Module 2

Sequence Data: Quality assessment and trimming

Goals: Learn how to assess sequence quality and trim/filter sequence reads to generate improved datasets for downstream analyses

2.1 Background

Before we start, we need to be sure we have a good grasp of the concepts of sequence reads, paired-ends, their relationships to the DNA fragments being sequenced, and to the original DNA samples. Figure 1 serves as a reminder of how the various entities are related to one another.

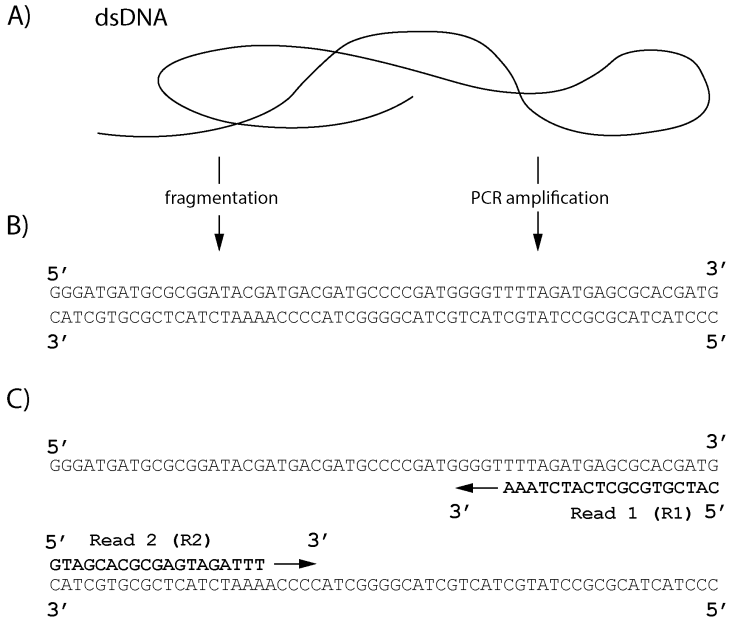

Figure 1. Summary of the essential steps involved in sequencing.

A) shows a double-stranded (ds) DNA molecule extracted from an organism, environmental sample, or a fossil.

B) The DNA is fragmented into amplifiable pieces (~300 bp–1.6 kb), or a specific target fragment is amplified, and adaptors (not shown) are added.

C) The two DNA strands are separated, and sequence is read from each strand in the 5' to 3' direction, producing two reads, R1 and R2. These are called "paired-end" reads because they are derived from opposite ends of a single dsDNA fragment.

2.2 The FASTQ sequence format

Most of us should be familiar with the FASTA sequence format. FASTQ sequences contain all the information normally found in FASTA sequence entries but also contain information about the quality of each base that was called. The first two lines of each FASTQ sequence entry contain the same information found in a FASTA sequence (header then sequence), except that FASTQ headers start with "@" instead of ">." Each FASTQ entry contains two additional lines: the first acts as a separator (usually a "+" symbol) and the second contains the quality values (represented by ASCI characters) for each nucleotide in line 2. Take a look at one of the Br80 *.fastq* files in your *sequences* directory to familiarize yourself with the format. Note how the number of quality values always matches the number of nucleotides.

In summary: a FASTA sequence entry has two lines that look like this:

>Header
Sequence….

And a FASTQ sequence entry has four line and looks like this:

@Header
Sequence…
Separator
Quality values…

2.3 Assess overall sequence quality with FastQC

- ☐ Make sure you are logged into your VM.
- ☐ **Change (cd) into the *sequences* directory**
- ☐ Use **FastQC** to perform quality analysis on the Br80 fastq datasets

```
• fastqc Br80_S1_L001_R1_001.fastq Br80_S1_L001_R2_001.fastq
```

- ☐ List the directory to see the outputs produced by **FastQC**

```
• ls -lrt
```

Note that here we are using two new options for the **ls** command (-r and -t). These tell ls to print the listing in order of file creation time and in reverse order (i.e. most recent files at the bottom). You will see that FastQC produced two html files and two zipped archives.

- ☐ Use **scp** to transfer the resulting *.html* files to your local machine. **Note that <u>you will need to run the command from your local machine</u>, so you must either open another terminal window or logout from your VM first.**

```
• scp myName@ip-xxx-xx-xx-xxx:sequences/*html .
```

- ☐ Navigate to the downloaded *.html* files on your local machine and open them by double-clicking on them. If you are unsure of where to find them, you can type **pwd** in the terminal window from which you issued the **scp** command and it will report the path to the files
- ☐ Look at **Basic Statistics**: This provides file name, type, encoding, number of sequences, sequence length range, sequences to be filtered, and overall %GC content.
- ☐ How many sequence reads were in for the R1 and R2 datasets? R1: _____,R2: _____
- ☐ What are the ranges in sequence lengths? low: _____,high: _____
- ☐ Click on **Per base sequence quality**: For each data point, the central red line is the median value; the yellow box represents the inter-quartile range (25-75%). The upper and lower whiskers represent the 10% and 90% points, and the blue connecting line represents the mean quality. A warning (yellow "!") is issued if the lower quartile for any base is less than 10, or if the median for any base is less than 25. This tab will signal failure (red "X") if the lower quartile for any base is less than 5 or if the median for any base is less than 20.
- ☐ Click on **Per tile sequence quality**: Illumina sequencing flow cells consist of "tiles" which are discrete areas that are imaged at each round of sequencing. This metric uses the flow cell coordinates in each read identifier to assess whether there might have been any systemic sequencing issues. Quality is represented using a color scale, with cold colors indicating quality higher than average for the given base and warm colors indicating quality lower than average. Base position is plotted on the x-axis and tile number is on the y-axis. (Note: 9 tiles were imaged with the MiSeq V2 chemistry used for the present run, and they are represented by indexes 1101 through 1109; 19 tiles are imaged with V3 chemistry.) A warning is issued if, at any base, a tile has a mean Phred score below 2 less than the mean quality across all tiles at that base. A failure is raised if, at any base, a tile has a mean Phred score below 5 less than the mean quality across all tiles at that base.
- ☐ Click on **Per sequence quality scores**: This plots a frequency distribution for average quality across the entire sequence. A warning is raised if the most frequently observed mean quality is below 27 (this equates to a 0.2% error rate). An error is raised if the most frequently observed mean quality is below 20 (1% error rate).
- ☐ Click on **Per base sequence content**: the relative frequency of each base should reflect the overall frequency of these bases in the acquired sequence data. Strong position-specific biases usually indicate an overrepresented, possibly contaminating sequence. Consistent bias across all positions either indicates true genome bias (i.e., AT-/GC-richness) or signals a systematic sequencing problem. This module issues a warning if the difference between A and T, or G and C is greater than 10% in any position. This module will indicate an error if the difference between A and T, or G and C is greater than 20% in any position.
- ☐ Click on **Per sequence GC content**: the overall GC content should reflect the GC content of the underlying genome. This module issues a warning if more than 15% of the reads deviate from the normal distribution; and a "failure" if the deviation involves more than 30% of the reads. An unusually shaped distribution could indicate the presence of AT-/GC-rich genomic "islands" or a contaminated library (e.g. fungal DNA contaminated with bacteria).
- ☐ Click on **Per base N content**: the % of reads with Ns at each base position. This module raises a warning if any position shows an N content of >5%. This module will raise an error if any position shows an N content of >20%.

- ☐ Click on **Sequence length distribution**: generates a graph showing the distribution of fragment sizes in the file under study. This module will raise a warning if all sequences are not the same length or an error if any of the sequences have zero length.
- ☐ Click on **Sequence duplication levels**: Plots the % of reads that occur as exact duplicates. Due to the supposedly random nature of shearing/tagmentation, most reads should have slightly different sequence start positions. Large numbers of exact sequence duplicates implicate PCR amplification as a culprit. A warning is given at >20% duplication, and an error is thrown at >50%.
- ☐ Click on **Overrepresented sequences**: Lists specific sequence(s) that are highly overrepresented in the dataset. All of the sequences which make up more than 0.1% of the total are reported. This module will issue a warning if any sequence is found to represent more than 0.1% of the total. This module will issue an error if any sequence is found to represent more than 1% of the total.
- ☐ Click on **Adapter content:** Occasionally, adapters may be missed by the de-multiplexing program, sometimes due to sequencing errors and sometimes because the fragment being sequenced is much shorter than the read length (typically 250 bp or 300 bp for MiSeq) which leads to the 3' adapter being sequenced. **FastQC** looks for matches to four common adapters within reads. The module will issue a warning when a sequence is found in more than 5% of reads and will report failure when a sequence is found in more than 10% of all reads.

Note that the quality of the sequences drop off dramatically as the sequence reads extend past 280 bp. This is not surprising because these data were generated using the 600 cycle MiSeq flowcell, which was fairly new technology at the time these data were generated.

Possible problems with biased sequence composition were detected at the starts and ends of the reads. Front-end bias is probably due to the tagmentation process. (In practice, the transposon that cuts the DNA and adds adapter sequences operates preferentially in certain genomic contexts.) This is unlikely to cause problems during assembly (after all, this represents "true" genomic sequence) and can therefore be ignored. On the other hand, the sequence bias at the ends of the R2 reads point to frequent occurrence of homopolymer tracts due to poor sequence resolution. Therefore, the R2 reads should be quality-trimmed prior to downstream analyses.

- ☐ Now run **FastQC** on the "Lh88405" sequences in the same directory.

Here, the Lh88405 dataset has exceptional sequence quality throughout the sequence reads, which is not surprising given that only 150 bp of data were acquired from each DNA strand. However, a significant proportion of reads have adapter contamination at their 3' ends, and this contamination must be trimmed before sequence assembly.

Question: Thinking about the various processes involved in NGS library construction, what would be the primary reason that so many reads have adaptor sequences at their ends? _____

_____

2.4 Trimming/filtering poor quality sequence data with Trimmomatic

High quality sequence data is important for all types of downstream analyses. **FastQC** provides a good overview of sequence quality and alerts us to potential problems with our data, such as short sequence lengths, unusual base composition indicative of sequence adaptor contamination and quality decline near the ends of the sequences. Now we will use the free utility, **Trimmomatic,** to filter and trim reads to address these quality issues.

Bolger et al. (2014) Trimmomatic: A flexible trimmer for Illumina sequence data. Bioinformatics doi: 10.1093.

http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/TrimmomaticManual_V0.32.pdf

Usage:

```
java -jar <path to trimmomatic.jar> PE [-threads <threads>] [-phred33 |
-phred64] [-trimlog <logFile>] <input 1> <input 2> <paired_output_1>
<unpaired_output_1> <paired_output_2> <unpaired_output 2> <trim_options>
```

In the above command, we introduce square and angle brackets for the first time. In this example, the text enclosed in square brackets ("[" and "]") are "options" that specify various user-specified parameters that affect program behavior. The brackets are omitted from the actual command. As implied by their name, these parameters are "optional" and if they are not specified, the program, assumes specific default settings. Parameters in angle brackets are required, and in this example show where we must insert specific inputs to **Trimmomatic**. In the actual command, you will omit the brackets and substitute the terms inside them with your specific filenames. This should become clear when you compare the provided usages with the actual commands we will be using.

We will use **Trimmomatic**'s nifty paired-end feature (PE mode) that separates reads based on whether corresponding forward and reverse sequences are available. This is useful because some downstream analysis programs (e.g., the **MaSuRCA** assembler) will only allow paired-end reads as input, or they process paired-ends and unpaired reads separately.

Because we are running Trimmomatic in paired-end mode, we will need to provide both forward (R1) and reverse (R2) read filenames as arguments to the program

☐ Use **Trimmomatic** to trim poor quality sequences from the Br80 dataset. **Remember, enter the entire command on a single line. Type very carefully and use tab completion where possible.**

```
• java –jar trimmomatic–0.38.jar PE –threads 2
  –phred33 –trimlog Br80_errorlog.txt
  Br80_S1_L001_R1_001.fastq
  Br80_S1_L001_R2_001.fastq
  Br80_S1_L001_R1_paired.fastq
  Br80_S1_L001_R1_unpaired.fastq
  Br80_S1_L001_R2_paired.fastq
  Br80_S1_L001_R2_unpaired.fastq
  CROP:280 SLIDINGWINDOW:20:20 MINLEN:150
```

`PE`	run in paired-end mode (create paired-read and unpaired-read output files)
`–trimlog`	creates a log of all read trimmings
`–phred33`	use the phred33 quality scale
`CROP`	clip read to specified length by removing reads from 3' end
`SLIDINGWINDOW`	clip read once mean quality in window of specified size falls below target value (average Phred score ≤ 20 in 20 nucleotide window)
`MINLEN`	discard reads that are shorter than the specified length

You can tell if you issue the correct commands if the program reports that TrimmomaticPE was started with a set of arguments, no errors (exceptions) are reported, and the command line prompt does not immediately reappear (indicating that the program is processing data). Once the trimming is complete, the program will report some summary statistics indicating how many reads were trimmed/retained after the trimming/filtering steps.

☐ Now run **FastQC** on the four trimmed output files (the ones with *paired* and *unpaired* in their names) to assess how well **trimmomatic** performed in removing poor quality sequence.

2.5 Trimming adaptor sequences

Next, we will use **Trimmomatic** to clip off sequences that correspond to adaptors that were added onto our DNA fragments during library construction. To do this we provide **Trimmomatic** with a fasta file that contains the sequences of possible adaptors.

☐ Use the command line to take a quick look at the contents of the *adaptors.fa* file (it should be in your current working directory).

Note that this file does not contain all possible adaptor sequences and additional ones may need to be added to the file depending on the specific type/manufacturer of the kit that was used for NGS library production.

☐ Use **Trimmomatic** to remove adaptor contamination (and poor quality regions) from the Lh88405 dataset. **Remember, enter the entire command on a single line. Type very carefully and use tab completion when typing names of input files.**

```
• java –jar trimmomatic–0.38.jar PE –threads 2
  –phred33 –trimlog Lh88405_errorlog.txt
  Lh88405_1.fq.gz
  Lh88405_2.fq.gz
  Lh88405_1_paired.fq
  Lh88405_1_unpaired.fq
  Lh88405_2_paired.fq
  Lh88405_2_unpaired.fq
  ILLUMINACLIP:adaptors.fa:2:30:10 SLIDINGWINDOW:20:20 MINLEN:100
```

`ILLUMINACLIP` clip off sequences that match adaptors in the provided reference file

☐ Run **FastQC** on the four trimmed output files to assess how well **trimmomatic** performed in removing the adaptor contamination.

2.6 Optional exercises

Using the command line to convert between sequence formats:

☐ Generate a *.fasta* file from a *.fastq* file using only Linux commands. (Don't just find a program that already does this conversion for you.) See if you can accomplish this in one line, using piped commands.

Hint: You already know commands (**grep, awk, sed**) that will allow you to do accomplish this task. Use the **man** pages (or Google/ChatGPT) to find out how to extend the functions of these programs to generate the necessary output.

Warning: You must understand the possible content of all lines in the *.fastq* file to be sure that your script functions correctly.