QAA_report

Abraham Solomon

2024-09-11

Part 1

The per base quality distribution for both *control* reads and both *both* reads meets satisfactory targets and is consistent with the substantial absence of N content per base (Fig. 1, Fig 2).

The per base quality score distribution for each file generated by my in house `qual_score_dist.py` python script is effectively equivalent to the distributions produced by FastQC. The total run time was approximately equivalent because while FastQC has capability to process all four files simultaneously with multithreading unlike my `qual_score_dist.py` python script, it also generates more plots than my `qual_score_dist.py` python script such as per base N content which evens out the total run time to ~8 minutes.

With zero overrepresented sequences for read 1 control, read 1 experimental and read 2 experimental reads, the data appears to be consistently high quality.

Part 2

3_2B_control libraries were prepared using an Illumina TruSeq paired-end library kit making each read strand specific where R1 and R2 correspond to the forward and reverse reads respectively. However, because *32_4G_both* samples were not prepared using a strand specific library, R1 and R2 reads are not strand specific and thus we expect to see equivalent adapter trimming for R1 and R2. After adapter trimming with Cutadapt, quality filtering with Trimmomatic to remove artifacts from sequencing, and plotting the distributiobn of read lengths for each sample type we find that the reverse read for *3_2B_control* (R2) maintained a higher number of reads with larger lengths suggesting less adapter content and higher quality than the forward read. It is expected that the forward read would have

Plotting the read length distribution for *32_4G_both* samples confirmed the postulate that these reads were not strand specific because all reads, both R1 and R2, were equally trimmed. By overlaying the read length distribution of R1 and R2 at 50% opacity we see that only one line is shown and in purple suggesting a perfect mixing of R1 and R2 for the frequency of different read lengths.

Part 3

All sample reads were aligned to the *Mus musculus* mouse reference genome using STAR and the number of transcripts that map to genes in this reference genome were counted using two methods HTSeq-count and an in-house sequence feature map counting script called `mapcounts.py` (Table 1, Table 2).

If the total number of features (mapped and unmapped) for either the forward or reverse read are summed, doubled and equal the number of features counted with `mapcounts.py` then the original library used was strand-specific giving rise to stranded RNA-seq reads. This is because the stranded option from HTSeq-count is checking for only the forward or reverse strand to the reference genome but `mapcounts.py` is not and so to get the total number of genes that were counted, you must double all outputs from HTSeq-count. If the HTSeq-count doubled total does not match `mapcounts.py` then the library was not stranded. As shown in

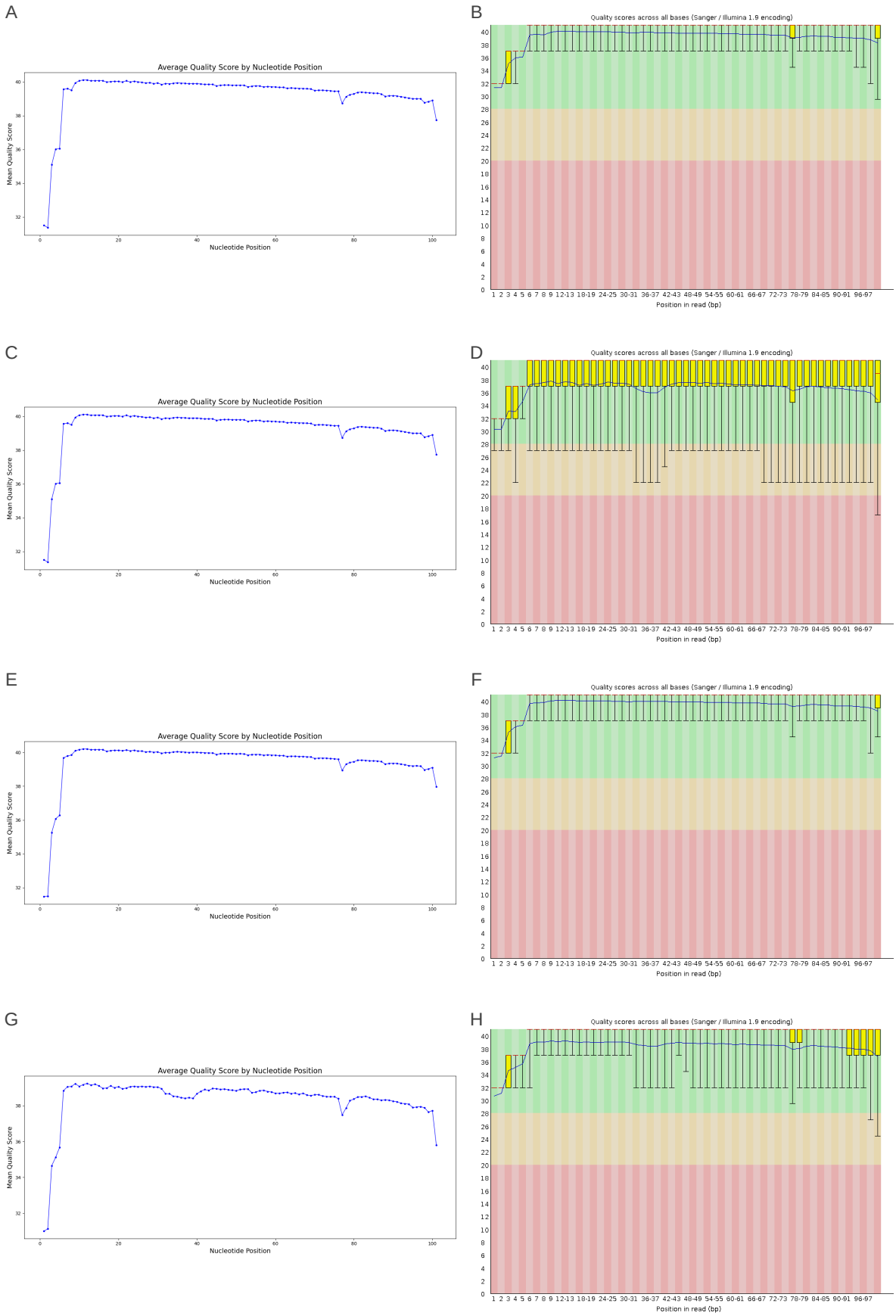

Figure 1: Figure 1. Per base quality score distribution comparison between by FastQC and an in-house python script. **(A, C, E, G)** were generated by my in-house python script. **(B, D, F, H)** were generated by FastQC. **(A, B)** is *3_2B_control_S3 R1* **(C, D)** is *3_2B_control_S3 R2* **(E, F)** is *32_4G_both_R1*, **(G, H)** is *32_4G_both_R2*.

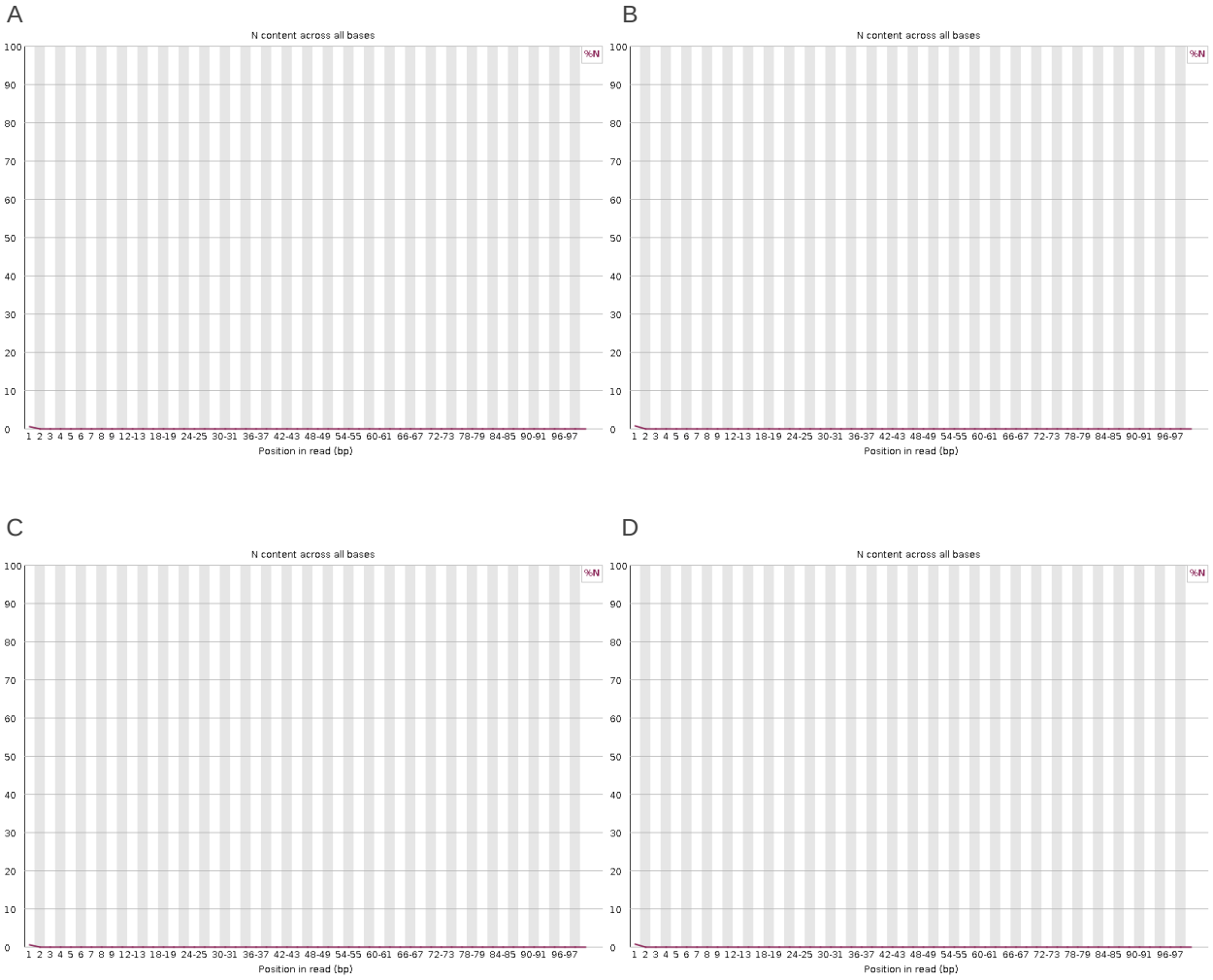

Figure 2: Figure 2. N content across all bases for **(A)** *sample 3_2B_control_S3_R1*, **(B)** *3_2B_control_S3_R2*, **(C)** *32_4G_both_S23_R1*, **(D)** *32_4G_both_S23_R2* generated by FastQC.

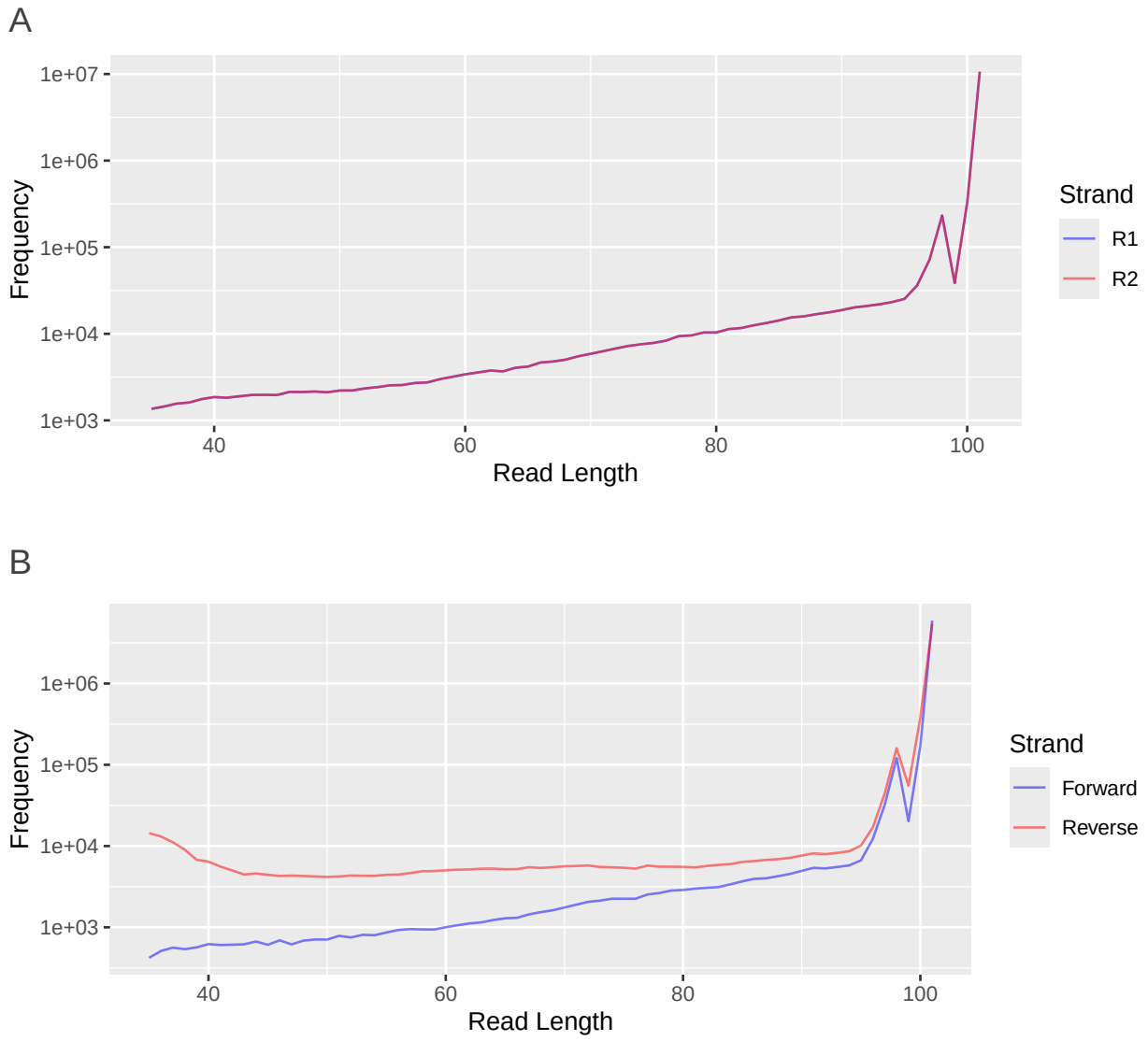

Figure 3: Figure 3. **(A)** Read Length distribution for 3_2B_control and **(B)** 32_4G_both after trimming the adapter sequences with cutadapt and quality filtering with Trimmomatic. The library of **(A)** is strand non-specific so read 1 (R1) and read 2 (R2) are not forward or reverse.

Table 1 shows that when the total gene count for the forward read is doubled, it equals the `mapcounts.py` total allowing us to confidently say that *3_2B_control* is stranded. Additionally in Table 2, we see a large difference in genes mapped to the reference genome between the forward and reverse reads providing further evidence that this library was stranded.

The same cannot be said for *32_4G_Both* reads because the percentage of genes that map the mouse reference genome are equal between R1 and R2. Additionally, the same rule of gene count totals does not apply.

Ultimately, I would suggest that we move forward with analyzing this data because it has maintained high quality reads with low N content, low adapter content, and trimming has not reduced the number of genes that can map to the genome too much.

Table 1: The number of transcripts successfully mapped to genes in the mouse reference genome for forward versus reverse reads counted with HTSeq-count and `mapcounts.py`.

HTSeq.count	X3_2B_control	X32_4G_both
HTSeq-count forward	6428019	12021662
HTSeq-count reverse	1389506	12021660
`mapcounts.py`	12856038	23600576

Table 2: Percent of genes successfully mapped to the mouse reference genome counted with HTSeq-count.

HTSeq.count	Percent.Stranded.Forward	Percent.Reverse
3_2B_control	3.45	15.95
32_4G_both	1.84	1.84

Table 3: The number of transcripts that did and did not successfully map to genes in the mouse reference genome for forward and reverse reads counted with `mapcounts.py`.

mapcounts.py	X3_2B_control	X32_4G_both
Mapped	12359963	168582
Unmapped	496075	23431994
Total	12856038	23600576