**Recombination Analysis Instructions**

1. Trim reads with appropriate software
2. Concatenate unzipped fastq files (with non-duplicate names 🡪 if fastq reads have duplicate names, rename).
3. Bowtie-build (NOT bowtie2) Reference\_Index for reference genome. Reference genome must have 150 “A” pad at end of fasta sequence.
4. Run ViReMa:

python2 ViReMa\_0.20/ViReMa.py Reference\_Index sample.fastq sample.sam --Output\_Dir sample\_virema/ --Output\_Tag sample\_virema -BED --p 4 --MicroInDel\_Length 5 --Defuzz 0 -FuzzEntry

1. Samtools transformations:

samtools view -b sample.sam > sample.bam

samtools sort -o sample\_sort.bam sample.bam

samtools index -b sample\_sort.bam sample\_sort.bam.bai

samtools idxstats sample\_sort.bam

samtools depth -a -m 0 sample\_sort.bam > sample\_coverage.txt

1. Pull the following files:

sample\_virema\_Virus\_Recombination\_Results.txt

sample\_virema\_Virus\_Recombination\_Results.bed

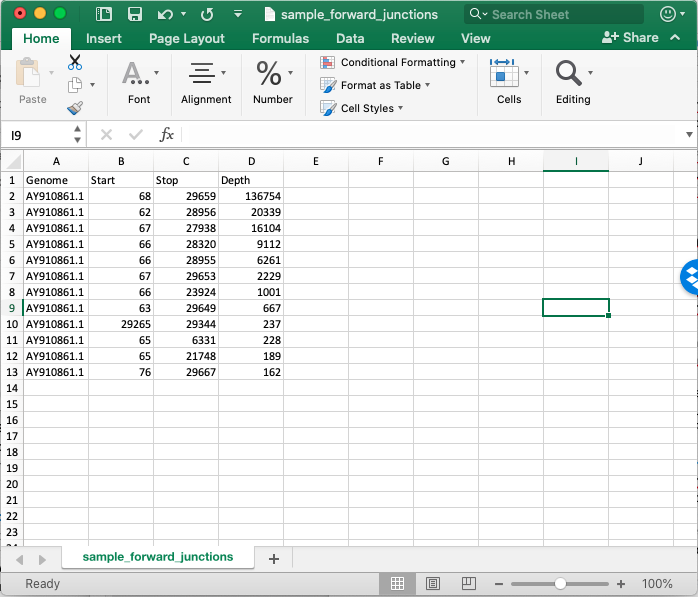
sample\_coverage.txt

1. Calculate junction frequency (junctions per 10,000 mapped nucleotides)

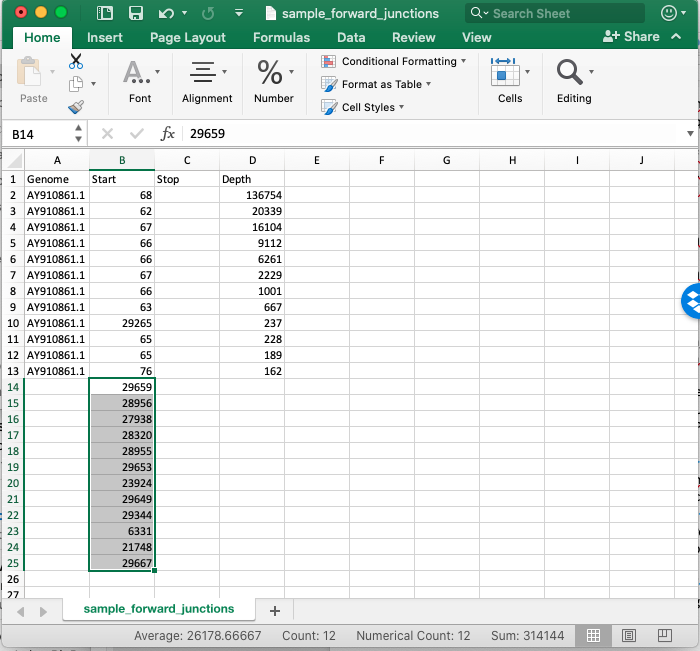
Junction frequency = (Sum of junction depths / sum of all position depths)\*10000

Junction depth is the 5th column of the BED file. Position depth is the 3rd column of the coverage file.

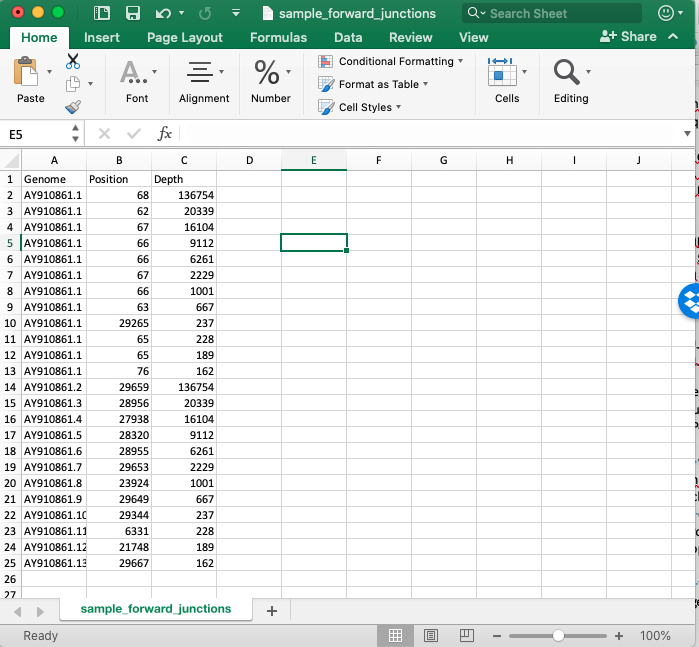
1. Record # of detect junctions (# of rows in the BED file)
2. Generate junction plots (Junction\_Pattern\_Plots.R) using BED File as input
3. Calculate recombination frequency at each position:
   1. Take forward junction file (output of Junction\_Pattern\_Plots.R) and create a 3-column data frame with names “Position” “Genome” “Depth”.
      1. Step 1: Slice Genome, Start, Stop, Depth columns



* + 1. Step 2: In Excel, cut the stop positions and append underneath the start positions in column 2.



* + 1. Step 3: Delete Stop column and fill Genome column.
    2. Copy Depth column and append underneath Depth column values in exact same order. Rename Start column to “Position”



* + 1. Save as sample\_forward\_single.txt
  1. Use PRF\_calculation.R. Input files are sample\_forward\_single.txt and coverage file.

1. DVG v. sgmRNA filtering (Input files is sample\_forward.txt)
   1. sgmRNA\_filtering\_SARS2.R (windows are determined by Kim et al, *Cell*, 2020 and match NC\_045512.2 and MT020881.1).
   2. Explanation of output files:
      1. “Canonical” sgmRNAs are the most abundant of a species
      2. “Alternative” sgmRNAs are all other less abundant species
      3. Sample\_canonical\_sgmRNAs.txt = tab-delineated file of the different canonical sgmRNA junction species with columns matching “Genome”, “Start”, “Stop”, and “Depth”
      4. Sample\_alternative\_sgmRNAs.txt = tab-delineated file of each distinct junction corresponding to a different alternative sgmRNA with columns matching “Genome”, “Start”, “Stop”, and “Depth”
      5. Sample\_total\_sgmRNAs.txt = tab-delineated file of all sgmRNA-forming junctions
      6. Sample\_alt\_sgmRNA\_summary.txt = tab-delinated file of the type of sgmRNA species (sgmRNA2, sgmRNA3, etc) and the total depth for each population
      7. Sample\_DVGs.bed.txt = BED-formatted file of junctions forming DVGs
   3. To calculate junction frequency of canonical sgmRNAs, alternative sgmRNAs, and DVGs (as well as specific sgmRNA species), divide the total depth of a population by the total mapped nucleotides in a population (from the coverage file) and multiply by 10000.
   4. To calculate proportions of populations, calculate % of population, with the denominator being the “Total recombined nucleotides”, ie depth of canonical sgmRNA depth + alternative sgmRNA depth + DVG depth.
2. % ACGU (input file = sample\_DVGs.bed.txt) (UNWEIGHTED by depth)

sort -k4 -rn sample.bed.txt > sample\_sort.bed.txt

python Rec\_Site\_Extraction.py --Window 40 sample\_sort.bed.txt reference.fa sample\_output.txt

* 1. The output is a tab-delineated file with both start and stop sequences listed in columns 10 and 11, respectively.
  2. If you define a window of 40, each sequence string will be 81 in length. The first 40 nucleotides are upstream of the junction site and the last 40 are downstream. The 1 nucleotide in the middle is the junction-forming nucleotide.
  3. %ACGT.R
     1. Make sure to the change the nrow variable in the matrix to match the number of nucleotides in your sequences. And the position labels of the output file. I always label so the +1 position is the junction forming in the resulting DVG sequence, ie Start sequences have +21 to -20 to account for junction-forming nucleotide being a part of the upstream sequence and Stop sequences have +20 to -21 to account for junction-forming nucleotide being a part of the downstream sequence.

1. uHomologyPlots.py (input is sample\_Virus\_Recombination\_Results.txt)
2. Shannon\_Entropy.py (input is sampe\_Virus\_Recombiantion\_Results.txt)
   1. If desired, remove library headers to make single list of data to get single Shannon entropy index. Use average depth across virus in input as “considering WT.”