**Introduction to Next Generation Sequencing: Cleaning**

**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

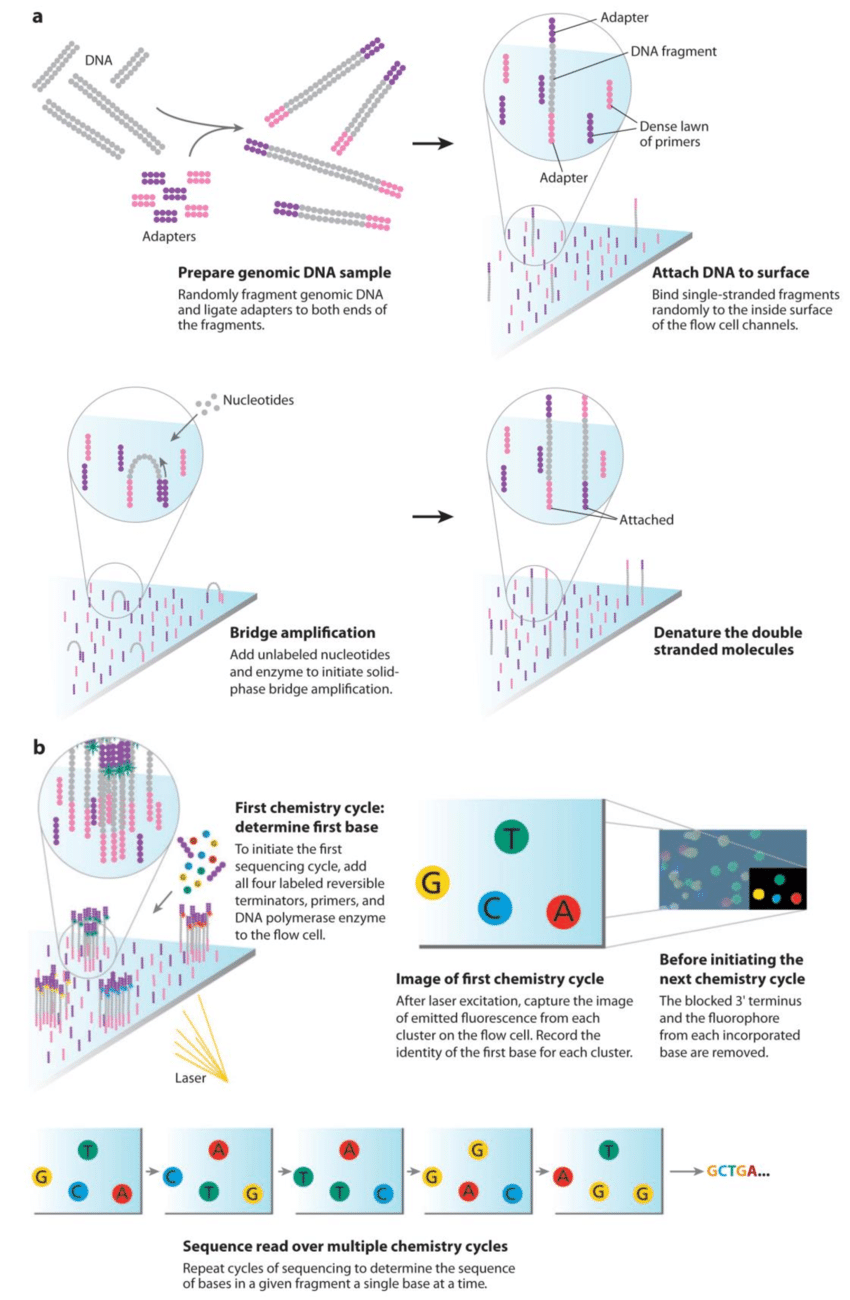
sudo apt install ProgramName

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.

**Introduction**

The ability to sequence DNA in an efficient and reliable way was one of the major advancements in the field of biology during the latter part of the 20th Century. The key breakthrough in this process was the invention of the so-called Sanger Sequencing method. While Sanger Sequencing was a huge improvement over existing methods and opened the world of DNA sequencing to research groups with even modest resources, it had some major limitations. The first limitation was the maximum length of the sequence that could be read. Sanger sequencing at most can sequence a few thousand base pairs at a time. Second, because Sanger Sequencing relied on the use of primers to designate the region in the genome to be sequenced, it meant that a researcher already had to know something about the DNA sequence, limiting the kinds of research questions that could be asked.

In the 1990s and early 2000s several new techniques were developed, and gradually became commercially available, that were able to get around the limitations of Sanger Sequencing. Collectively, these new sequencing methods are known as “Next Generation Sequencing” (NGS) methods, or “Second Generation Sequencing” methods (Roche Pyrosequencing, Illumina, PacBio, and Oxford Nanopore). These methods vary quite a bit, but have in common the ability to sequence regions with no prior knowledge, to sequence a large number of DNA segments in parallel, and to sequence longer sections of DNA, either as a result of the method itself or through computer algorithms that assemble a series of short sequences into a single long sequence. Illumina sequencing, which we will use today, can be seen below.



**doi.org/10.13140/RG.2.2.29564.08327**

The results of NGS sequencing also differ from the results of Sanger Sequencing. The typical output of a Sanger Sequenced DNA fragment was a FASTA file. This file has two parts, the name of the sequence, preceded by a “carrot” (>), with the sequence being placed on the next line and any subsequent lines necessary.

**FASTA Format**

>SEQUENCE\_NAME

TAAGTGAATTTTGTTTCATTAGGGCGAAGTTCTGTAAAAACCTCTGCTTTCTTTAAAA

NGS methods generate a slightly different file called a FASTQ. Essentially it is a FASTA file but with the addition of quality scores for each base. In other words, how confident we are that the base given in the output was the actual base in the sequence. Line 1 has the name of the DNA sequence preceded by the “at’ symbol. The next line or lines are the sequence itself. The third line is the name of the sequence again, this time preceded by the “plus” sign. And line 4 displays the quality score for each base in the sequence. Together these four lines constitute a “read” or single sequence of DNA with a quality score. For most NGS methods, the FASTQ file that is generated will be quite large, containing millions of “reads” one after another all in FASTQ format.

**FASTQ Format**

@SEQUENCE\_NAME

GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAG

+SEQUENCE\_NAME(Optional)

!''\*((((\*\*\*+))%%%++)(%%%%).1\*\*\*-+\*''))\*\*55CCF>>>>>>CCCCCC

**FASTQ Quality Scale**

!"#$%&'()\*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^\_`abcdefghijklmnopqrstuvwxyz{|}~

Lowest Quality ----------------------------------------------------------------------- Highest Quality

**Starting Files: Genome Assembly**

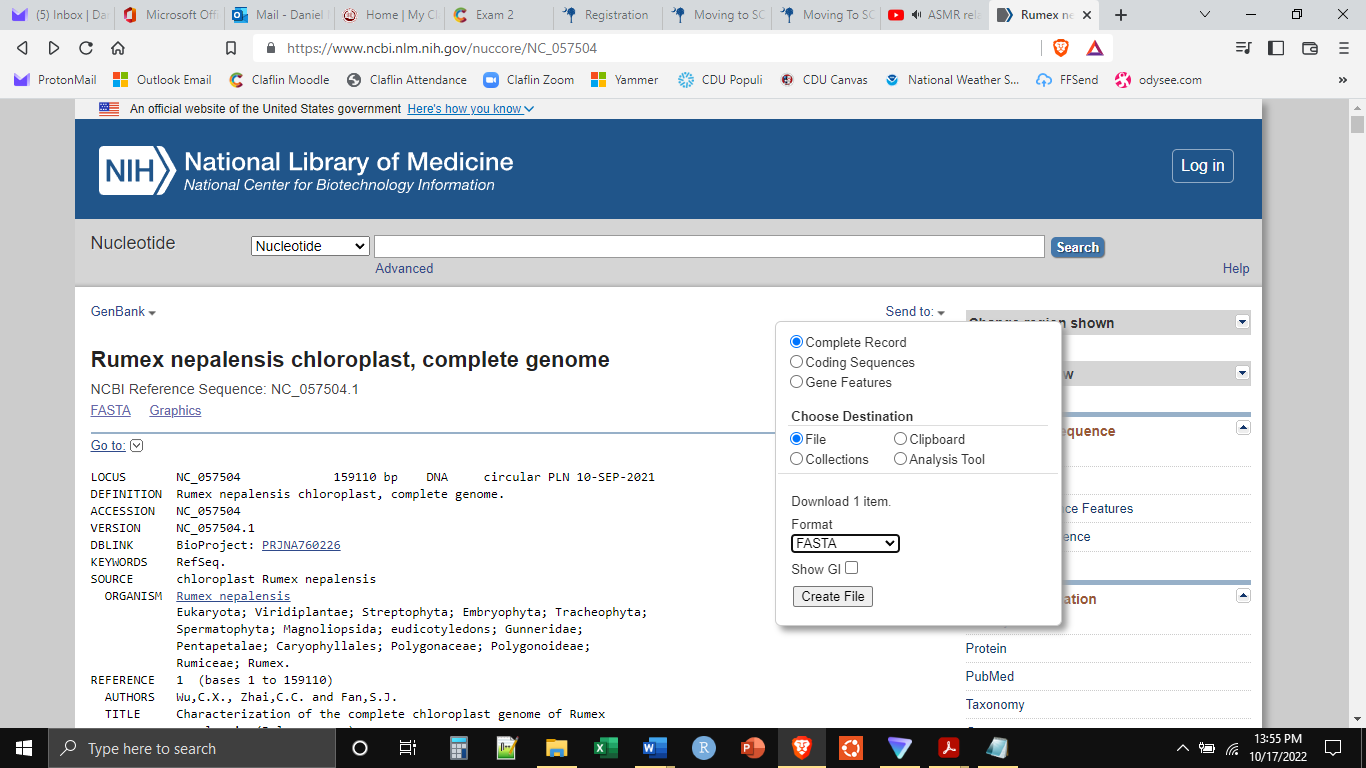
Taken together, the millions or hundreds of millions of reads that result from a sequencing analysis are often referred to as a “read archive”. It is from a read archive, or multiple read archives, that a chloroplast genome, mitochondrial genome, or nuclear genome can be assembled. There are two ways to assemble a genome 1) de novo, and 2) on a reference. In reference aided assembly a preexisting sequence from the same genome (nuclear, chloroplast, mitochondrial) of a closely related species is used as a template (the ‘reference”). The reads from the read archive are “mapped”, that is, aligned, to the reference genome. This method is useful as it is computationally less intensive and had make up for having poor quality reads. The disadvantages are that the new genome might be assembled erroneously, matching the reference genome in ways that it shouldn’t, but which the mapping algorithm forced it to. Reference-aided assemble also requires that you possess a reference genome from a closely related species, which may not be true.

When reference-aided assembly is not possible, de novo assembly is used. Here, each read is aligned to all other reads to form clusters of reads that overlap (“contigs”). The goal being that all the reads overlap creating a single long sequence. In practice, de novo assembly is rarely able to produce a single sequence and the result is a genome assembled into a number of pieces. This is especially true if the quality of the reads is not very good. But, de novo sequencing may be able to detect variations in the genome that reference-aided assembly could not.

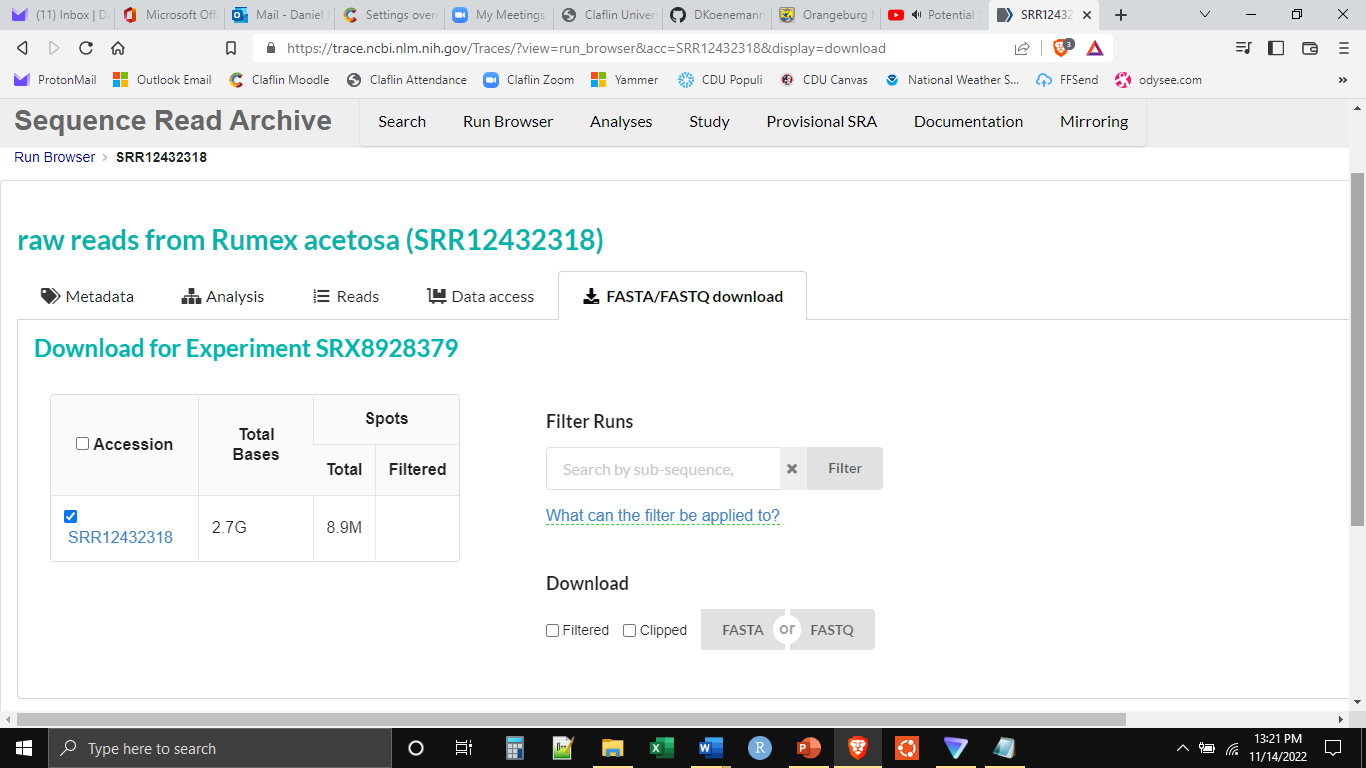
We will be doing a reference-aided assembly. This means that we need to start with two things, 1) a reference plastome (an already assembled chloroplast genome as closely related to the chloroplast genome we wish to assemble as possible), and 2) a read archive from the species whose chloroplast genome we would like to assemble. Ideally, the reference plastome should be annotated, but there are work-arounds if it is not.

We can obtain these two starting materials from public repositories. The NCBI GenBank contains genomes that have already been assembled, as well as Sanger Sequences, all of which have been published by researchers. These files will be in FASTA format. The other repository is the Sequence Read Archive (SRA), which includes raw read libraries from previously accomplished studies. Let’s begin by downloading the two files we will need. First, make a new folder on your desktop for all of the files for today.

The reference can be downloaded from Genbank (https://www.ncbi.nlm.nih.gov/genbank/). Then search for record “NC057504”. Select “Send to” and then specify “File” in FASTA format. Then click “Create File”. Open the file and rename the sequence (the part after the “>”) to be “RumexNepalensisReference”. Also, rename the file to “RumexNepalensisReference.fasta”.



Next, let’s download the read archive we will be using. Go to the Sequence Read Archive website (https://www.ncbi.nlm.nih.gov/sra), and search for record “SRX8928379”. Scroll to the bottom of the page and click on “Run SRR12432318”. This will take you to a new page. Here click on “Download FASTA/FASTQ”, and then click on the FASTQ button.



After downloading, please rename the file “RumexAcetosaArchive.fastq.gz”. This file is in the compressed version of FASTQ (.fastq.gz). In most cases, the two file types can be used interchangeably. The compressed version can be decompressed using the following command in the terminal.

First, make sure “gzip” is installed by executing:

sudo apt install gzip

Then execute the following:

gunzip RumexAcetosaArchive.fastq.gz

The expected output is a file named “RumexAcetosaArchive.fastq”. The compressed .fastq.gz files are not human readable. An uncompressed .fastq file is human readable but very, very large. The best way to examine them is through a terminal. Use the following commands to view the first or last twenty lines of the file.

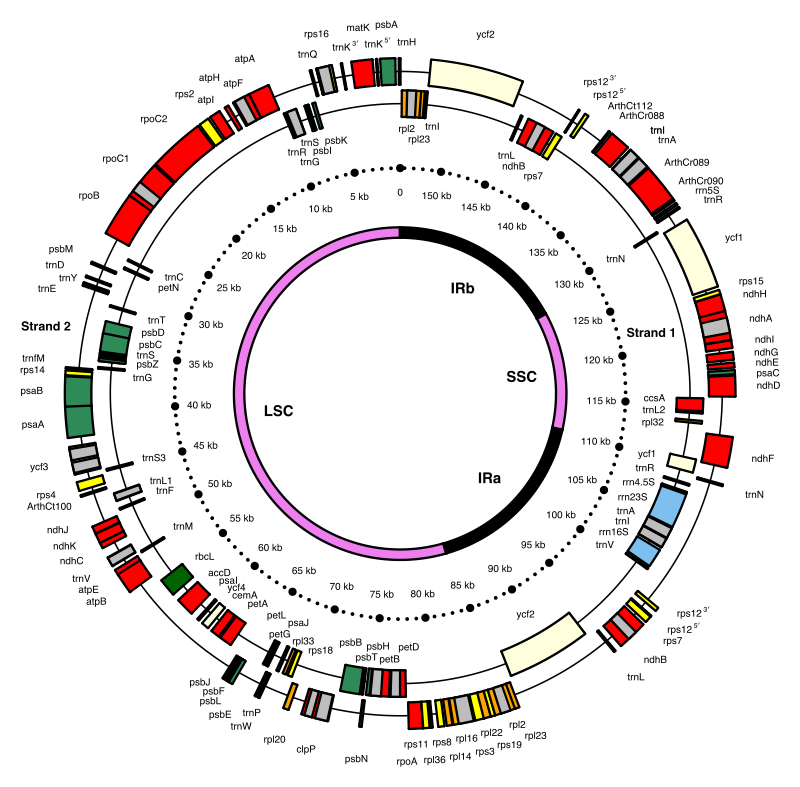
head -20 RumexAcetosaArchive.fastq

tail -20 RumexAcetosaArchive.fastq

less RumexAcetosaArchive.fastq

**Trim Reference**

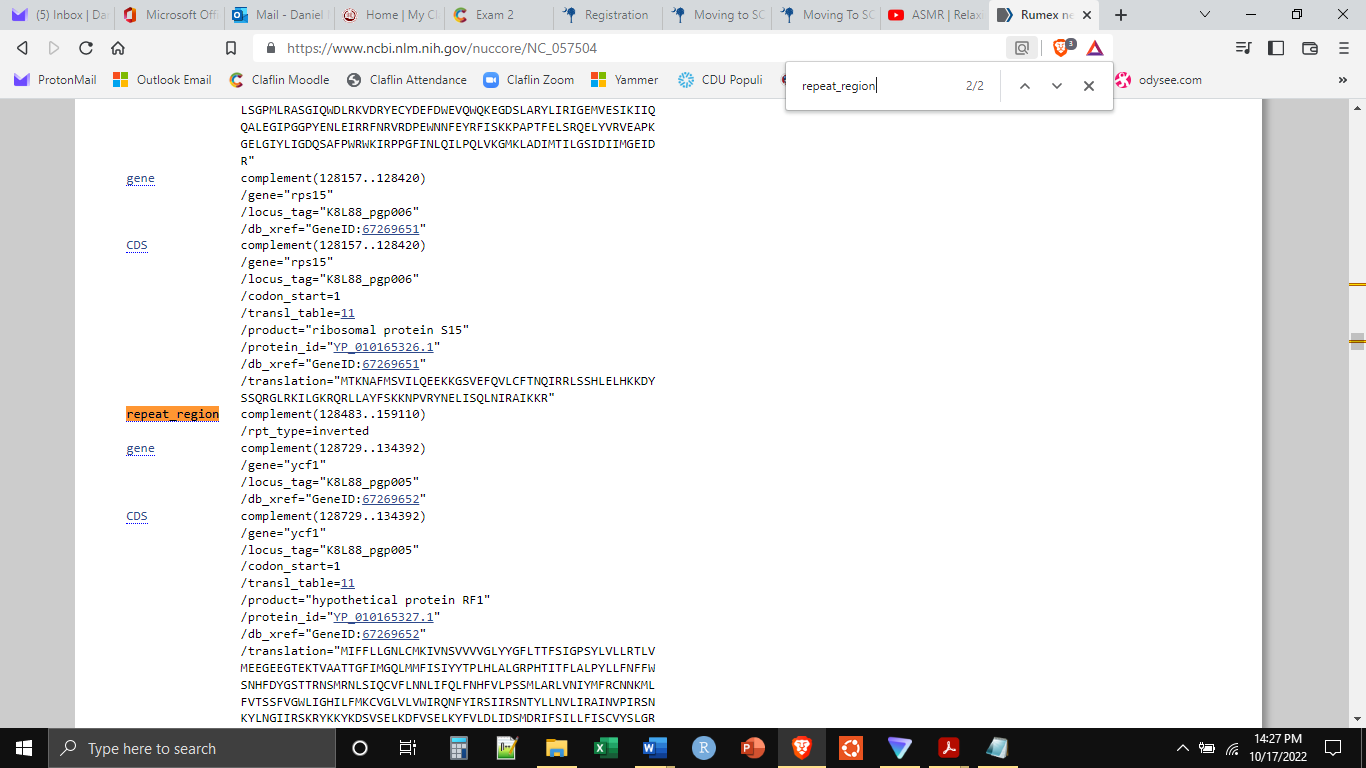
Recall that chloroplast genomes have four sections: the large single copy region, the small single copy region, and two inverted repeat sections. As the names suggest, the inverted repeat sections are identical, but inverted. These two sections code for the same genes in the same order, but reversed.



**Emmanuel Douzery. Wikimedia Commons.**

We are using a computer to map (align) each of the reads to the reference genome. Since any read from within the inverted repeat regions can align to two places, this will confuse the computer mapping algorithm and it will exclude these reads. This is undesirable. Thus, for plastome (chloroplast genome) assemblies, one of the inverted repeat sections of the reference plastome needs to be removed before we start. This is accomplished using bedtools (“bedtools” must be installed on the machine). bedtools is able to extract a portion of a FASTA sequence. The genetic coordinates defining the large single copy region, the small single copy region, and one of the repeats needed for the .bed file are obtained from the plastome annotation on GenBank.

Go back to GenBank and search for record “NC057504”. Use CTRL+F to search on the page and search for “repeat\_region”. There should be two hits in the annotation, indicating the boundaries of the two regions. In this case the second inverted repeat is the last part of the sequence. Thus we can take the whole sequence up to the start of this region.



Now that we know the region of the reference we want to keep, we need to make the BED file. BED files are simple, tab-delimited files that can be created, viewed, and edited with a simple text editor. Four our purposes, they are four columns. The first column indicates the name of the “chromosome” you are defining coordinates for. The second and third columns indicate the start and stop positions of the region you are extracting. In this case, the genetic region in the .bed file should encompass the part of the plastome we want to keep (LSC, SSC, IR1). The fourth column indicates the sequence name the new output FASTA should have (the part after the “>”). IMPORTANT, the sequence name given to the “chromosome” in the .bed file must match the sequence name in the .fasta reference plastome file exactly (the part after the “>” in the .fasta).

Open a simple text editor (Notepad, Notepad++, WordPad, Notes) and type the following into the first line of the document (each group of text is separated by a tab):

RumexNepalensisReference 1 128482 RumexNepalensisReference\_Trimmed

Save this file as “ReferenceBED.bed” in the same folder as the other files for today. Now we can write and execute the code necessary to use the BED file to cut out a section of the FASTA sequence. Open the terminal (if on a MAC) or Ubuntu (if on Windows or Chromebook). Navigate to the proper working directory (the folder on the desktop with all the files for today). Make sure bedtools is installed by executing:

sudo apt install bedtools

The execute the following (all on one line):

bedtools getfasta -fi RumexNepalensisReference.fasta -bed ReferenceBED.bed -fo RumexNepalensisReference\_Trimmed.fasta -name

The result should be a new file named “RumexNepalensisReference\_Trimmed.fasta”. If we open and inspect it, we will see that the sequence name has been changed to “RumexNepalensisReference\_Trimmed”, and the file should be substantially smaller.

**Quality Control**

The next step is to do quality control on the read libraries. This is focused on doing two things. First, any reads that consist mostly of low-quality base scores, should simply be removed from the dataset. There are so many reads in a read archive, that even removing many low-quality reads will not reduce the archive size much. Second, the sequencing process requires the addition of primers and other fragments of DNA to the genomic sequences. Since these added DNA pieces were not part of the genetic material of the living organism, we want to make sure they are all cut off.

Quality control has at least two steps. In the first step, we generate a report to assess the quality of the reads in the archive. This is accomplished through the program fastqc. We will use the default conditions for the analysis.

Make sure that fastqc is installed, by executing the following:

sudo apt install fastqc

Then execute the following code to run fastqc on our read archive:

fastqc RumexAcetosaArchive.fastq

The expected output will be two files “RumexAcetosaArchive\_fastqc.html” and “RumexAcetosaArchive\_fastqc.zip”. The two files contain the same information. Opening the .html file in an internet browser will open a summary report of the quality control results. Examples of what good and bad Illumina data look like can be found on Moodle (“BadIlluminaData\_Example.html” and “GoodIlluminaData\_Example.html”). In most cases, problems uncovered by fastqc can be solved by trimgalore.

Trimgalore will clean the read libraries, specifically by removing any adaptors that remain on the reads, as well as remove any reads that have low quality base scores.

Make sure that trimgalore is installed, by executing the following:

Sudo apt install trim\_galore

Then execute the following code to clean the read libraries. This code will use the default conditions for the analysis.

trim\_galore RumexAcetosaArchive.fastq

Depending on how deep the sequencing and thus how big the read archive, trimgalore may take a long time to run. The expected output will be two files: “RumexAcetosaArchive.fastq\_trimming\_report.txt” – a report detailing the trimming that was done, and “RumexAcetosaArchive\_trimmed.fq” – a new read archive including only the high-quality reads with the adaptors removed. It is this \_trimmed.fq.gz file that you will carry forward. If you wanted to re-run RumexAcetosaArchive\_trimmed.fq.gz through fastqc again to see if it is improved, you could.

This has prepared us for next week, where we will be able to actually assemble the new chloroplast genome on the reference. Keep all of the files we generated this week as we will need them for next week.