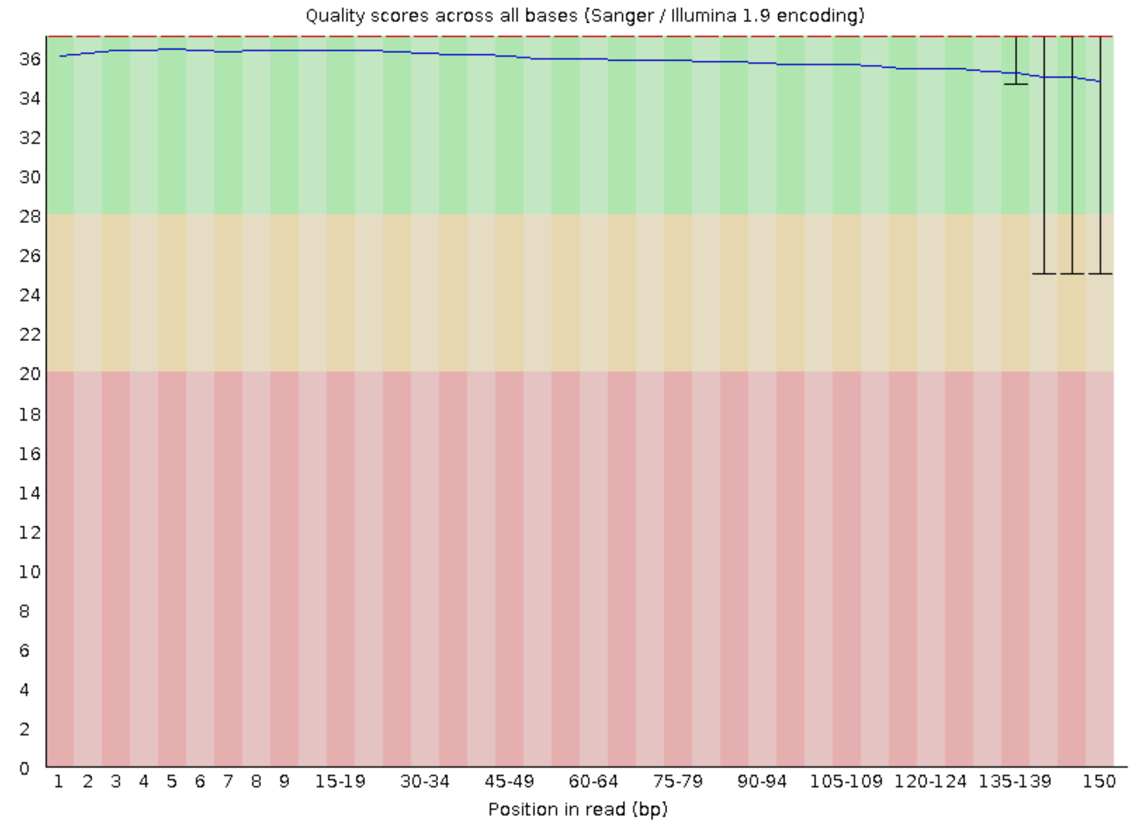
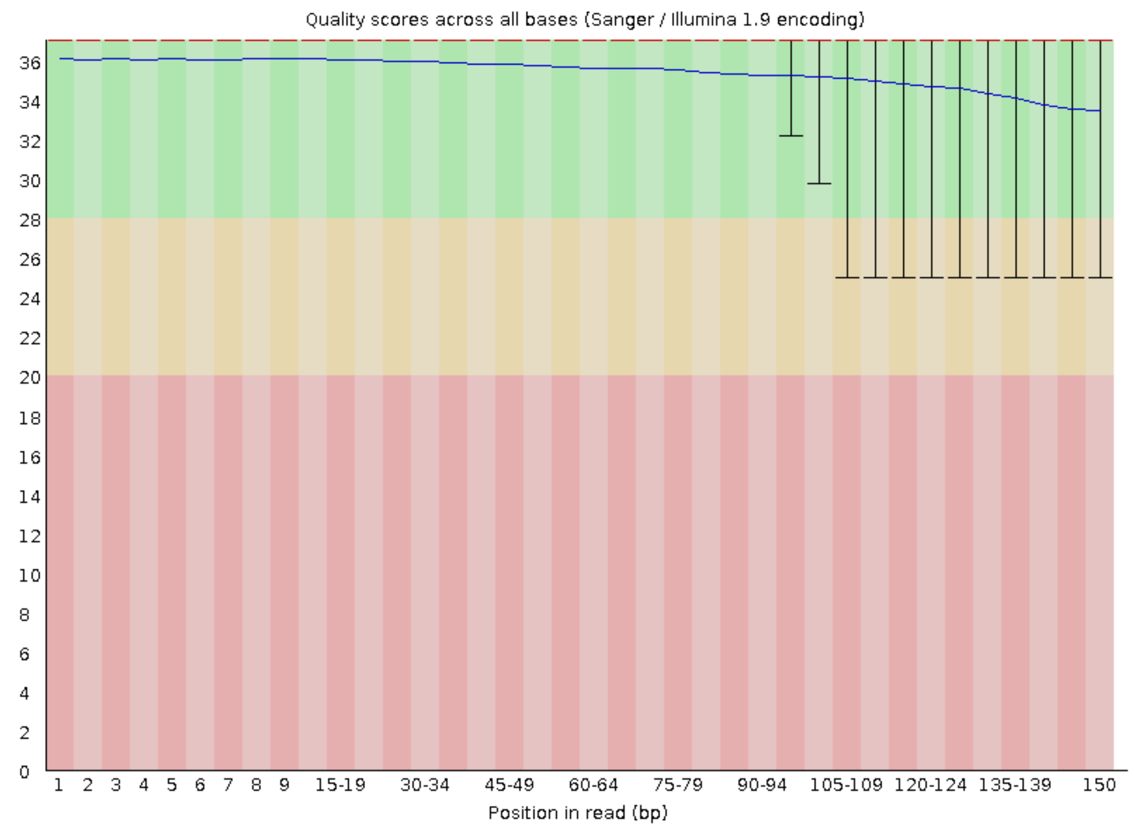
QAA Report

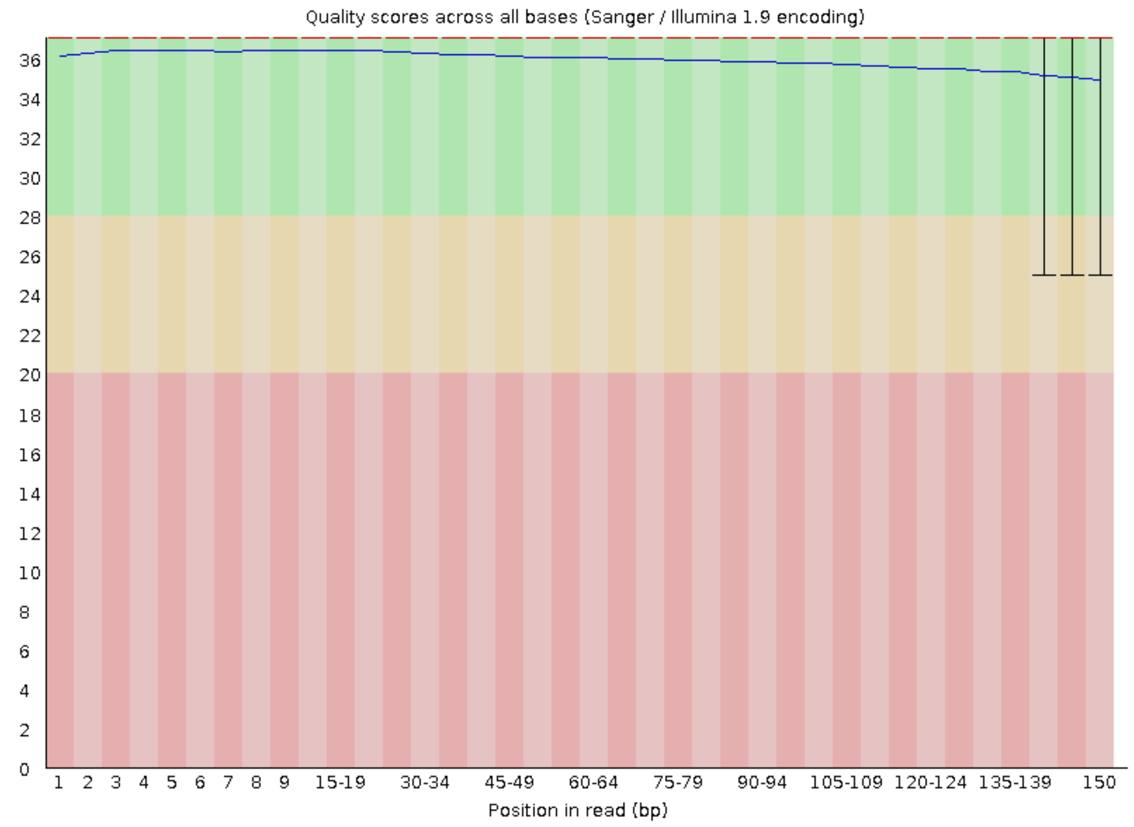
Ian Anderson

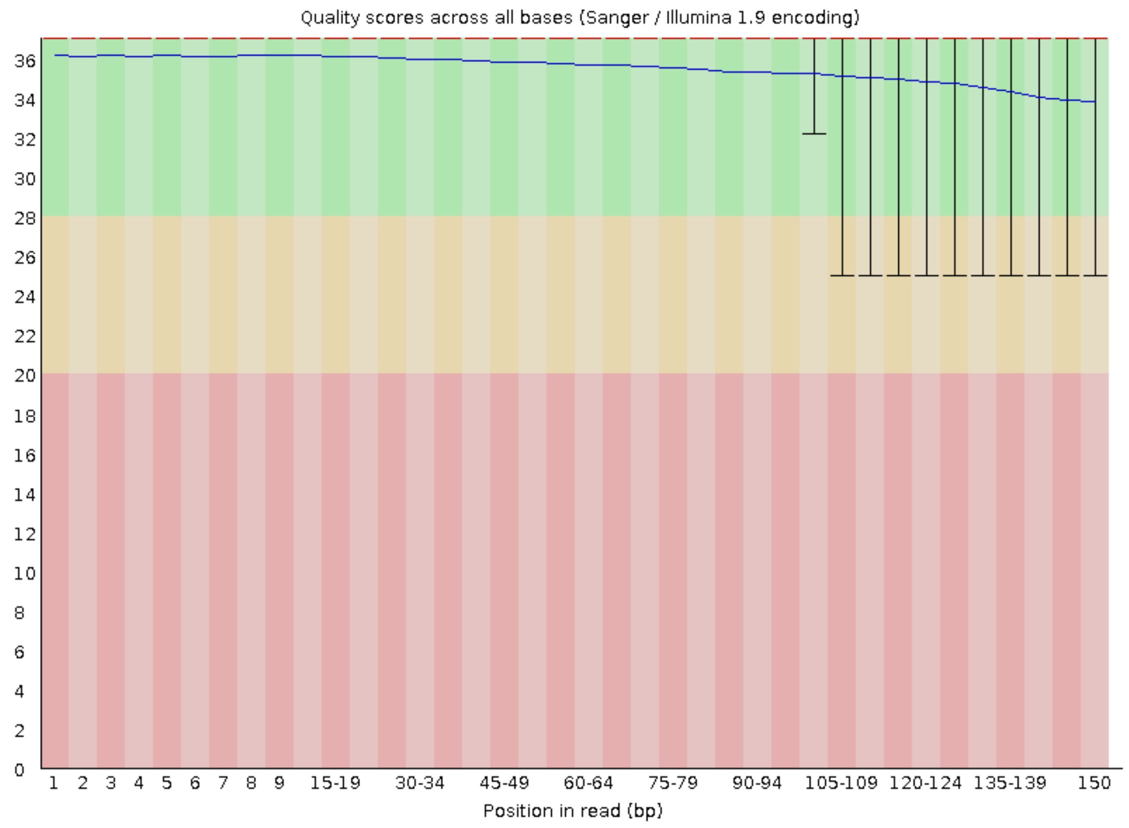
Part 1

1.1 Per-base quality score distributions (FastQC)

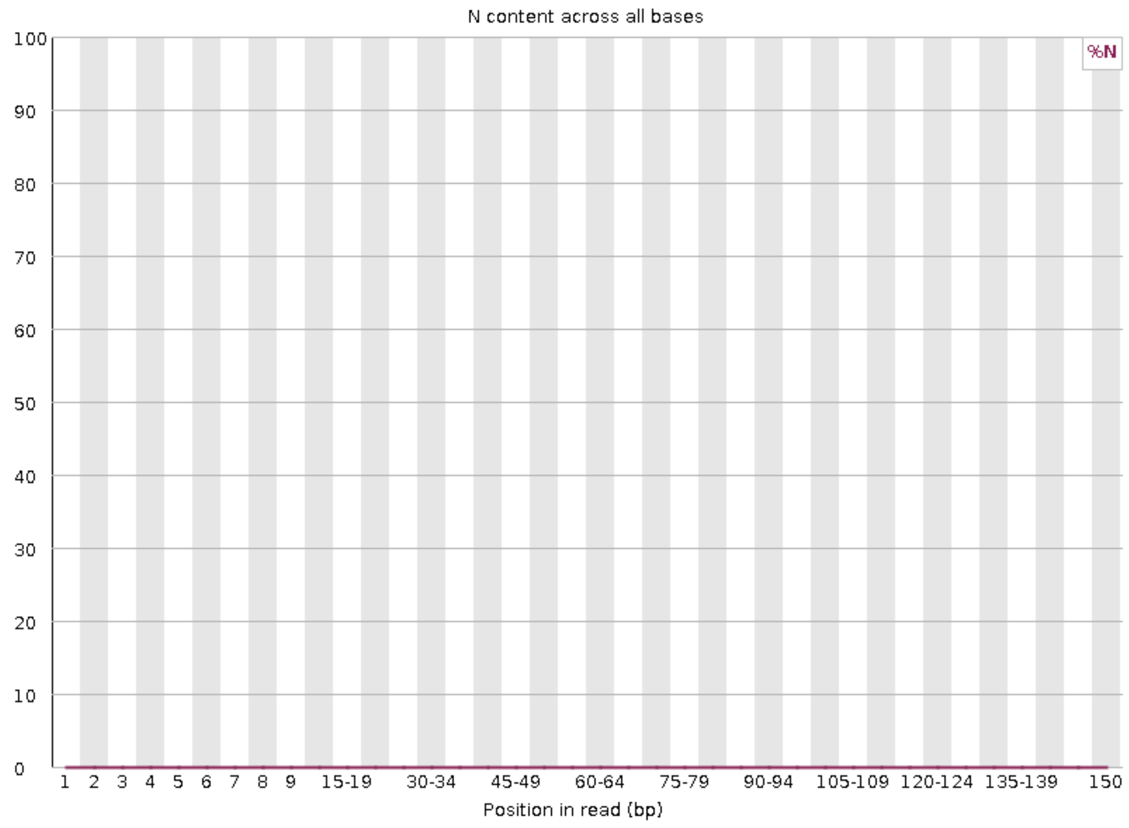


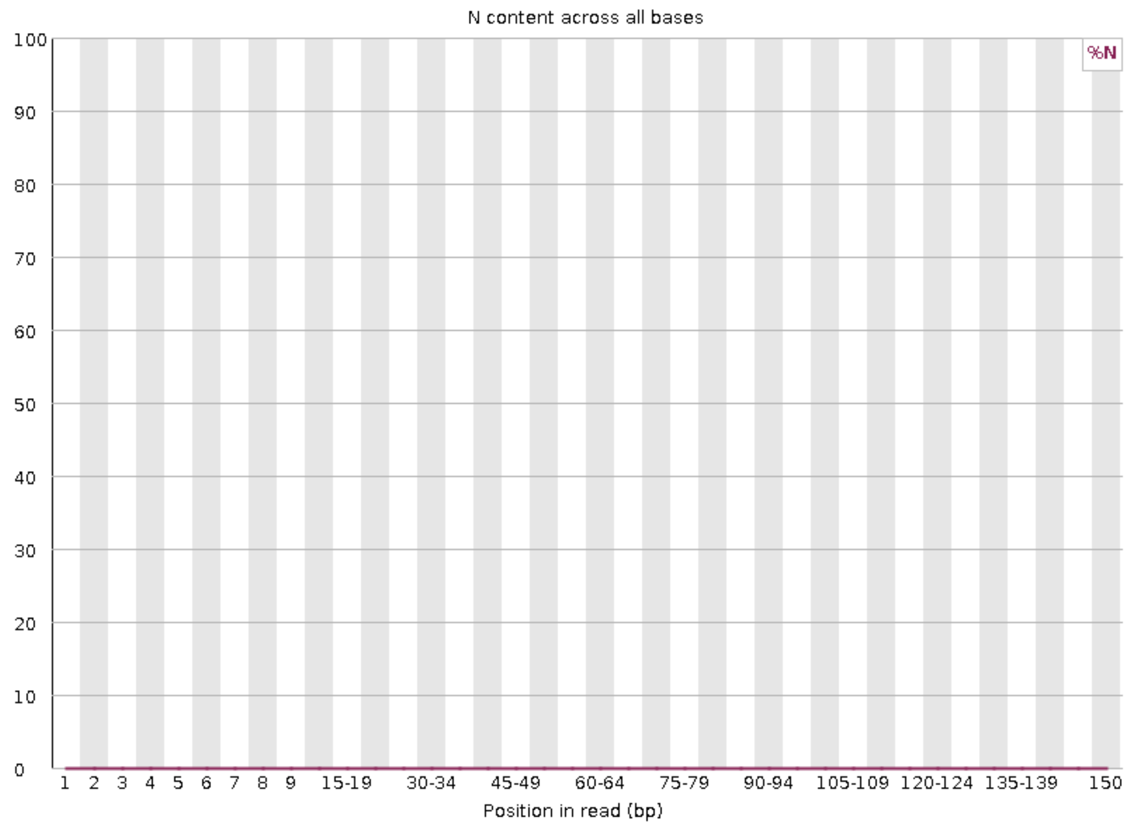

Quality scores across all bases (Sanger / Illumina 1.9 encoding)
36
34
32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
1 2 3 4 5 6 7 8 9 15-19 30-34 45-49 60-64 75-79 90-94 105-109 120-124 135-139 150
Position in read (bp)

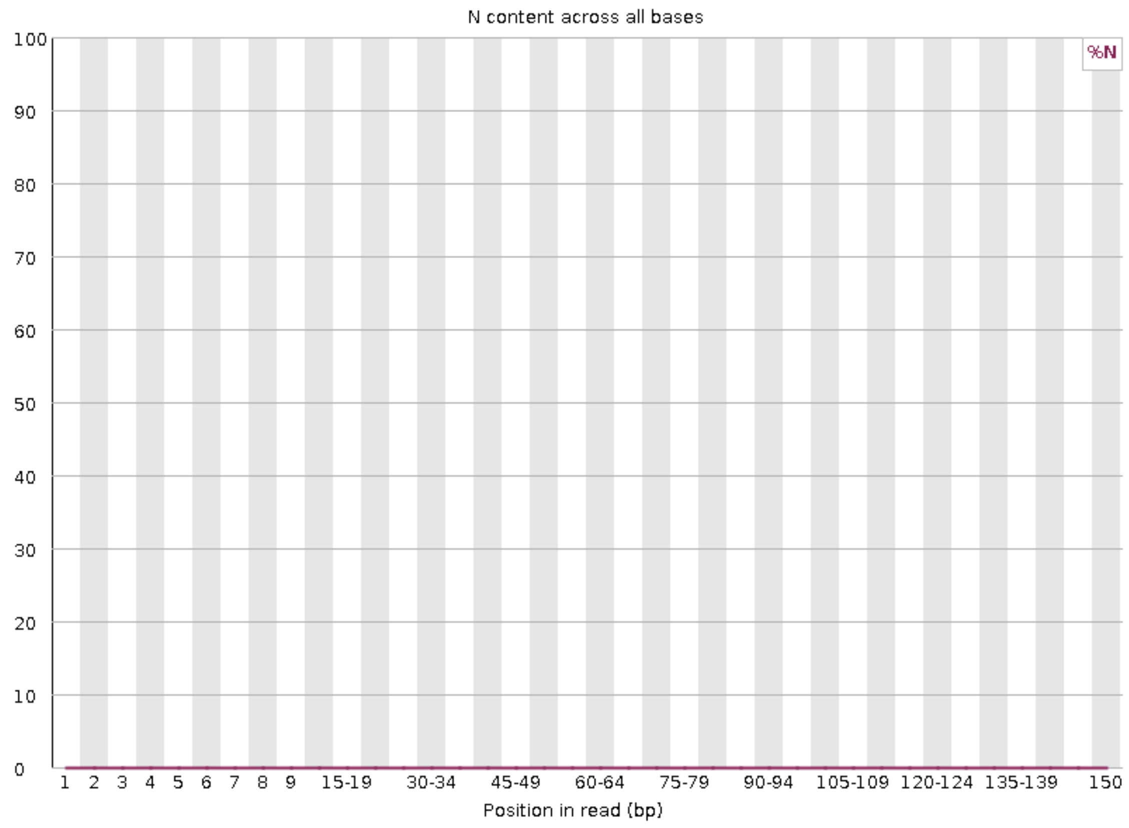

Quality scores across all bases (Sanger / Illumina 1.9 encoding)
36
34
32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
1 2 3 4 5 6 7 8 9 15-19 30-34 45-49 60-64 75-79 90-94 105-109 120-124 135-139 150
Position in read (bp)

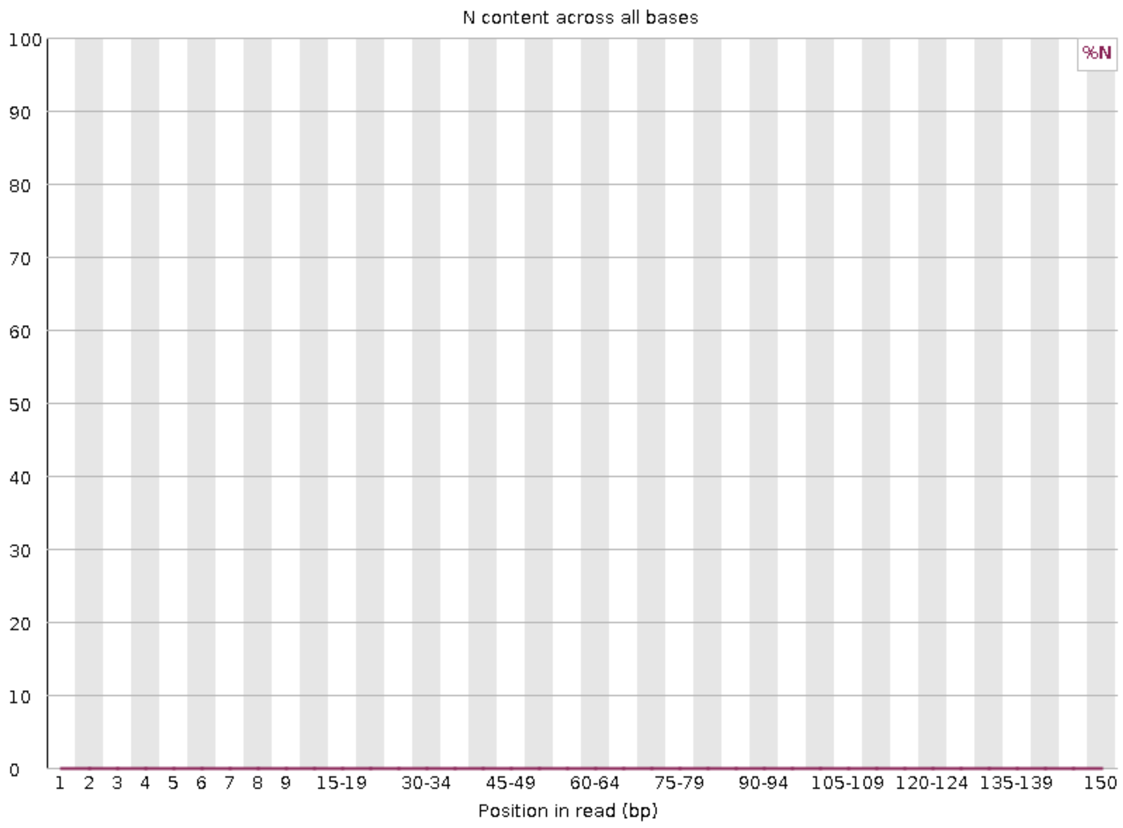

Quality scores across all bases (Sanger / Illumina 1.9 encoding)
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
1
2
3
4
5
6
7
8
9
15-19
30-34
45-49
60-64
75-79
90-94
105-109
120-124
135-139
150
Position in read (bp)

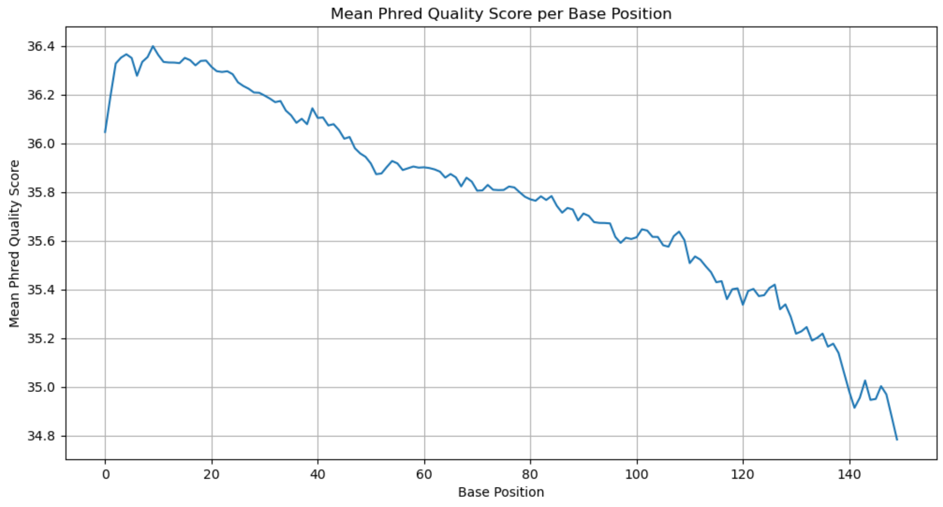
1.2 Per-base N content (FastQC)

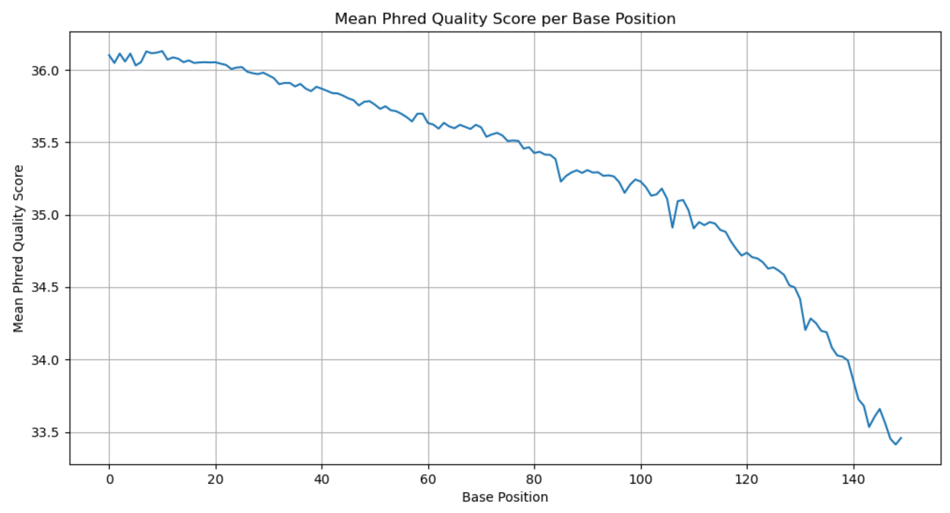


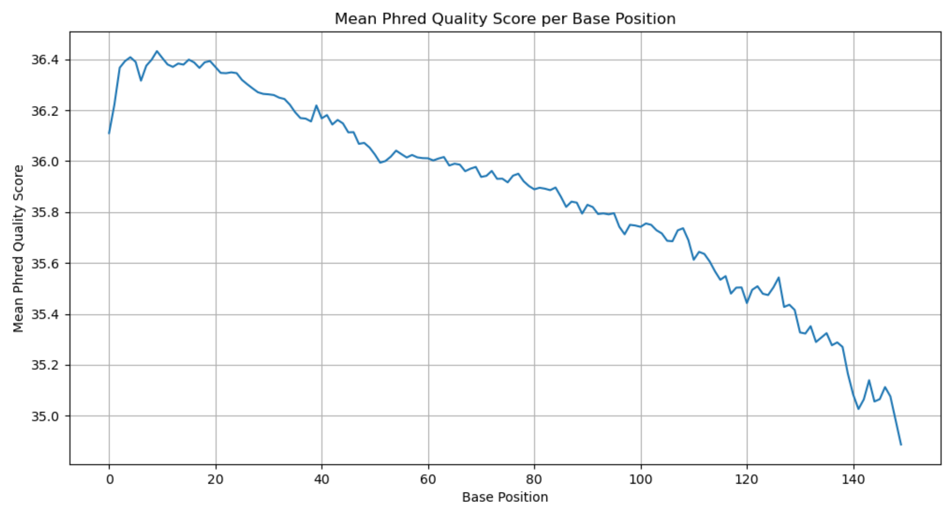

N content across all bases
%N
Position in read (bp)

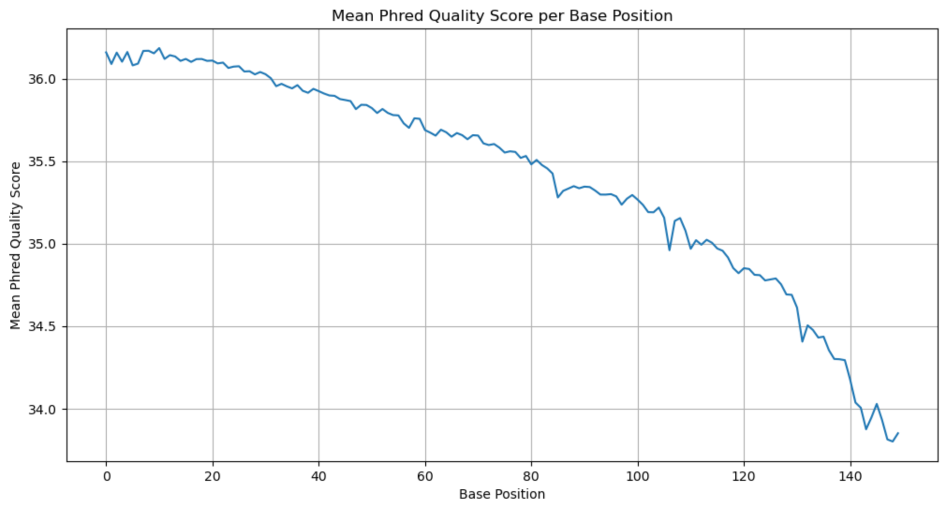

N content across all bases
%N
Position in read (bp)


N content across all bases
%N
Position in read (bp)
0
10
20
30
40
50
60
70
80
90
100
1 2 3 4 5 6 7 8 9 15-19 30-34 45-49 60-64 75-79 90-94 105-109 120-124 135-139 150


Mean Phred Quality Score per Base Position
Mean Phred Quality Score
Base Position
34.8
35.0
35.2
35.4
35.6
35.8
36.0
36.2
36.4
0 20 40 60 80 100 120 140


Mean Phred Quality Score per Base Position
Mean Phred Quality Score
Base Position


Mean Phred Quality Score per Base Position
Mean Phred Quality Score
Base Position



The near-zero N content is consistent with the high base-quality profiles. Bases are confidently called (not "N") across the reads, including at the ends where quality dips slightly but remains high.

The FastQC quality plots and my plots are similar. They all have very high scores at the beginning of that reads (~Q36) with a gradual decline toward the end of the reads, and R2 tails off slightly more than R1. Per-base N content is ~0% in FastQC, which matches the high quality seen in both sets of plots. Small visual differences are expected because FastQC shows median with IQR (box-and-whiskers) while mine plot the mean, FastQC bins later cycles (e.g., 105–109) which smooths the right side. Overall, the results are consistent; any offsets are methodological, not biological. FastQC ran a bit faster because it's an optimized, compiled Java program, whereas my Python script is single-threaded and spends more time in gzip decompression and Python loops. FastQC typically uses more CPU and RAM but finishes sooner; the Python script uses less CPU/RAM but takes longer.

Both libraries meet the bar for downstream RNA-seq. Quality scores are high across most bases with only a small drop at the end. The N content is ~0% across positions, the Adapter Content panel doesn't show a worrying rise, there aren't big unknown "overrepresented" sequences, duplication levels are typical for RNA-seq (representing highly expressed transcripts), the GC content looks close to what we expect, and the read lengths are consistent. Overall, the data look clean and are good to use for downstream analysis.

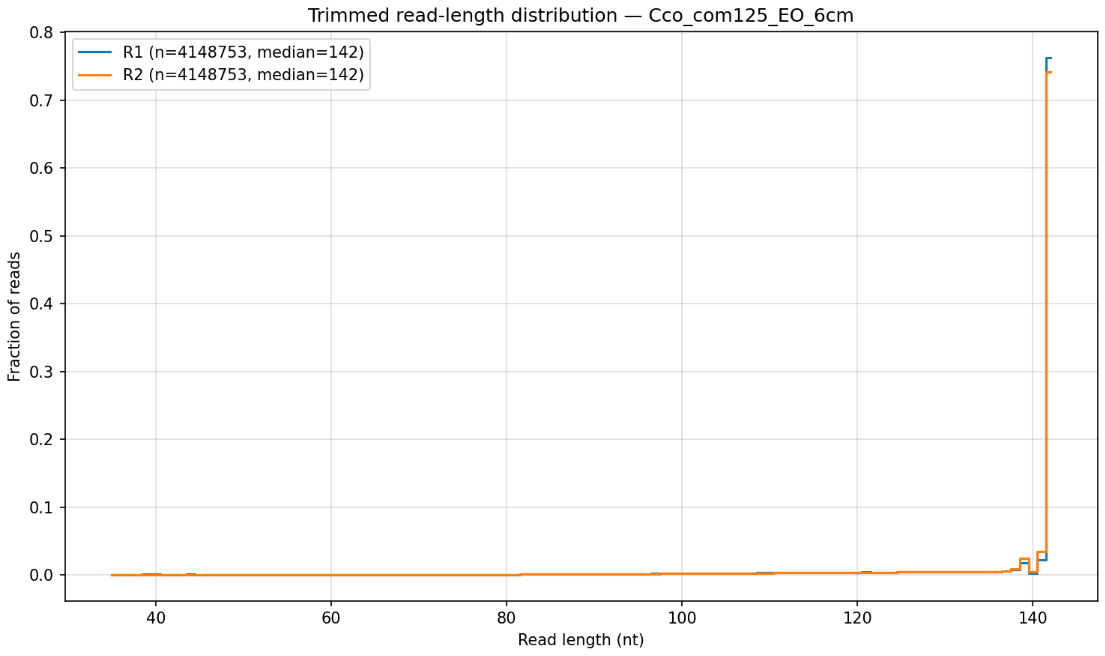
Part 2

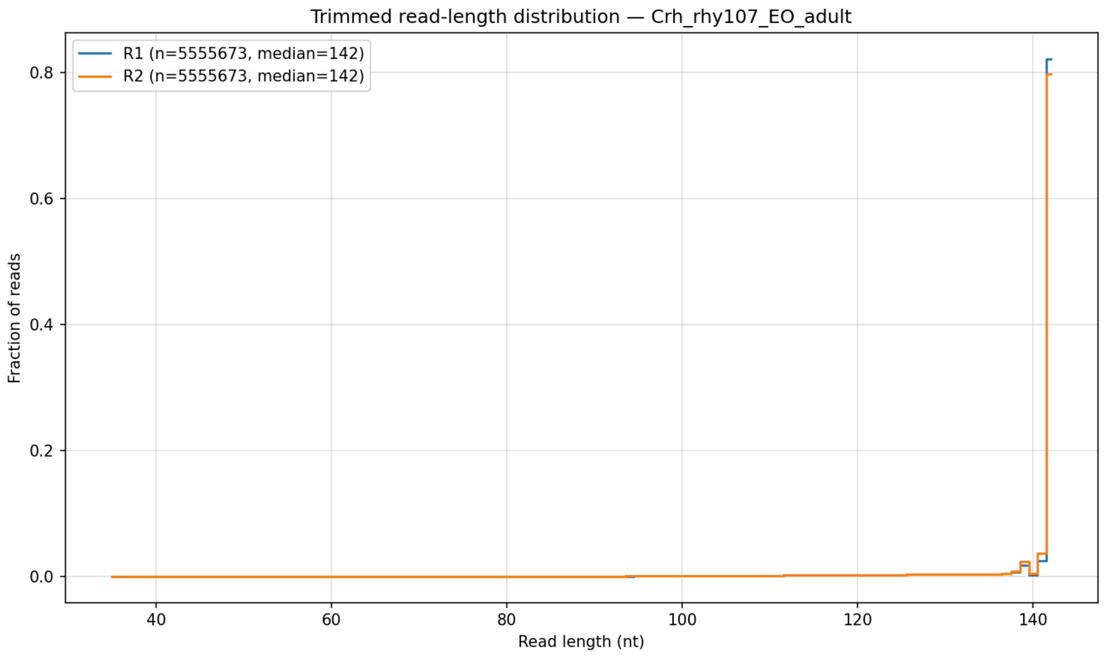
Proportion of reads trimmed by Cutadapt:

Sample	Read	% Trimmed
Cco_com125_EO_6cm	R1	18.2%
	R2	18.5%

Sample	Read	% Trimmed
Crh_rhy107_EO_adult	R1	12.5%
	R2	13.0%

Trimmed read-length distributions (R1 vs R2 overlaid)





R1 and R2 look basically the same. Both end up around 142 nt, so most trimming came from the fixed HEADCROP:8, not from cutting off adapters. In general, R1 and R2 should have about the same amount of adapter trimming because adapters show up when the DNA/RNA fragment is shorter than the read, which affects both reads equally. Any tiny differences are usually just because R2's end is a bit lower quality, not because it has more adapters.

Part 3

I had to copy the Campylomormyrus compressirostris genome fasta and gff file from Talapas because the Dryad download wouldn't work for me. I kept getting a "ERROR 403: Forbidden." error. I installed the gffread package and used it to transform the gff file into a gtf. Then I created two STAR databases by using the fasta and gtf files (one for each sample needing alignment). Next I used STAR alignment to align the Cco_com125_EO_6cm and Crh_rhy107_EO_adult paired and trimmed fastq files to the two STAR databases I created.

Mapped / Unmapped Read Counts (PS8)

Sample	File	Unique mapped	Unique unmapped
`Cco_com125_EO_6cm`	`Cco_com125_EO_6cm.out.sorted.rmdup.sam`	1,903,088	958,755
`Crh_rhy107_EO_adult`	`Crh_rhy107_EO_adult.out.sorted.rmdup.sam`	2,855,485	913,124

HTseq-count counts per file

calc	assigned	total	percent
Cco_com125_EO_6cm Stranded = yes	93,309	4,508,188	2.07%
Cco_com125_EO_6cm Stranded = reverse	1,942,036	4,508,188	43.08%
Crh_rhy107_EO_adult Stranded = yes	153,957	6,360,672	2.42%
Crh_rhy107_EO_adult Stranded = reverse	3,344,574	6,360,672	52.58%

These data are strand-specific (reverse-stranded). With htseq-count, –stranded=reverse assigned 43.08% of reads to genes vs 2.07% with –stranded=yes for the Cco_com125_EO_6cm dataset. With htseq-count, –stranded=reverse assigned 52.58% of reads to genes vs 2.42% with –stranded=yes for the Crh_rhy107_EO_adult dataset. Given the kit's dUTP chemistry, this matches expectations. We will use htseq-count –stranded=reverse for downstream differential expression.