**Instructions for running Python Scripts associated with Stroik et al**

**THETA PROCESSIVITY V83**

Script is designed to search for slippage of polymerase theta at sites of homopolymeric repeats (‘TTT’ repeats in this case). The code is highly specialized for the substrates we designed for this experiment and would need significant modification to be adapted for other DNA repair substrates.

Input is .csv file with a list of all .csvs you wish to analyze in an experiment. Output will be a unique file for each input file including counts and indel frequency at each TTT repeat position. Script will also output a compiled .csv for the full experiment for ease of comparing indel frequencies at all positions across different samples in an experiment.

1. Enter name of .csv file with list of all .csv files you wish to analyze in a single experiment. When the program is complete, it will print:

Process finished with exit code 0

If the code ever prints “Process finished with exit code 1”…something went wrong and it will give you an error code that should give you some idea of the issue.

**TMEJ PREDICTOR V1**

Script is designed to predict possible TMEJ products at the input locus, including microhomology-mediated deletions (MHD), inverted repeat TINS (irTINS), and strand switching TINS (ssTINS). These product classes are all defined in **Stroik S\*, Luthman AJ\*, and Ramsden DA (2023) Environ Mol Mutagen. PMID: 37438951 DOI: 10.1002/em.22564.** Direct repeat TINS (drTINS) are not worth attempting to predict as they originate from failed MHD events. You can anticipate drTINS priming from the same microhomologies identified as MHD products in blue on the graph.

Several of our lab’s most common loci are already included in the code (Lines 157-195). Script will prompt manual input of locus name. If it matches one of ours, script will run to completion with no further input. In order to customize script to a locus not yet listed, if locus input name does not match the script will prompt you to manually input either left and right flank sequences, or the full locus sequence and Cas9-gRNA sequence in 5’-NGG-3’ orientation. If locus sequence and guide option are chosen, script will identify cutsite from this information and proceed with no further input necessary.

1. Choose locus – remove the pound sign to “un-comment” each line and add pound signs in front of whichever locus definitions you wish to turn off
   1. OPTION 1 – select pre-entered locus and guide sequence (lines 157-195). I included several of our most common analysis sites as examples
   2. OPTION 2 – enter locus and guide sequences manually in either NGG PAM orientation (lines 200-204). The locus sequence and raw reads must be in the same orientation. I have provided a separate small module titled “library\_revcom.py” that will convert all reads in a library to the reverse complement sequences if you need to do so to simplify analysis.
   3. OPTION 3 – Manual input of left and right flanks (lines 197-198). Again, the sequence orientation needs to match the orientation of the raw sequence reads
2. Hit enter…and you’re done! When the program is complete, it will print:

Process finished with exit code 0

If the code ever prints “Process finished with exit code 1”…something went wrong and it will give you an error code that should give you some idea of the issue.

This program will output a .csv file named ‘TMEJ\_predictor\_[locus\_name].csv’. Predicted TINS products are slightly more complicated to convey information via text. We have found this script very useful in automatically predicting TMEJ outcomes and using this to guide our experiments so we know which products to look for by sequencing or quantitative/digital Please note that this script defines ALL POSSIBLE outcomes of TMEJ by searching for microhomologies in the flanks adjacent to a Cas9 double strand break. Not all of these products will occur and some of them may be incredibly rare. We recommend using this as a guide as to what products to consider, and nothing more.

Due to the somewhat complicated format of the output .csv, I have also developed an R script to plot predicted outcomes (also uploaded to Github in the same project directory). You will need the following packages to properly display the graph in R as I intend it: ggplot2, ggiraph, and ggrepel. The charts can get quite busy with all predicted outcomes, so I used ggiraph to make it interactive and allow users to look at individual classes or “families” of products by hovering over them. To run the R script, make sure the output .csv from the Python script is placed in the same folder as your R packages, and change the file name to match.



