read assignments: Thomas 28_4D_mbnl_S20_L008 3_2B_control_S3_L008

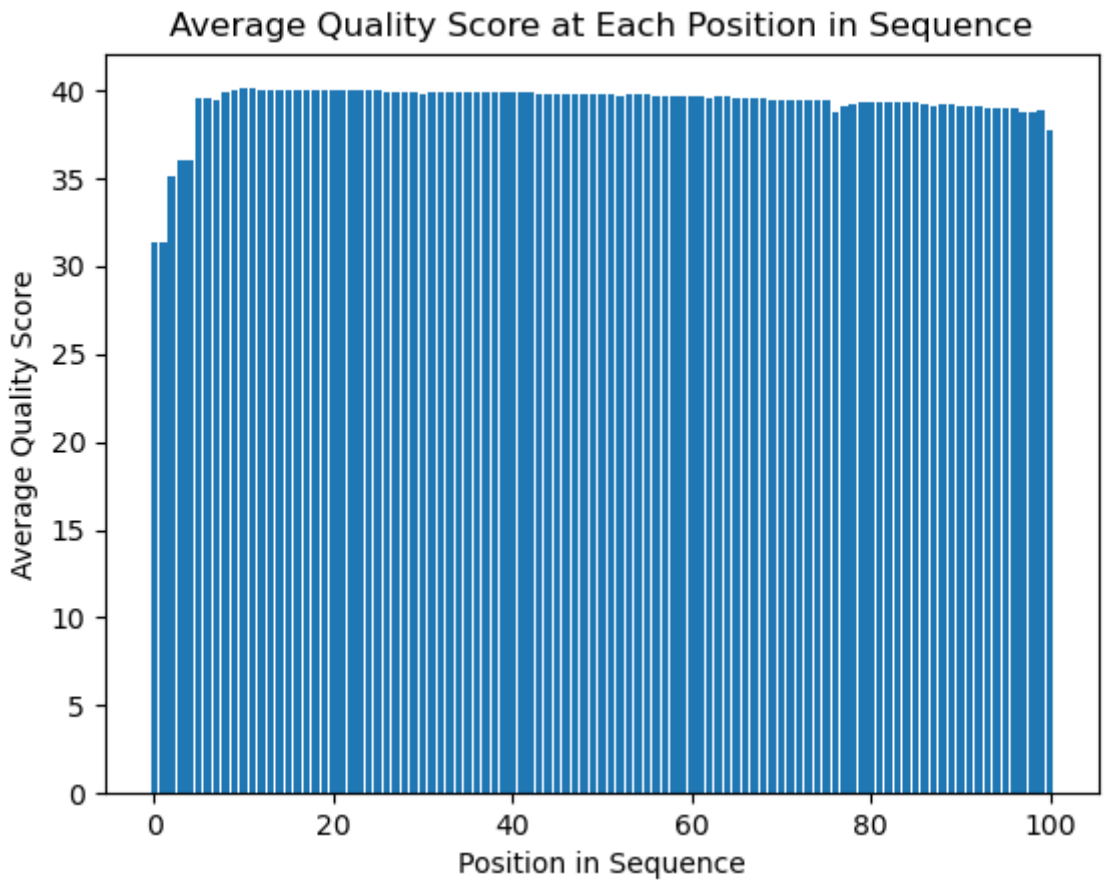
Part 1

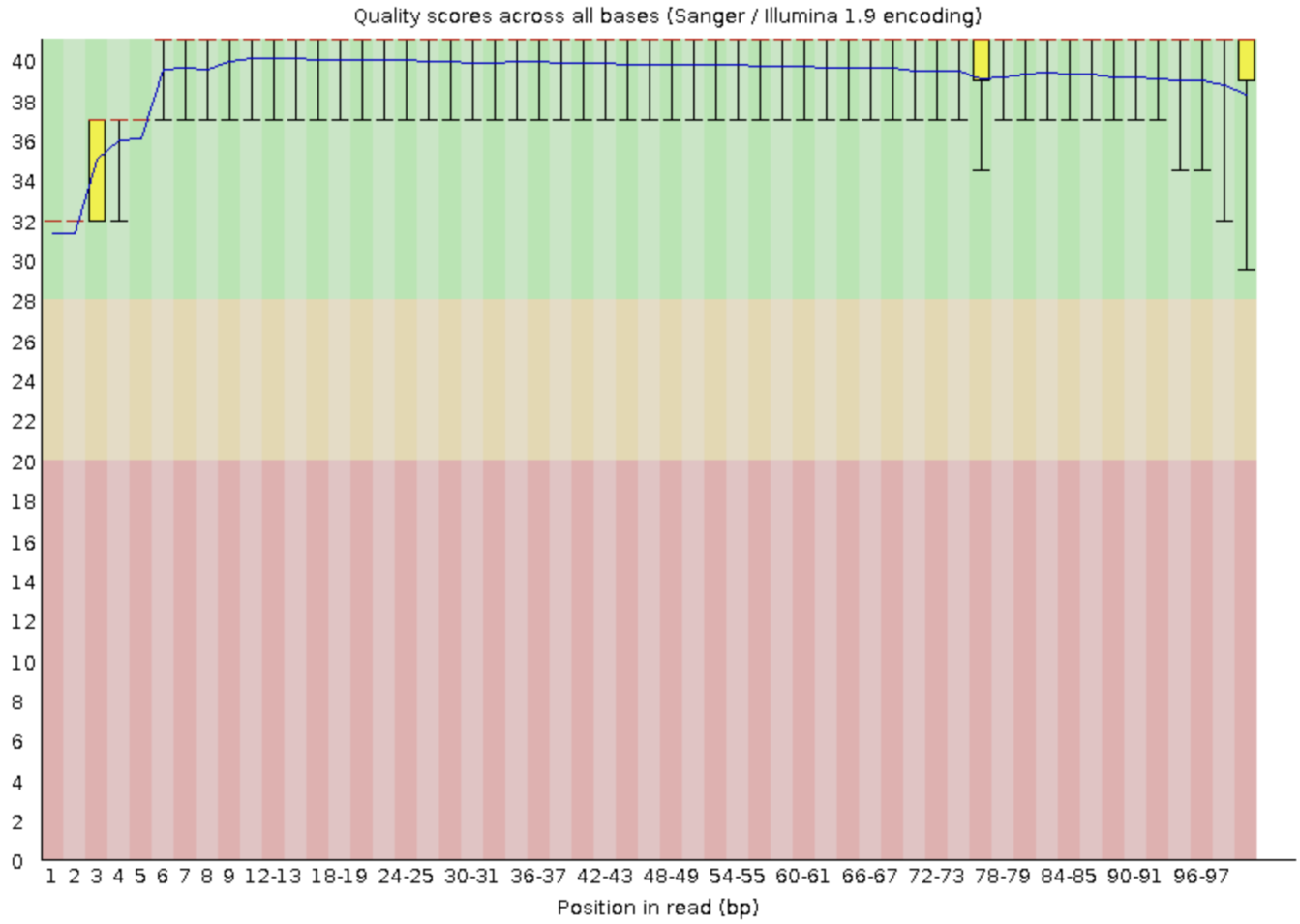
In comparing the charts generated by fastqc with those generated from my quality-score graphs, it appears that these graphs are mostly the same, with some deviation in the drops of quality experienced at the ends of the reads. In the graphs generated by fastqc, it appears that these drops are larger than they are in the graphs I created. However, the overall trends are consistent between these two graphs.

Overall, the quality score appears to be consistently above 35 in both read 1 files, with some dips at the beginning and end of the reads. however, the quality is significantly less consistent and lower quality in both R2 files, with some dips falling as low as a quaity score of about 16. It appears, then, that the R1 files are much higher quality on a per-base basis, while the R2 files offer data that is less consistently high quality.

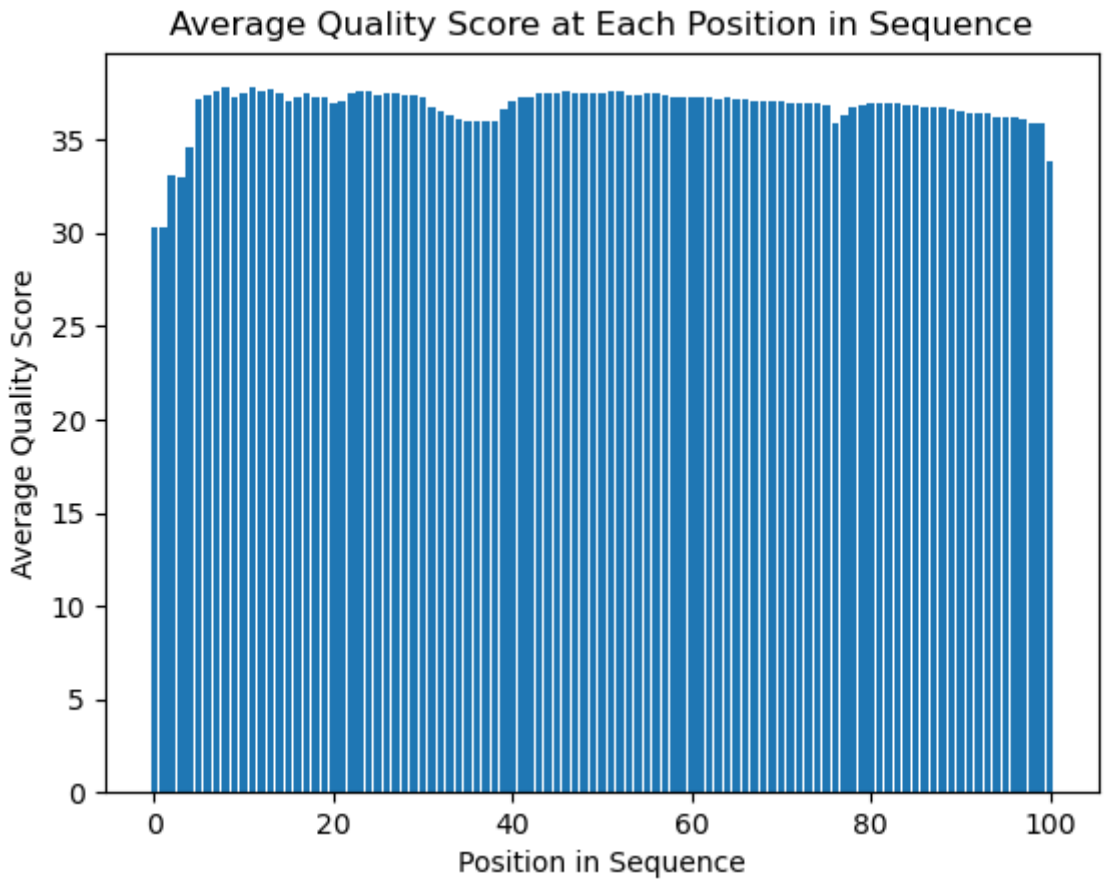
3_2B_control_S3_L008_R1 Quality Score Distributions

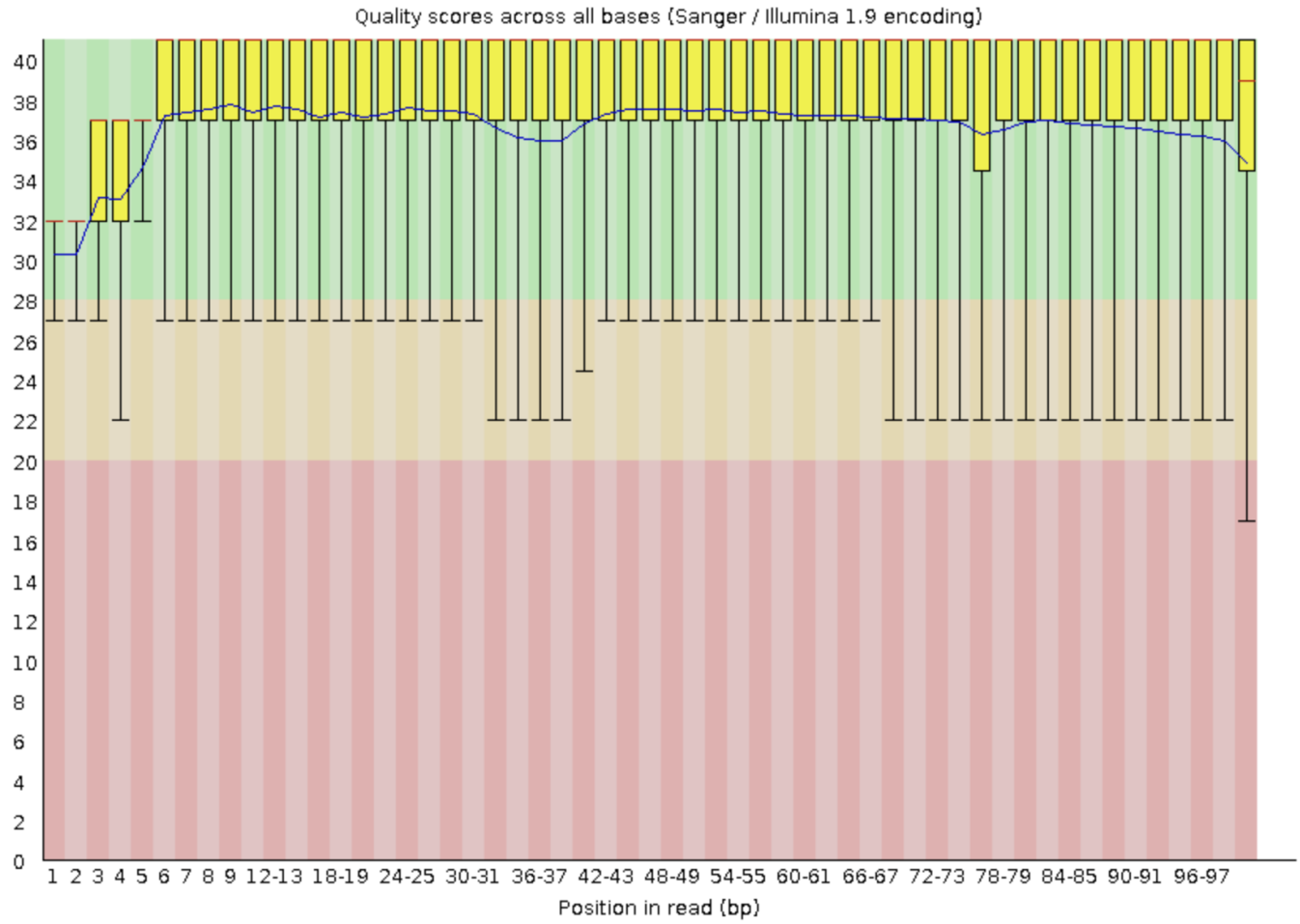
(top distribution generated via python script from demux assignment part 1, bottom via fastqc)



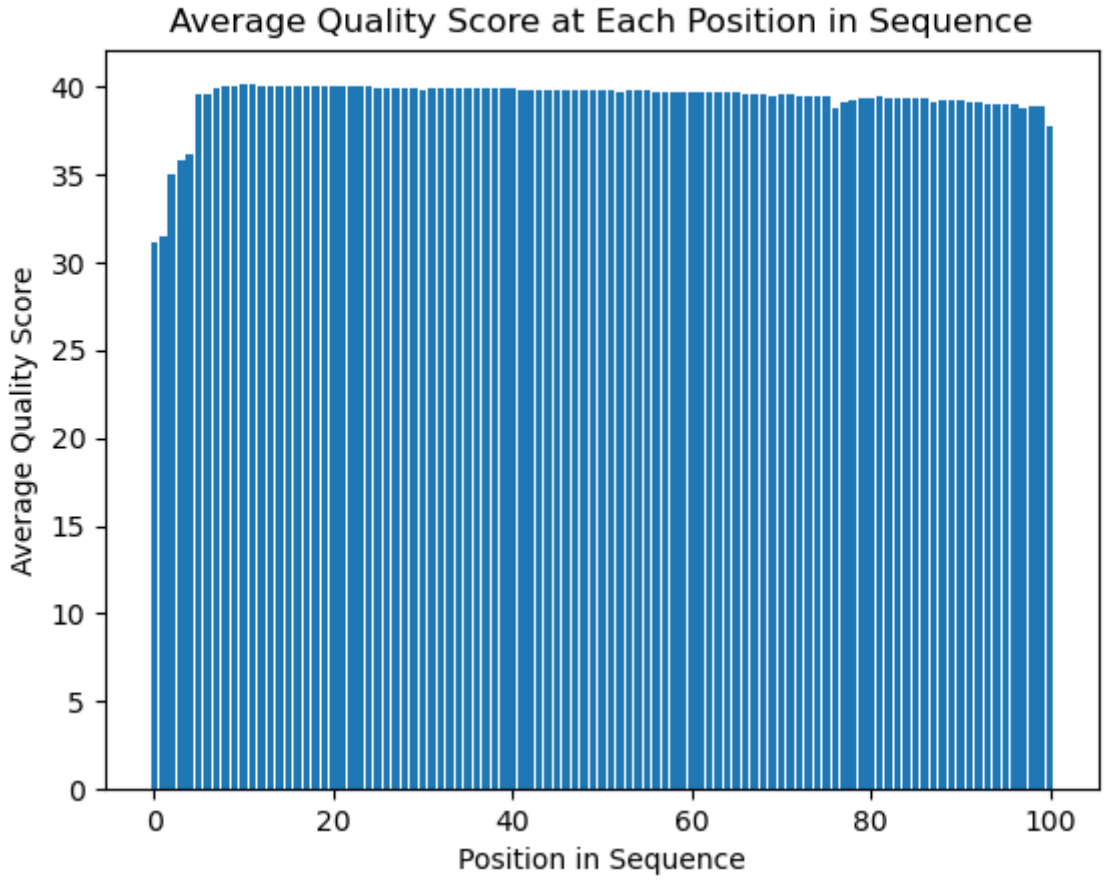


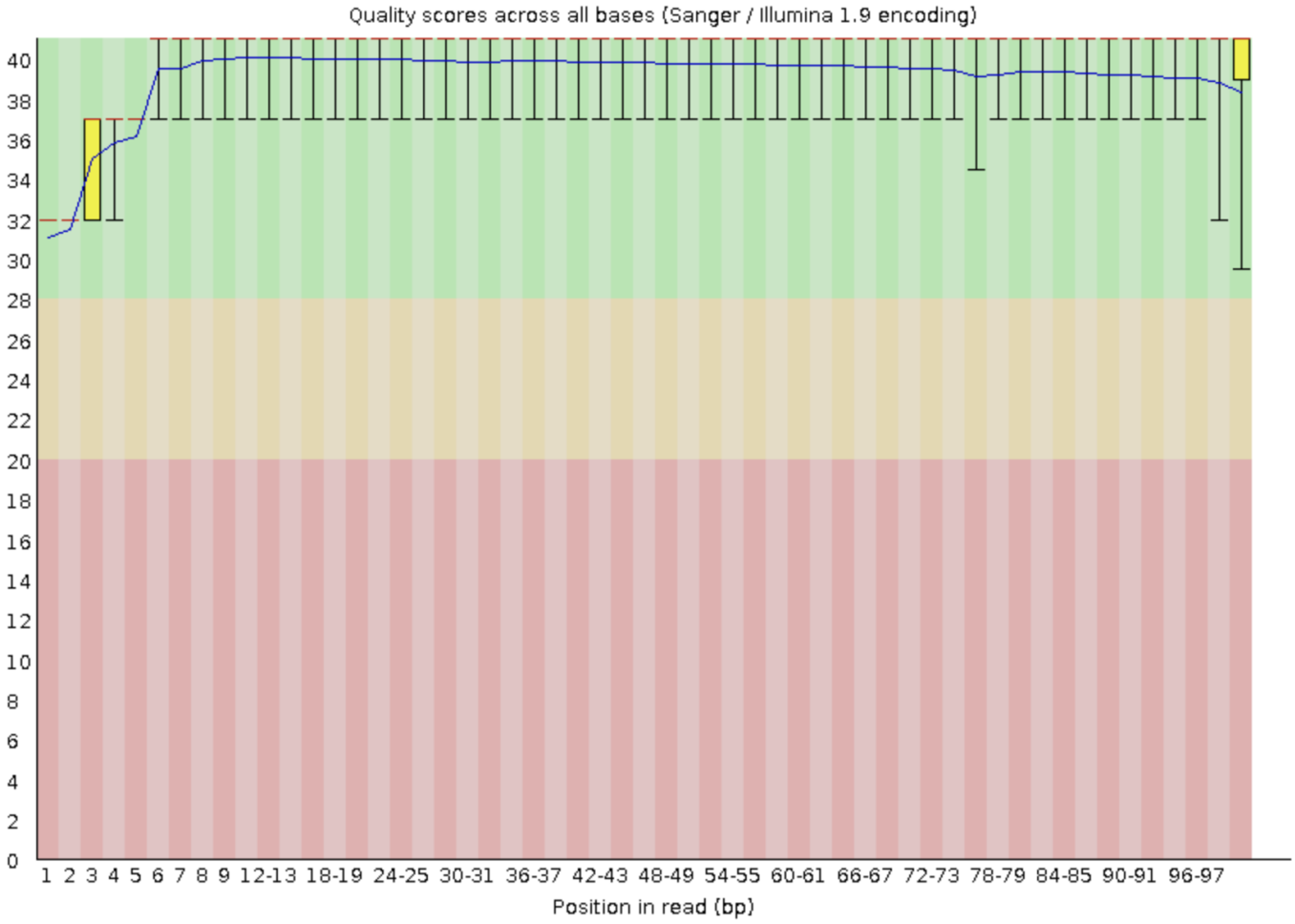
3_2B_control_S3_L008_R2 Quality Score Distributions



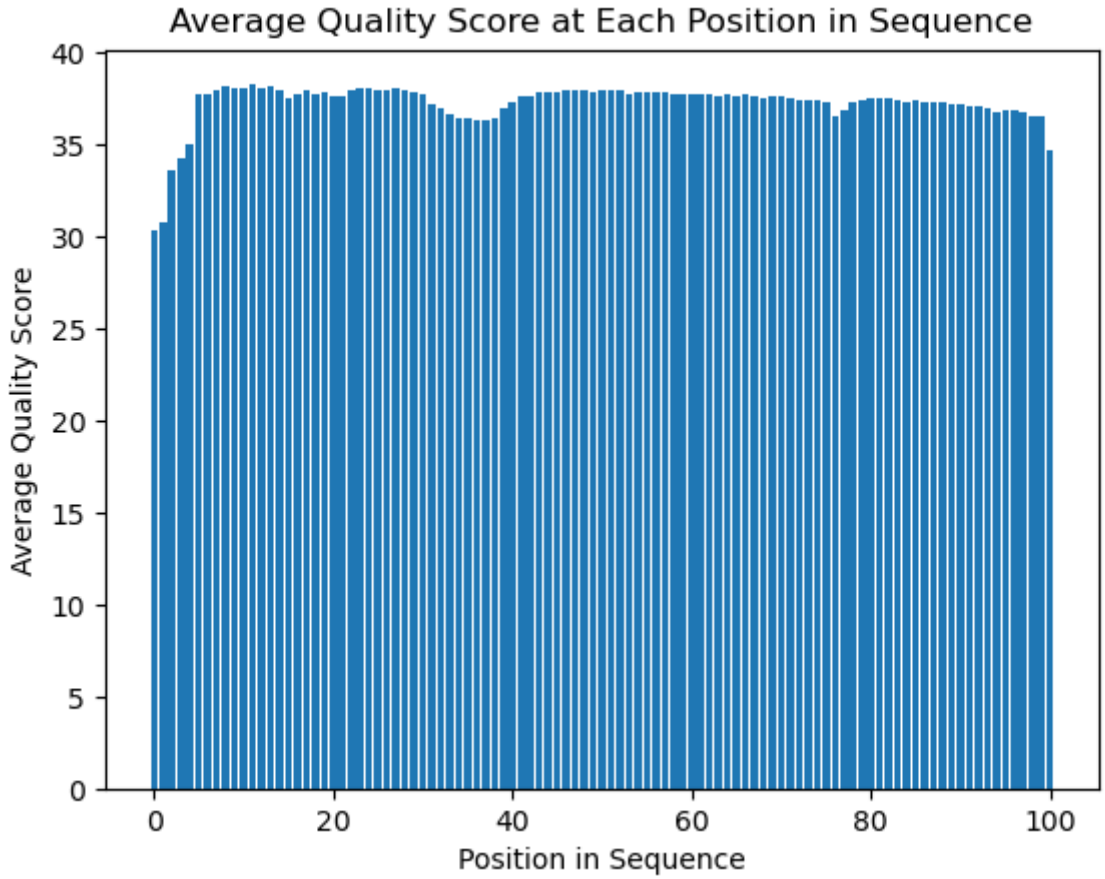


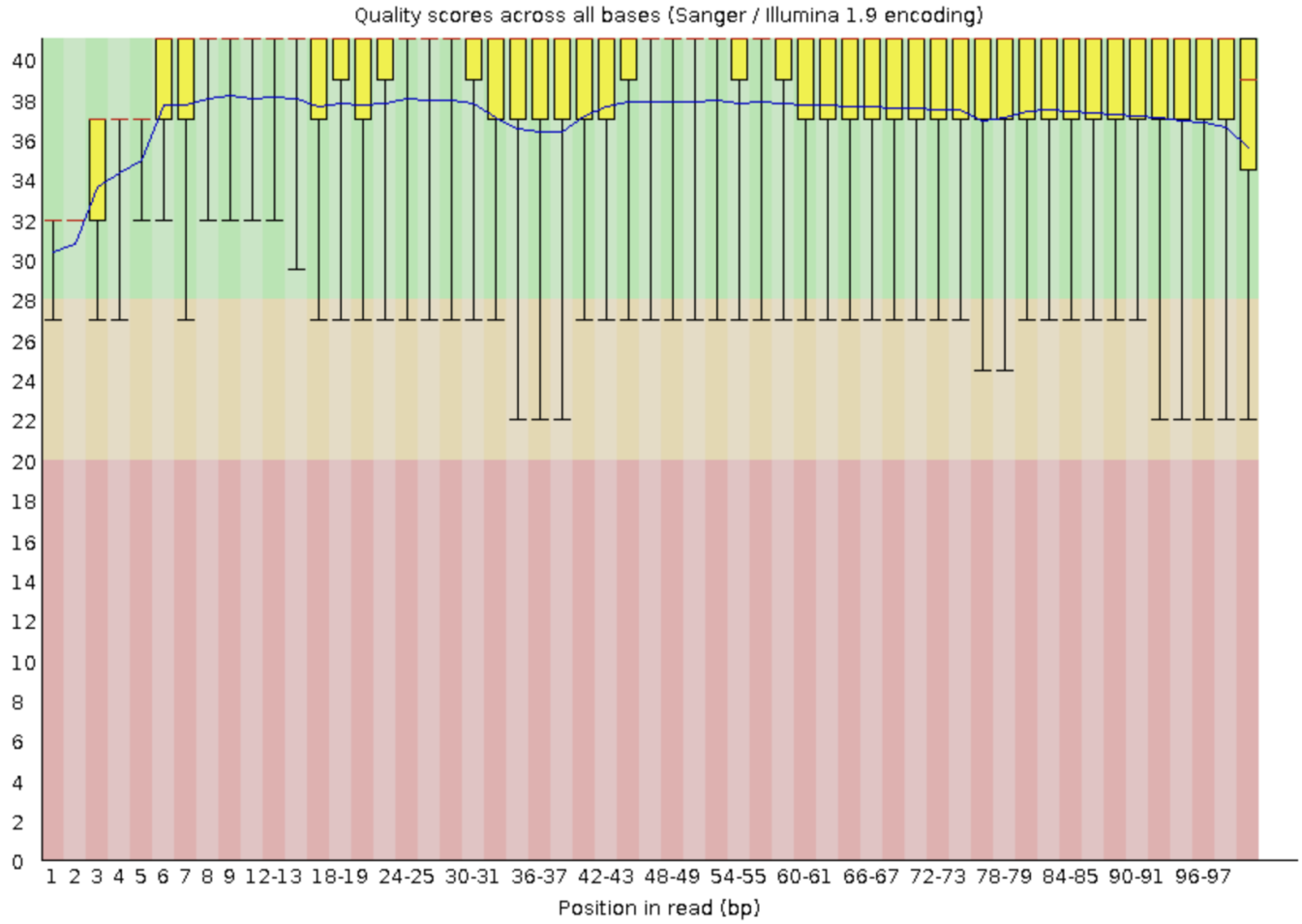
28_4D_mbnl_S20_L008_R1 Quality Score Distributions





28_4D_mbnl_S20_L008_R2 Quality Score Distributions





Part 2

cutadapt:

used command:

```
cutadapt -a AGATCGGAAGAGCACACGTCTGAACTCCAGTCA -A
AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT -o 28_4D_R1.fastq -p 28_4D_R2.fastq
/projects/bgmp/shared/2017_sequencing/demultiplexed/28_4D_mbnl_S20_L008_R1_001.f
astq.gz
/projects/bgmp/shared/2017_sequencing/demultiplexed/28_4D_mbnl_S20_L008_R2_001.f
astq.gz
```

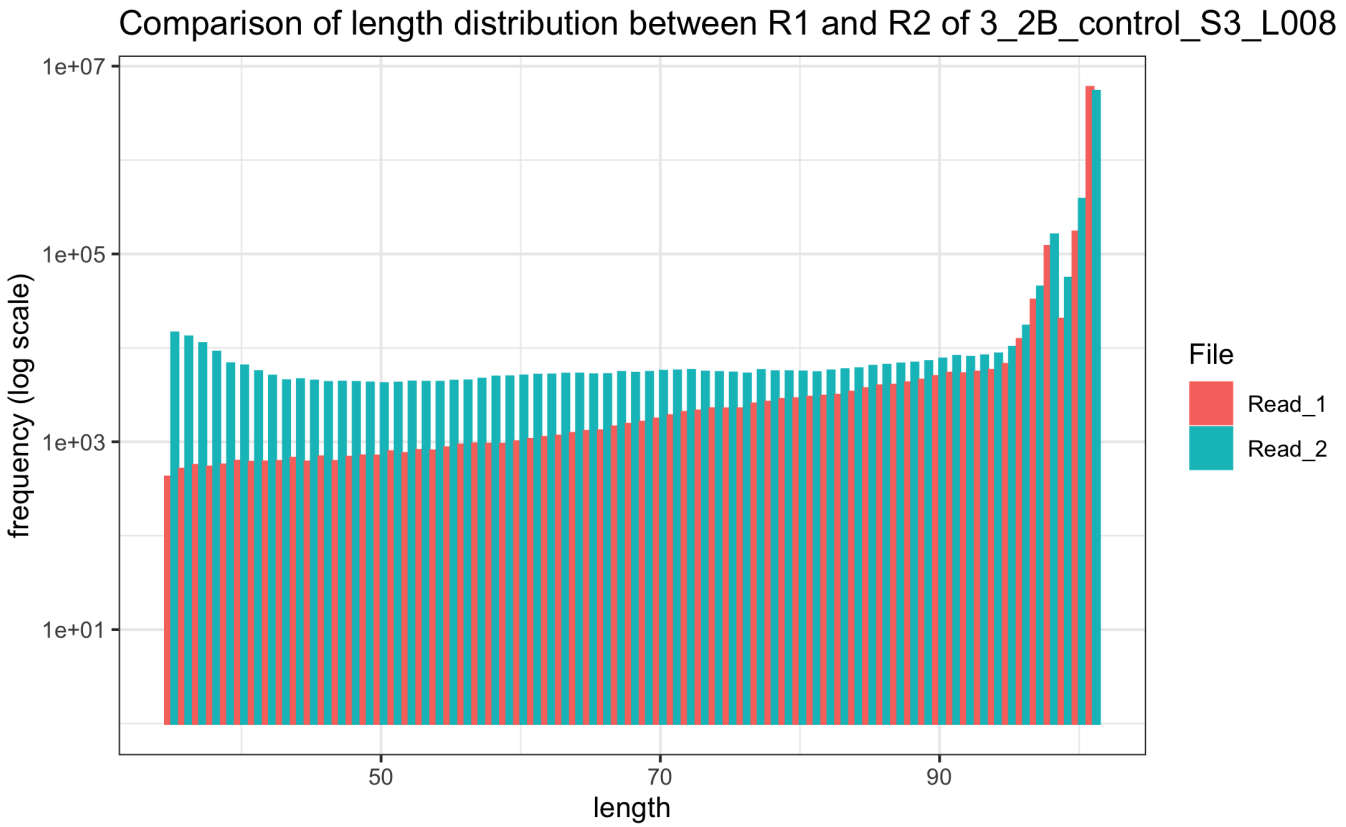
28_4D: From the read 1 file, it appears that 6.0% of the reads had the adapter and were trimmed. From read 2, 6.8% of the reads had the adapter and were trimmed.

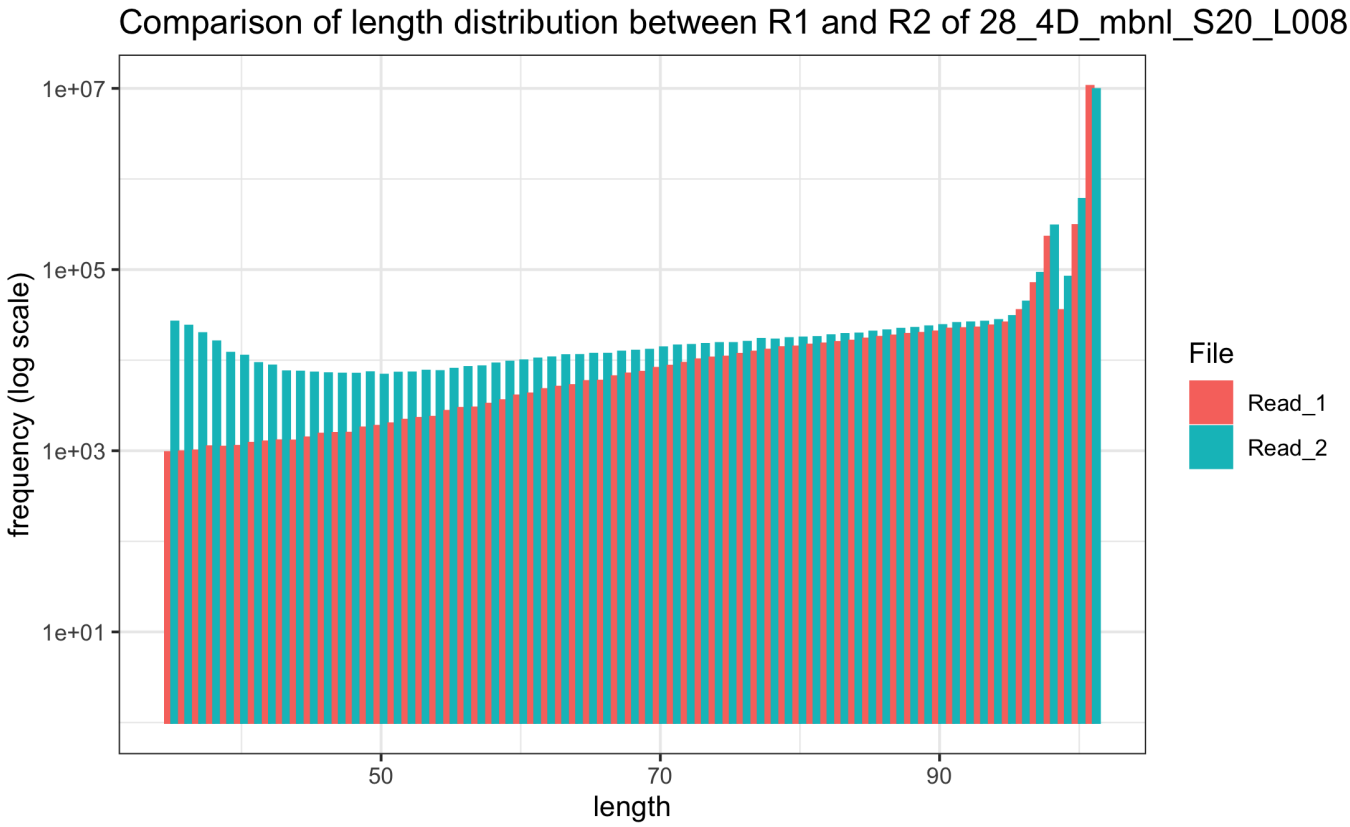
3_2B: From the read 1 file, it appears that 3.2% of the reads had the adapter and were trimmed. From read 2, 3.9% of the reads had the adapter and were trimmed.

command used:

```
/usr/bin/time -v trimmomatic PE 28_4D_R1.fastq 28_4D_R2.fastq
28_4D_R1trimmed.fastq.gz 28_4D_R1_untrimmed.fastq.gz 28_4D_R2trimmed.fastq.gz
28_4D_R2_untrimmed.fastq.gz LEADING:3 TRAILING:3 SLIDINGWINDOW:5:15 MINLEN:35
```

R plots of trimmed read length distributions for R1 and R2 of each library





Part 3

3_2B:

mapped:6140102

unmapped:239784

total:6379886

28_4D:

mapped:11264152

unmapped:384245

total:11648397

htseq-count command used: `/usr/bin/time -v htseq-count --stranded=yes Aligned_3_2b.Aligned.out.sam Mus_musculus.GRCm39.104.gtf`

In determining the strandedness of the reads, I compared the summary statistics provided at the end of the htseq-count output between the two conditions (stranded vs unstranded). In the case of both outputs, there was a roughly ten-fold increase in reads with no feature present in the stranded counts. While initially, I thought that this was definite evidence of non-strand-specific libraries, the fact that the number of ambiguous reads increases so much leads me to believe the opposite, that these are strand-specific RNA-seq libraries, my reasoning being that the ambiguous reads increased by many more fold, and that the ambiguity (ambiguous counts) was more indicative of a poor fit than absolutes (no feature).

HTSeq-Count Results:

Stranded;

3_2B:

Category	Count
__no_feature	5651011
__ambiguous	5177
__too_low_aQual	15299
__not_aligned	239784
__alignment_not_unique	281854

28_4D:

Category	Count
__no_feature	10345760
__ambiguous	8533
__too_low_aQual	22843
__not_aligned	384245
__alignment_not_unique	539306

Unstranded;

3_2B:

Category	Count
__no_feature	503343
__ambiguous	311325
__too_low_aQual	15299
__not_aligned	239784
__alignment_not_unique	281854

28_4D:

Category	Count
__no_feature	822053
__ambiguous	542147
__too_low_aQual	22843
__not_aligned	384245
__alignment_not_unique	539306