length that is not a multiple of three, any bases after the last full triplet codon are ignored. The required input is a SampleName.trimmed.fastq file, which we generally produce after trimming a paired-end merged file using Cutadapt. The primary output file is a FASTA-format file with translated reads. In addition, this script will identify all of the unique translated reads in the file, along with the number of counts for each unique sequence. These will be saved in three text files, with one sequence per line. One file has the full list of sequences/counts, a second file only has those sequences without stop codons, and the third file only has those sequences with one or more stop codons. The input and output file names must be changed within the script, and the input files should be in the same working directory as the script.

AA-count-nostop.py

This is the standard script that we use to generate the position-specific amino acid counts table from an X5-Y-X5 library screen. The required input is a SampleName.translate.fasta file that has been generated by the translateUnique.py script. The output is a tab-delimited text file that contains an 11x21 matrix, where each column represents a position in an 11-residue peptide (-5 to +5 around a central tyrosine), and each row represents an amino acid (alphabetical order) or the stop codon (21st row). This script only counts sequences in the translated reads file that are 11 residues long, contain a central tyrosine, and do not contain any stop codons. The input and output file names must be changed within the script, and the input files should be in the same working directory as the script.

AA-count-full.py

This script is analogous to AA-count-nostop.py, except that it counts all 11-residue sequences, including those that contain non-central tyrosine residues and stop codons. The input and output file names must be changed within the script, and the input files should be in the same working directory as the script.

AA-count-1stop.py

This script is analogous to AA-count-nostop.py, except that it exclusively counts 11-residue sequences that contain a central tyrosine residue and exactly one stop codon. The input and output file names must be changed within the script, and the input files should be in the same working directory as the script.

batch-translateUnique.py

This is the batch version of the translateUnique.py script, which can be used to translate multiple trimmed sequence files with one command. Any .trimmed.fastq files that are in the same working directory as the script will be analyzed, and a corresponding translated files and lists/counts of unique sequences will be generated.

batch-AA-count-nostop.py

This is the batch version of the AA-count-nostop.py script, which can be used to process multiple translated sequence files with one command. Any .translated.fasta files that are in the same working directory as the script will be analyzed, and a corresponding text file with a counts table will be generated.

batch-AA-count-full.py

This is the batch version of the AA-count-full.py script, which can be used to process multiple translated sequence files with one command. Any .translated.fasta files that are in the same working directory as the script will be analyzed, and a corresponding text file with a counts table will be generated.

batch-AA-count-1stop.py

This is the batch version of the AA-count-1stop.py script, which can be used to process multiple translated sequence files with one command. Any .translated.fasta files that are in the same working directory as the script will be analyzed, and a corresponding text file with a counts table will be generated.

**Scripts and files for scoring sequences using X5-Y-X5 library data**

score\_peptide\_nostop.py

This script is used to score a list of peptides using the specificity profiles (position-weighted scoring matrix) from a screen with the X5-Y-X5 library. The script requires three input files: (1) the counts matrix for an unselected (reference) sample, generated using the AA-count-nostop.py script, (2) the counts matrix for a selected (test) sample from kinase or SH2 profiling, generated using the AA-count-nostop.py script, and (3) a text file containing a list of peptides to score, with one peptide per line (see example file ‘peptides.txt’). The peptides should be the same length as the number of columns in the scoring matrix (11 residues). The input and output file names must be changed within the script, and the input files should be in the same working directory as the script.

peptides.txt

A sample file containing the sequences of 12 peptides that were used for kinase validation experiments in this study.

**Scripts and files for processing pTyr-Var library screening data**

countPeptides.py

This script is used to count the number of reads in a trimmed sequence file associated with every peptide sequence in a supplied list. The script takes, as input, a file named SampleName.trimmed.fastq, translates every read, and assesses whether that translated sequence exists in a provided list of peptide sequences. The resulting list of sequences and their counts is produced as a text file. The pTyr-Var\_full.txt file is given as an example list that contains all 9,898 unique sequences in the pTyr-Var Library. The input and output file names must be changed within the script, and the input files should be in the same working directory as the script.

countPeptides-var-ref.py

This script is a variation on countPeptides.py that takes three input files: (1) the list of variants in the pTyr-Var Library (pTyr-Var\_variant.txt), (2) a matched file containing the corresponding reference sequences (pTyr-Var\_reference.txt), and (3) a SampleName.trimmed.fastq file from a pTyr-Var screen. The resulting output is a tab-delimited text file where each line contains a different peptide pair, and the columns are: (1) the variant sequence, (2) variant counts, (3) reference sequence, (4) reference counts. The input and output file names must be changed within the script, and the input files should be in the same working directory as the script.

batch-countPeptides.py

This is the batch version of the countPeptides.py script, which can be used to process multiple trimmed sequence files with one command. Any .trimmed.fastq files that are in the same working directory as the script will be analyzed, and a corresponding text file with a counts list will be generated.

batch-countPeptides-var-ref.py

This is the batch version of the countPeptides-var-ref.py script, which can be used to process multiple trimmed sequence files with one command. Any .trimmed.fastq files that are in the same working directory as the script will be analyzed, and a corresponding text file with a counts list will be generated.

pTyr-Var\_full.txt

This text file contains a list of all unique peptide sequences in the pTyr-Var Library.

pTyr-Var\_variant.txt

This text file contains a list of all variant peptide sequences in the pTyr-Var Library.

pTyr-Var\_reference.txt

This text file contains a list of all reference peptide sequences in the pTyr-Var Library.