QAA

James Adler

9/8/2021

Part 1 - Read quality score distributions

Files - referred to as ‘Fox’ and ‘Control’ files throughout report

Fox Files

- /projects/bgmp/shared/2017_sequencing/demultiplexed/31_4F_fox_S22_L008_R1_001.fastq.gz
- /projects/bgmp/shared/2017_sequencing/demultiplexed/31_4F_fox_S22_L008_R2_001.fastq.gz

Control Files

- /projects/bgmp/shared/2017_sequencing/demultiplexed/23_4A_control_S17_L008_R1_001.fastq.gz
- /projects/bgmp/shared/2017_sequencing/demultiplexed/23_4A_control_S17_L008_R2_001.fastq.gz

1. Fastqc run

Fox Read 1 Results:

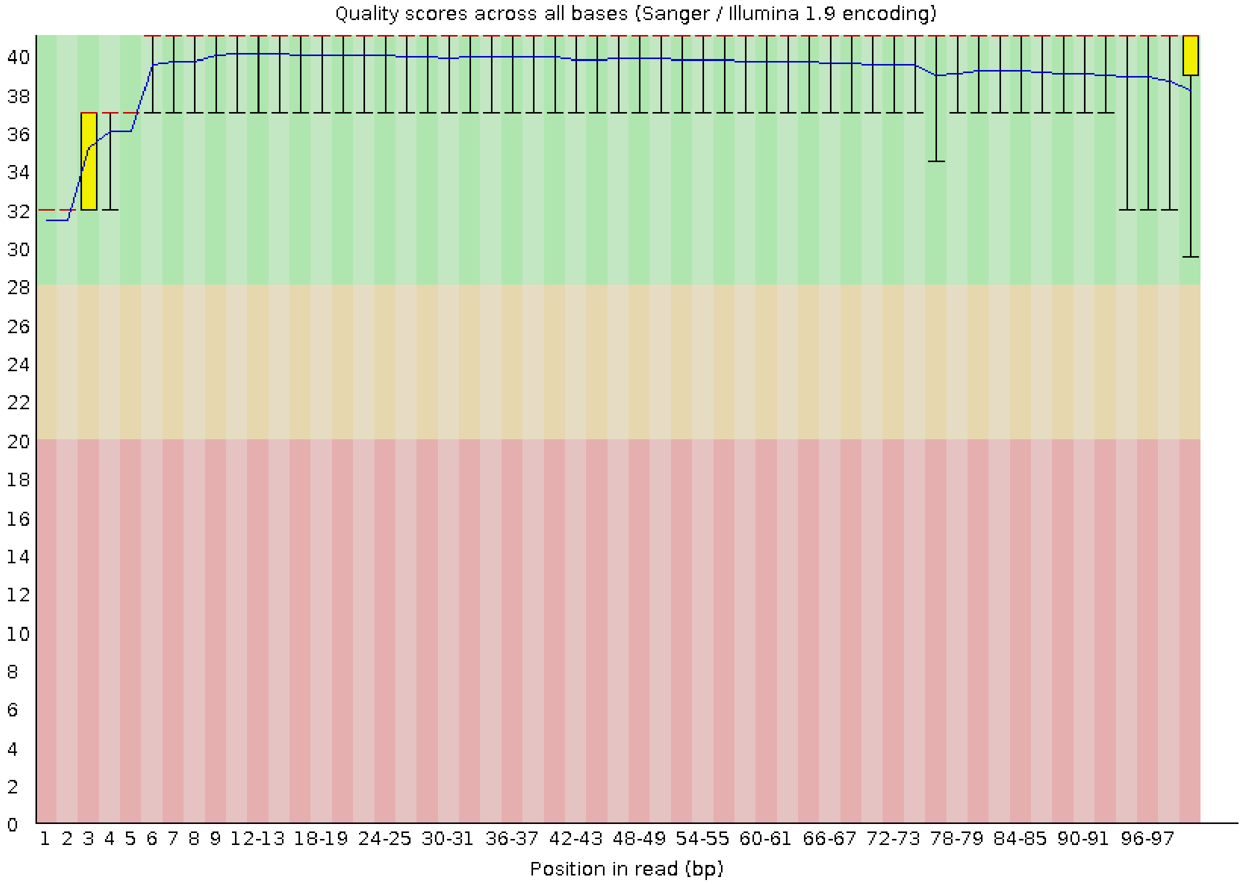

Figure 1: Per-base quality content ‘31_4F_fox_S22_L008_R1_001’

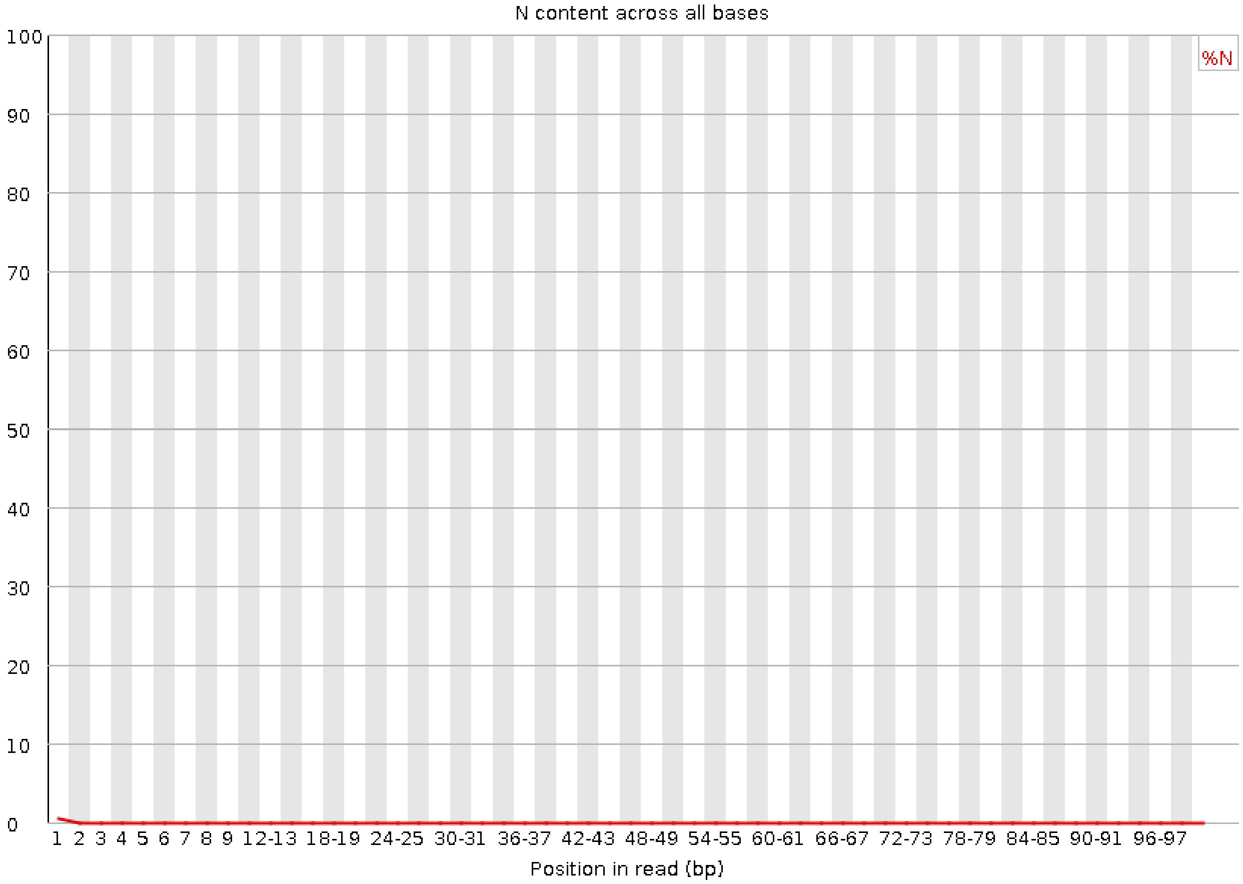

Figure 2: Per-base N content ‘31_4F_fox_S22_L008_R1_001’

Fox Read 2 Results:

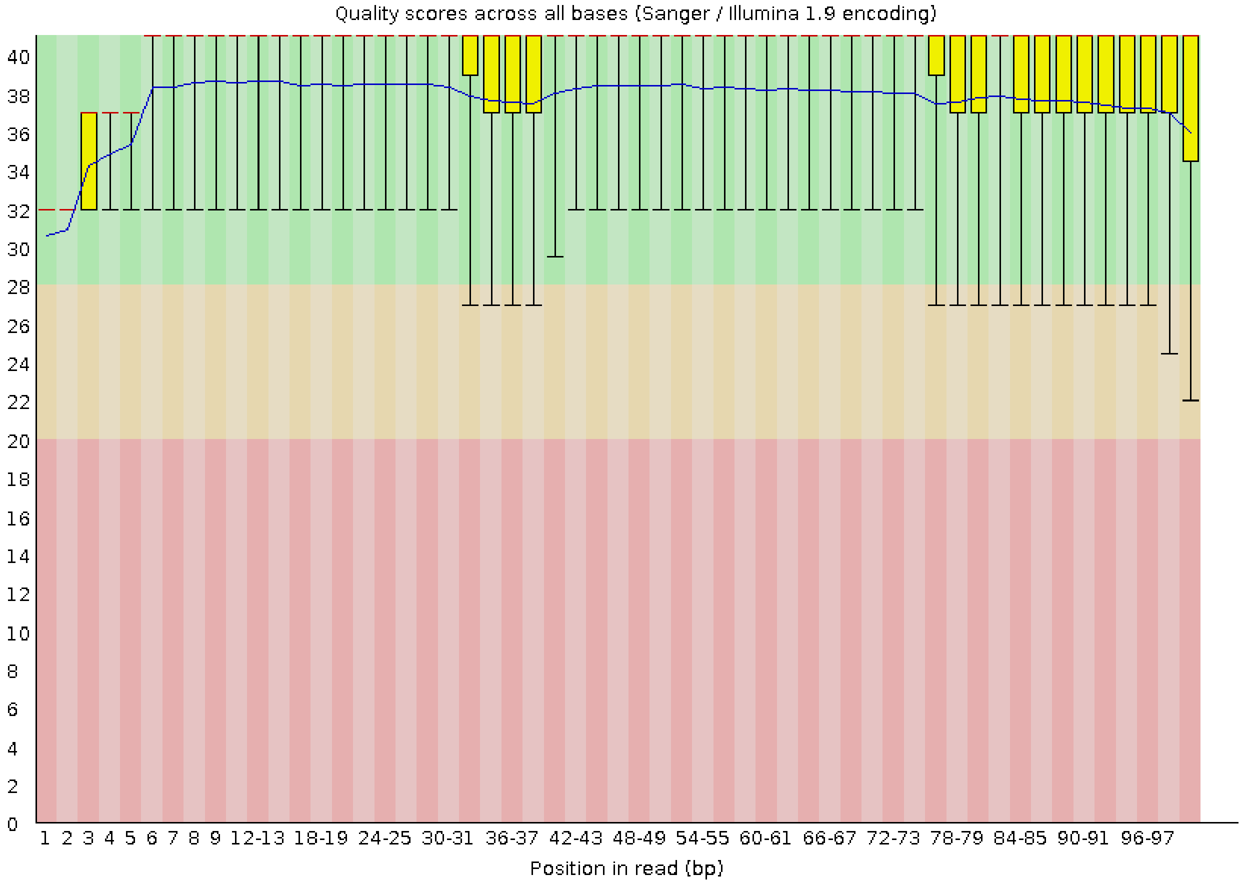

Figure 3: Per-base quality content '31_4F_fox_S22_L008_R2_001'

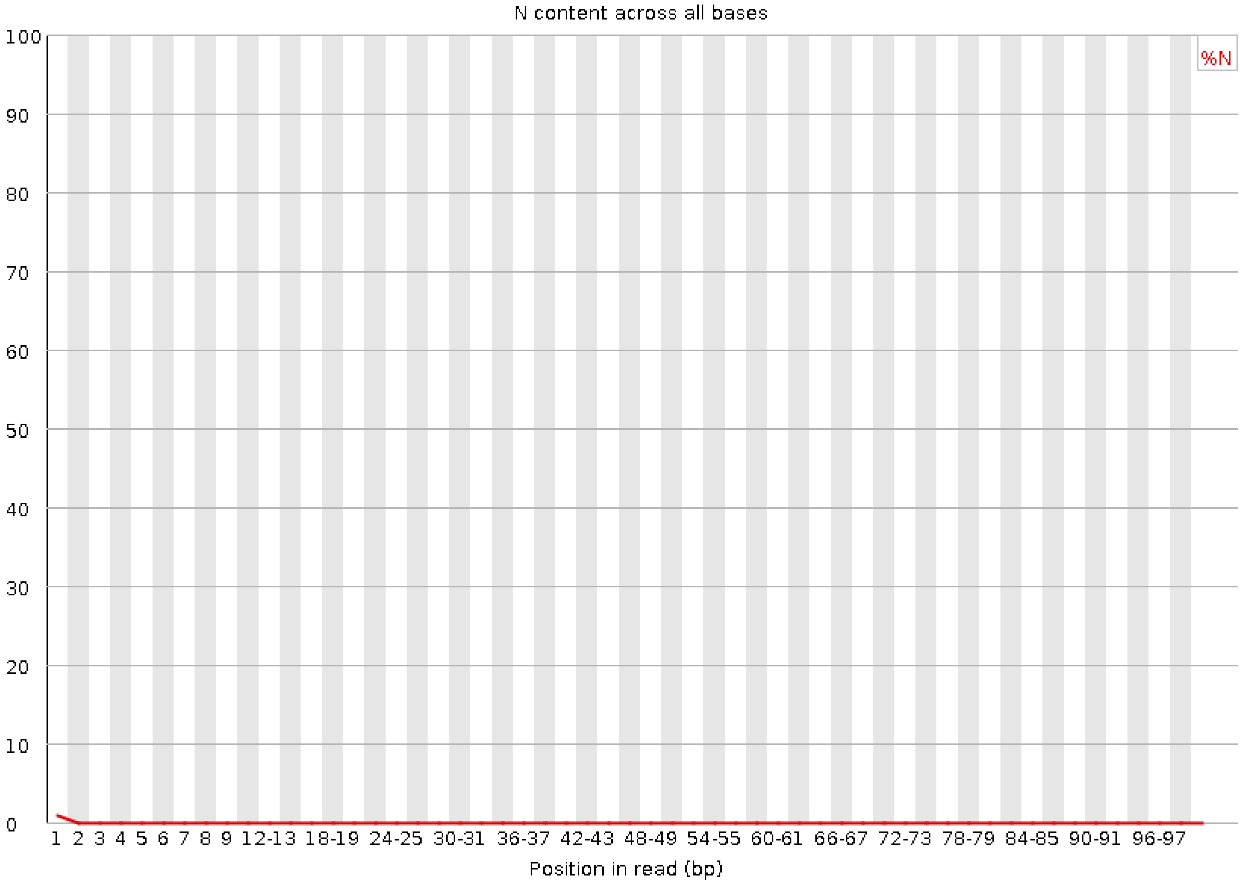

Figure 4: Per-base N content ‘31_4F_fox_S22_L008_R2_001’

Control Read 1 Results:

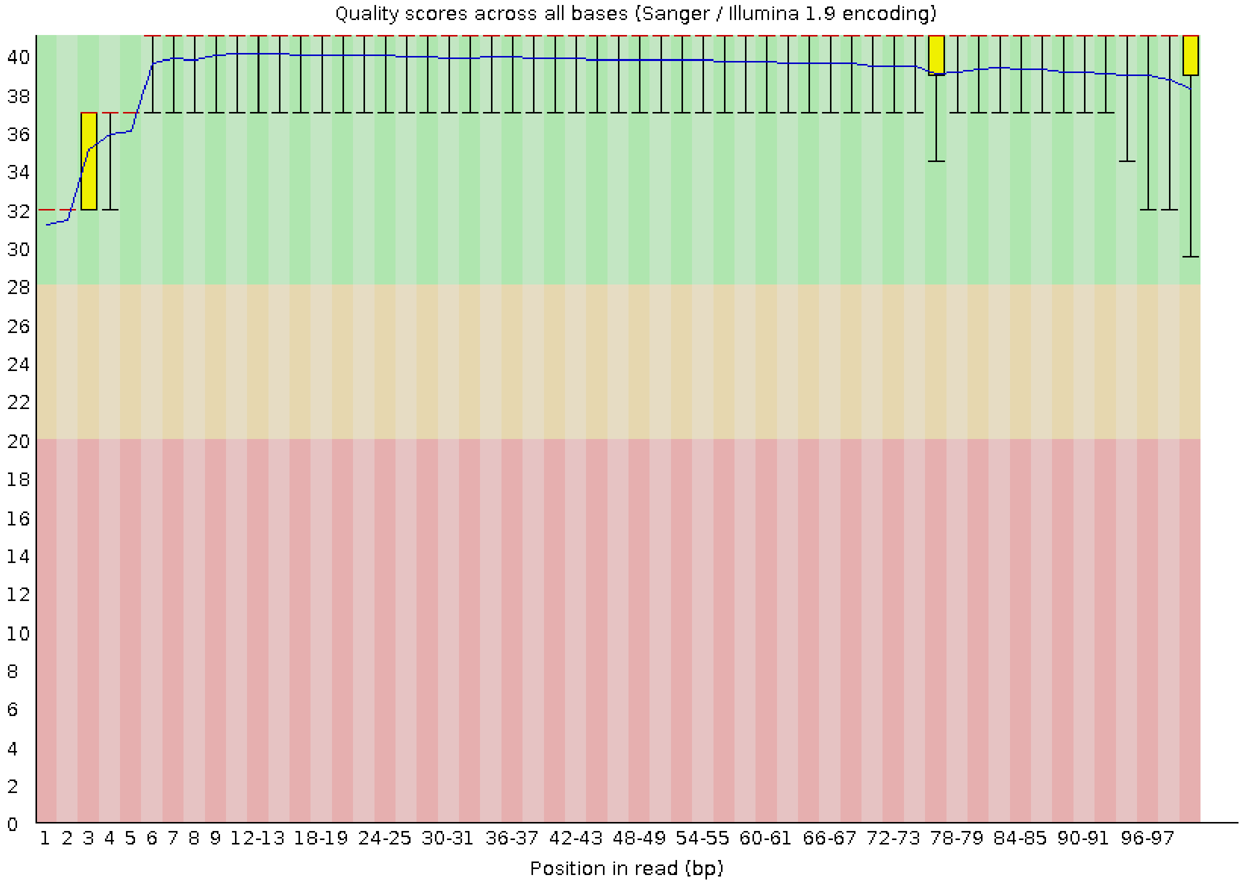

Figure 5: Per-base quality content ‘23_4A_control_S17_L008_R1_001’

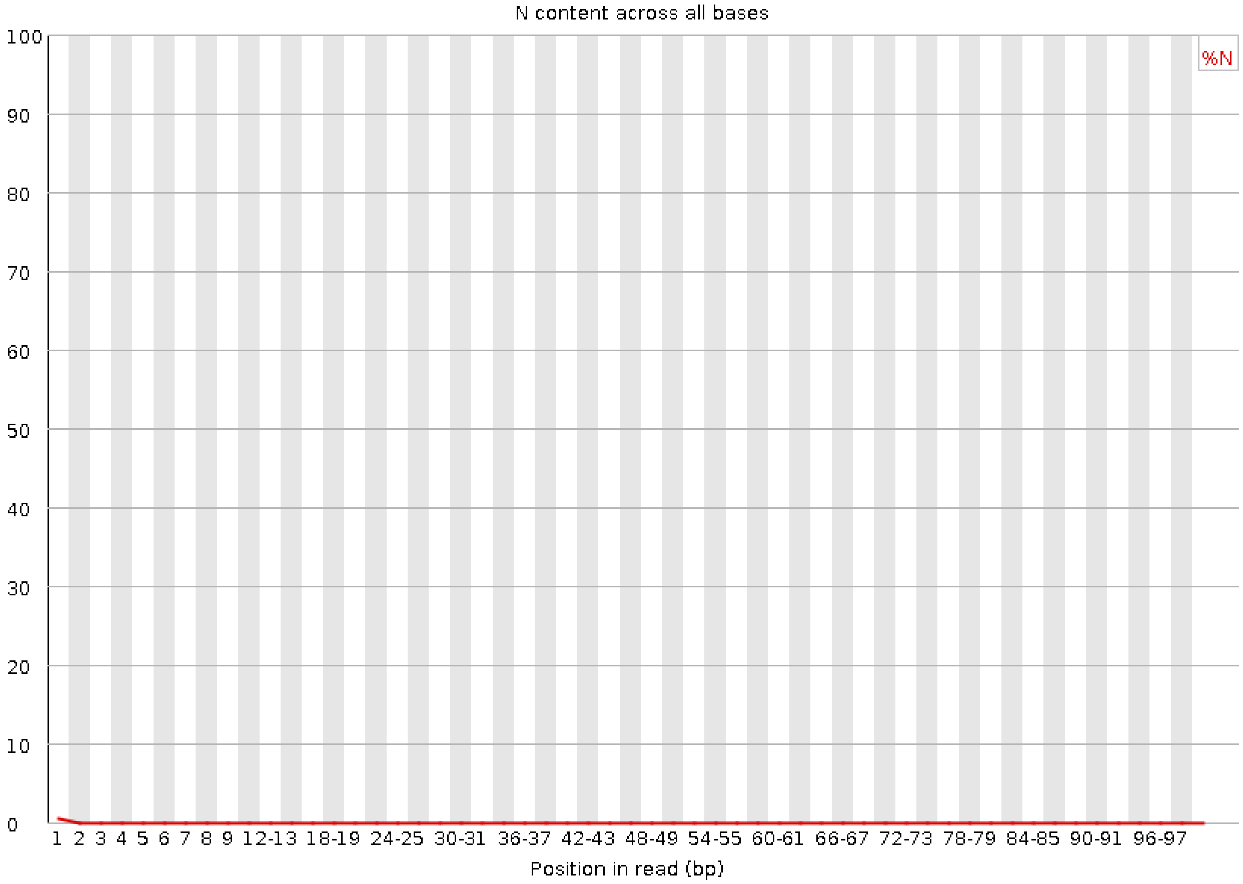

Figure 6: Per-base N content ‘23_4A_control_S17_L008_R1_001’

Control Read 2 Results:

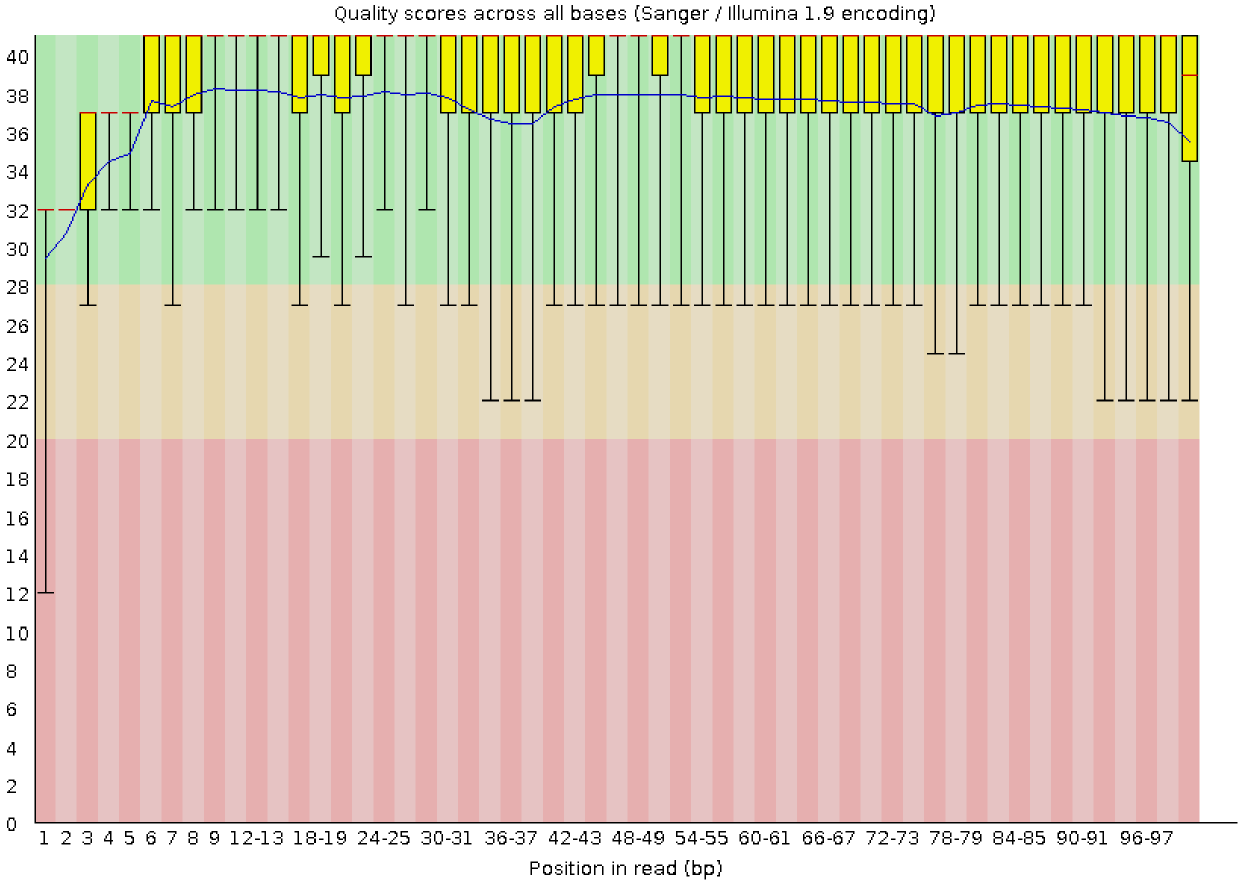

Figure 7: Per-base quality content '23_4A_control_S17_L008_R2_001'

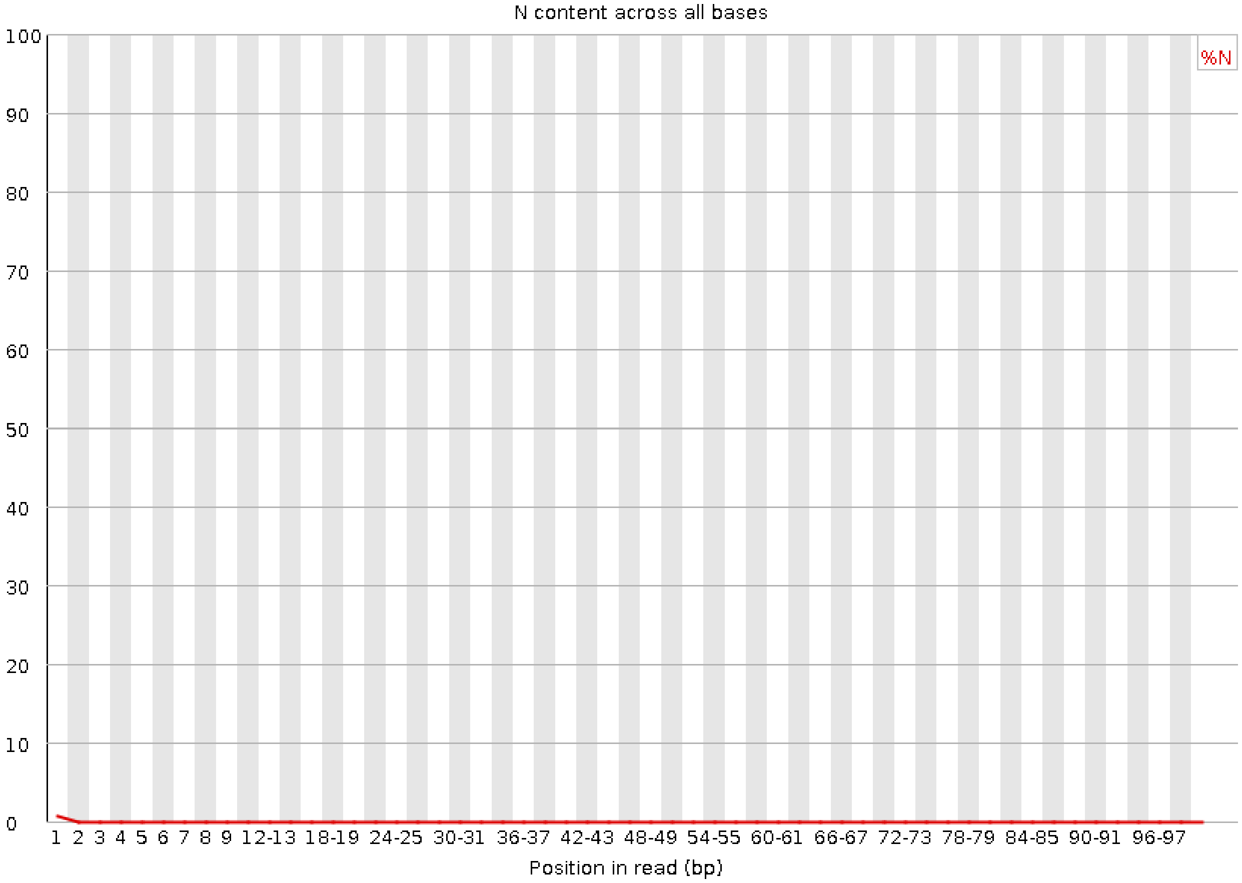

Figure 8: Per-base N content '23_4A_control_S17_L008_R2_001'

The per-base N quality graphs are consistent with the per-base quality graphs in that base 1 has higher N content relative to the other bases. This is in alignment with the lower per-base quality score relative to the other bases.

2. Generate histograms with personal python script

Fox Read 1:

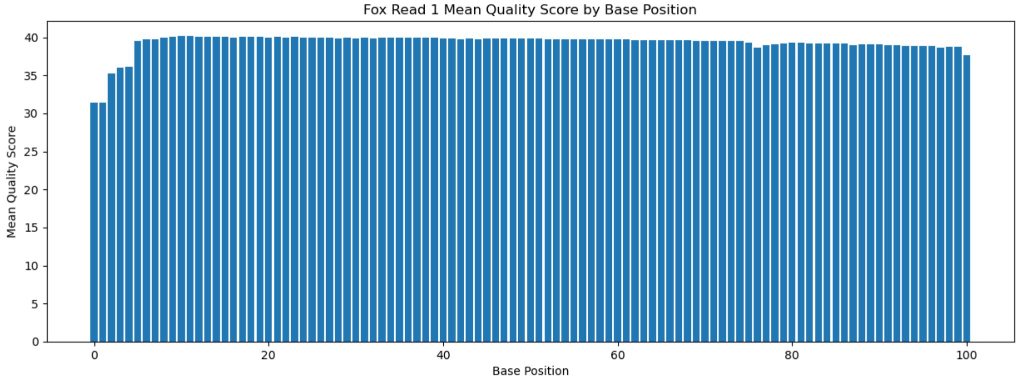

Figure 9: Distribution of lengths for untrimmed reads '31_4F_fox_S22_L008_R1_001'

Fox Read 2:

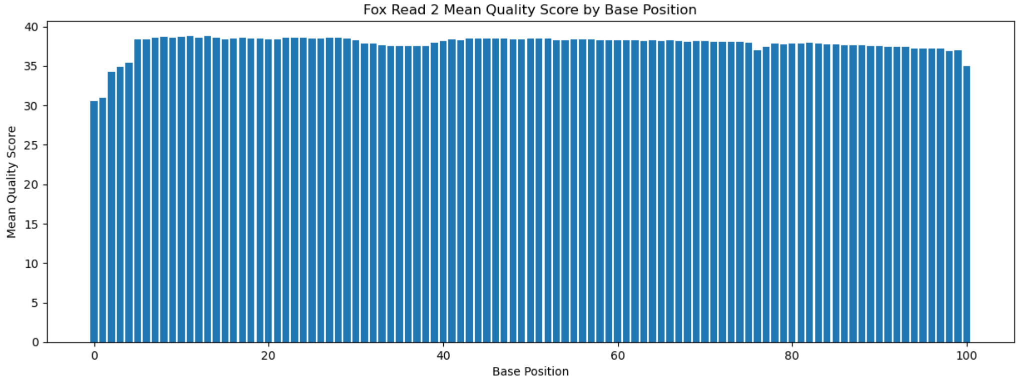

Figure 10: Distribution of lengths for untrimmed reads '31_4F_fox_S22_L008_R2_001'

Control Read 1:

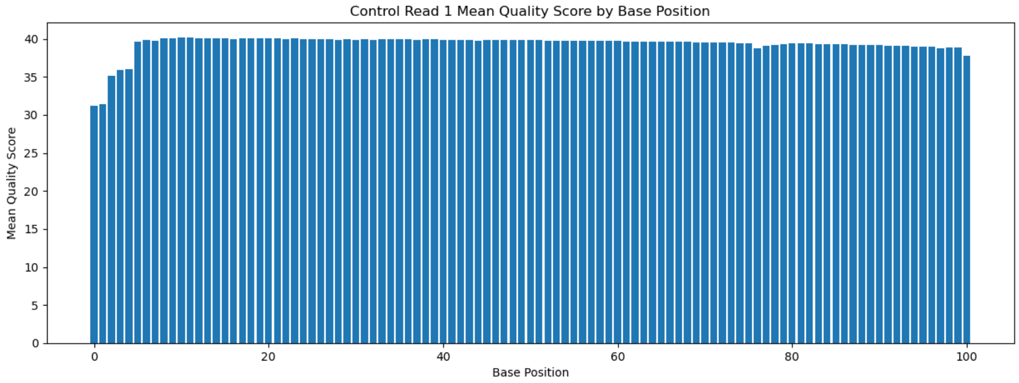

Figure 11: Distribution of lengths for untrimmed reads '23_4A_control_S17_L008_R1_001'

Control Read 2:

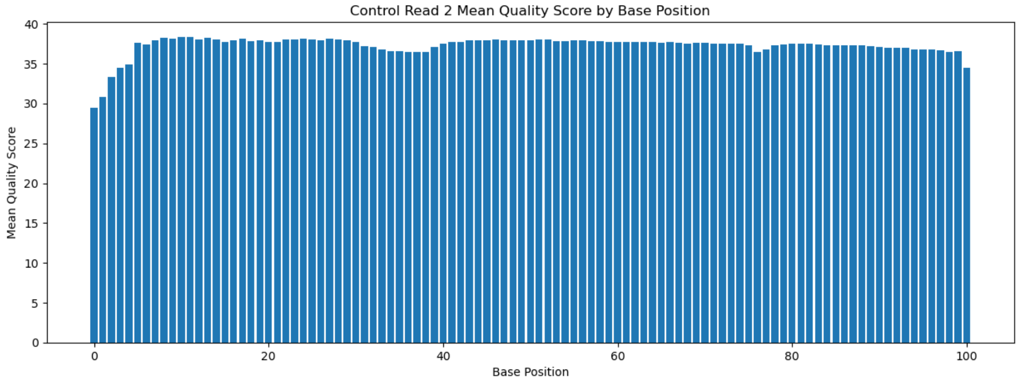

Figure 12: Distribution of lengths for untrimmed reads '23_4A_control_S17_L008_R2_001'

Fastqc vs personal python script: Yes, the output and runtimes do differ. While the shape of the mean quality score at each base appears to be identical between the two plots, the fastqc graphs include box and whiskers, providing additional information on the range of quality scores at each base. The red lines in these plots indicate the median at each base, and the upper and lower ends of the whiskers indicate the 10% and 90% points at each base. The fastqc charts also include green (28-41), yellow (20-28), and red (0-20) areas that correspond to quality level. This is helpful in providing the viewer with a quick reference of quality level at each base.

Fox files: The histogram generator (personal python script) runs for the fox files each took about 3 minutes 45 seconds and only produced the one histogram. The fastqc runs can be run with multiple CPUs using the -t flag. Running the two files with -t 8 results in total runtime of 29 seconds and produces substantially more information relative to our histogram generator script.

Control files: The histogram generator (personal python script) runs for the control files each took about 40 minutes. Running fastqc on the two controls files with 8 CPUs results in total runtime of 3 minutes 45 seconds and produces substantially more information relative to our histogram generator script.

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

Fox files: The quality of each of the read files is high. Neither was flagged for poor quality. The mean score is greater than 28 (green zone) for each of the read files. The `31_4F_fox_S22_L008_R2_001.fastq.gz` file does have some whiskers that drop down below quality score of 28 (yellow zone), indicating that there are values at the corresponding bases that fall below the quality level of 28. These drops in quality primarily occur between bases 30-37 and 78-end-of-read, but the mean and majority of quality scores at each of the bases fall between 34-41.

Control files: The quality for each of the read files is high, though R2 does have many whiskers extending below 28 (yellow zone) and one whisker, at base 1, that extends below 20 (red zone), indicating that there is a portion of the lowest 10% of quality scores at these corresponding bases that fall below these levels. The mean quality score across all bases are all above 28 (green zone), though the 1st base mean quality score is lower at around 30, lower than the rest of the mean quality scores in this read and the lowest value that we seen among all four files.

Part 2 - Adaptor trimming comparison

5. Adapter trim with cutadapt

What proportion of reads were trimmed?

Fox File Adapter Trim Stats:

Total read pairs processed: 3,788,343

Read 1 with adapter: 456,168 (12.0%)

Read 2 with adapter: 482,503 (12.7%)

Control File Adapter Trim Stats:

Total read pairs processed: 44,303,262

Read 1 with adapter: 1,359,563 (3.1%)

Read 2 with adapter: 1,657,295 (3.7%)

Confirm adapter sequences were removed:

```
zcat trimmed_31_4F_fox_S22_L008_R1_001.fastq.gz | grep "AGATCGGAAGAGCACACGTCTGAACTCCAGTCA"

zcat trimmed_31_4F_fox_S22_L008_R2_001.fastq.gz | grep "AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT"

zcat trimmed_23_4A_control_S17_L008_R1_001.fastq.gz | grep "AGATCGGAAGAGCACACGTCTGAACTCCAGTCA"
zcat trimmed_23_4A_control_S17_L008_R2_001.fastq.gz | grep "AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT"
```

Determination of adapter sequences: I utilized the 'Overrepresented sequences' tab from the fastq report to assist in determining adapter sequences. Fastqc suggested Truseq was used for sequencing. /i looked up adapter trimming for TruSeq and was directed to an Illumina webpage, `https://support.illumina.com/bulletins/2016/12/what-sequences-do-i-use-for-adapter-trimming.html`, that provided the sequences to use for adapter trimming. This information is available in the cutadapt documentation, as well.

Results of adapter check: All returns to the check commands returned nothing, confirming that adapter sequences are not present in the adapter-trimmed files.

Reasoning for choosing commands for adapter check: I chose to use a command that checks for the adapter sequence in the finished file because I want to confirm that the adapter sequence was removed from the raw reads before being output to the adapter trimmed files.

7. Distribution of length of trimmed sequences in each read for each group

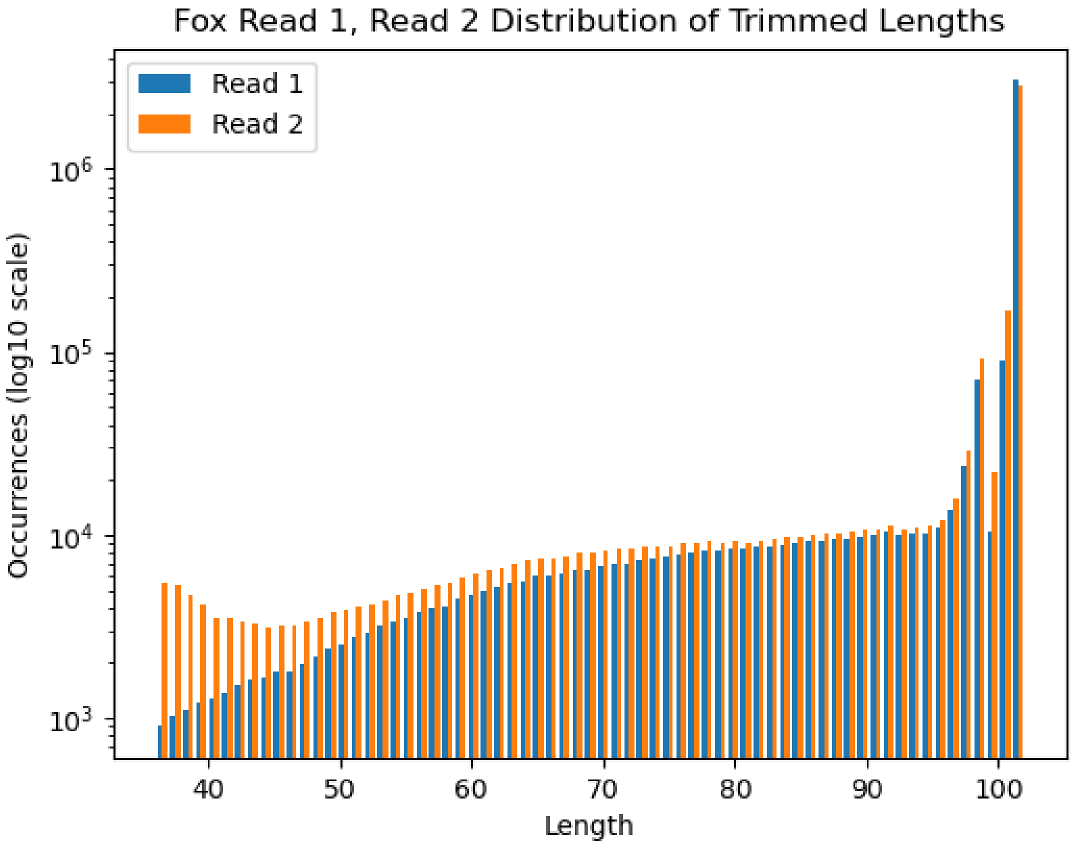

Figure 13: Distribution of lengths of '31_4F_fox_S22_L008_R1_001' and '31_4F_fox_S22_L008_R2_001' reads following adapter and quality trimming.

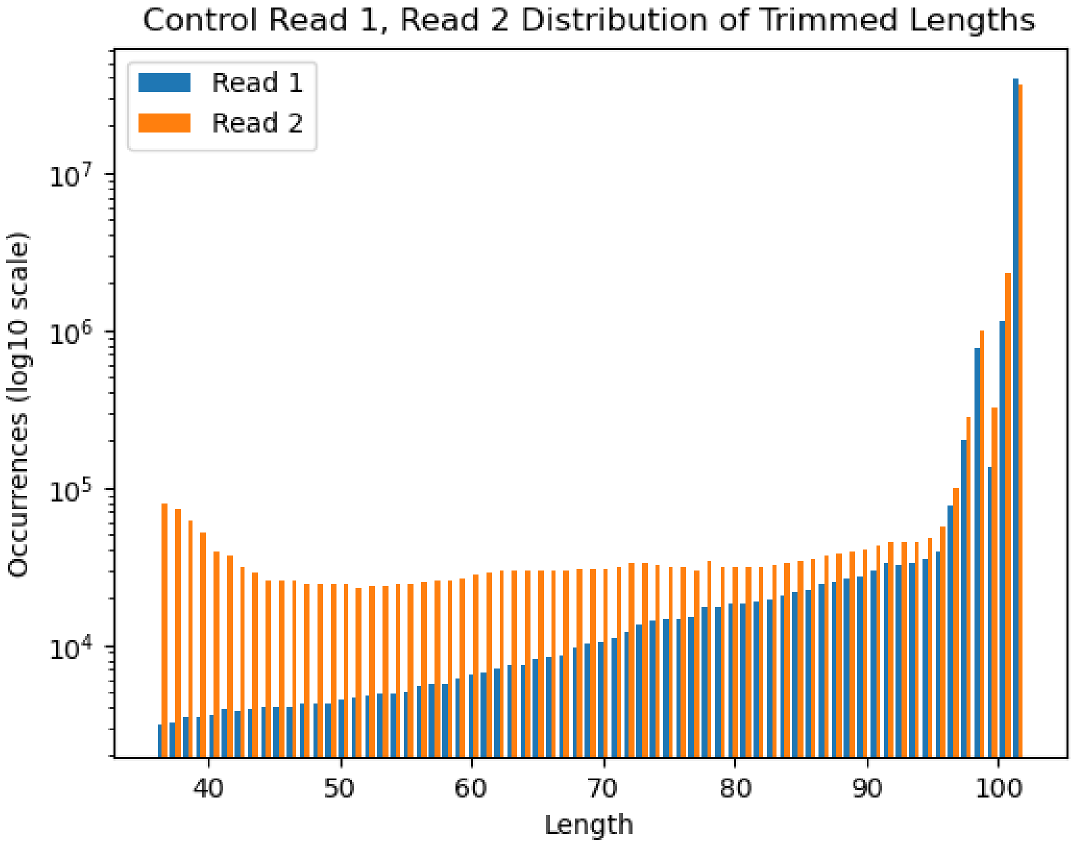

Figure 14: Distribution of lengths of ‘23_4A_control_S17_L008_R1_001’ and ‘23_4A_control_S17_L008_R2_001’ reads following adapter and quality trimming.

R1 or R2 adapter-trimmed at different rates? I would expect R2 to be adapter-trimmed at a higher rate. Because this read sits on the sequencer for a longer period of time, the strand is exposed to a greater amount of harsh chemicals and there are more opportunities for breakdown of the DNA molecules, subsequently leading to a higher proportion of adapter sequences present in the reads.

Part 3 - Alignment and strand-specificity

10. Mapped vs. unmapped reads

Mapped vs. Unmapped Fox .sam:

file: /projects/bgmp/jadler2/bioinfo/Bi623/QAA/align/fox/Aligned.out.sam

mapped reads: 6969878

unmapped reads: 225938

Mapped vs. Unmapped Control .sam:

file: /projects/bgmp/jadler2/bioinfo/Bi623/QAA/align/control/Aligned.out.sam

mapped reads: 79473045

unmapped reads: 4640081

12. Demonstrate whether data are strand-specific RNA-seq libraries or not

Determine number of mapped reads in each file:

```
control_reverse.genecount
65575631 reads mapped to features / 81286723 total reads = 80.7%

control_stranded.genecount
2173292 / 81286723 = 2.67%

control_unstranded.genecount
64101648 / 81286723 = 78.9%

fox_reverse.genecount
5686010 / 6695837 = 84.9%

fox_stranded.genecount
282362 / 6695837 = 4.2%

fox_unstranded.genecount
5654669 / 6695837 = 84.5%
```

Stranded or unstranded library prep: I propose the data are from stranded library preps because the proportion of reads mapped in the 'reverse' direction for the 'fox' and 'control' reads, 84.9% and 80.7% respectively, are roughly equal to the proportion of reads mapped in each 'unstranded' run, 84.5% and 78.9% respectively, whereas, in the 'stranded' (forward) run only 4.2% and 2.67% of the reads mapped in the 'fox' and 'control' runs, respectively. If the prep were unstranded, we would expect roughly equal proportions of mapped reads in the 'stranded' (forward) and 'reverse' runs.