RNA-seq Quality Assessment Assignment

2023-09-11

Contents


```
knitr::opts_chunk$set(echo = TRUE)
```

Assigned Libraries

7_2E_fox_S6_L008_R1_001_fastqc.zip, 7_2E_fox_S6_L008_R2_001_fastqc.zip, Undetermined_S0_L008_R1_001_fastqc.zip, Undetermined_S0_L008_R2_001_fastqc.zip

##Part 1 – Read quality score distributions

---

1. FastQc generated per-base quality score distributions for R1 & R2 / per-base N content. Comment on whether or not they are consistent with the quality score plots.

The quality per-base quality score distributions for R1 & R2 of both my libraries show acceptable quality score means with a with low per base N content. Looking at the figures attached below we can see that the mean quality score is signified by the blue line going across the

which means that almost all of the reads that were sequenced was interpreted to make a valid base call.

2. Describe how the FastQC quality score distribution plots compare to your own. If different, propose an explanation. Also, does the runtime differ? If so, why?

my FastQc plots for the 4 reads:-

Ideally, quality scores should be high (Phred scores > 30), looking at the 4 graphs the mean quality score which is signified by the blue line.

7_2_R1_per_base_quality.png- has a quality score of 38 with some starting at 32 which is considered high quality

7_2_R2_per_base_quality.png- has a quality score of 36 with some starting at 30 which is considered high quality

Undetermined_R1_per_base_quality.png- has a quality score of 36 with some starting at 30 which is considered high quality

Undetermined_R2_per_base_quality.png- has a quality score of 34 with some starting at 29(outlier) which is considered high quality

All were pretty similar to the figures that I have generated.

yes, FastQc is much faster in generating the plots for the quality score distribution because it uses multi-threading to parallelize computations on modern multi-core processors, which can significantly speed up the analysis process.

3. Comment on the overall data quality of your two libraries. Go beyond per-base q-score distributions. Make and justify a recommendation on whether or these data are of high enough quality to use for further analysis.

for the following 2 different library reads:-

7_2_R1_per_base_quality.png & 7_2_R2_per_base_quality.png - This library collectively had a high quality score looking at both the mean and median values showen in the graph. I would recomend this data to be pushed up to further analasis

Undetermined_R1_per_base_quality.png & Undetermined_R2_per_base_quality.png - This library collectively had a high quality score looking at both the mean and median values showen in the graph. I would recomend this data to be pushed up to further analasis

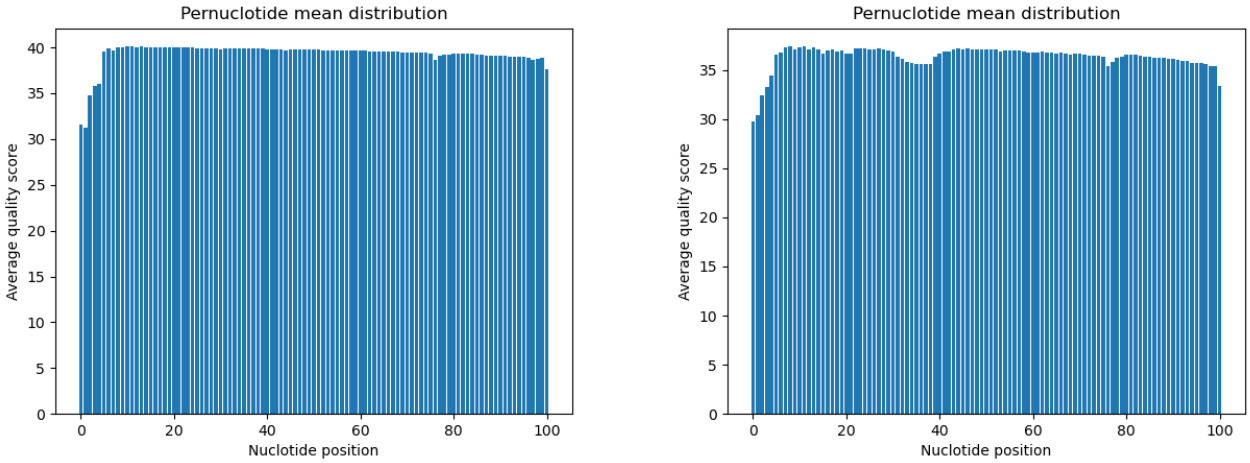

Figure 1: FastQC Generated Plots:7-2E-fox-S6-L008-R1-R2

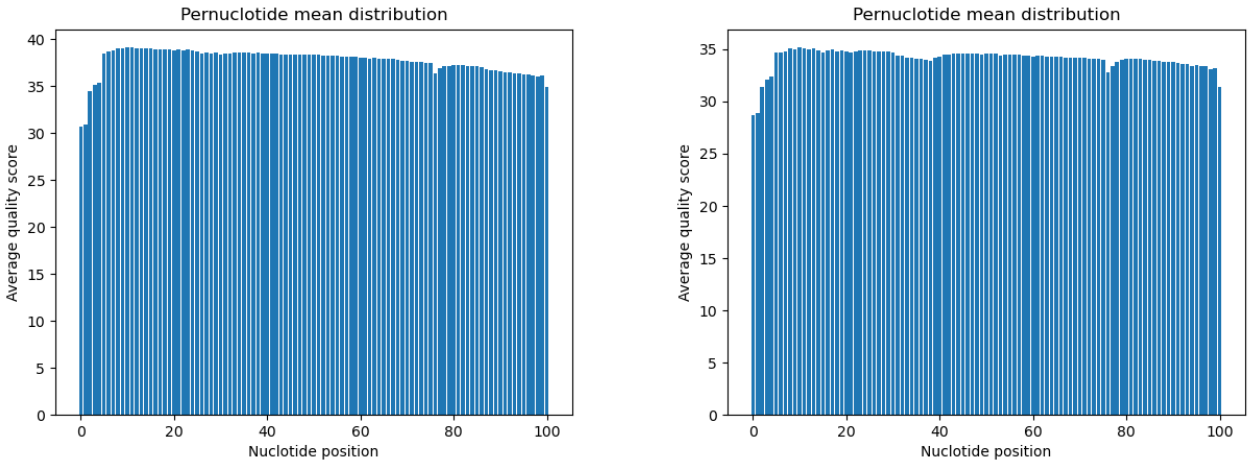

Figure 2: FastQC Generated Plots:Undetermined-S0-L008-R1-R2

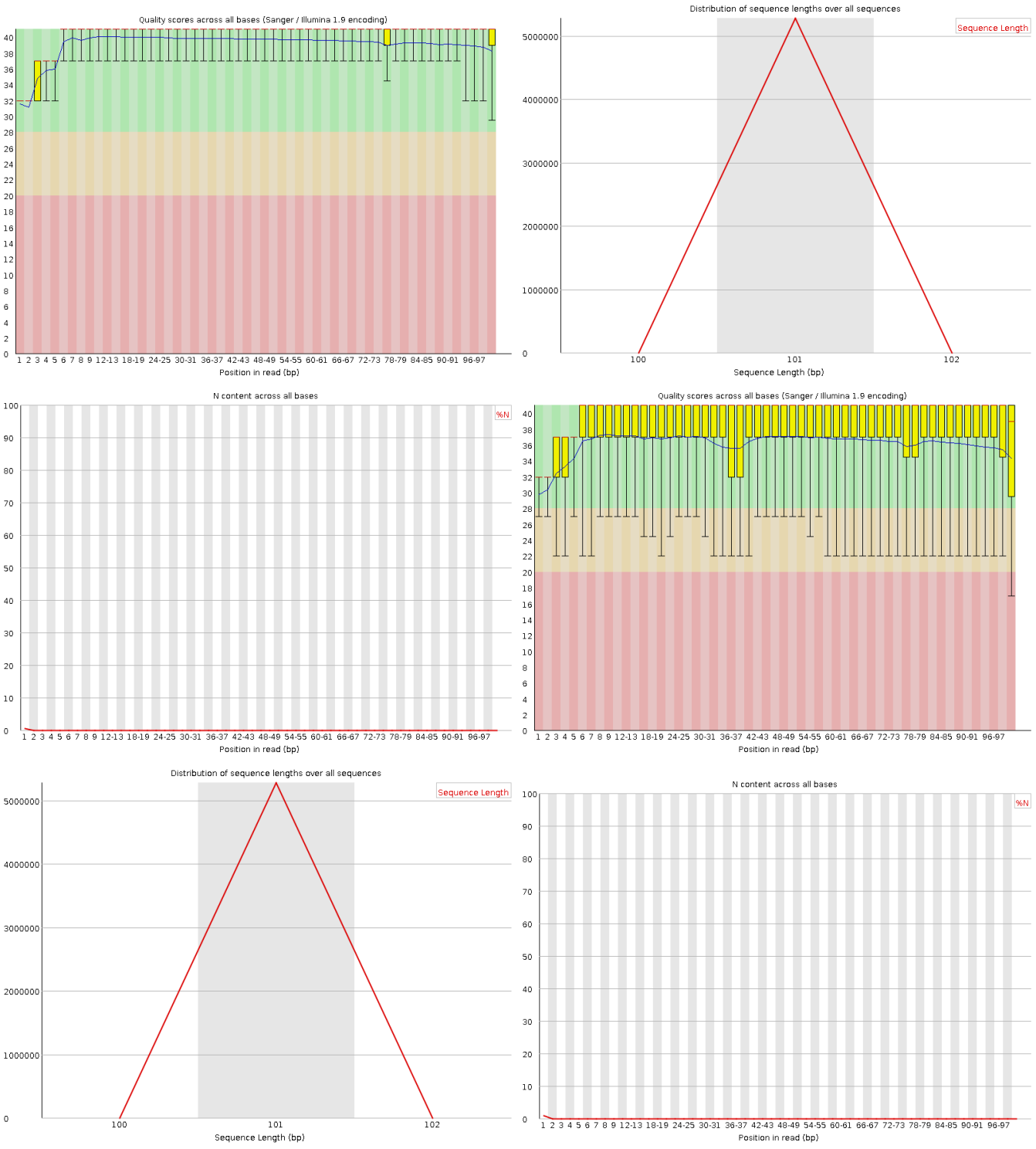

Figure 3: FastQC Generated Plots:7-2E-fox-S6-L008-R1-R2

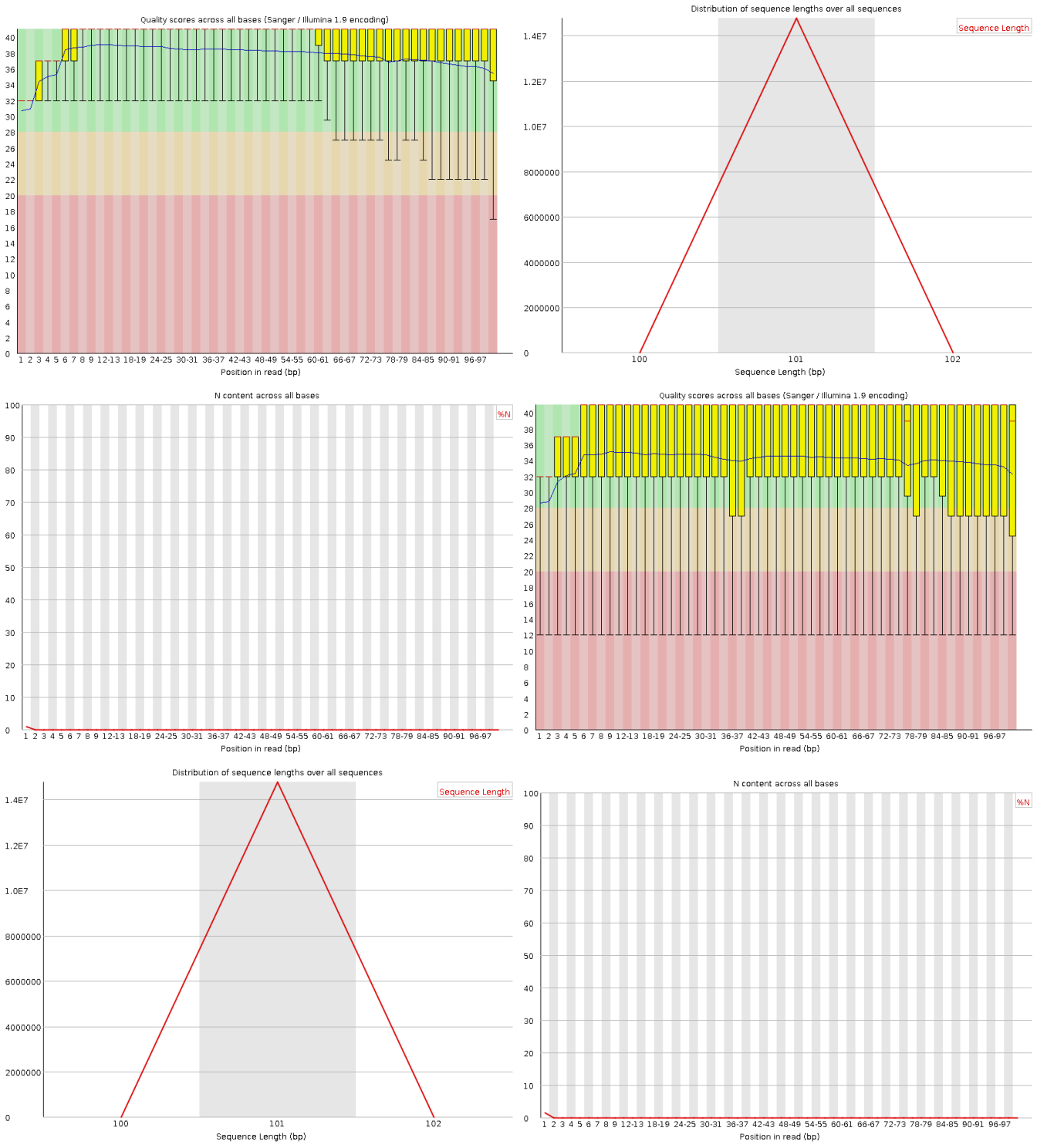

Figure 4: FastQC Generated Plots:Undetermined-S0-L008-R1-R2

Fig.2- FastQc generated quality score distribution and Distribution of sequence length

Part 2 – Adaptor trimming comparison

conda enviroments

cutadapt –version 4.4 trimmomatic -version 0.39

we can cheak for adapters through the command line by

```
zcat<file>|grep("AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC")|wc-l
```

cutadapt
-a AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC
-A AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTAGATCTCGGTGGTCGCCGTATCATT
-o trimmed.1.fastq.gz -p trimmed.2.fastq.gz
reads.1.fastq.gz reads.2.fastq.gz

Once we had our quality score reads and have determined that they were sufficient to move on we moved on to cutting adapters using the above command for paired end reads. I have listed out my results below:-

Cutadapt 7_2E_fox_S6_L008_R1_001.fastq.gz and R2 summary

=== Summary ===

Total read pairs processed: 5,278,425 Read 1 with adapter: 173,473 (3.3%) Read 2 with adapter: 212,512 (4.0%) Pairs written (passing filters): 5,278,425 (100.0%)

Cut adapt undetermined R1 & R2 summary

=== Summary ===

Total read pairs processed: 14,760,166 Read 1 with adapter: 543,021 (3.7%) Read 2 with adapter: 607,660 (4.1%) Pairs written (passing filters): 14,760,166 (100.0%)

–Trimmomatic

**Comment on whether R1s are trimmed more extensively than R2s, or vice versa. Comment on whether you expect R1s and R2s to be adapter-trimmed at different rates and why.

trimmomatic PE -phred33

LEADING:3 TRAILING:3 SLIDINGWINDOW:5:15 MINLEN:35<- special specifications

We expect for both reads to be trimmed at an equal amount because the reads are reverse transcriptomes of each other which are the same length and so are the adapters that we are attaching in both sides,but looking at our data read2 was more trimmed that read1 for both of my libraries. This is shown in the graphes generated below

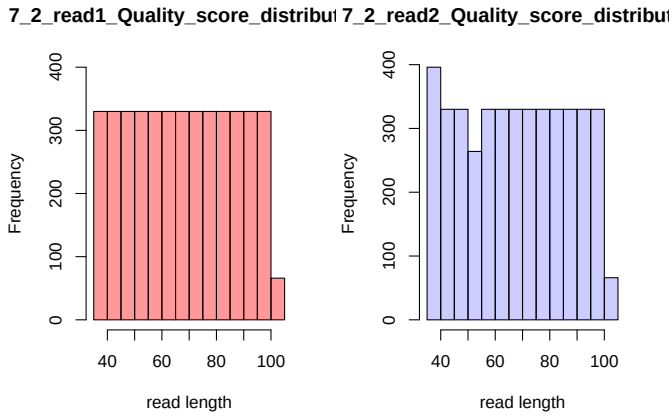

Figure 5: FastQC Generated Plots: 7-2E-fox-S6-L008-R1-R2-Quality-score-distribution

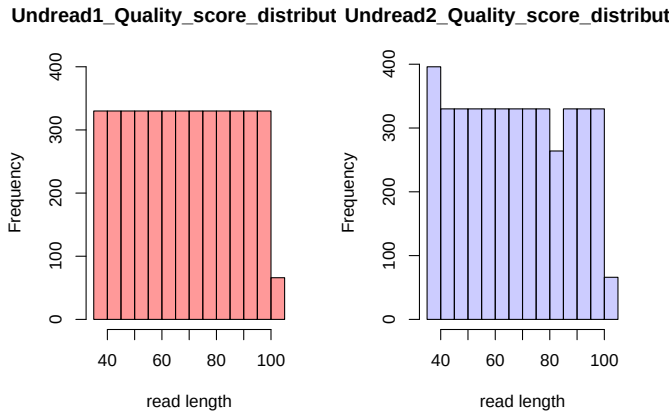

Figure 6: FastQC Generated Plots: Undetermined-S0-L008-R1-R2-Quality-score-distribution

Part 3 – Alignment and strand-specificity

7_2aligned_readAligned.out.sam

Input Read Pairs: 5278425 Both Surviving: 4882703 (92.50%) Forward Only Surviving: 388376 (7.36%) Reverse Only Surviving: 3803 (0.07%) Dropped: 3543 (0.07%) TrimmomaticPE: Completed successfully

Mapped – 9424740 Unmapped - 340666

Undetermined_aligned_readAligned.out.sam

Input Read Pairs: 14760166 Both Surviving: 12160071 (82.38%) Forward Only Surviving: 2511252 (17.01%) Reverse Only Surviving: 31174 (0 .21%) Dropped: 57669 (0.39%) TrimmomaticPE: Completed successfully

Mapped – 15584495 Unmapped - 8735647

**Demonstrate convincingly whether or not the data are from “strand-specific” RNA-Seq libraries. Include any commands/scripts used. Briefly describe your evidence, using quantitative statements.

To count the mapped and unmapped of fw and rv reads for both libraries

Command :-

```
cat <filename> | 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
7_2E_fox_S6_L008	179673	4027046	4206719	0.042710 (4.27%)	0.9572 (95.72%)
Undetermined_S0_L008	306331	6617259	6923590	0.044244 (4.42%)	0.955755 (95.57%)

I propose that these data are strand-specific, because 95.72% of the reads are rv mapped, as opposed to forward strand. looking at the differences between these results the reads are strand specific