Bi623_QAA_EWong_Report_20230915

2023-09-12

Contents

Bi623 QAA Report Assigned reads:

28_4D_mbnl_S20_L008

3_2B_control_S3_L008

QAA Pt 1-Read quality score distributions

Description:

Plots included for part 1 of QAA include: FastQC Plots, per-base quality score distributions for R1 and R2 reads, per-base N content, and plots produced from Demultiplexing.

Path to FastQC plotting script: https://github.com/evelyn-n-wong/QAA/blob/master/fastqc_plot_script_pt1.sh

For the python script that demultiplex_comparison.sh (path: pt1_output/demultiplex_output/demultiplex_comparison.sh) references on GitHub, see: https://github.com/evelyn-n-wong/Demultiplex/tree/master/Assignment-the-first/Part_1/Bi622_Pt1_Qscore_Dist.py

Pages 2-9 are all FastQC produced graphs. Included graphs are: - adapter content - duplication levels - kmer profiles - per base n content - per base quality - per base sequence content - per sequence gc content - per sequence quality - per tile quality - sequence length distribution

##1. FastQC output plots:

28_4D_mbnl_S20_L008_R1_001

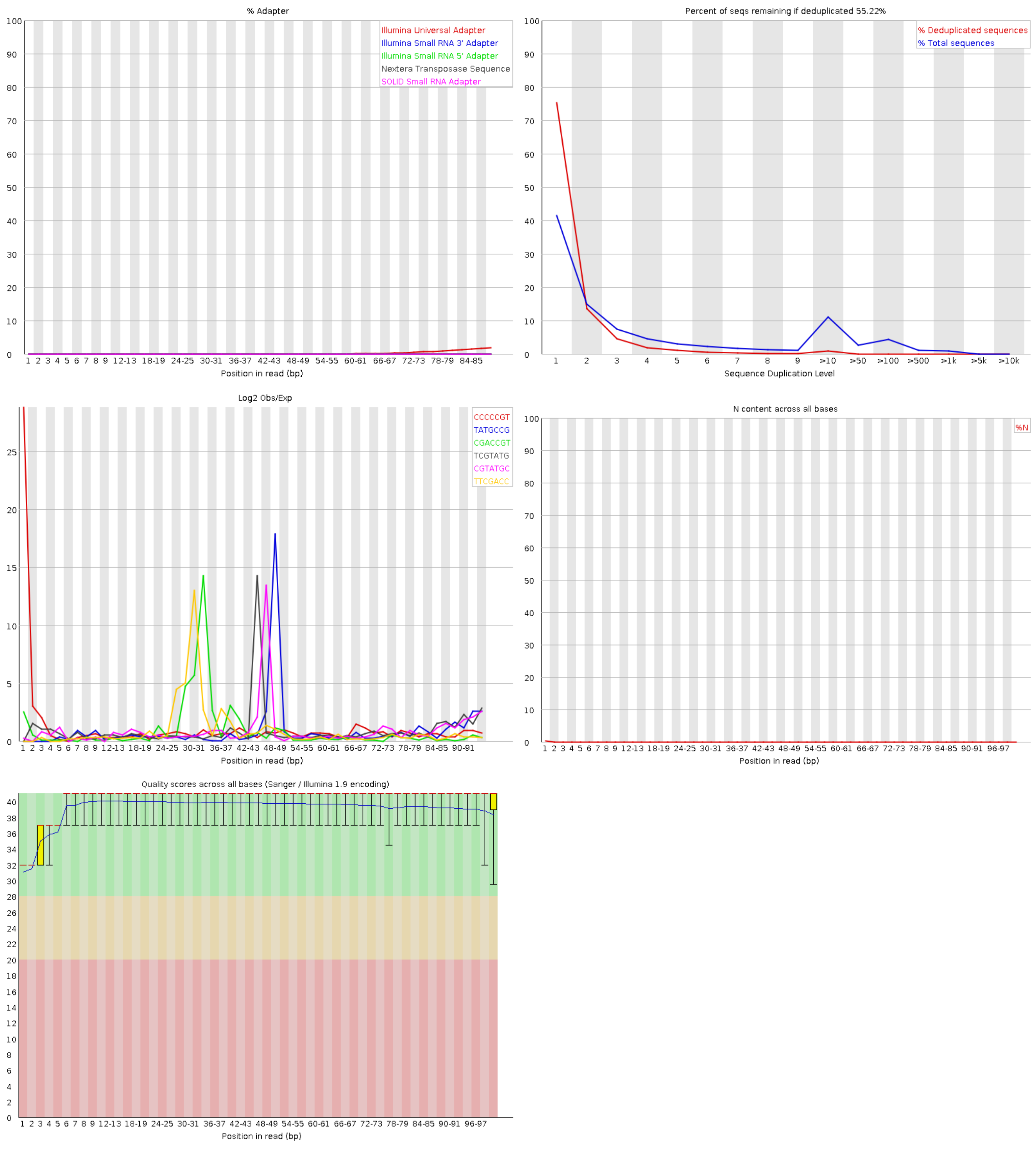

Figure 1: FastQC Generated Plots: 28-4D-mbnl-S20-L008-R1-001

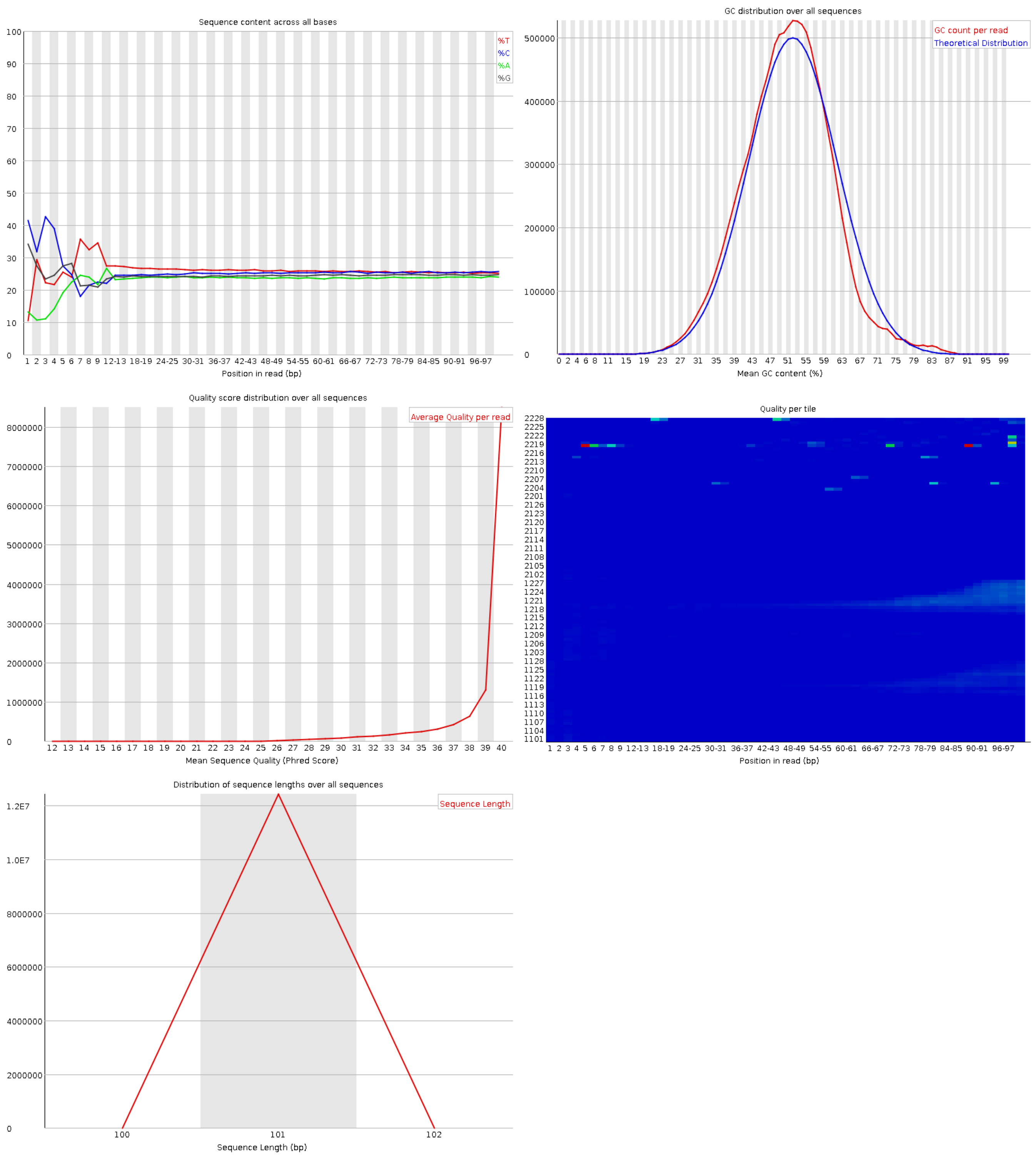

Figure 2: FastQC Generated Plots: 28-4D-mbnl-S20-L008-R1-001

28_4D_mbnl_S20_L008_R2_001

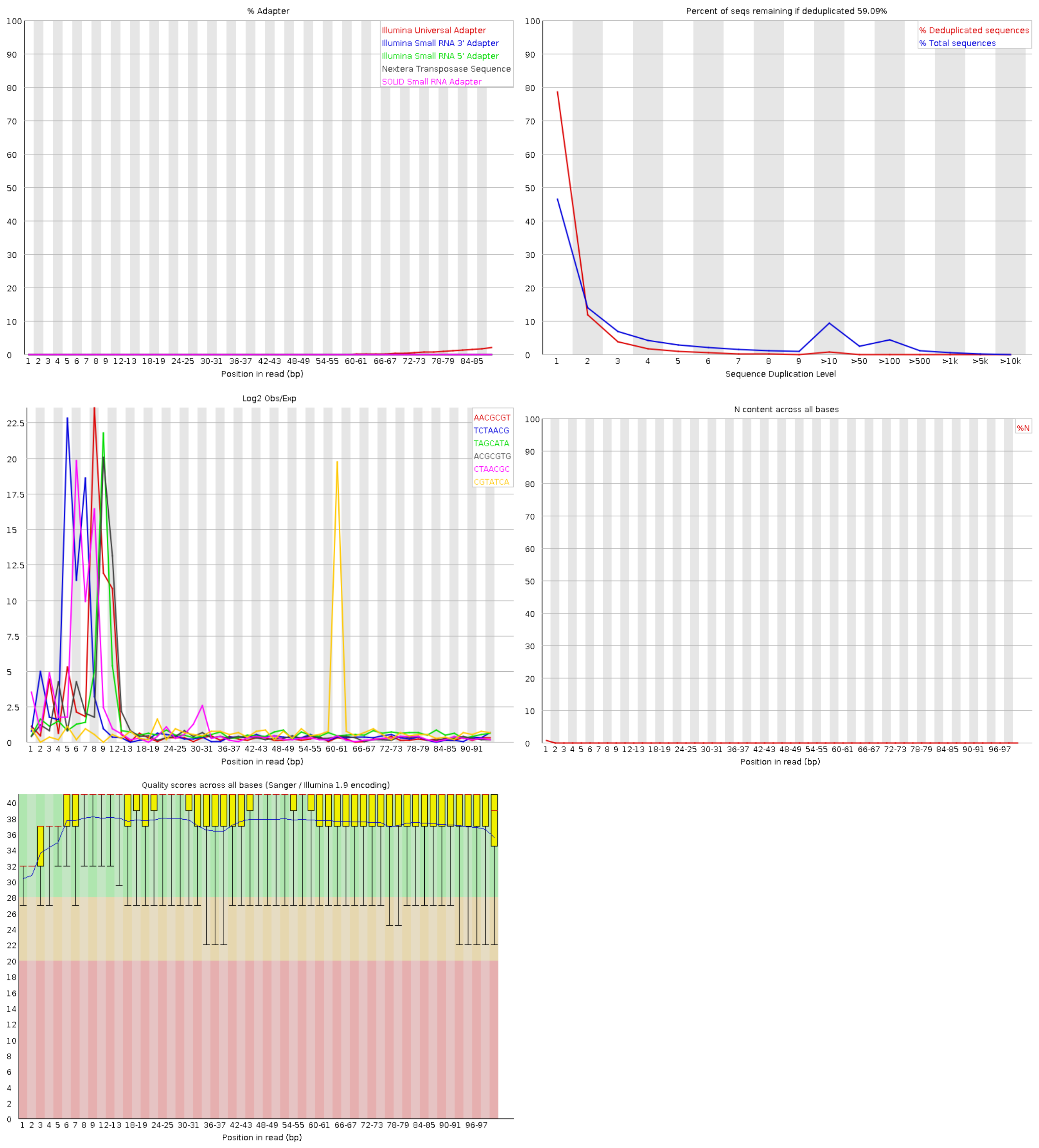

Figure 3: FastQC Generated Plots: 28-4D-mbnl-S20-L008-R2-001

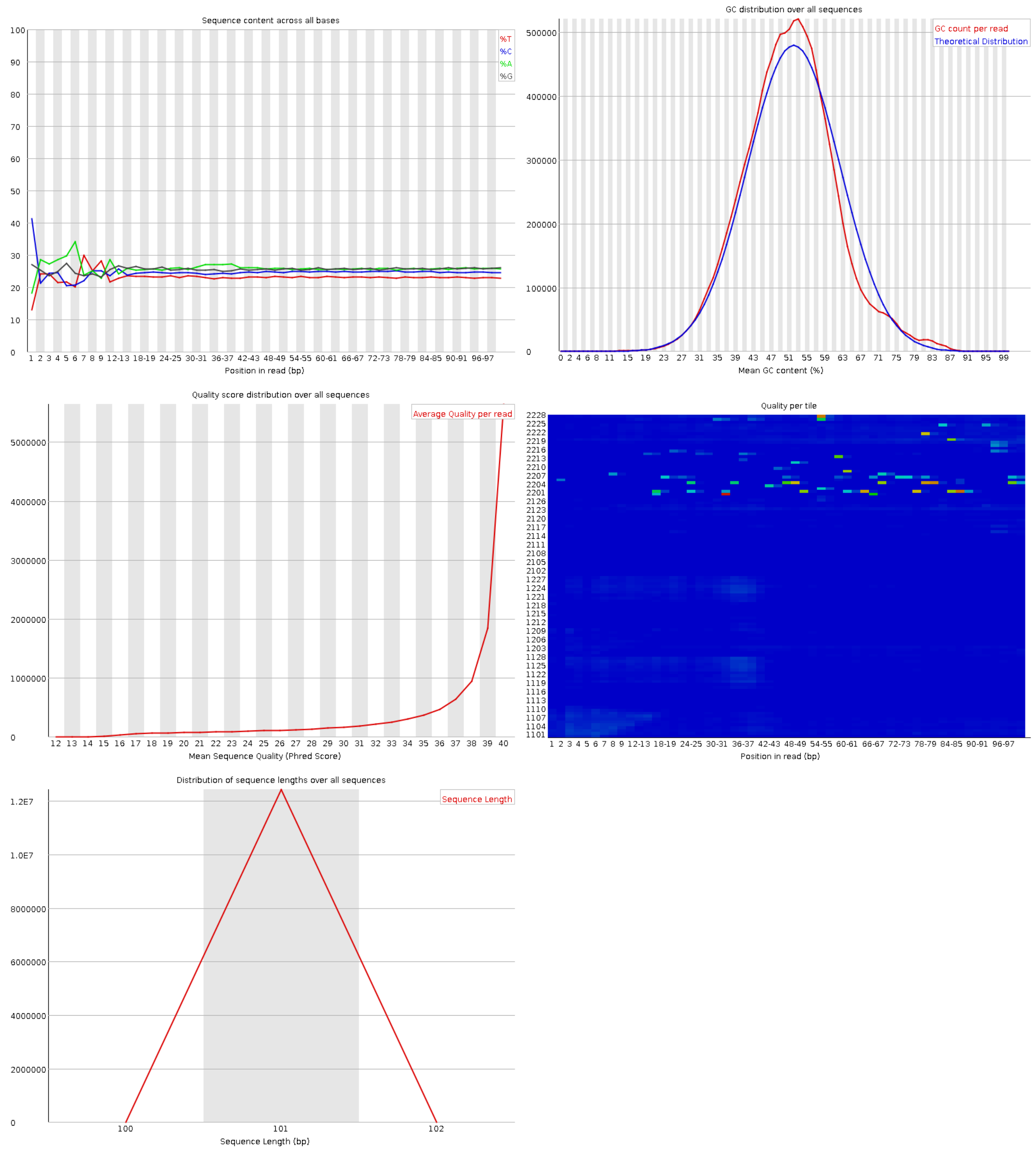

Figure 4: FastQC Generated Plots: 28-4D-mbnl-S20-L008-R2-001

3_2B_control_S3_L008_R1_001

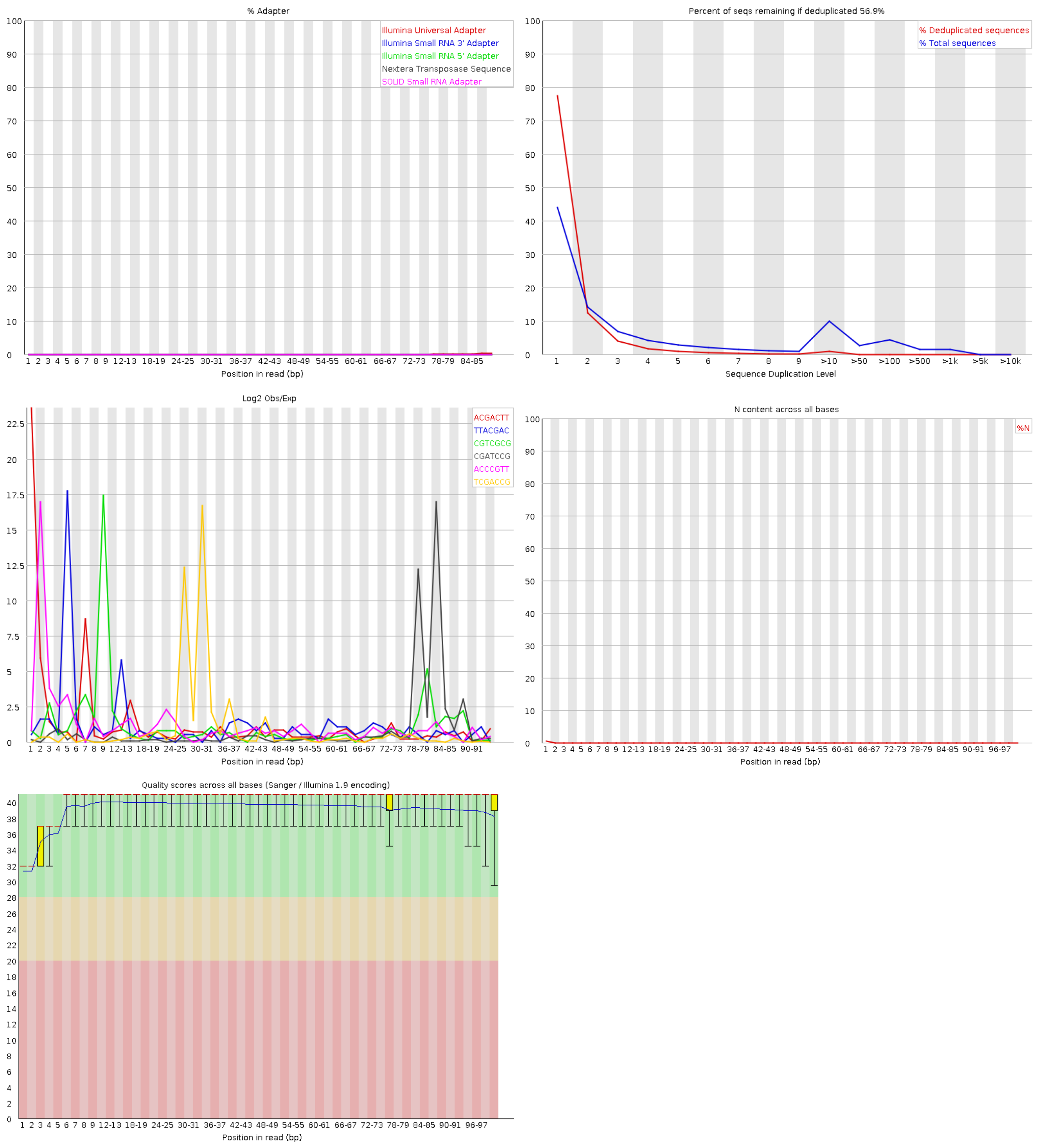

Figure 5: FastQC Generated Plots: 3-2B-control-S3-L008-R1-001

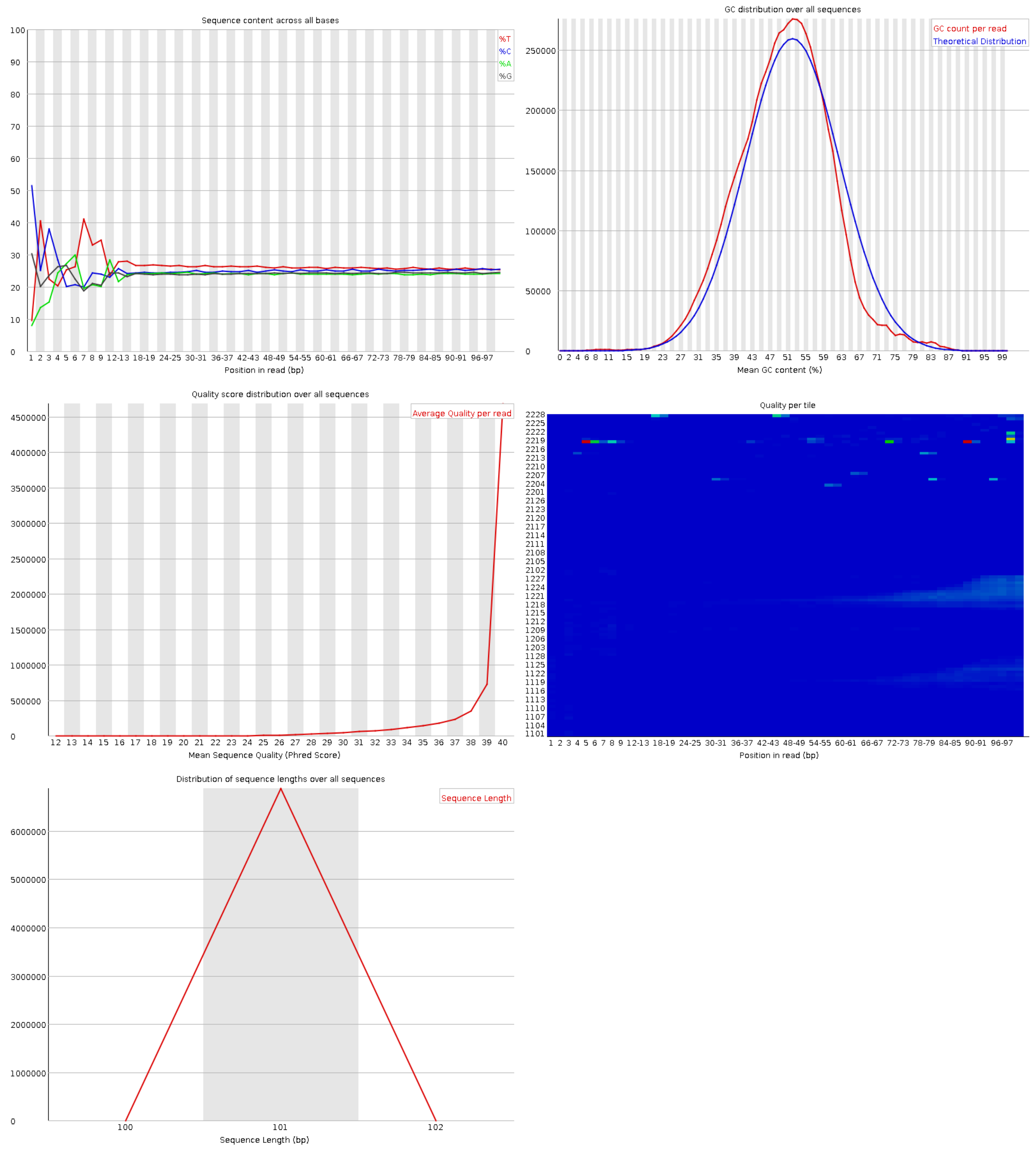

Figure 6: FastQC Generated Plots: 3-2B-control-S3-L008-R1-001

3_2B_control_S3_L008_R1_002

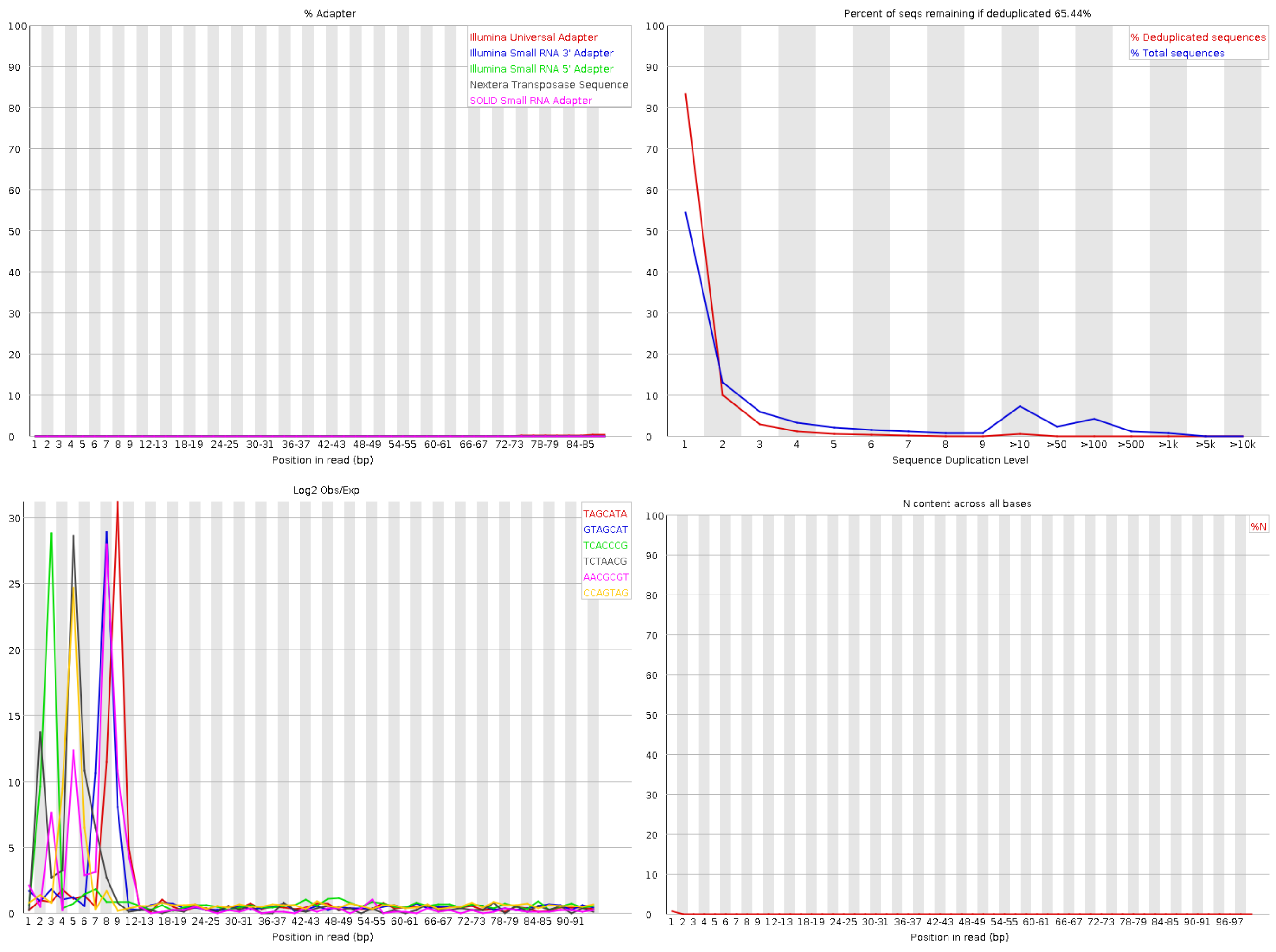

Figure 7: FastQC Generated Plots: 3-2B-control-S3-L008-R1-002

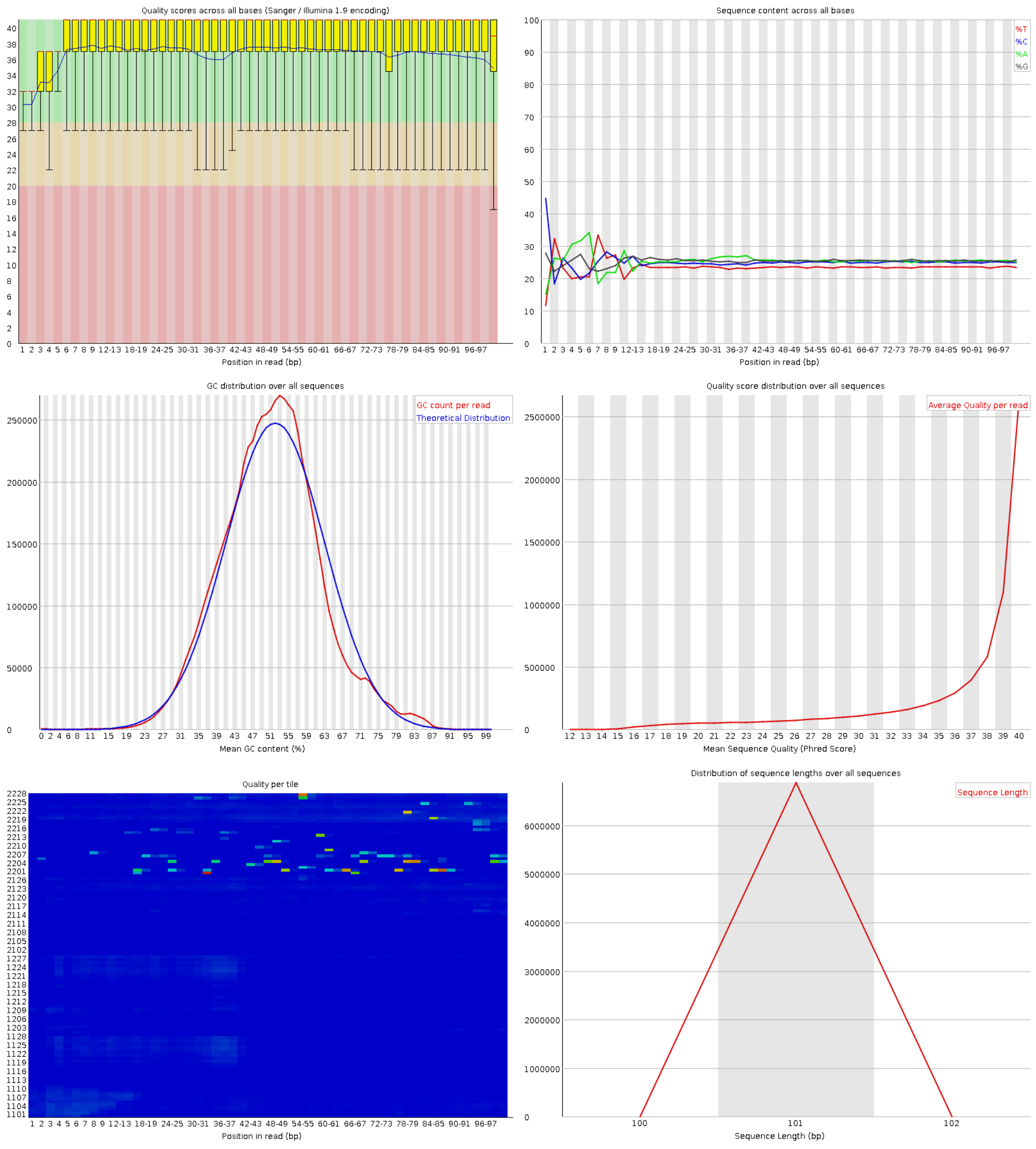

Figure 8: FastQC Generated Plots: 3-2B-control-S3-L008-R1-002

##2. Demultiplex Plots

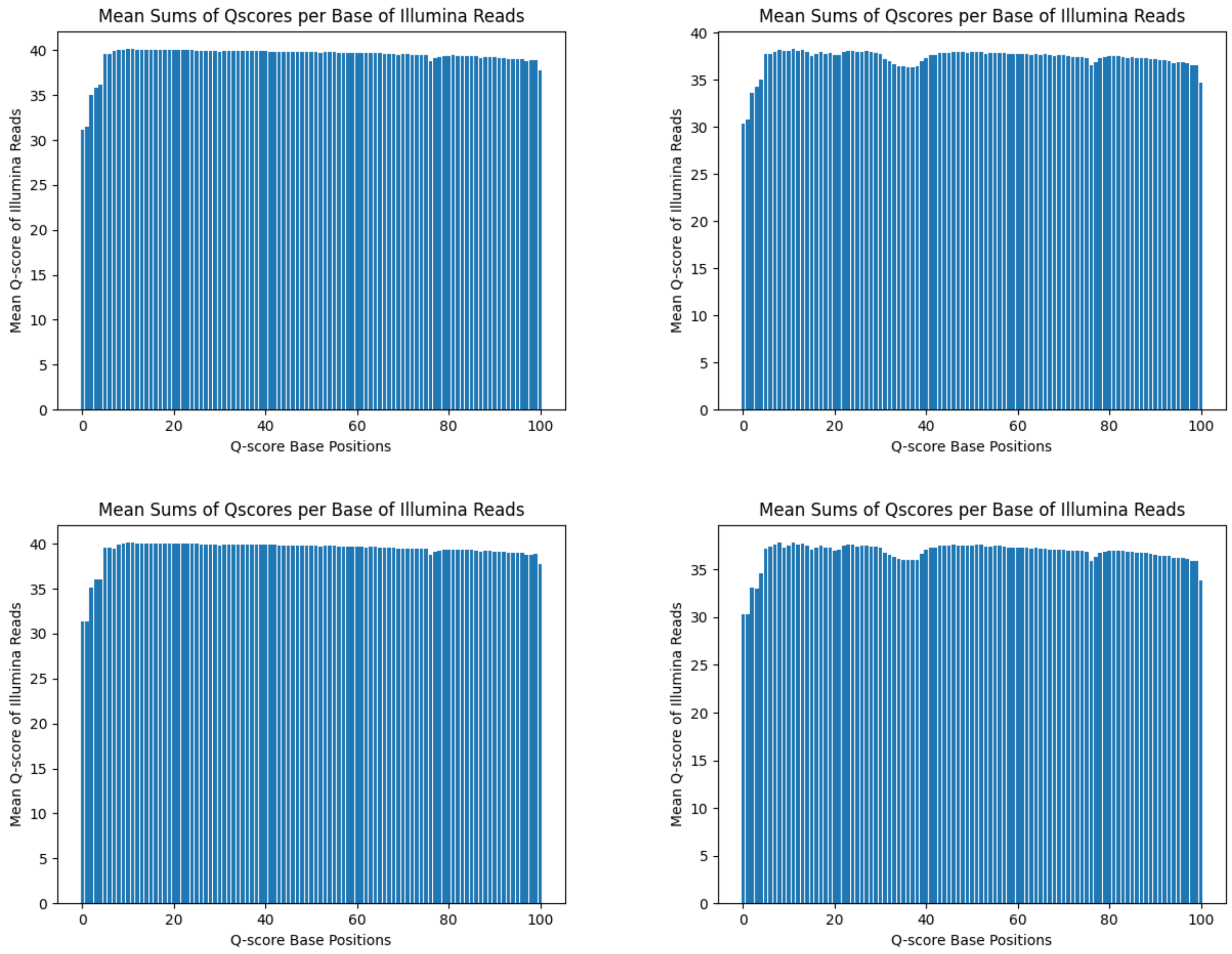

Figure 9: Demultiplex Generated Plots: 28-4D-mbnl-S20-L008-R1-001 and 3-2B-control-S3-L008-R1/R2

Following running the demultiplex script, we are able to see that the general shape of plots generated were the same. Those produced by FastQC do contain the interquartiles defined by the lines as seen in quality scores across all bases. The majority of the lower quartiles end up in the green for Read 1s in both the shorter 3_2b_control and longer 28_4_mbnl reads. We observe that Read 2s do end up toward yellow as we head to the end of the sequences which is expected. Looking at the number of adapters, we also see that they populate toward the end. The plots of the per-base N content and quality score plots are consistent.

In terms of running FastQC and Demulitplex, FastQC is faster by far when comparing the elapsed time required for the reads in both the longer 28_2_mbnl sequences and shorter 3_2b_control than during demultiplex processing from my demultiplex script.

Each file is processed serially. See below for a table of the elapsed time between FastQC and demultiplex.

Program	Elapsed Time for R1 28_4_mbnl (mm:ss)	Elapsed Time for R2 28_4_mbnl (mm:ss)	Elapsed Time for R1 3_2b_control (mm:ss)	Elapsed Time for R2 3_2b_control (mm:ss)
FastQC	1:04.75	1:05.19	0:36.18	0:38.30
Demultiplex	3:49.27	3:49.82	2:09.91	2:08.46

Mean quality distribution scores are consistent and don't fall below Q20. Looking at the sequence quality scores for each read on the summary.txt files from FastQC, they all pass. Meanwhile, looking at per-tile sequence quality as we go from Read 1 to Read 2 seems to "fail" for R1 and "warn" for R2 showing an increase in strictness or improvement. Also looking at the average quality score per read, most of them fall around the range of ~40. This suggests that the data is high-quality enough to use for further analysis.

QAA Pt 2-Adaptor trimming comparison

cutadapt ver. 4.4 was utilized to trim adapter sequences from 28_4D and 3_2B sequences.

Adapters were identified after looking ath the adapter_content.png and referring to https://knowledge.illumina.com/library-preparation/general/library-preparation-general-reference_material-list/000001314. Illumina universal adapters were listed as follows for paired reads:

Read 1: AGATCGGAAGAGCACACGTCTGAACTCCAGTCA

Read 2: AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT

Command used:

```
cutadapt -a <Read 1 adapter> -A <Read 2 adapter> -o <output1.fastq> <output2.fastq> <input1.fastq>
<input2.fastq>
```

Example:

```
cutadapt -a AGATCGGAAGAGCACACGTCTGAACTCCAGTCA -A AGATCGGAAGAGCGTCGTG-
TAGGGAAAGAGTGT -o /projects/bgmp/evew/bioinfo/Bi623/QAA/pt1_output/demultiplex_output/28_4D_mbnl_S20
-p /projects/bgmp/evew/bioinfo/Bi623/QAA/pt1_output/demultiplex_output/28_4D_mbnl_S20_L008_R2_001_out.fas
/projects/bgmp/shared/2017_sequencing/demultiplexed/28_4D_mbnl_S20_L008_R1_001.fastq.gz
/projects/bgmp/shared/2017_sequencing/demultiplexed/28_4D_mbnl_S20_L008_R2_001.fastq.gz
```

After using default settings with cutadapt, trimmomatic was used to quality trim the reads further. The proportion of reads that were trimmed:

Sequences	Read 1	Read 2	Total Base Pairs Processed	Total Written (filtered)
28_4D_mbnl_S20_L008	743,440	841,389	2,510,610,732 bp	2,489,647,234 bp (99.2%)
3_2B_control_S3_L008	219,477	268,119	1,388,448,818 bp	1,384,906,999 bp (99.7%)

Following Trimmomatic:

Sequences	Input Read Pairs	Both Surviving	Forward Only Surviving	Reverse Only Surviving	Dropped
28_4D_mbnl_S20_L008	12428766	11725400 (94.34%)	677662 (5.45%)	8727 (0.07%)	16977 (0.14%)
3_2B_control_S3_L008	6873509	6428019 (93.52%)	436824 (6.36%)	4648 (0.07%)	4018 (0.06%)

Command used:

```
trimmomatic PE -phred33 [input] [output]
```

Also specified LEADING:3, TRAILING:3, SLIDINGWINDOW:5:15 (window size of 5 and required quality of 15), and MINLEN:35

There are four output files per sequence. See: “trimmomatic.sh” located at https://github.com/evelyn-n-wong/QAA/tree/master/pt2_output for the script used to run trimmomatic.

Below are the trimmed read length distributions plots for each sequence.

28_4D_mbnl_S20_L008 Reads 1 and 2

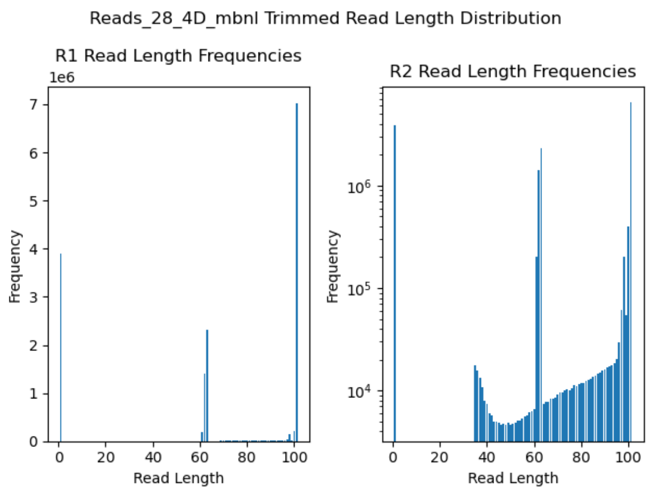

Figure 10: Reads-28-4D-mbnl Trimmed Read Length Distributions

3_2B_control_S3_L008 Reads 1 and 2

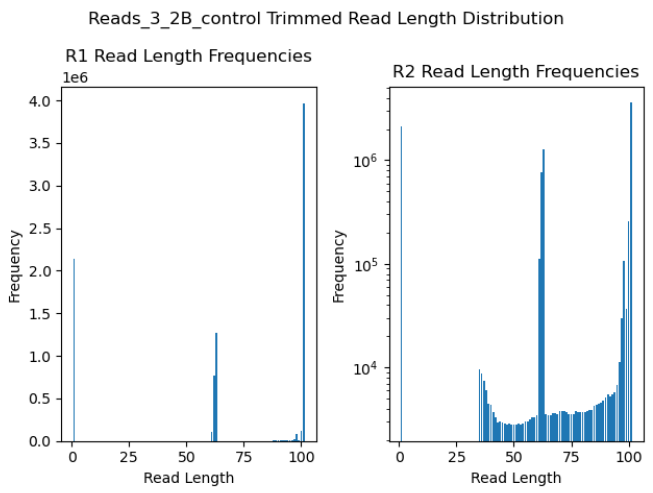

Figure 11: Read-3-2b-control Trimmed Read Length Distributions

Had expected R1s and R2s to be trimmed at similar rates, but Read 2 as shown in the graphs are trimmed further. Read 2 generally has lower quality (by nature of being sequenced later and possible exposure to DNA degradation) which we saw represented earlier in the FastQC graphs as well. Because trimmomatic also has a sliding window which reads R1 and then R2 sequentially, we do expect that Read 2 would also have shorter lengths which would result in Read2 being quality trimmed further.

Reutilized a PS8 assignment script to report number of mapped and unmapped reads in the 2 outputted SAM files:

Mouse_QAA_new_28_4D_mbnl_S20_L008_Aligned.out.sam

Batch script for running PS8 script: Mouse_QAA_STAR_parser.sh

Sequence	Number Mapped Reads	Number Unmapped Reads
28_4D_mbnl_S20_L008	22657634	793166
3_2B_control_S3_L008	12359959	496079

QAA Pt 3-Alignment and strand specificity

Used STAR to create a database for mus musculus reference genome and align reads. htseq was then used to count the reads that were mapped to features twice: once with –stranded=yes, and again with –stranded=reverse.

From Ensembl Release 110:

Mouse reference genome by chromosome (FASTA): Mus_musculus.GRCm39.dna.primary_assembly.fa.gz

Mouse reference genome by gene set (GTF): Mus_musculus.GRCm39.110.gtf.gz

Command for htseq:

htseq-count –stranded=

Example:

/usr/bin/time htseq-count –stranded=yes /projects/bgmp/evew/bioinfo/Bi623/QAA/STAR_out/Mouse_QAA_new_28_4
/projects/bgmp/evew/bioinfo/Bi623/QAA/STAR/Mus_musculus.GRCm39.110.gtf

Table of Counts produced from Ensembl 110 Mus Musculus following alignment with STAR and htseq. Below are the heads with counts and the tails also include count information for ___no_feature, ___ambiguous, ___too_low_aQual, ___not_aligned, and ___alignment_not_unique.

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
stranded_28_4d <- read.delim("/Users/evelynwong/bioinformatics/Bi623/QAA/fastq_plots/htseq_counts/stran
head(stranded_28_4d)
```

```
##   ENSMUSG00000000001 X5
## 1 ENSMUSG00000000003  0
## 2 ENSMUSG00000000028  4
## 3 ENSMUSG00000000031  0
## 4 ENSMUSG00000000037  0
## 5 ENSMUSG00000000049 21
## 6 ENSMUSG00000000056  4
```

```r
tail(stranded_28_4d)
```

```
##           ENSMUSG00000000001       X5
## 56940     ENSMUSG00002076992        0
## 56941            __no_feature 10337055
## 56942             __ambiguous     8572
## 56943         __too_low_aQual    22846
## 56944           __not_aligned   384252
## 56945 __alignment_not_unique   539378
```

```r
rev_stranded_28_4d <- read.delim("/Users/evelynwong/bioinformatics/Bi623/QAA/fastq_plots/htseq_counts/re
head(rev_stranded_28_4d)
```

```
##   ENSMUSG00000000001 X2896
## 1 ENSMUSG00000000003     0
## 2 ENSMUSG00000000028   987
## 3 ENSMUSG00000000031     0
## 4 ENSMUSG00000000037     0
## 5 ENSMUSG00000000049     0
## 6 ENSMUSG00000000056   301
```

```r
tail(rev_stranded_28_4d)
```

```
##           ENSMUSG00000000001  X2896
## 56940     ENSMUSG00002076992      0
## 56941            __no_feature 888880
## 56942             __ambiguous 188817
## 56943         __too_low_aQual  22846
## 56944           __not_aligned 384252
## 56945 __alignment_not_unique 539378
```

```r
stranded_3_2B <- read.delim("/Users/evelynwong/bioinformatics/Bi623/QAA/fastq_plots/htseq_counts/strande
head(stranded_3_2B)
```

```
##   ENSMUSG00000000001 X3
## 1 ENSMUSG00000000003  0
## 2 ENSMUSG00000000028  0
## 3 ENSMUSG00000000031  0
## 4 ENSMUSG00000000037  0
## 5 ENSMUSG00000000049 15
## 6 ENSMUSG00000000056  2
```

```r
tail(stranded_3_2B)
```

```
##           ENSMUSG00000000001      X3
## 56940     ENSMUSG00002076992       0
## 56941            __no_feature 5645316
## 56942             __ambiguous    5207
## 56943         __too_low_aQual   15298
## 56944           __not_aligned  239785
## 56945 __alignment_not_unique  281880
```

```
rev_stranded_3_2B <- read.delim("/Users/evelynwong/bioinformatics/Bi623/QAA/fastq_plots/htseq_counts/re
head(rev_stranded_3_2B)
```

```
##   ENSMUSG00000000001 X1475
## 1 ENSMUSG00000000003     0
## 2 ENSMUSG00000000028   549
## 3 ENSMUSG00000000031     0
## 4 ENSMUSG00000000037     0
## 5 ENSMUSG00000000049     1
## 6 ENSMUSG00000000056   147
```

```
tail(rev_stranded_3_2B)
```

```
##           ENSMUSG00000000001  X1475
## 56940     ENSMUSG00002076992      0
## 56941            __no_feature 527458
## 56942             __ambiguous 103113
## 56943         __too_low_aQual  15298
## 56944           __not_aligned 239785
## 56945 __alignment_not_unique 281880
```

I propose that these data are strand-specific RNA-seq libraries because for 28_4D 3.70% of the reads mapped to forward strand, and 82.55% mapped to reverse. The numbers similar for the 3_2B. The reverse read file had more reads mapping to features. The first read must map to the same strand as the feature and the second read to the opposite. By changing the –stranded setting to =yes as opposed to =reversed, the number of mapped reads as shown in the table below was affected.

Command:

cat stranded/rev_stranded genecount file | grep -v "___" | awk ' {sum+=$2} END {print sum}'

Sequence	Number of Reads Mapped FW	Number of Reads Mapped RV	Total Reads	Percentage Reads Mapped FW	Percentage Reads Mapped RV
28_4D_mbnl_S20_L008	433297	9701227	11725400	0.03695371 (3.70%)	0.82546773 (82.55%)
3_2B_control_S3_L008	240533	5260485	6428019	0.03741946 (3.74%)	0.81835799 (81.84%)