RNA-seq Quality Assessment Assignment

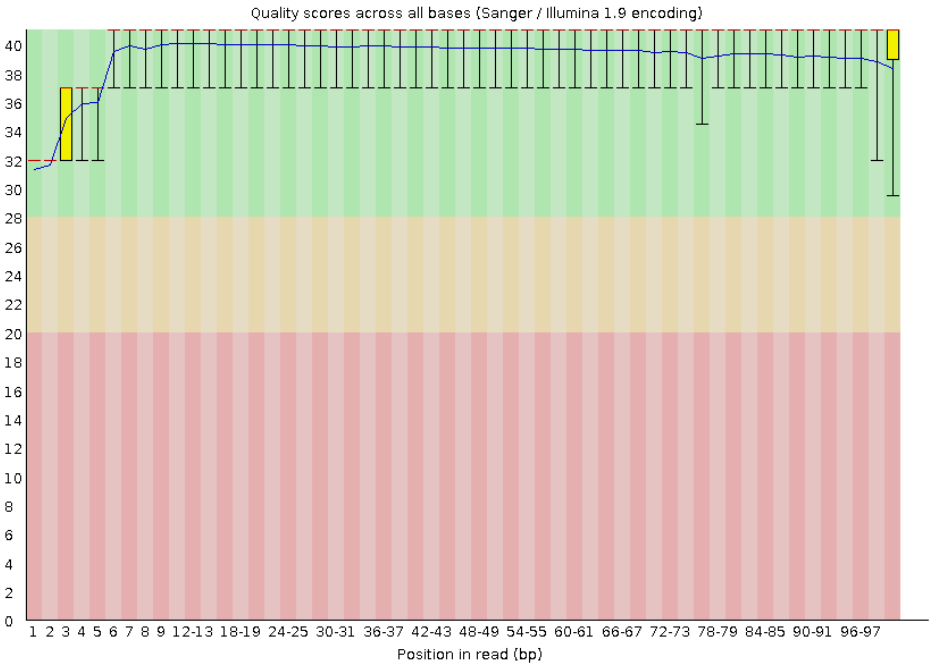
Justine Macalindong

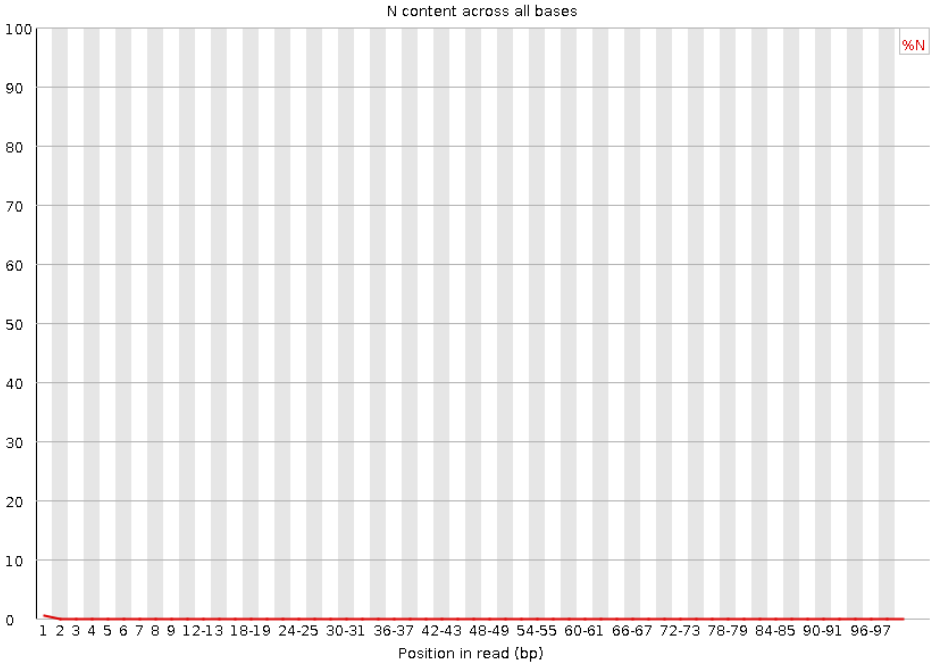
2022-09-07

Part 1 – Read quality score distributions

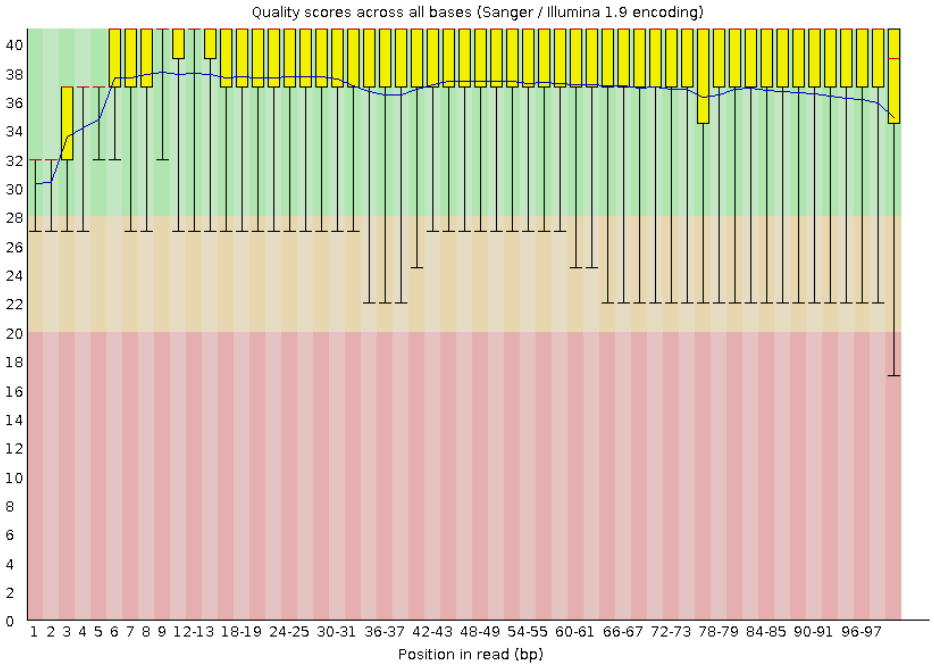
1. Using FastQC via the command line on Talapas, produce plots of quality score distributions for R1 and R2 reads. Also, produce plots of the per-base N content, and comment on whether or not they are consistent with the quality score plots.

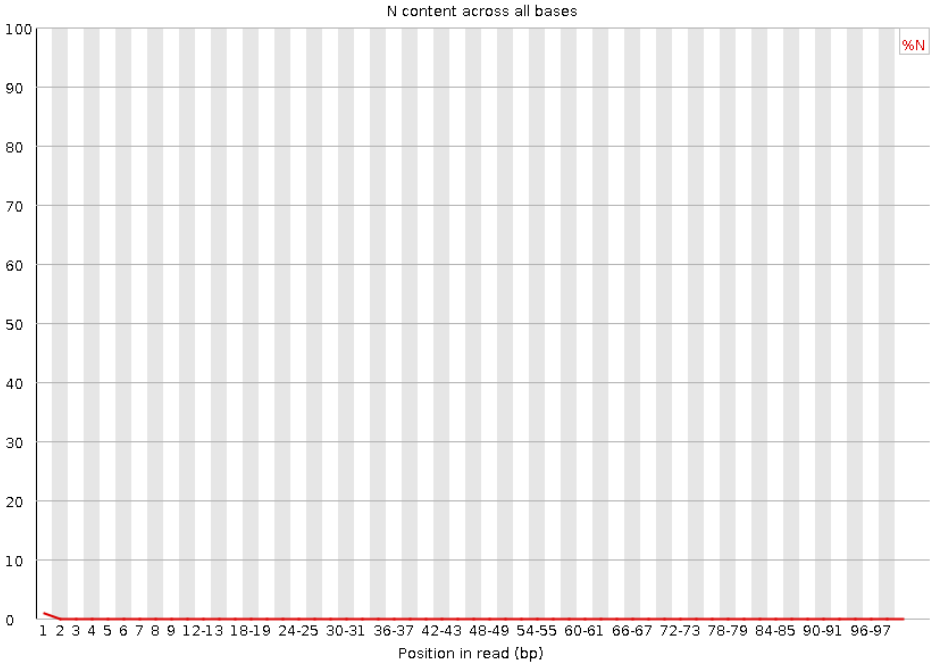
1_2A_control_S1_L008_R1_001.fastq.gz



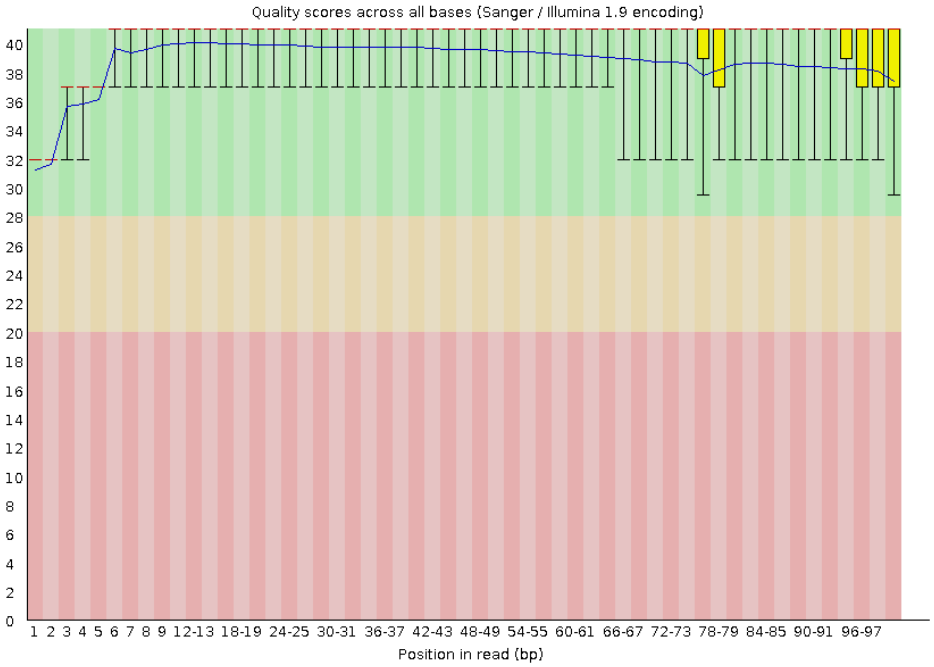


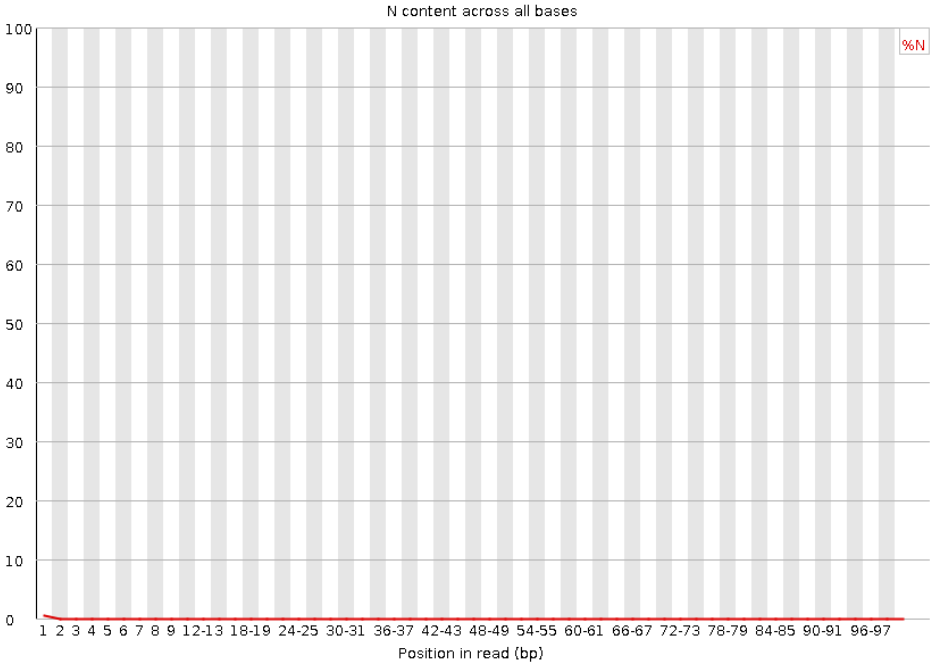
1_2A_control_S1_L008_R2_001.fastq.gz



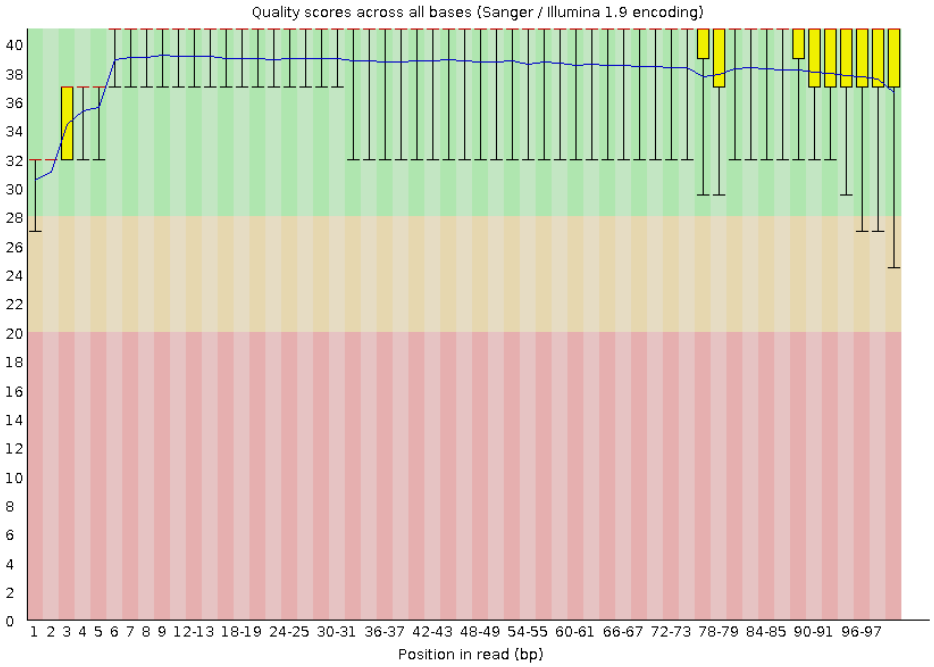


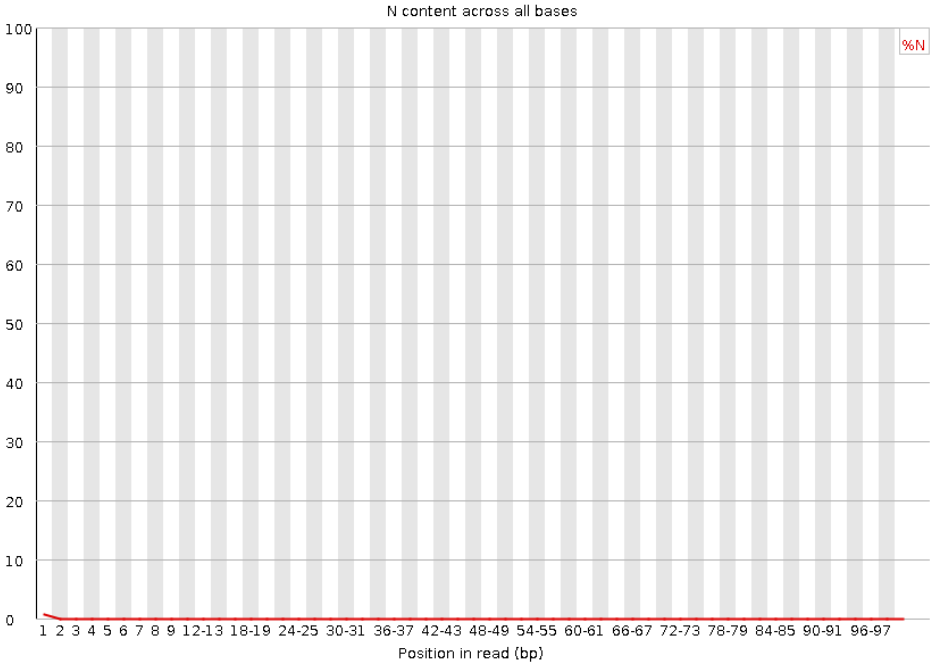
24_4A_control_S18_L008_R1_001.fastq.gz





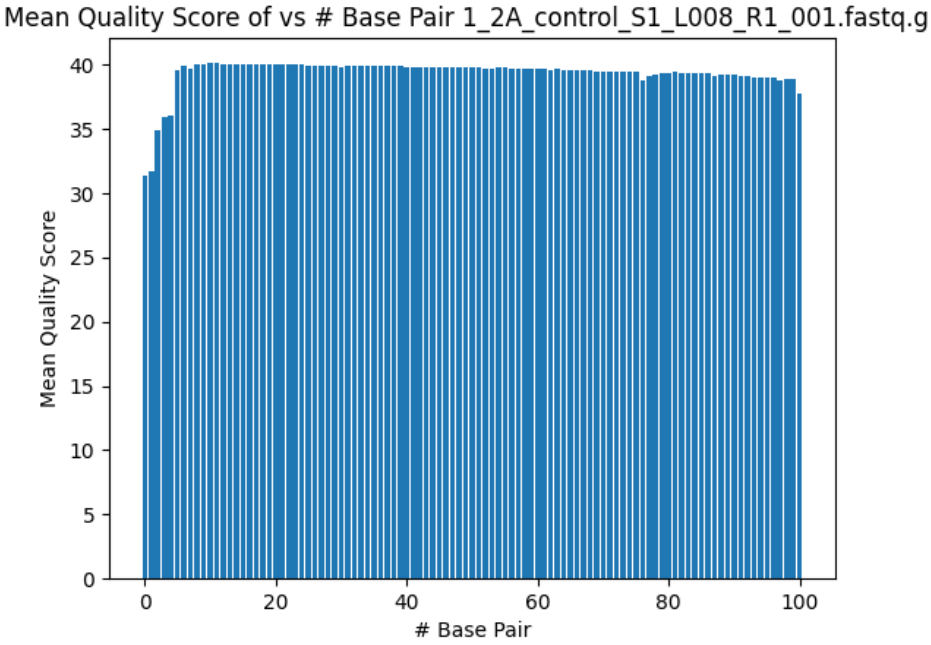
24_4A_control_S18_L008_R2_001.fastq.gz

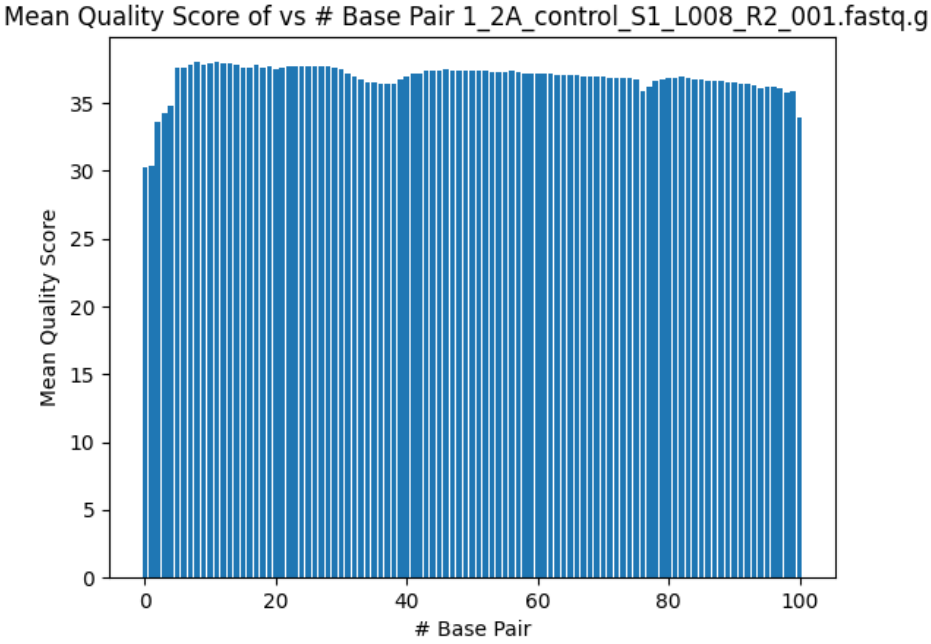


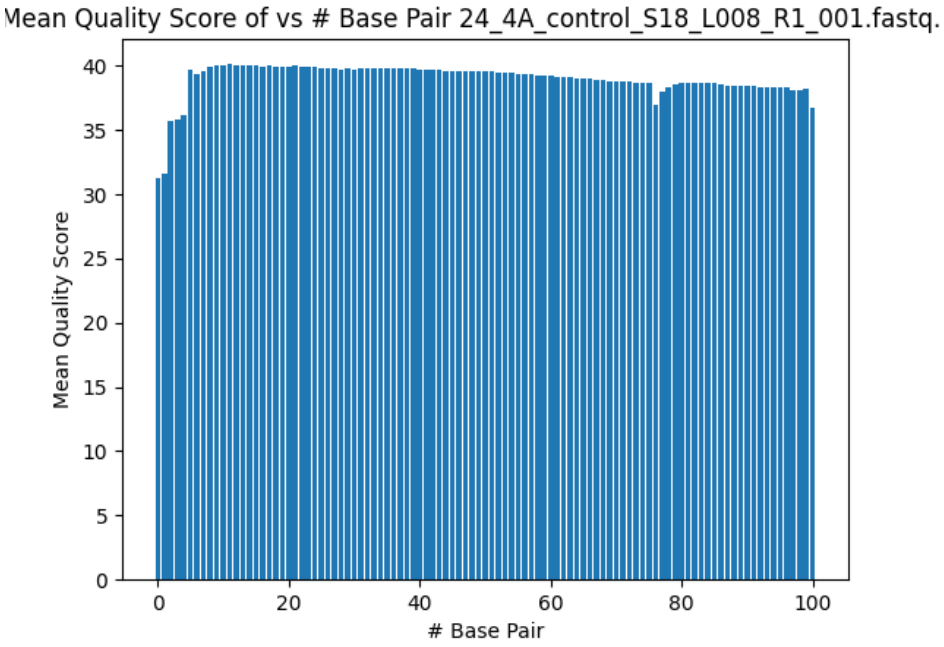


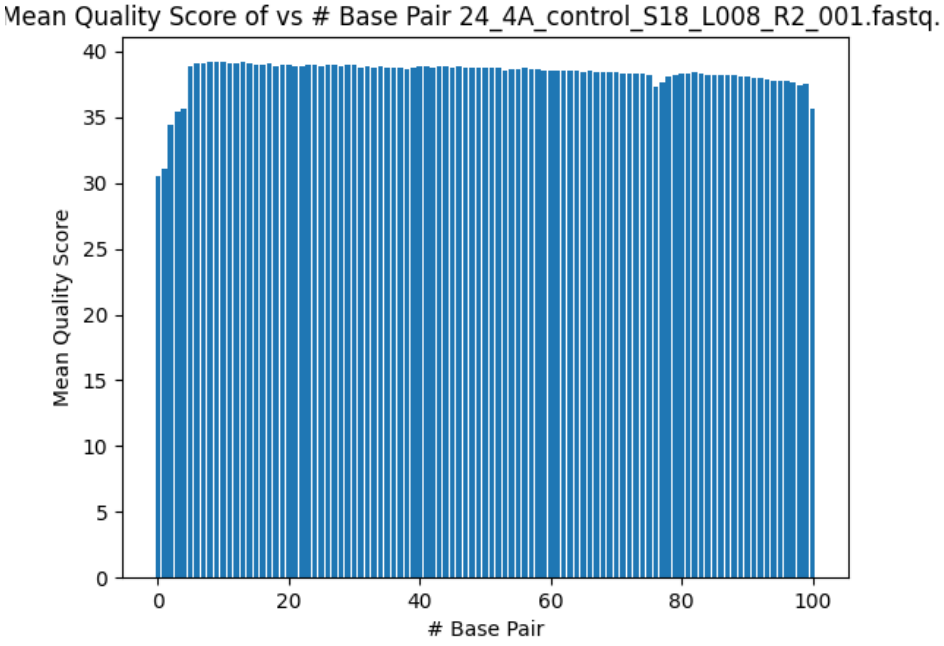
The quality score distribution and per-base N content graphs of all the files are consistent with one another. Because the reads are of high quality (>30), the per-base N content of each file is roughly zero throughout each graph. The highest point of each per-base N content graph correlates to the lowest quality score of their corresponding quality score distribution graph (the first base pair).

2. Run your quality score plotting script from your Demultiplexing assignment. (Make sure you're using the "running sum" strategy!!) Describe how the FastQC quality score distribution plots compare to your own. If different, propose an explanation. Also, does the runtime differ? If so, why?









The FastQC quality score distribution plots are very similar, if not exactly the same, as my quality score plotting script from my Demultiplexing assignment. When looking at the graphs side by side, you can observe the same dips in average quality at the same read position (bp).

The runtimes between read1 and read2 of each file were very similar and only differed by a few seconds. However, the 24_4A_control_S18_L008 files ran for ~30 minutes longer than the 1_2A_control_S1_L008 files.

```
1_2A_control_S1_L008_R1_001.fastq.gz: 2:38.28
```

```
1_2A_control_S1_L008_R2_001.fastq.gz: 2:40.78

24_4A_control_S18_L008_R1_001.fastq.gz: 3:17.85

24_4A_control_S18_L008_R2_001.fastq.gz: 3:15.78
```

After using wc -l on all of the files, I believe the 24_4A_control_S18_L008 files took longer to run because they are larger files and have 8,152,060 more lines than the 1_2A_control_S1_L008 files. Therefore, the 24_4A_control_S18_L008 files have 2,038,015 more records than the 1_2A_control_S1_L008 files.

```
1_2A_control_S1_L008: 33,911,436

24_4A_control_S18_L008: 42,063,496
```

3. Comment on the overall data quality of your two libraries.

The quality of my two libraries are very high as the average quality scores at each base all fall above 30, which is the cutoff I used to determine a high quality read in my demultiplexing assignment.

Part 2 – Adaptor trimming comparison

5. Using cutadapt, properly trim adapter sequences from your assigned files. Be sure to read how to use cutadapt. Use default settings. What proportion of reads (both R1 and R2) were trimmed?

I checked the FastQC data files for overrepresented sequences to find the adapters on my own and there were not any detected. I still ran cutadapt using the adapters provided (R1: AGATCGGAAGAGCACACGTCT-GAACTCCAGTCA and R2: AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT) and searched for them using grep.

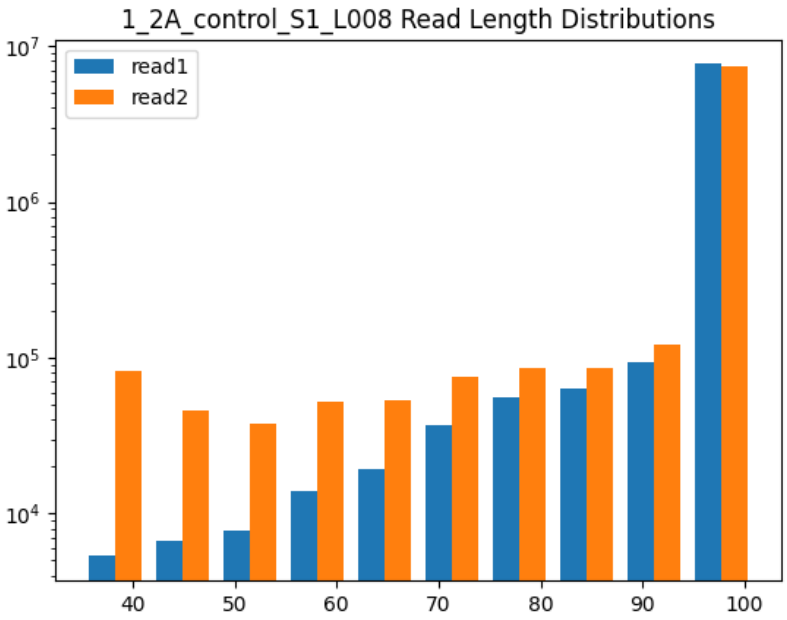
```
zcat 1_2A_control_S1_L008_R1_001.fastq.gz | grep "^AGATCGGAAGAGCACACGTCTGAACTCCAGTCA"
zcat 1_2A_control_S1_L008_R2_001.fastq.gz | grep "^AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT"
zcat 24_4A_control_S18_L008_R1_001.fastq.gz | grep "^AGATCGGAAGAGCACACGTCTGAACTCCAGTCA"
zcat 24_4A_control_S18_L008_R2_001.fastq.gz | grep "^AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT"
```

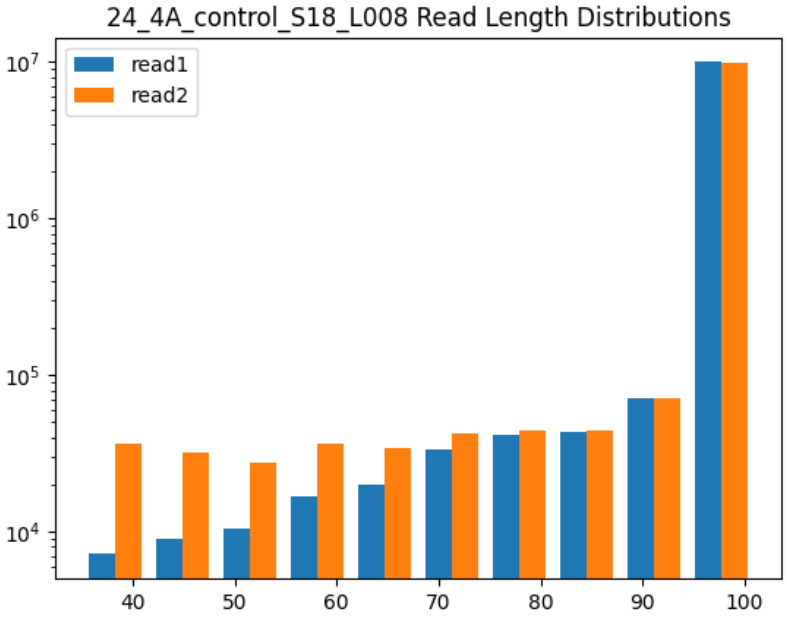
After looking at the output files from cutadapt, I found the number of times reads were trimmed.

```
Proportion of 1_2A_control_S1_L008 reads trimmed:
Read1: 468835 times / 8477859 * 100 = 5.53% trimmed
Read2: 537308 times / 8477859 * 100 = 6.34% trimmed

Proportion of 24_4A_control_S18_L008 reads trimmed:
Read1: 335742 times / 10515874 * 100 = 3.19% trimmed
Read2: 417709 times / 10515874 * 100 = 3.97% trimmed
```

7. Plot the trimmed read length distributions for both R1 and R2 reads (on the same plot). You can produce 2 different plots for your 2 different RNA-seq samples. There are a number of ways you could possibly do this. One useful thing your plot should show, for example, is whether R1s are trimmed more extensively than R2s, or vice versa. Comment on whether you expect R1s and R2s to be adapter-trimmed at different rates.





For both files, the R2s were trimmed more extensively than the R1s. I would expect the R1s and R2s to be adapter-trimmed at different rates, with R2 being trimmed more than R1 because R2 is usually lower quality due to degradation of sample and because it's run on the sequencer second.

Part 3 – Alignment and strand-specificity

10. Using your script from PS8 in Bi621, report the number of mapped and unmapped reads from each of your 2 sam files. Make sure that your script is looking at the bitwise flag to determine if reads are primary or secondary mapping (update/fix your script if necessary).

File	Number of Mapped Reads	Number of Unmapped Reads
1_2A_control_S1_L008	15627441	305245
24_4A_control_S18_L008	19780644	710220

12. Demonstrate convincingly whether or not the data are from "strand-specific" RNA-Seq libraries. Include any comands/scripts used. Briefly describe your evidence, using quantitative statements (e.g. "I propose that these data are/are not strand-specific, because X% of the reads are y, as opposed to z.").

1_2A_control_S1_L008

```
Total number of reads:
cat 1_2A_stranded_htseq | awk '{sum += $2} END {print sum}'
7966343

Proportion of mapped reads for "yes"
cat 1_2A_stranded_htseq | awk '$1 ~ "ENSMUSG" {sum += $2} END {print sum}'
312665/7966343 * 100 = 3.92%

Proportion of mapped reads for "reverse":
cat 1_2A_reverse_htseq | awk '$1 ~ "ENSMUSG" {sum += $2} END {print sum}'
6877200/7966343 * 100 = 86.33%
```

I propose that the 1_2A_control_S1_L008 data is strand-specific because 86.33% of the reads are mapped when stranded=reverse as opposed to 3.92% when stranded=yes. If the data was not strand-specific, you would expect to see the same percentages of mapped reads between stranded=reverse and stranded=yes.

24_4A_control_S18_L008

```
Total number of reads:
cat 24_4A_stranded_htseq | awk '{sum += $2} END {print sum}'
10245432

Proportion of mapped reads for "yes":
cat 24_4A_stranded_htseq | awk '$1 ~ "ENSMUSG" {sum += $2} END {print sum}'
356658/10245432 * 100 = 3.48%

Proportion of mapped reads for "reverse":
cat 24_4A_reverse_htseq | awk '$1 ~ "ENSMUSG" {sum += $2} END {print sum}'
8377321/10245432 * 100 = 81.77%
```

I propose that the 24_4A_control_S18_L008 data is strand-specific because 81.77% of the reads are mapped when stranded=reverse as opposed to 3.48% when stranded=yes. If the data was not strand-specific, you would expect to see the same percentages of mapped reads between stranded=reverse and stranded=yes.