RNA-seq Quality Assessment

Maura Kautz

September 10, 2024

Goal

The goal of this project was to use existing bioinformatic tools for the analysis of two RNA-seq libraries. The quality assessment was done in three parts including read quality score distributions, adaptor trimming, and alignment and strand-specificity analyses.

Part 1: Read Quality Score Distributions

FastQC Plot Results Compared to My Quality Score Plotting Script Results

Plots of per-base quality score distributions for samples:

- 4_2C_mbnl_S4_L008_R1_001
- 4_2C_mbnl_S4_L008_R2_001
- 34_4H_both_S24_L008_R1_001
- 34_4H_both_S24_L008_R2_001

The following FastQC graphs employ BoxWhisker plots to visualize the data distribution. The key for the plots is as follows:

- The central red line is the median value
- The yellow box is the inter-quartile range from 25-75%
- The upper and lower whiskers represent the 10% and 90% range points
- The blue line shows the mean quality
- The background color of the graphs divide the y-axis into very good quality scores (green), reasonable quality scores (yellow), and poor quality scores (red)

My quality score plotting script mean_qual.py produces only the blue line showing the mean quality scores across each base position.

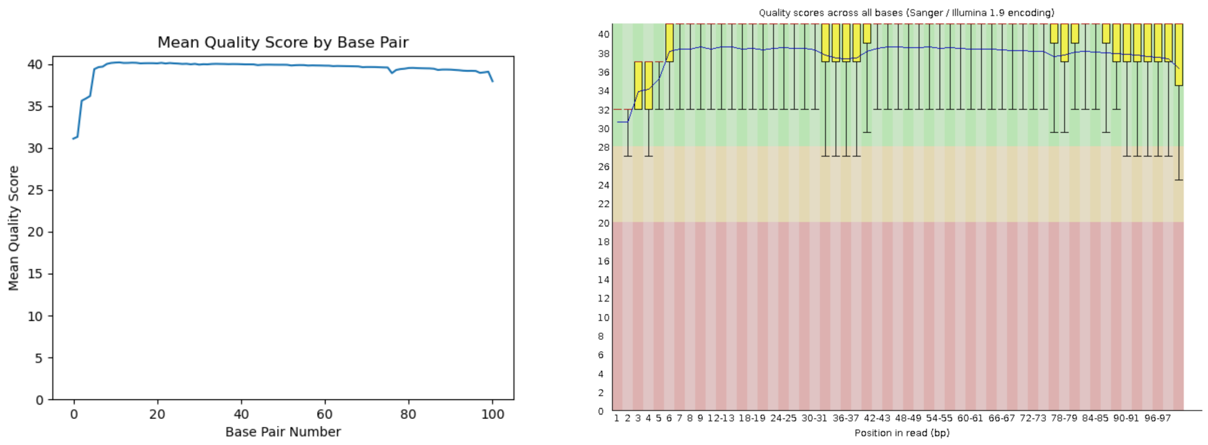


Figure 1: Sample 4_2C_mbnl_S4_L008_R1_001 Plotting Comparison. This is the range of quality values across all 101 bases at each position in the FastQ file after running both FastQC and mean_qual.py for read one (R1) of sample 4_2C_mbnl_S4. Following the mean quality score line for this library, an average quality score around 38 in the FastQC graph indicates high quality data with a base call accuracy of 98.984% or a 0.000158 probability of an incorrect base call. The whiskers that drop into the reasonable quality section of the graph show that some of the reads at that base have lower quality scores, but the majority of bases maintain high quality base calls. The graph produced by mean_qual.py follows a very similar pattern to that of FastQC, however it lacks the smaller details in quality score dips, such as that observed in the FastQC plot at the 36-37 bp range. The overall quality scoring of mean_qual.py produces a higher average as well. These plots visualize a data distribution with arguably high Phred +33 scores and indicates success in Illumina sequencing.

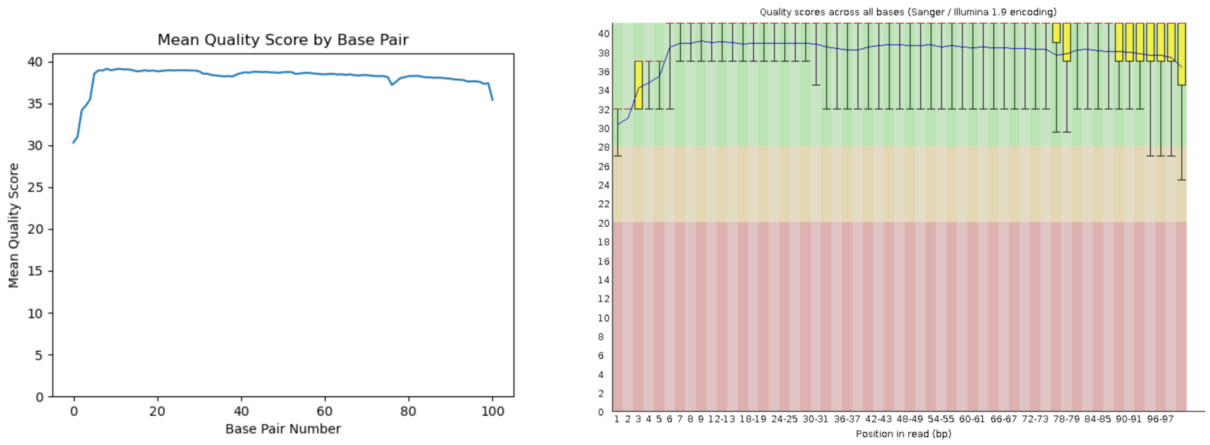
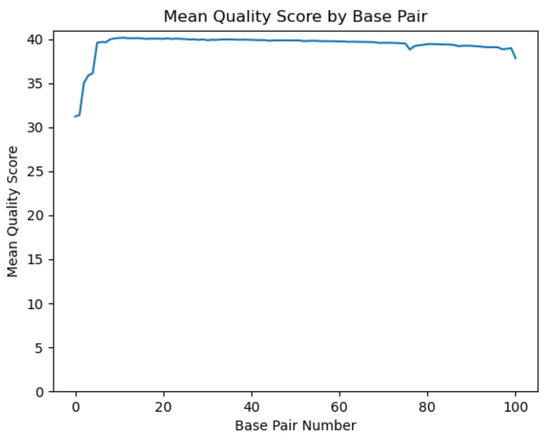


Figure 2: Sample 4_2C_mbnl_S4_L008_R2_001 Plotting Comparison. Similar to Figure 1, these plots visualize the quality score distribution of the second Illumina read (R2) for sample 4_2C_mbnl_S4. The distribution is essentially identical to that of read one, a predictable outcome as this sample is the result of paired-end sequencing. Overall, the lower whisker distribution across the FastQC plot is consistently in the very good quality score range, indicating that the overall base call quality is higher than that of read one, which will be expanded upon later. A dip in base call quality at the right end of the plot is explained by a decreased quality outcome as runs progress on Illumina sequencers, a common trend that is accounted for. The plot produced by mean_qual.py is slightly better than that produced for read one of the same sample. Some of the smaller intricacies observed in the FastQC plot are also seen in the mean_qual.py plot, such as the dip in quality score around the 37 bp range.



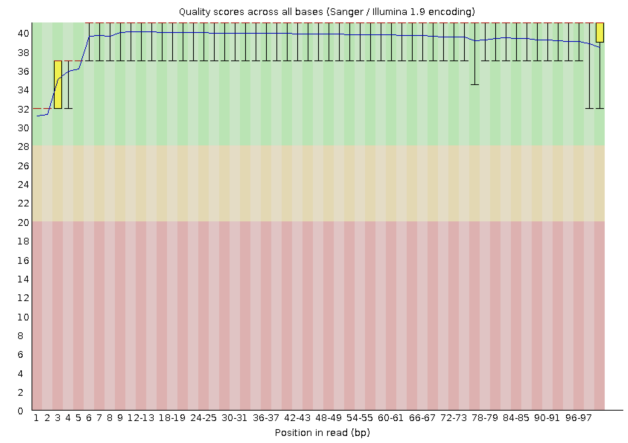
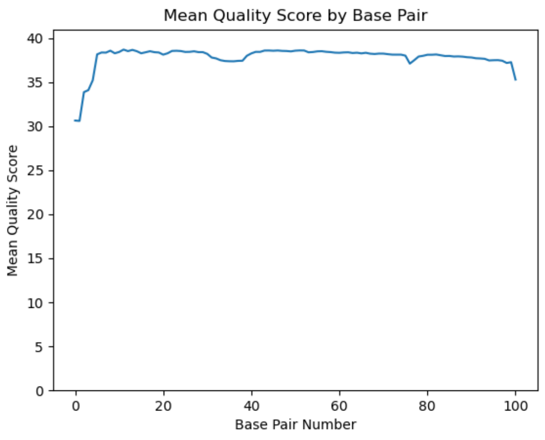


Figure 3: Sample 34_4H_both_S24_L008_R1_001 Plotting Comparison. These plots display the quality score distribution of all 101 bases from R1 of sample 34_4H_both_S24. As demonstrated by the mean quality score line and the lower whisker distribution in the FastQC plot, the average quality of base calls for this sample is extremely high. There is an apparent trend around the score of 40, which is indicative of a 99.99% base call accuracy. As explaned in Figure 2, the slight drop in quality towards the right end of the plot is a common occurence due to quality degradation as the run progresses on the sequencer. The mean_qual.py graph appears to produce a plot almost identical to that of FASTQC, with the average quality score remaining at teh 40 mark. While the FastQC program produces overall higher quality analysis, the mean_qual.py graph is also useful here as it shows essentially the same data outcome.



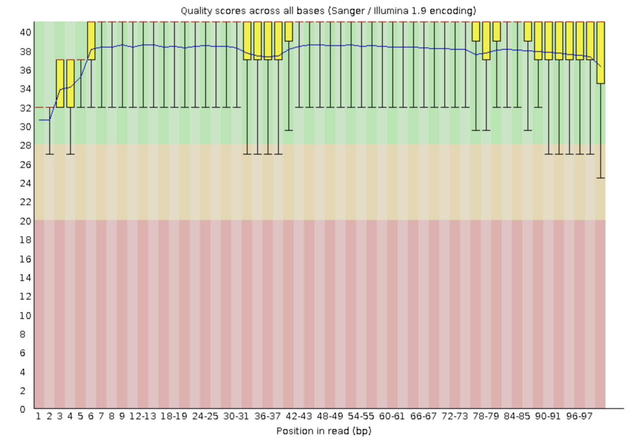


Figure 4: Sample 34_4H_both_S24_L008_R2_001 Plotting Comparison. Similar to Figure 3, the graphs above display the quality score distributions for R2 of sample 34_4H_both_S24. While the overall per base sequence quality is high, between 37 and 38, the lower whisker distribution is significantly lower than that for read one of the same sample. The lower whiskers in the FastQC plot indicate a larger range of base call qualities across all positions in the sequence, and an overall higher occurence of lower quality base calls. The Illumina sequencing data for R2 is usually slightly lower, as the cluster sizes decrease ring bridge amplification at the paired-end turnaround stage. The mean_qual.py distribution is very similar to that of the FastQC plot, and also displays an overall drop in quality.

Overall, the FastQC quality plots were very similar to those produced by my script. However, I would recommend using FastQC due to time and memory usage aspects. The FastQC software is overall much faster than mean_qual.py. It ran both 4_2C_mbnl reads together in under two minutes, whereas my script took over four minutes to run each read for 4_2C_mbnl separately. Memory usage was similar for both scripts, considering that the FastQC run for both 34_4H_both reads used 98% of the CPU allocated for the job, while the run for read one of 34_4H_both had 95% CPU usage. The CPU usage is not a very insightful metric in this case, as only one core was required for these jobs. The time difference is expected though, as FastQC is optimized to be as efficient as possible. While my code works similarly to FastQC, it is not optimized and only produces one plot at a time, compared to the multiple graph output of FastQC.

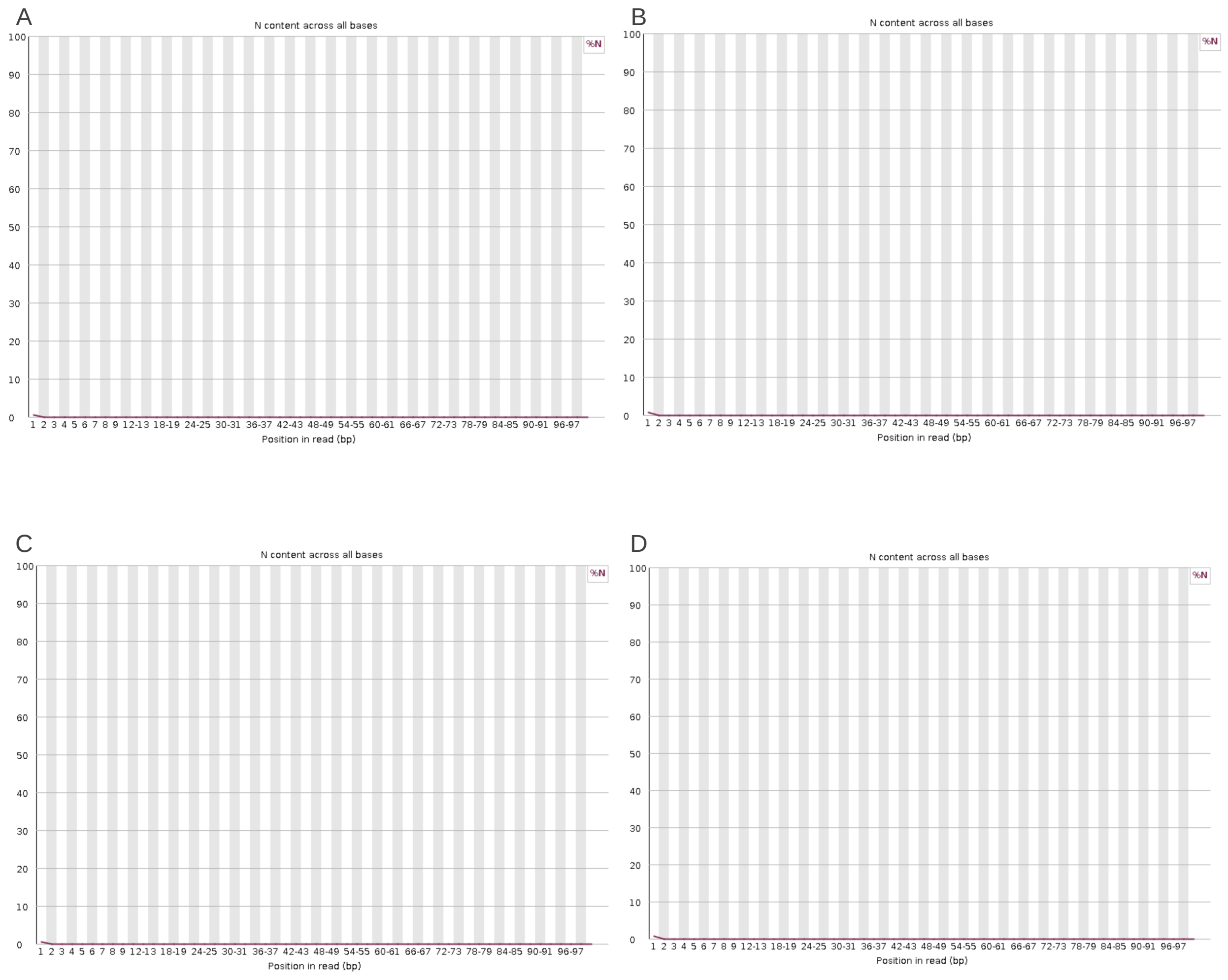


Figure 5: Comparison of Per Base N Content. Plot A is associated with R1 of sample 4_2C_mbnl and B with R2. Plots C and D are associated with sample 34_4H_both, R1 and R2 respectively. Due to high sequencing accuracy, there is very low N content displayed.

Overall, both libraries have high data quality that suggests further analysis would be beneficial. Looking at the per-base quality score distributions in Figures 1, 2, 3, and 4 for reads in both samples, the overall mean qualities at all base positions are considered high on a Phred+33 scale. The lower quality dips at the beginning and ends of the sequences are expected due to Illumina software and sequencing flaws, but these dips are not low enough to impact overall data quality. The N content across all bases can also be considered in this recommendation. In Figure 5, it is observed that there is little to no N content at all base positions.

An N will only appear when a sequencer is unable to make a base call with sufficient confidence. So, the low N content helps in understanding the overall data quality for both libraries. The slight uptick in N content at the beginning of the sequences is expected as the reads at the beginning of the sequences are on the edges of Illumina flow cells and therefore makes imaging more difficult. The combination of low N content and high quality score distributions makes these libraries good candidates for further investigation.

Part 2: Adapter Trimming Comparison

Cutadapt Trimming Results

Cutadapt was used to trim adapter sequences from both libraries. The following proportion of reads were trimmed:

Table 1. Proportion of reads that were trimmed - CutAdapt Output

Library	Read 1	Read 2
4_2C_mbnl	6.2%	6.9%
34_4H_both	9.1%	9.8%

A sanity check was performed and it was confirmed that Adapter 1 was only found in R1 of both libraries, and Adapter 2 was only found in R2 of both libraries.

Trimmomatic Quality Trimming Results

Trimmomatic was used on the cutadapt output to quality trim the data.

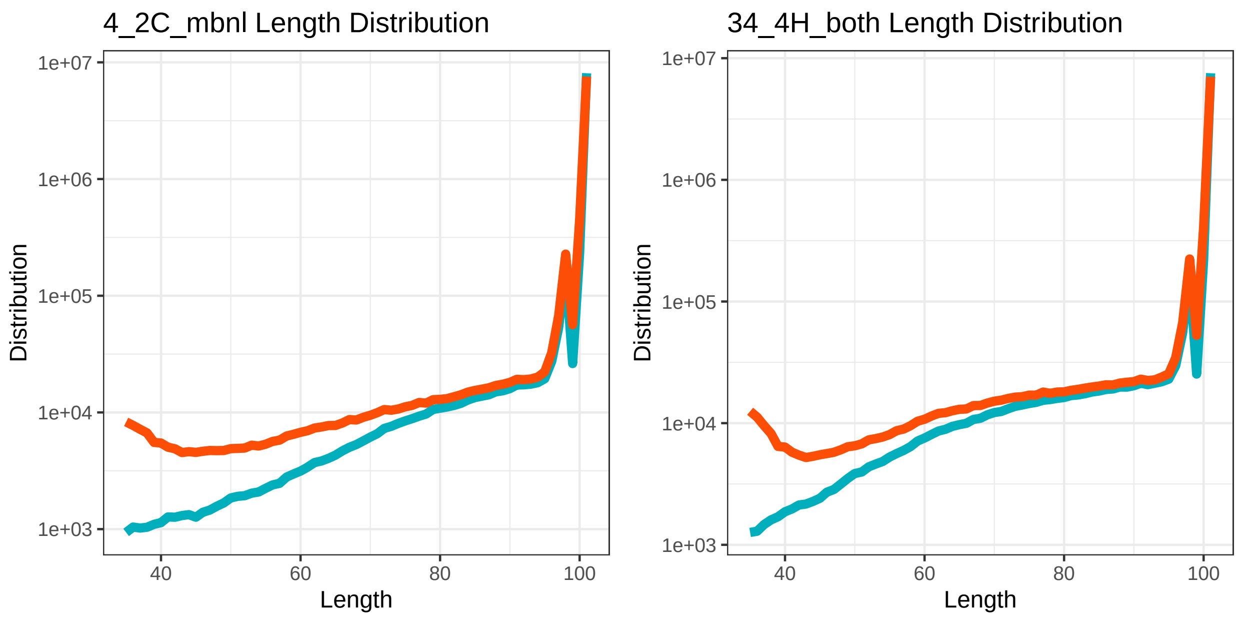


Figure 6: Trimmed Read Length Distributions for Samples 4_2C_mbnl and 34_4H_both. In the plots above, the blue line represents R1 and the orange line represents R2. It is observed that after adapter and quality trimming, there is more variable distribution in read lengths as the lower quality data is removed. The distribution is normalized to a log10 scale to better visualize the smaller data points.

The trimmed read length distributions give insight into how much the data was impacted by adapter removal and quality control. It is observed that R2 for both samples appear to have a larger distribution of reads with a shorter length. This may be because degradation begins to occur on the flow cell as Illumina progresses, such as chemical and imaging errors that result in R2 having overall slightly lower quality and the need for more quality trimming. R2 is also more susceptible to having the sequencer read into the adapter sequence if the DNA fragment is shorter, causing adapter trimming to create this distribution.

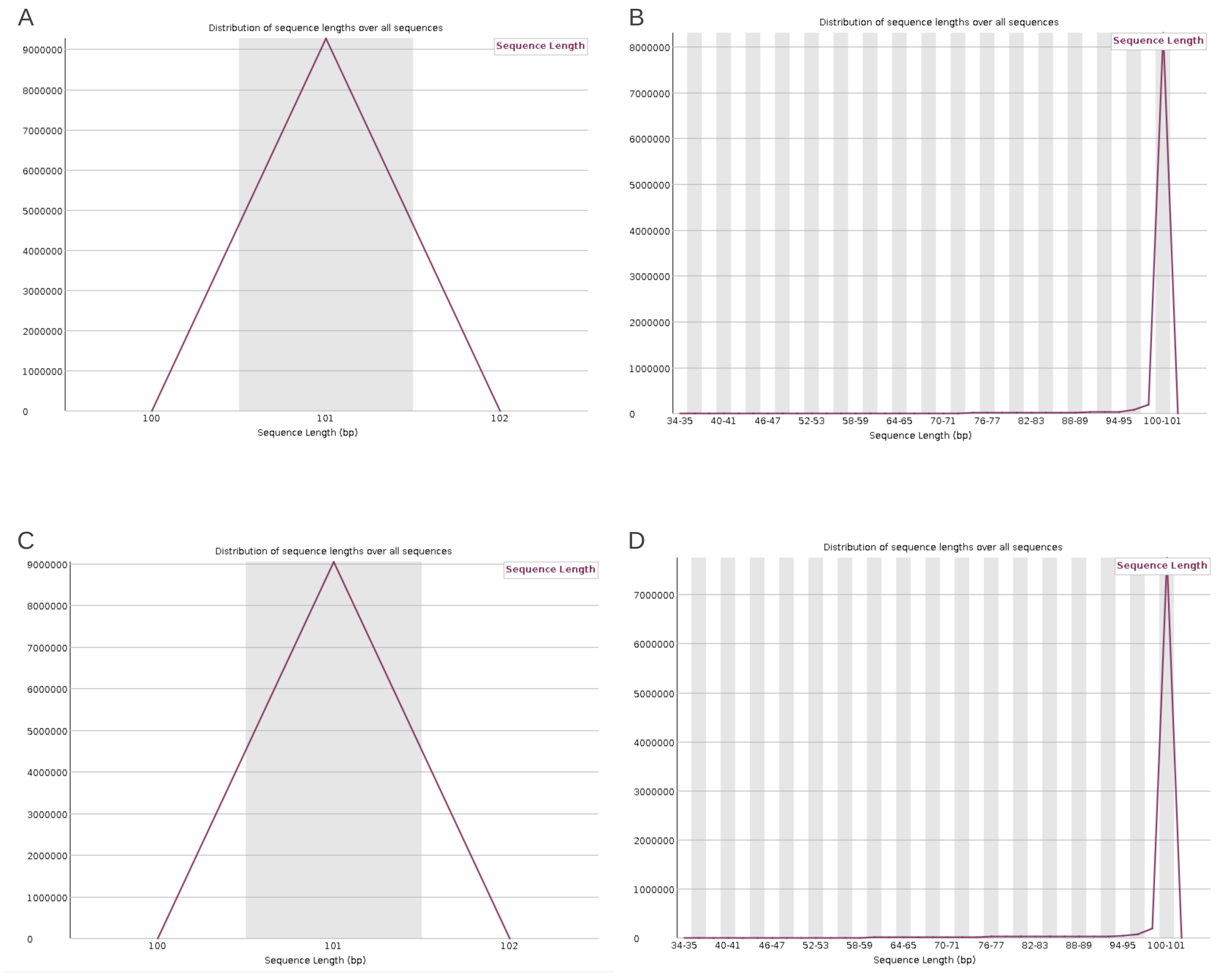


Figure 7: Comparison of Post Trimming Length Distribution. A refers to R1 of sample 4_2C_mbnl in FastQC before trimming, and B is R1 of the same sample after trimming. C and D refer to R1 of sample 34_4H_both in FastQC before and after trimming, respectively. The distributions are impacted by overall data qaulity.

The comparison of read length distribution is drastically different between the untrimmed and trimmed data. Before trimming, there were no reads with a sequence length less than 101. This shows raw data that was produced to specifically have a read length of 101, which is observed in the sharp increase and decrease between lengths 100 and 102. In the trimmed data sets, the read length distribution has much more variability. In Figure 6, the log(10) normalization shows the small data points at many different length distributions. As discussed earlier, data cleanup including adapter removal and trimming creates reads of different sequence lengths, as it is only including the highest quality relevant data. The main difference between the untrimmed and trimmed read length distributions is the level of data quality.

Part 3 - Alignment and Strand Specificity

Mapped and Unmapped Reads

A database was built to compare the trimmed data to. The number of mapped and unmapped reads were produced as a result of my read_count.py script.

Table 2. Proportion of Mapped and Unmapped Reads

Results	4_2C_mbnl	34_4H_both
Unmapped Reads	788079	483616
Mapped Reads	17172681	16822666
Total Reads	17,960,760	17,306,282
Percent of Reads Mapped	95.61%	97.21%

Almost all of the reads from both libraries were mapped to the reference genome database that was built from Ensembl. The small number of unmapped reads could be a result of sequencing errors.

Determined the percentage of reads mapped to features in both the stranded and reversed output files from htseq-count, to make an inference about strand specificity.

Table 3. Proportion of Forward Stranded vs Reverse Stranded Reads

Results	4_2C_mbnl	34_4H_both
Forward Stranded	3.71%	5.46%
Reverse Stranded	80.58%	83.37%

I propose that these data are strand specific because 80.58% of the reads in sample 4_2C_mbnl are labeled as reverse stranded, versus 3.71% of reads noted as forward stranded. This indicates that the RNA-seq data is reverse stranded or that the sequences of read 2 align to the RNA strand, not those of read 1. Similarly, 83.37% of sample 34_4H_both are reverse stranded compared to 5.46% being forward stranded. If all of the reported data was displayed as very low percentages, that would be an indication that the libraries were not stranded.