**Introduction to Next Generation Sequencing: Assembly**

**Preliminary**

As a general rule, the Linux terminal, the Mac terminal, and the Linux subsystem terminal for Windows (Ubuntu) are interchangeable, and any of the three can be used. Also as a general rule, the naming conventions of files in these labs matter. Using files not named based on the templates in this workflow may result in unexpected behaviors.

Some programs will need to be installed along the way. To install programs through the terminal, use the command:

sudo apt install ProgramName

If problems arise, try:

sudo apt-get update && apt-get ProgramName

sudo apt install ProgramName

You will probably get errors along the way because you don’t have some of the dependencies to “make” a file. If you see something like “fatal error: NNNNNNNN No such file or directory”. Copy and paste the error into your browser and somebody from GitHub or StackOverflow will tell you what you need to install (basically some other software development programs). Install these and try again, you may get further into the build process and then get a new error. Copy-paste, find, and install the new programs and try again. It’s an iterative process. I will be there to help with the process.

**Starting Files**

We will start with the files we ended with last week. Specifically, the trimmed reference genome “RumexNepalensisReference\_Trimmed.fasta”, and the cleaned read archive “RumexAcetosaArchive\_trimmed.fq”.

**Map Reads to Reference**

The next step is to map the cleaned read libraries to the reference. This is accomplished in the terminal by the program BWA (Burrows-Wheeler Aligner), which has several mapping algorithms. We will use the BWA-MEM algorithm with the default settings.

Make sure BWA is installed, by executing:

sudo apt install bwa

We will use the default conditions for bwa-mem. Two commands are necessary, one to index the .fasta reference file, and a second to perform the mapping (both are executed all on one line).

bwa index RumexNepalensisReference\_Trimmed.fasta

bwa mem RumexNepalensisReference\_Trimmed.fasta RumexAcetosaArchive\_trimmed.fq > RumexAcetosaArchive\_Trimmed\_SAM.sam

Depending on how big the read archive and how long the reference, BWA may take a long time to run (days). I have chosen a smallish read archive (the smallest I could find!). Additionally, the resulting .sam files will be very large when mapping deep libraries.

There will be a number of files generated by the indexing of the .fasta reference. These files are used by bwa, some are human readable, some are not. The expected output of bwa will be one .sam file for every read archive. The .sam file contains information about the mapping of each read to the reference, as well as the quality of that mapping. The SAM file is not human readable.

**Convert SAM to BAM**

The .sam file needs to be converted to a .bam file. The information will be the same, indicating where each read in the archive matches the reference, or, if the read does not match the reference at all. But the BAM file is the format that is needed for the next step. This task is accomplished by samtools. The input files are the .sam files generated as output from BWA.

Make sure that samtools is installed, by executing:

sudo apt install samtools

The conversion to a .bam file is then accomplished by the following code (all on one line):

samtools view -S -b RumexAcetosaArchive\_Trimmed\_SAM.sam > RumexAcetosaArchive\_Trimmed\_BAM.bam

The expected output will be a .bam file. The BAM file is also not human readable.

**Sort the BAM**

The .bam file must be sorted to put the reads in the order that they map to the reference. This is necessary to calculate a number of metrics about the mapping. This task is also accomplished with samtools. The input files are the output from the previous step (BAM file).

We have already installed samtools, so no need to install anything new. The command to sort the BAM file is accomplished by the following code (all on one line). Notice also the weird syntax, the outfile (following -o) is defined before the infile, which is defined last.

samtools sort -o RumexAcetosaArchive\_Trimmed\_SortedBAM.bam RumexAcetosaArchive\_Trimmed\_BAM.bam

The expected output will be one sorted BAM file. The sorted BAM file is also not human readable.

**Check Coverage**

The mapping is complete and the files have been converted into a form that is useful for our next steps. The next step is to check the mapping to examine how good it is. This task is also accomplished through samtools, and in two ways. The first check is a visual assessment of the mapping. There are two commands. The .sorted.bam file must be first indexed:

samtools index -b RumexAcetosaArchive\_Trimmed\_SortedBAM.bam

And then samtools tview is used:

samtools tview RumexAcetosaArchive\_Trimmed\_SortedBAM.bam

Immediately the terminal will turn into an interactive viewer showing the mapped reads and their position relative to the reference. You can scroll this viewer using the arrow keys. It is not necessary to view the entire mapping. Look for indels and examine to see if there are large numbers of SNPs at the edges of the indels (a lot more than you see elsewhere in the sequence). If so, this is likely an artifact and the bwa parameters (open indel penalty and/or match quality necessary for a successful read mapping) need to be adjusted. To exit the viewer do CTRL+c or CMD+c (for Mac).

The next way to examine the quality of the mapping is through an examination of coverage statistics. There are two major statistics we will look at. First, we will look at the percent of the reference covered by the read archive. Second, we will look at average coverage depth. Coverage depth is the number of reads that match to a particular site in the reference.

This task is accomplished through samtools (and an R script). First, we need to use samtools to extract the coverage depth of the reads mapped at every site/position on the reference.

samtools depth RumexAcetosaArchive\_Trimmed\_SortedBAM.bam -aa > Coverage.csv

The expected output file will have three tab-delimited columns. The first column indicates the read archive, the second indicates the position in the mapping, and the third column indicates the coverage at that position.

Once these file has been generated, the next step is to generate the basic statistics. Older versions of samtools did this automatically but the most recent versions have removed this functionality. As a result, we need to do this ourselves with an R script. I will do this, but the script used can be found on Moodle.

**Make Pileup**

If the coverage looks acceptable, the next step is to make the pileup file, a preliminary step to generating the .fasta sequence of the mapping. This task is also accomplished by samtools. The input file will be the sorted BAM file from the previous step. Execute the following code (all on one line):

samtools mpileup -ABf RumexNepalensisReference\_Trimmed.fasta RumexAcetosaArchive\_Trimmed\_SortedBAM.bam > RumexAcetosaArchive\_Trimmed.pileup

The .pileup file is human readable and can be inspected by using:

head -20 RumexAcetosaArchive\_Trimmed.pileup

tail -20 RumexAcetosaArchive\_Trimmed.pileup

**Collapse Pileup to Fasta**

The final step is to collapse the .pileup files into simple .fasta files, the complete assembly! This task is accomplished with angsd. Unfortunately, the installation and execution of this program is more complicated. It cannot be installed directly on the system. Instead, we will download and compile the program in its own directory (folder). To install and compile/make the program (each command on its own line):

wget http://popgen.dk/software/download/angsd/angsd0.938.tar.gz

tar -xzvf angsd0.938.tar.gz

cd htslib;make;cd ..

cd angsd

make HTSSRC=../htslib

The expected result should be two folders “angsd” and “htslib”. The necessary input files need to be moved into the “angsd” folder. These are three files: 1) the .pileup file, 2) the reference plastome .fasta file, and 3) the indexed reference plastome (.fasta.fai). The indexed reference fasta should have already been generated in a previous step. Then, navigate into the “angsd” folder in the terminal and execute the following code (all on one line):

./angsd -doFasta 2 -pileup RumexAcetosaArchive\_Trimmed.pileup -nInd 1 -doCounts 1 -fai RumexNepalensisReference\_Trimmed.fasta.fai -out RumexAcetosa\_FinalAssembly

In the output files there should be “RumexAcetosa\_FinalAssembly.fa”. This is the final assembly in FASTA format. This .fa file can be viewed in any text editor (provided it isn’t too large of a file).

Keep all of the files we generated this week as we will need them for next week.