Assignment - QAA

Angela Crabtree

9/1/2021

```
library(ggplot2)
library(tidyr)
```

Part 1 – Read quality score distributions

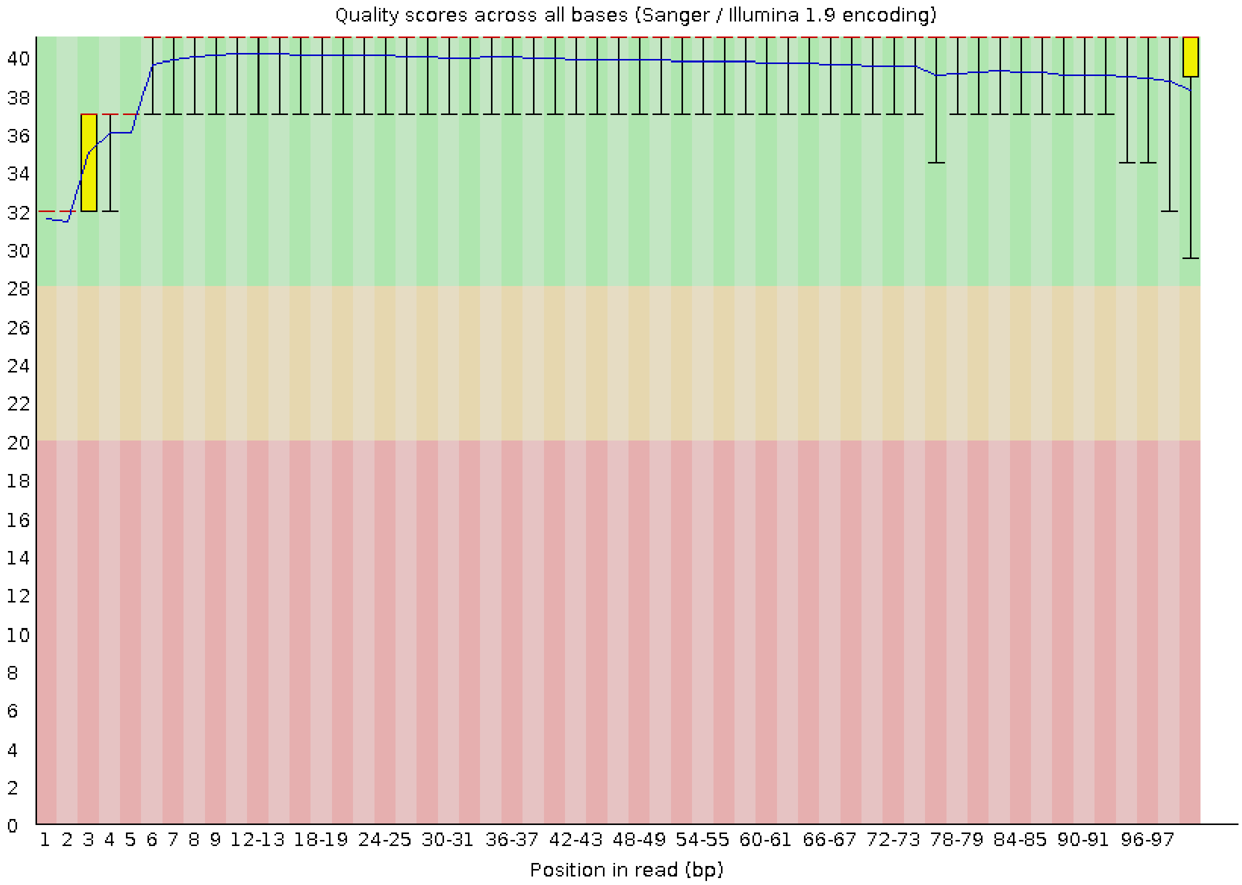
Samples

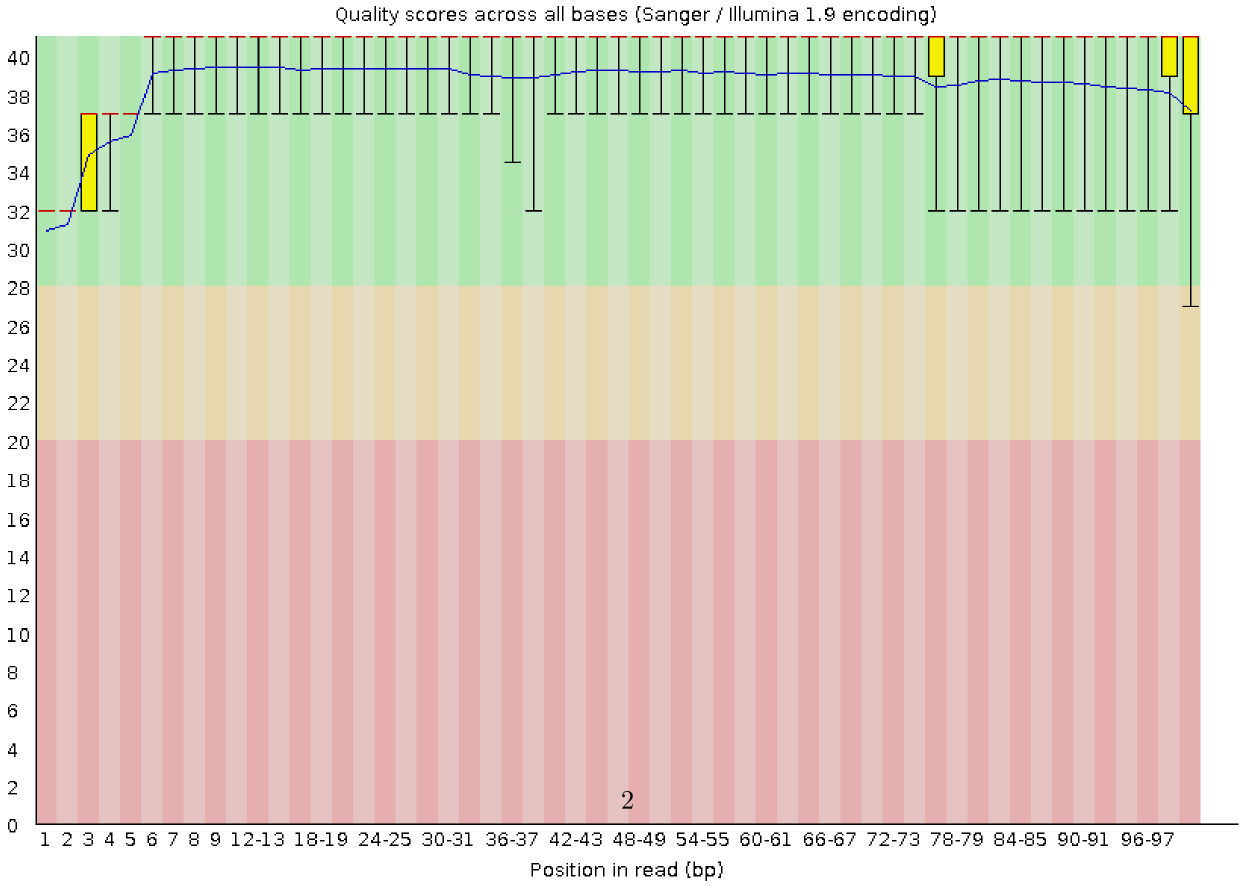
- 2_2B_control
- Undetermined

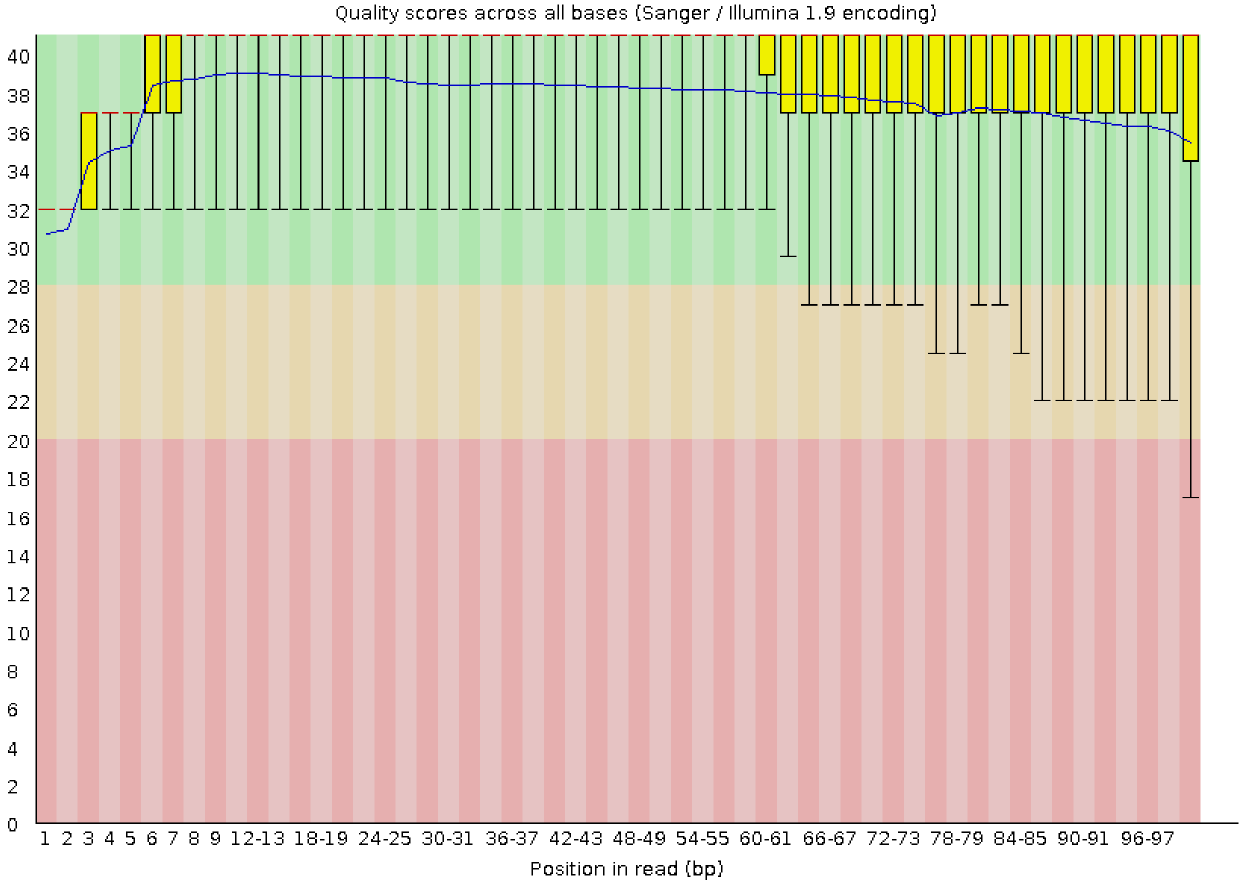
Running FastQC

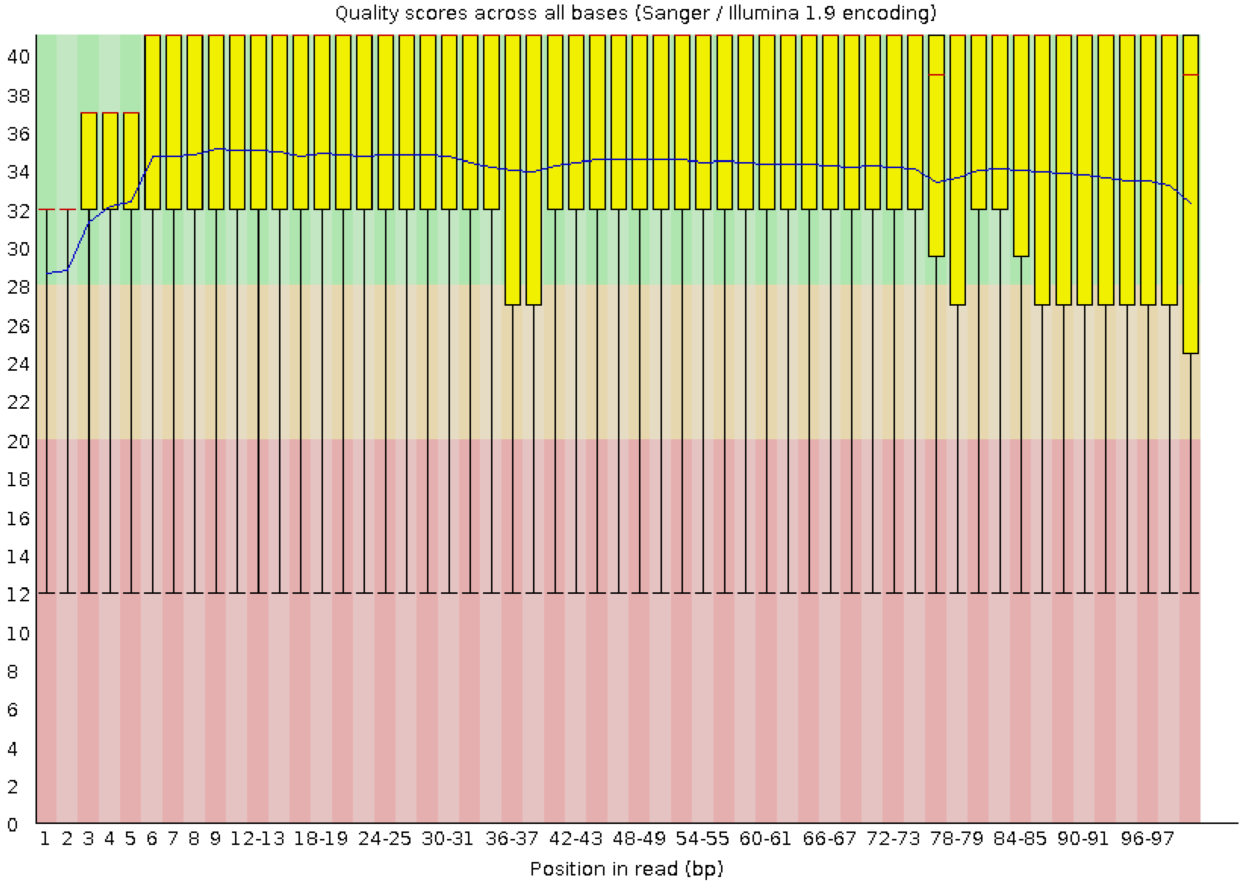
```
/usr/bin/time -v fastqc \
  -o . \
  /projects/bgmp/shared/2017_sequencing/demultiplexed/2_2B_control_S2_L008_R1_001.fastq.gz

/usr/bin/time -v fastqc \
  -o . \
  /projects/bgmp/shared/2017_sequencing/demultiplexed/2_2B_control_S2_L008_R2_001.fastq.gz

/usr/bin/time -v fastqc \
  -o . \
  /projects/bgmp/shared/2017_sequencing/demultiplexed/Undetermined_S0_L008_R1_001.fastq.gz

/usr/bin/time -v fastqc \
  -o . \
  /projects/bgmp/shared/2017_sequencing/demultiplexed/Undetermined_S0_L008_R2_001.fastq.gz
```

Quality score distributions





Quality scores across all bases (Sanger / Illumina 1.9 encoding)
40
38
36
34
32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
1 2 3 4 5 6 7 8 9 12-13 18-19 24-25 30-31 36-37 42-43 48-49 54-55 60-61 66-67 72-73 78-79 84-85 90-91 96-97
Position in read (bp)



Per-base N content

The per-base N content is consistent with the quality score distributions.

Running in-house script for histogram generation

```
/usr/bin/time -v ./histograms.py \
  -f /projects/bgmp/shared/2017_sequencing/demultiplexed/2_2B_control_S2_L008_R1_001.fastq.gz \
  -r 101 -o ../img/2_2B_control_R1_hist.png

/usr/bin/time -v ./histograms.py \
  -f /projects/bgmp/shared/2017_sequencing/demultiplexed/2_2B_control_S2_L008_R2_001.fastq.gz \
  -r 101 -o ../img/2_2B_control_R2_hist.png

/usr/bin/time -v ./histograms.py \
  -f /projects/bgmp/shared/2017_sequencing/demultiplexed/Undetermined_S0_L008_R1_001.fastq.gz \
  -r 101 -o ../img/Undetermined_R1_hist.png

/usr/bin/time -v ./histograms.py \
  -f /projects/bgmp/shared/2017_sequencing/demultiplexed/Undetermined_S0_L008_R2_001.fastq.gz \
  -r 101 -o ../img/Undetermined_R2_hist.png
```

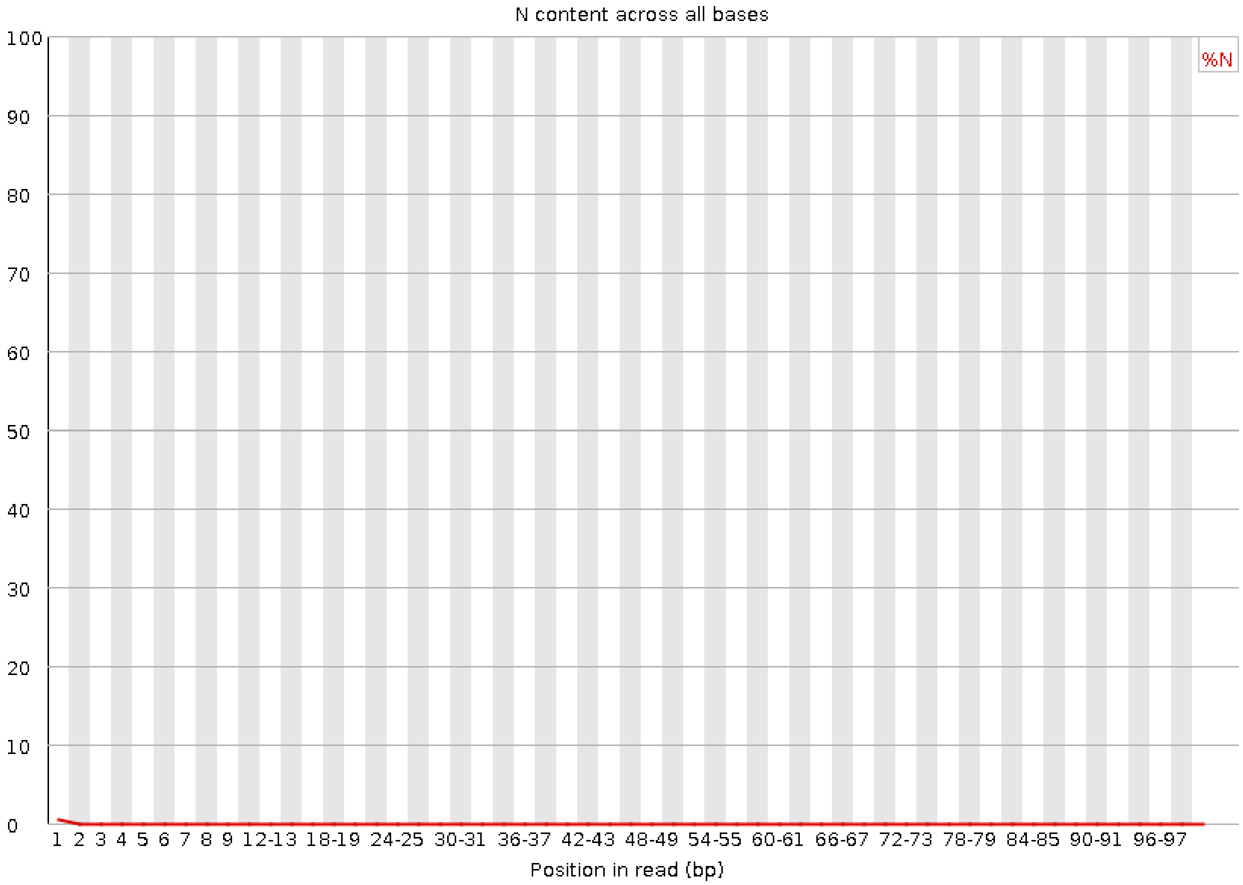


Figure 1: 2_2B_control R1

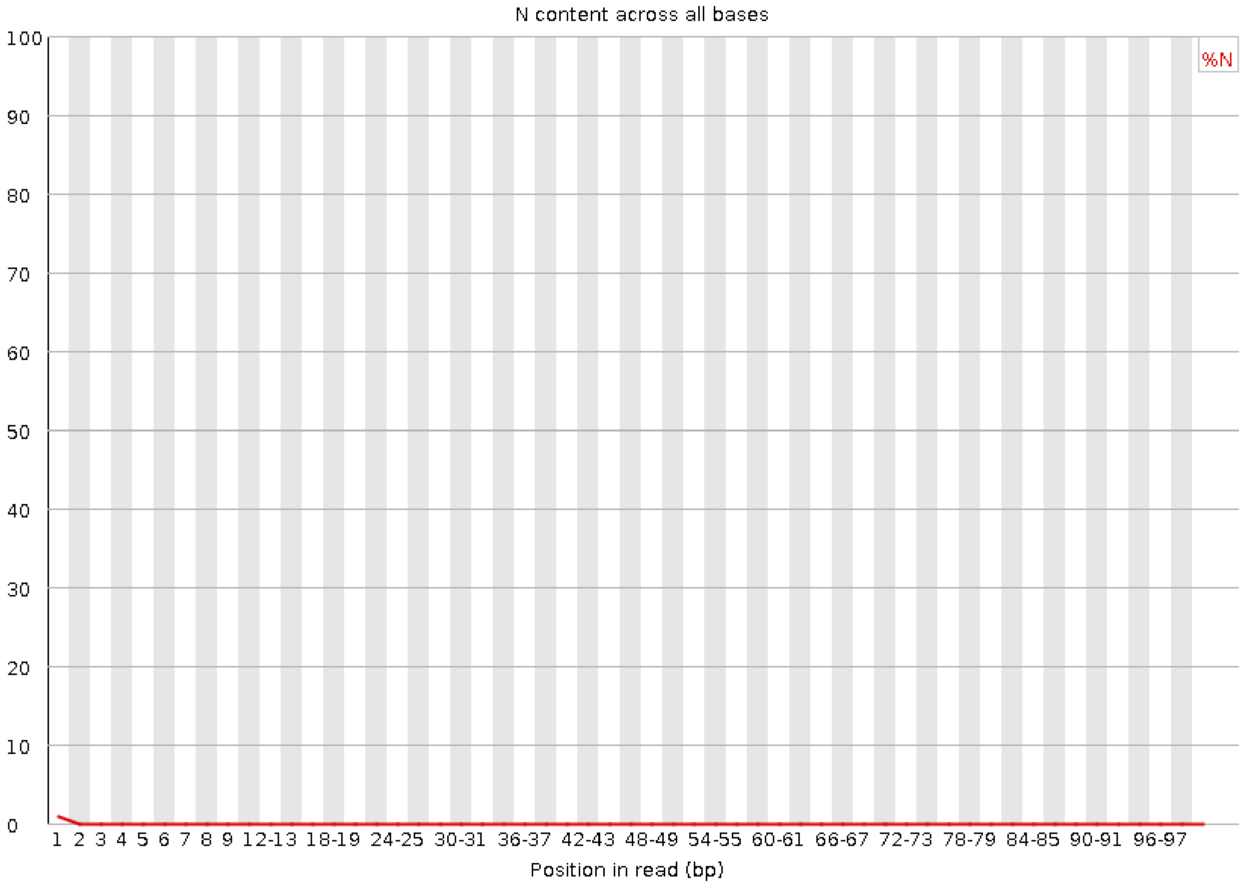


Figure 2: 2_2B_control R2

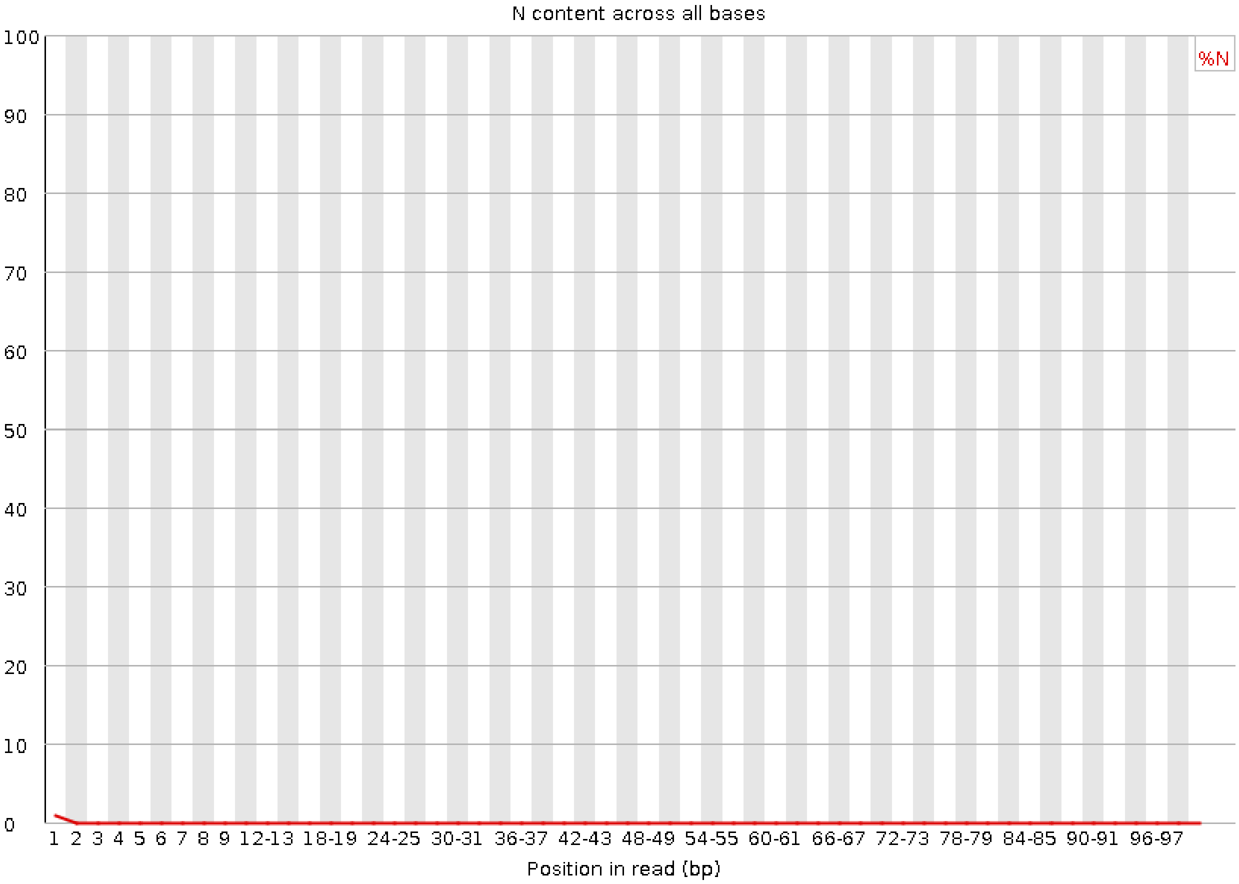


Figure 3: Undetermined R1

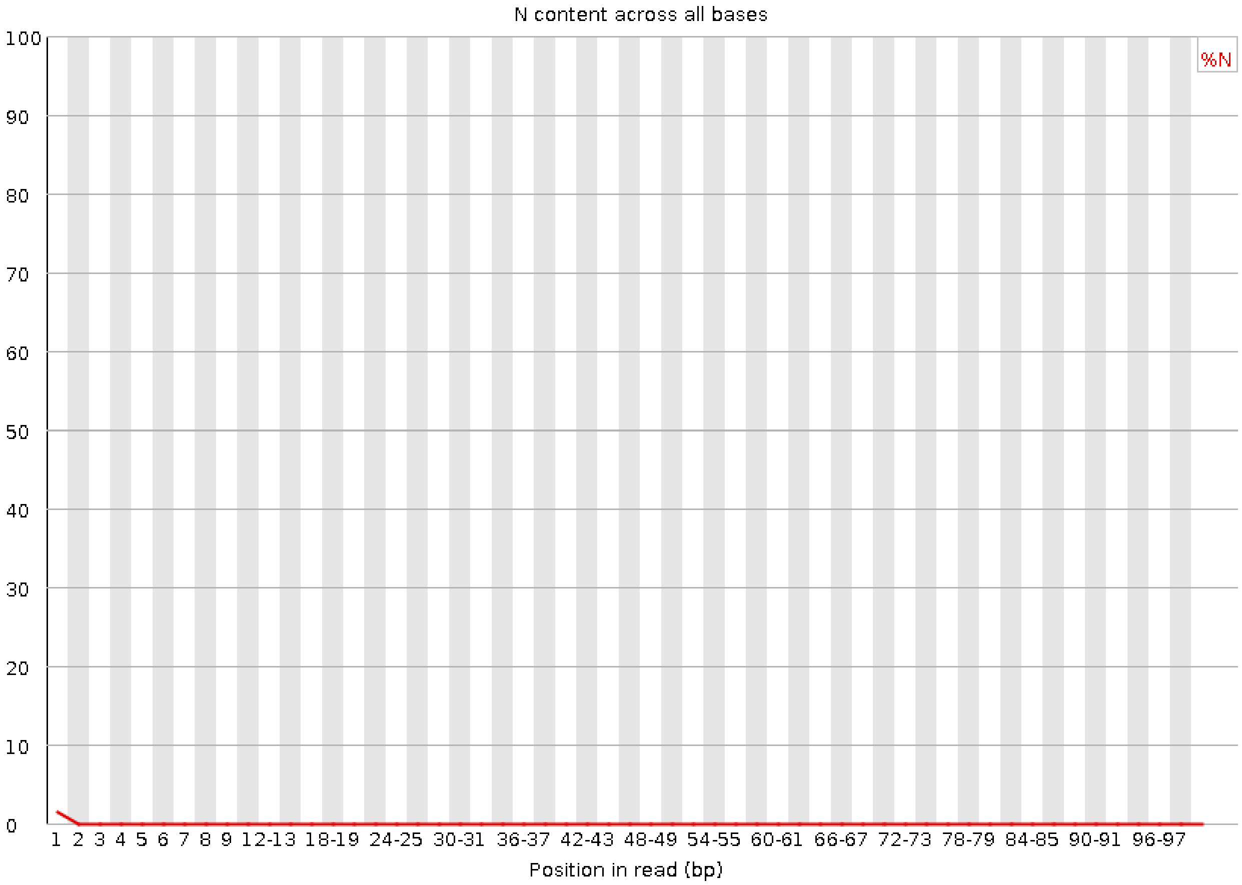


Figure 4: Undetermined R2

Quality Score Plot (using in-house script)

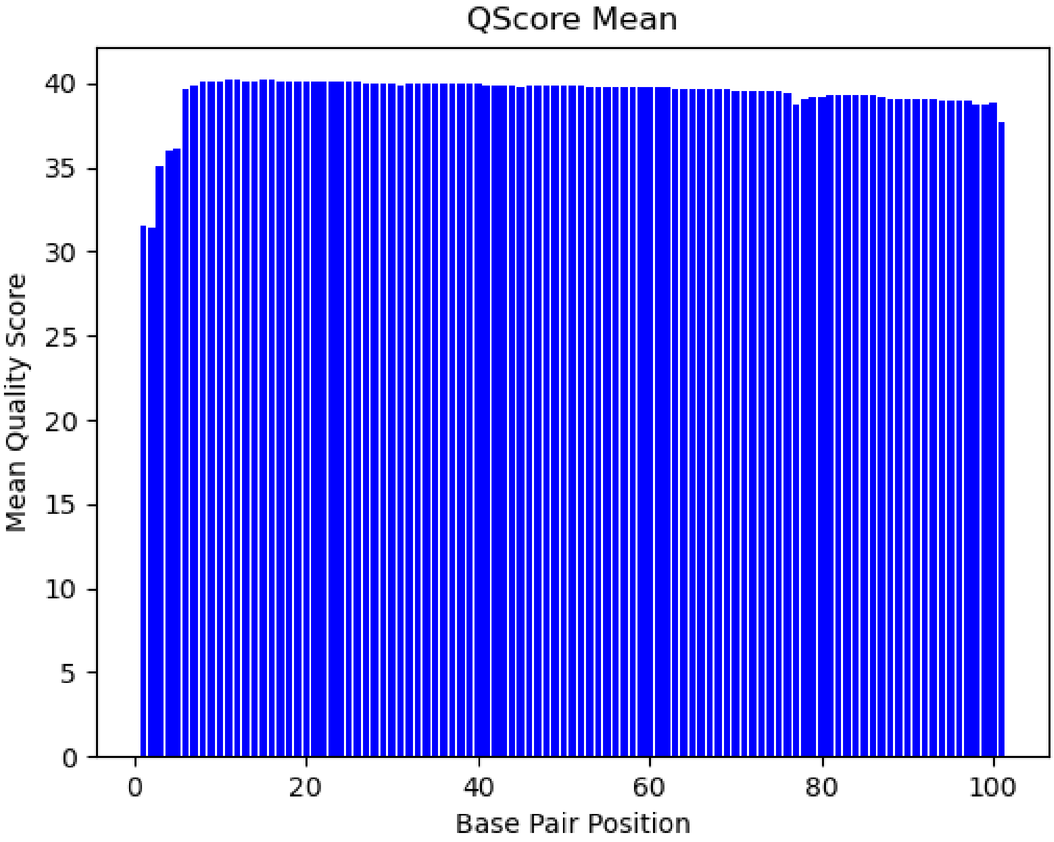


Figure 5: 2_2B_control R1

These quality score distributions generated from the in-house script are very similar to the plots generated by FastQC. The in-house python script ran much slower than FastQC because FastQC is written in java which is a compiled language and is therefore more efficient to run than python (an interpreted language).

The overall data quality of the 2_2B_control library is great, while the quality of the Undetermined library is worse but probably acceptable for most applications. R2 reads of the Undetermined library were quite poor.

Part 2 – Adaptor trimming comparison

Illumina TruSeq Adapter Sequences found here

Running Cutadapt

```
cutadapt -a AGATCGGAAGAGCACACGTCTGAACTCCAGTCA -A AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT \
  -o 2_2B_control.1.fastq.gz \
```

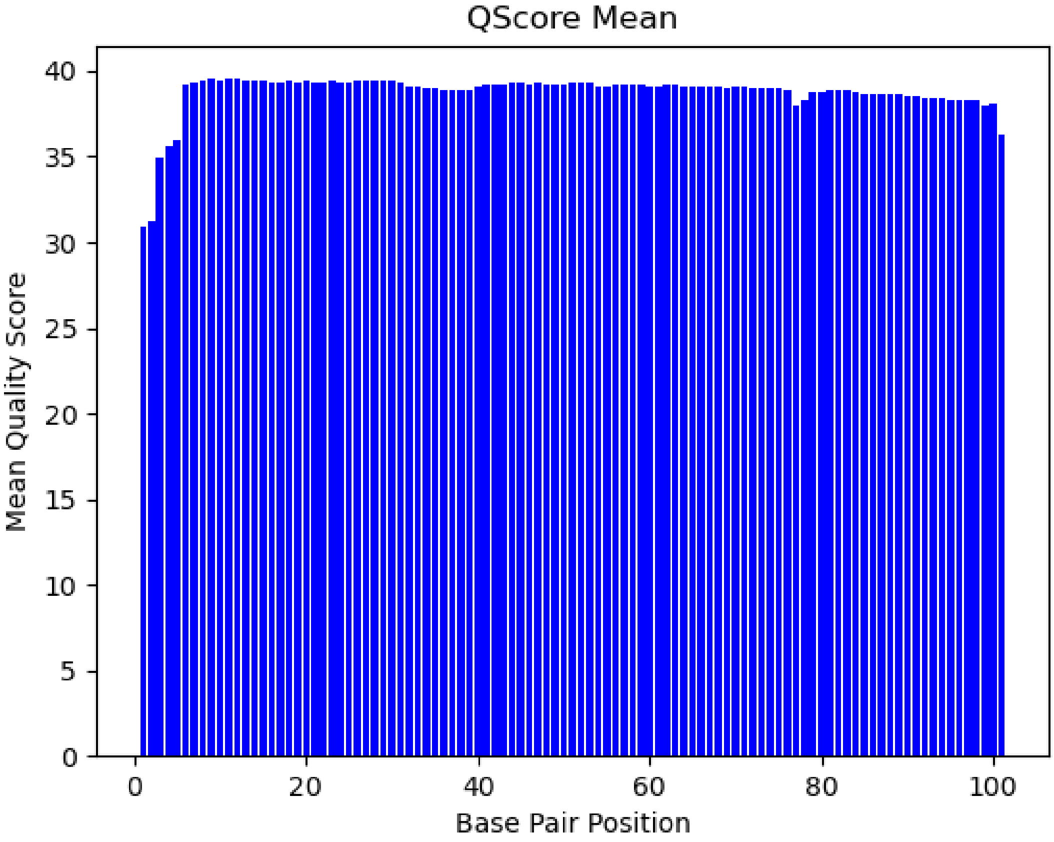


Figure 6: 2_2B_control R2

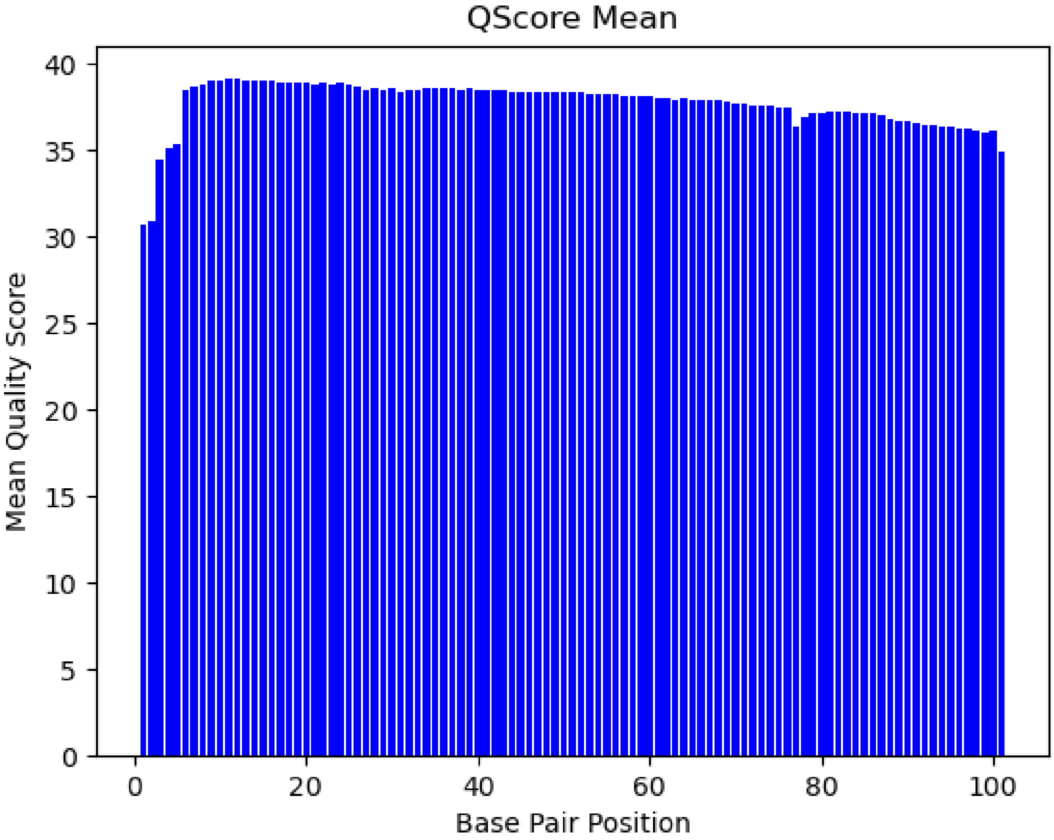


Figure 7: Undetermined R1

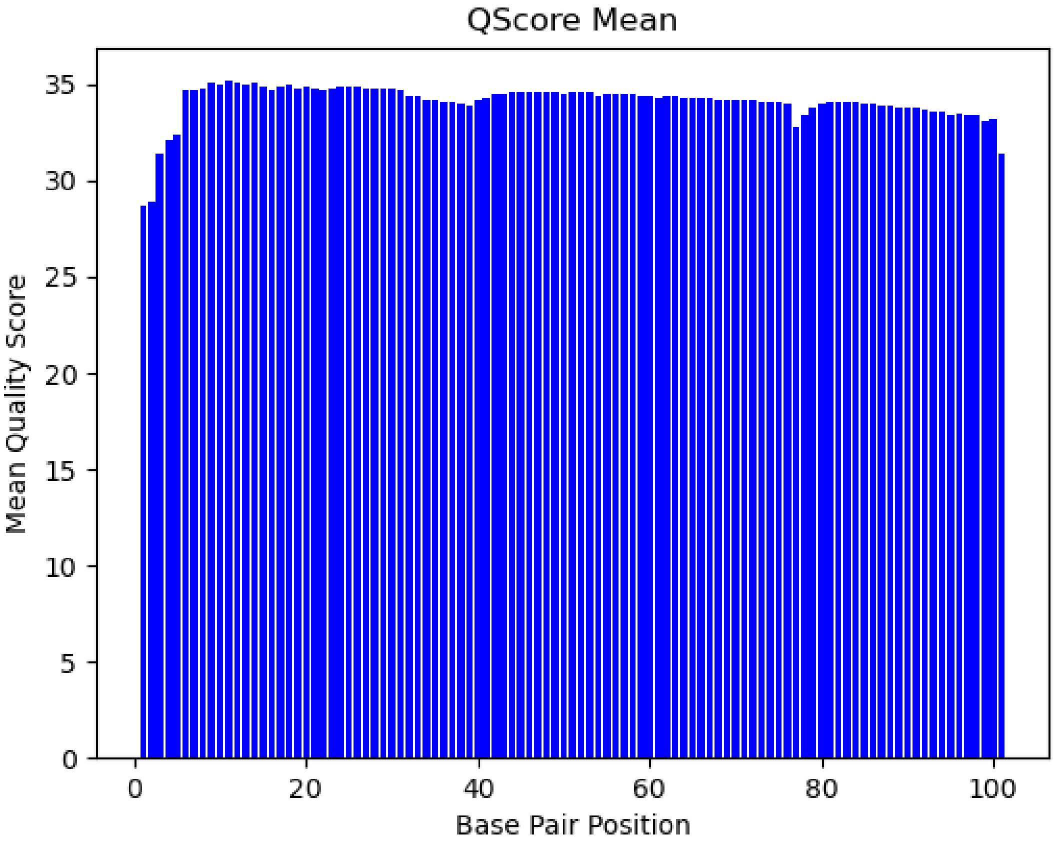


Figure 8: Undetermined R2

```
  -p 2_2B_control.2.fastq.gz \
  /projects/bgmp/shared/2017_sequencing/demultiplexed/2_2B_control_S2_L008_R1_001.fastq.gz \
  /projects/bgmp/shared/2017_sequencing/demultiplexed/2_2B_control_S2_L008_R2_001.fastq.gz

cutadapt -a AGATCGGAAGAGCACACGTCTGAACTCCAGTCA -A AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT \
  -o Undetermined.1.fastq.gz \
  -p Undetermined.2.fastq.gz \
  /projects/bgmp/shared/2017_sequencing/demultiplexed/Undetermined_S0_L008_R1_001.fastq.gz \
  /projects/bgmp/shared/2017_sequencing/demultiplexed/Undetermined_S0_L008_R2_001.fastq.gz
```

Running Trimmomatic

```
trimmomatic PE -threads 4 \
  ../cutadapt/2_2B_control.1.fastq.gz \
  ../cutadapt/2_2B_control.2.fastq.gz \
  2_2B_control.1.trim.fastq.gz 2_2B_control.1.untrim.fastq.gz \
  2_2B_control.2.trim.fastq.gz 2_2B_control.2.untrim.fastq.gz \
  LEADING:3 \
  TRAILING:3 \
  SLIDINGWINDOW:5:15 \
  MINLEN:35

trimmomatic PE -threads 4 \
  ../cutadapt/Undetermined.1.fastq.gz \
  ../cutadapt/Undetermined.2.fastq.gz \
  Undetermined.1.trim.fastq.gz Undetermined.1.untrim.fastq.gz \
  Undetermined.2.trim.fastq.gz Undetermined.2.untrim.fastq.gz \
  LEADING:3 \
  TRAILING:3 \
  SLIDINGWINDOW:5:15 \
  MINLEN:35
```

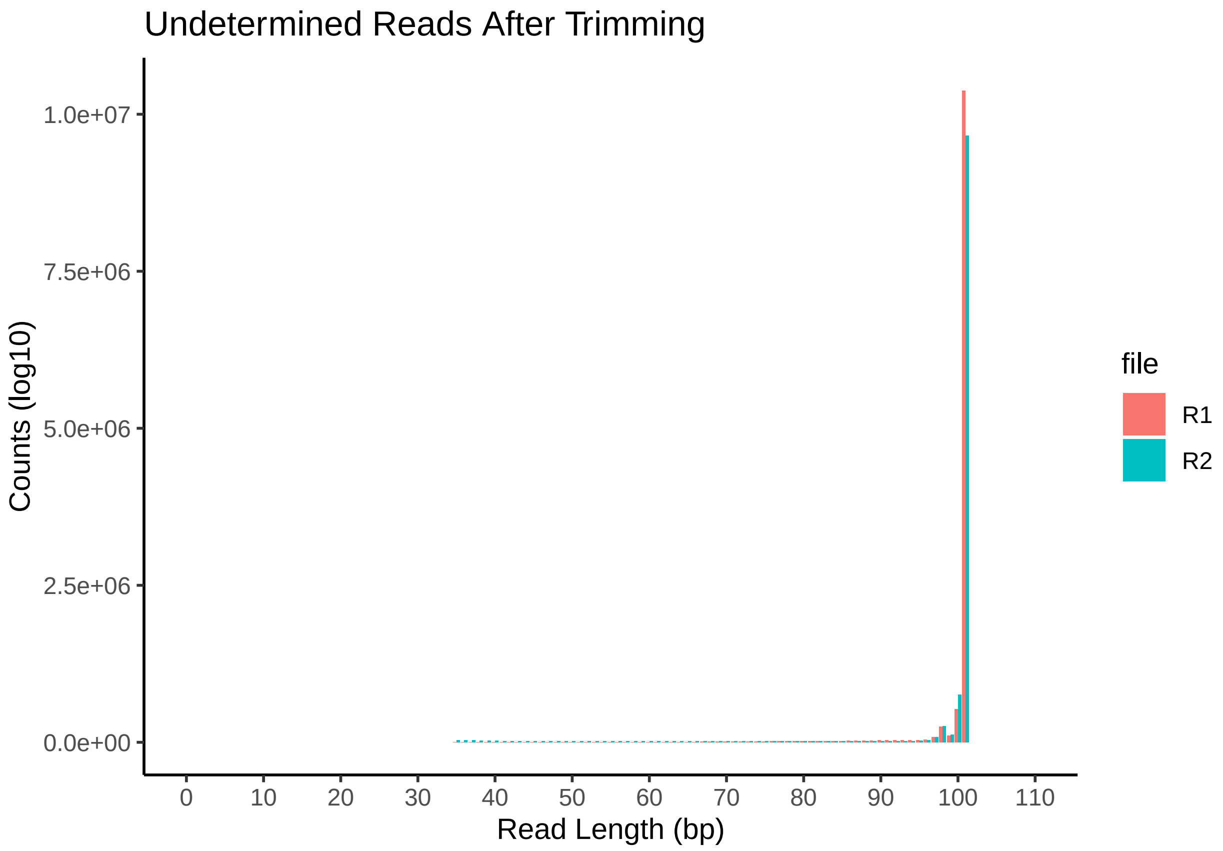
Outputting Distribution Data for R

```
zcat 2_2B_control.1.trim.fastq.gz | awk 'NR % 4 == 2 { print length() }' | sort -n | uniq -c > 2_2B_con

zcat 2_2B_control.2.trim.fastq.gz | awk 'NR % 4 == 2 { print length() }' | sort -n | uniq -c > 2_2B_con

zcat Undetermined.1.trim.fastq.gz | awk 'NR % 4 == 2 { print length() }' | sort -n | uniq -c > Undeterm

zcat Undetermined.2.trim.fastq.gz | awk 'NR % 4 == 2 { print length() }' | sort -n | uniq -c > Undeterm
```

```r
r1 = read.table("data/Undetermined_R1_distr.txt", header = F, sep="")
r2 = read.table("data/Undetermined_R2_distr.txt", header = F, sep="")
r1$file = "R1"
r2$file = "R2"
r1r2 = rbind(r1, r2)
colnames(r1r2)=c("rcount","rlength","file")
head(r1r2)
```

```
##   rcount rlength file
## 1   2617      35   R1
## 2   2799      36   R1
## 3   2979      37   R1
## 4   3086      38   R1
## 5   3253      39   R1
## 6   3223      40   R1
```

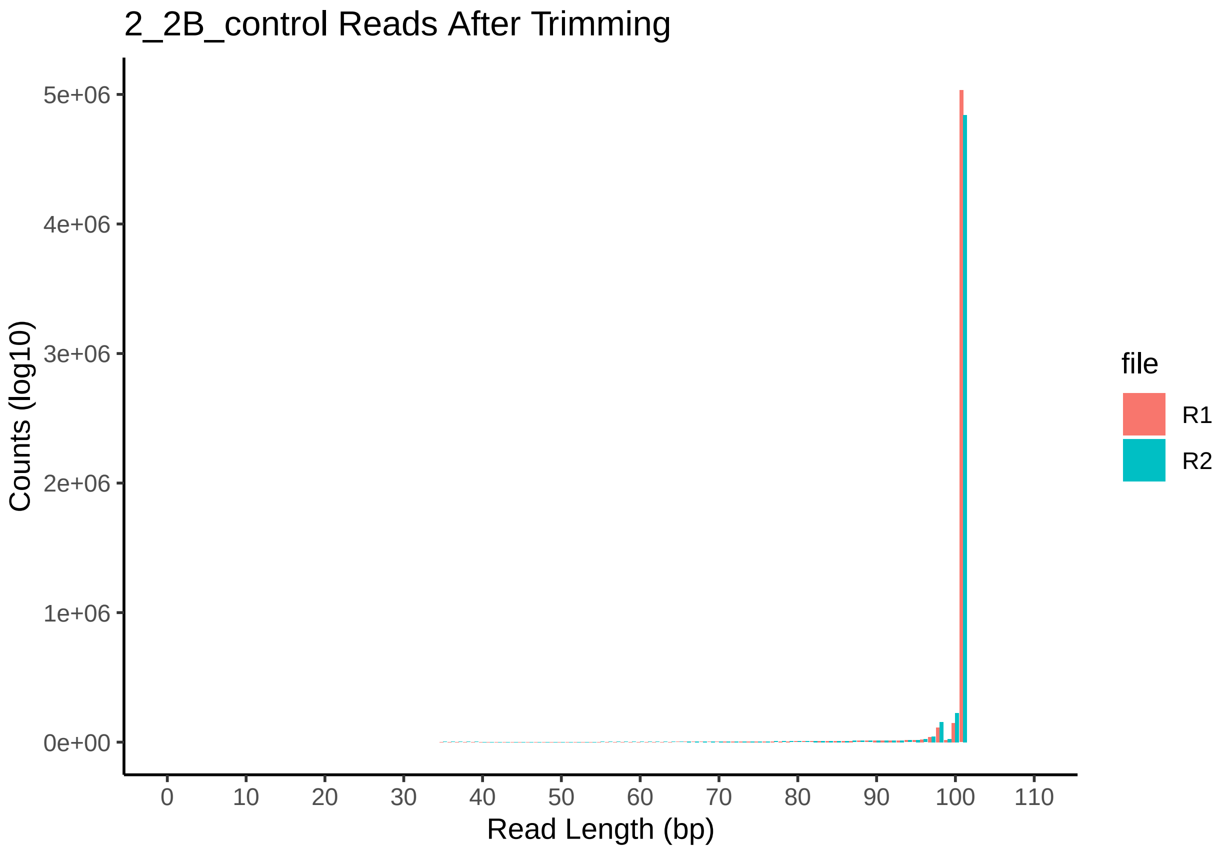
```r
ggplot(r1r2, aes(x=rlength, y=rcount, fill=file)) +
  geom_bar(stat="identity", position=position_dodge()) +
  theme_classic() +
  scale_x_continuous(breaks = seq(0, 110, by=10), limits=c(0,110)) +
  labs(title="Undetermined Reads After Trimming",
       x="Read Length (bp)", y="Counts (log10)", color="Read File")
```



```r
r1 = read.table("data/2_2B_control_R1_distr.txt", header = F, sep="")
r2 = read.table("data/2_2B_control_R2_distr.txt", header = F, sep="")
r1$file = "R1"
r2$file = "R2"
r1r2 = rbind(r1, r2)
colnames(r1r2)=c("rcount","rlength","file")
head(r1r2)
```

```
##   rcount rlength file
## 1    723      35   R1
## 2    826      36   R1
## 3    890      37   R1
## 4    942      38   R1
## 5   1010      39   R1
## 6   1148      40   R1
```

```
ggplot(r1r2, aes(x=rlength, y=rcount, fill=file)) +
  geom_bar(stat="identity", position=position_dodge()) +
  theme_classic() +
  scale_x_continuous(breaks = seq(0, 110, by=10), limits=c(0,110)) +
  labs(title="2_2B_control Reads After Trimming",
       x="Read Length (bp)", y="Counts (log10)", color="Read File")
```



R2 reads should be of lower quality than R1 reads because they are sequenced after many cycles have already been completed on the sequencer, and after there has been another round of bridge amplification. This is reflected in the distribution of read lengths after trimming, above.

Part 3 – Alignment and strand-specificity

SLURM: STAR alignment & HTSeq

```
#!/bin/bash
#SBATCH --partition=bgmp            ### Partition (like a queue in PBS)
#SBATCH --job-name=qaa              ### Job Name
#SBATCH --output=star-%j.log        ### File in which to store job output
#SBATCH --error=star-%j.err         ### File in which to store job error messages
```

```
#SBATCH --time=0-20:00:00           ### Wall clock time limit in Days-HH:MM:SS
#SBATCH --nodes=1                   ### Number of nodes needed for the job
#SBATCH --cpus-per-task=8           ### Number of CPUs to be used per task
#SBATCH --account=bgmp              ### Account used for job submission
#SBATCH --mail-user=acrabtre@uoregon.edu    ### email for job submission notifications
#SBATCH --mail-type=ALL             ### specifies types of notification emails to send

## load conda environment
conda activate qaa

## assign variables
d=/projects/bgmp/acrabtre/bioinfo/Bi623/QAA/star
f_read=/projects/bgmp/acrabtre/bioinfo/Bi623/QAA/trim/Undetermined.1.trim.fastq.gz
r_read=/projects/bgmp/acrabtre/bioinfo/Bi623/QAA/trim/Undetermined.2.trim.fastq.gz
prfx="Undetermined_Mmus_"
samfile=$d/${prfx}Aligned.out.sam

cd $d

## run STAR to generate genome indexes
/usr/bin/time -v STAR \
    --runThreadN 8 \
    --runMode genomeGenerate \
    --genomeDir $d/mmus \
    --genomeFastaFiles $d/mmus/Mus_musculus.GRCm39.dna.primary_assembly.fa \
    --sjdbGTFfile $d/mmus/Mus_musculus.GRCm39.104.gtf

## run STAR to assemble reads to reference genome
/usr/bin/time -v STAR --runThreadN 8 --runMode alignReads \
    --outFilterMultimapNmax 3 \
    --outSAMunmapped Within KeepPairs \
    --alignIntronMax 1000000 --alignMatesGapMax 1000000 \
    --readFilesCommand zcat \
    --readFilesIn $f_read $r_read \
    --genomeDir $d/mmus \
    --outFileNamePrefix $d/$prfx

## run python script to get counts of mapped and unmapped reads
python /projects/bgmp/acrabtre/bioinfo/Bi623/QAA/scripts/inspector_sam.py $samfile > $d/${prfx}map_count

## run HTseq to count reads that map to features
htseq-count --stranded=no $samfile $d/mmus/Mus_musculus.GRCm39.104.gtf > $d/${prfx}hts.genecounts
htseq-count --stranded=yes $samfile $d/mmus/Mus_musculus.GRCm39.104.gtf > $d/${prfx}hts_str.genecounts

######################## RUN STAR ON SAMPLE 2 ########################

## assign variables
d=/projects/bgmp/acrabtre/bioinfo/Bi623/QAA/star
f_read=/projects/bgmp/acrabtre/bioinfo/Bi623/QAA/trim/2_2B_control.1.trim.fastq.gz
r_read=/projects/bgmp/acrabtre/bioinfo/Bi623/QAA/trim/2_2B_control.2.trim.fastq.gz
prfx="2_2B_control_Mmus_"
samfile=$d/${prfx}Aligned.out.sam

## run STAR to assemble reads to reference genome
```

```
/usr/bin/time -v STAR --runThreadN 8 --runMode alignReads \
    --outFilterMultimapNmax 3 \
    --outSAMunmapped Within KeepPairs \
    --alignIntronMax 1000000 --alignMatesGapMax 1000000 \
    --readFilesCommand zcat \
    --readFilesIn $f_read $r_read \
    --genomeDir $d/mmus \
    --outFileNamePrefix $d/$prfx

## run python script to get counts of mapped and unmapped reads
python /projects/bgmp/acrabtre/bioinfo/Bi623/QAA/scripts/inspector_sam.py $samfile > $d/${prfx}map_coun

## run HTseq to count reads that map to features
htseq-count --stranded=no $samfile $d/mmus/Mus_musculus.GRCm39.104.gtf > $d/${prfx}hts.genecounts
htseq-count --stranded=yes $samfile $d/mmus/Mus_musculus.GRCm39.104.gtf > $d/${prfx}hts_str.genecounts
```

Read counts for each genecounts file (use to determine if library was stranded):

```
$ cat 2_2B_control_hts_counts.txt | grep -v "^__" | awk '{sum+=$2} END {print sum}'
4702902

$ cat 2_2B_control_Mmus_hts_counts_str.txt | grep -v "^__" | awk '{sum+=$2} END {print sum}'
216262

$ cat Undetermined_Mmus_hts_counts.txt | grep -v "^__" | awk '{sum+=$2} END {print sum}'
6414111

$ cat Undetermined_Mmus_hts_counts_str.txt | grep -v "^__" | awk '{sum+=$2} END {print sum}'
299593
```

Total number of reads respectively for each sample file:

```
$ cat 2_2B_control_hts.genecounts | awk '{sum+=$2} END {print sum}'
5652541

$ cat Undetermined_Mmus_hts_counts.txt | awk '{sum+=$2} END {print sum}'
12160073
```

htseq-count –stranded=yes

- 2_2B_control: htseq mapped ~3.8% of total reads to reference genome
- Undetermined: htseq mapped ~2.5% of total reads to reference genome

htseq-count –stranded=no

- 2_2B_control: htseq mapped ~83% of total reads to reference genome
- Undetermined: htseq mapped ~52% of total reads to reference genome

> I propose this library is not stranded because the number of mapped reads (in gene counts files) after running HTSeq are extremely low. For example, in the "Undetermined" sample, only 2.5% of the reads mapped to the reference genome when htseq assumed the library was stranded, whereas 52% of the reads were mapped when the library was considered non-stranded.

2_2B_control

- mapped reads: 11078823 (98%)
- umapped reads: 226259 (2%)

Undetermined

- mapped reads: 15584518 (64%)
- umapped reads: 8735628 (36%)

Additionally, the number of mapped and unmapped reads were calculated from the SAM file for both libraries (above). The proportion of mapped read counts from htseq-count non-stranded option more closely refected the proportion of mapped read counts calculated directly from the SAM file.