Illinois Institute of Technology

Long Assignment Report

BIOL 550: BIOINFORMATICS
Fall 2022

Instructed by : Prof. Dr. Jean-François Pombert

Submitted by :

Kajol Shah - A20496724
Yalitza Jimenez - A20488132
Johanna Steinbrecher - A20474010

Problem Statement: We are given a genomic dataset from collaborators consisting of Illumina (paired ends) and Oxford Nanopore sequencing data and we need to assemble and annotate the genome. Collaborators are expecting data from a *Staphylococcus aureus* strain.

Part I. Read Processing

i. - Investigate read quality of the Illumina and Oxford Nanopore datasets

A FastQC report provides an overview of the basic quality control of a sequence data set. The per base sequence displays where the distribution of quality scores at each position are read, providing an overview of the overall sequence. It is also assorted in color, such as the low quality in red, medium quality in orange, and high quality in green. Per sequence quality score assists in providing the average quality score with the number of sequences in the data set. As for per base sequence content and GC content, it displays the proportion of each base position within each of the four DNA bases and the overall GC distribution of the sequence. In sequence length distribution, plot generates sequence fragments that are placed in uniform length, but some may vary in length. As for the overrepresented sequence, the table will assist in portraying any contaminants that have been identified within the sequence. Moreover, each genomic data set, Illumina and Oxford Nanopore, underwent through a FastQC report in order to identify the quality of the overall sequence.

▪ Illumina Read Quality Results

"FastQC" command was applied to the Illumina reads as below:

```
$ cp -r /media/Data_1/F22_BIOL550_LA /home/F22_T01
$ chmod 777 F22_BIOL550_LA
$ chmod +x *fastq.gz
$ fastqc -t 4 *.fastq.gz
```

The command execution process and output were as follows:

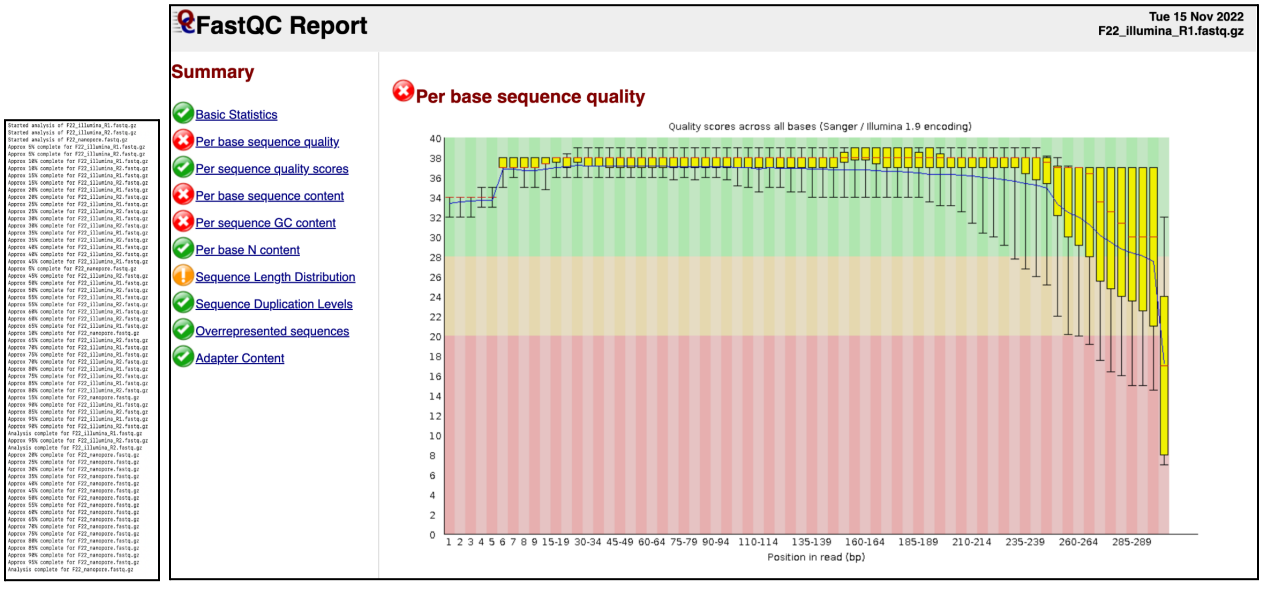


Figure 1. FastQC report results for R1 Illumina

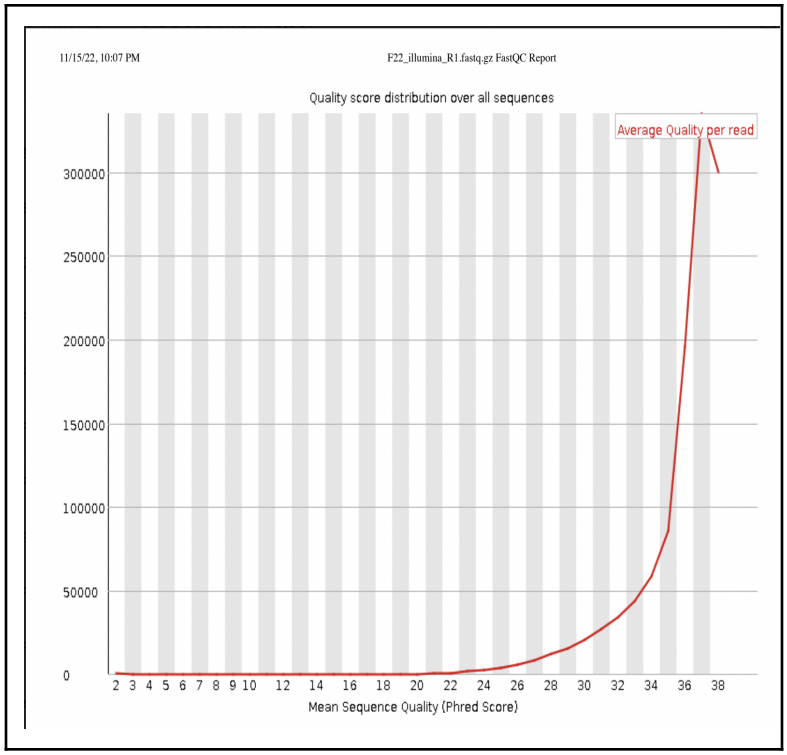


Figure 2. FastQC for R1 Illumina - Sequence Quality

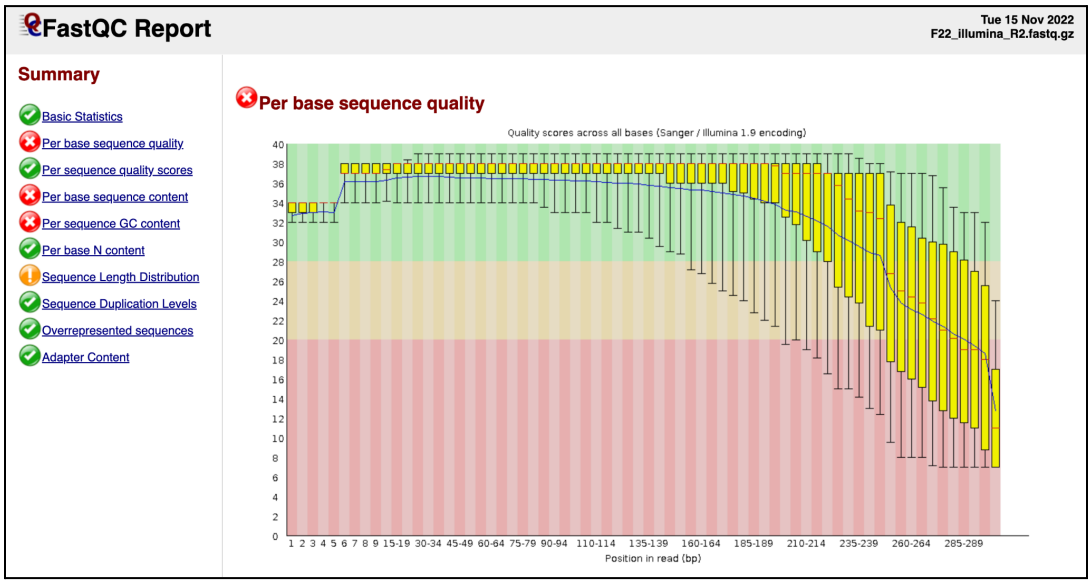


Figure 3. FastQC report results for R2 Illumina

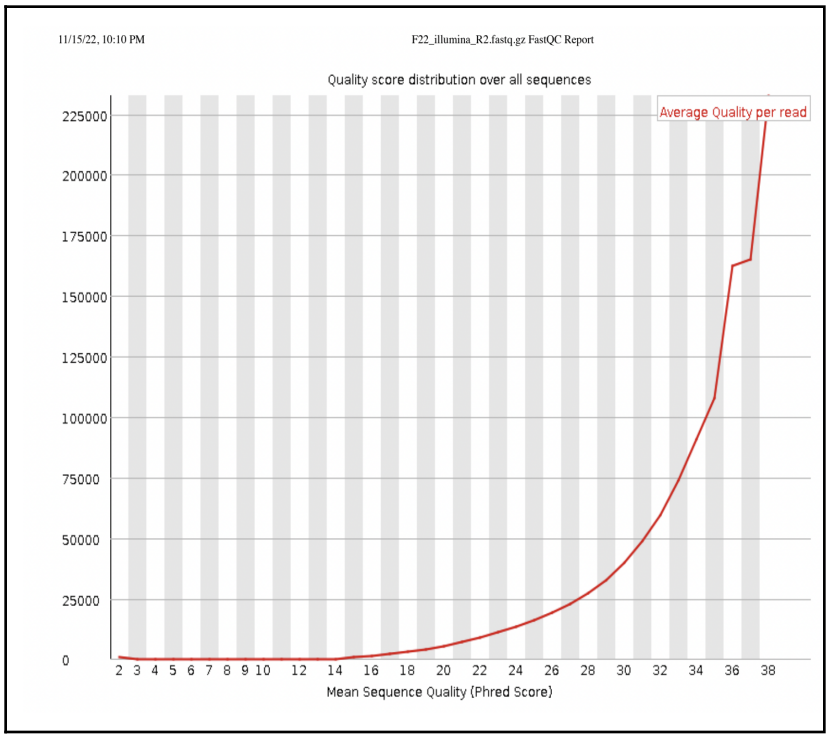


Figure 4. FastQC for R2 Illumina - Sequence Quality

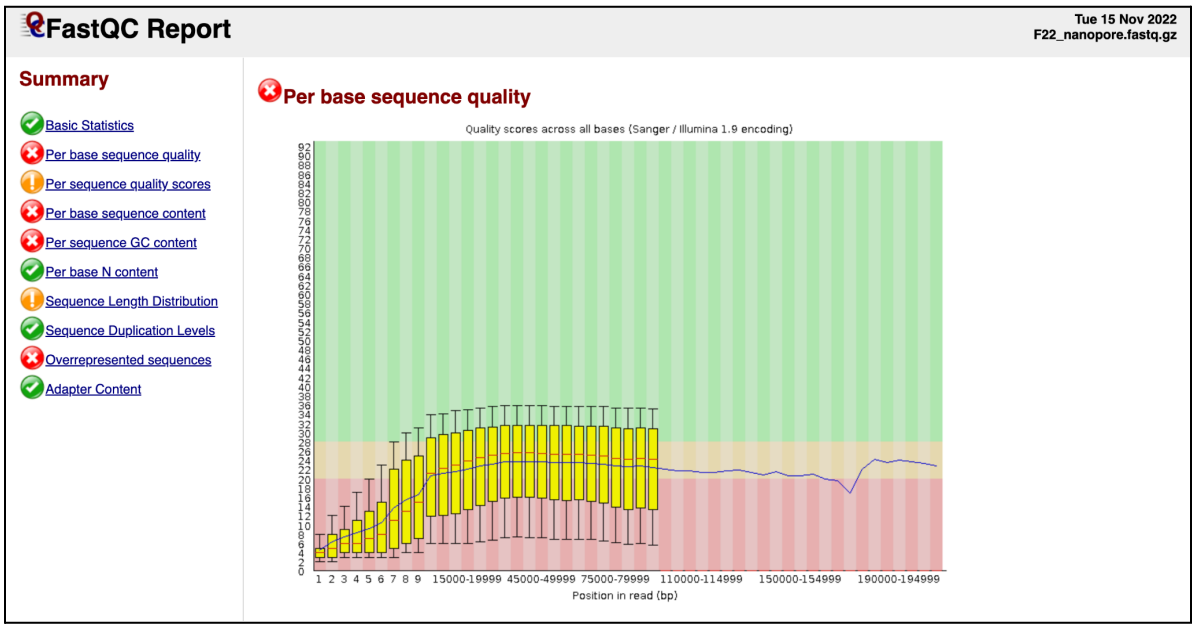


Figure 5. FastQC report results for Nanopore

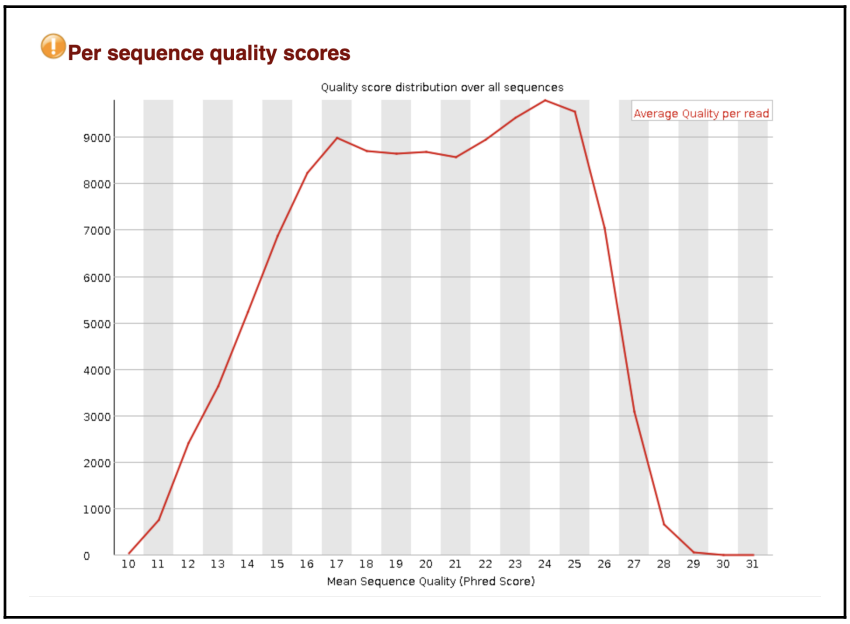


Figure 6. FastQC for Nanopore - Sequence Quality

In the FastQC files, both Illumina R1 and R2 files, *figure 1* and *figure 3*, needed adjustments in the categories of per base sequence quality, base sequence content, sequence GC content, and sequence length distribution. Nanopore, shown in *figure 5* also portrayed similar categories that need adjustment, in addition to sequence quality score and overrepresented sequences. Adapter Content for both Illumina and Nanopore datasets was in green, which means it was fine. Also, after observing the mean sequence quality of both the datasets, to improve the quality of reads, we used Nanofilt and Fastp for Nanopore and Illumina datasets respectively and filtered it by using a good quality score.

ii. - Plot the read length distribution / calculate the metrics of the nanopore dataset.

▪ Length Distribution and Metrics of Nanopore Dataset

In order to determine the length distribution, the "read_len_plot_.py" script was applied, with resulting outputs captured below. As expected, the length of the nanopore reads was much longer than illumina (The N50 score for nanopore was colossal compared to illumina: nanopore was ~10K whereas illumina did not exceed 300). The total number of reads for illumina was much greater than nanopore.

```
$ read_len_plot.py -f F22_nanopore.fastq.gz
```

Output:

```
[biF22_14@Mozart F22_BIOL550_LA]$ read_len_plot.py -f F22_nanopore.fastq.gz

Working on F22_nanopore.fastq.gz...

Metrics for F22_nanopore.fastq.gz:

Total bases:    1,092,535,846
# reads:        119,363
Longest:        206,017
Shortest:       1,000
Average:        9,153
Median:         15,721
N50:            10,530
N75:            7,834
N90:            5,732

[biF22_14@Mozart F22_BIOL550_LA]$
```

$ read_len_plot.py -f F22_illumina_R1.fastq.gz

Output:

```
[biF22_14@Mozart F22_BIOL550_LA]$ read_len_plot.py -f F22_illumina_R1.fastq.gz

Working on F22_illumina_R1.fastq.gz...

Metrics for F22_illumina_R1.fastq.gz:

Total bases:    290,576,132
# reads:        1,161,675
Longest:        300
Shortest:       35
Average:        250
Median:         299
N50:            275
N75:            251
N90:            209

[biF22_14@Mozart F22_BIOL550_LA]$
```

$ read_len_plot.py -f F22_illumina_R2.fastq.gz

Output:

```
[biF22_14@Mozart F22_BIOL550_LA]$ read_len_plot.py -f F22_illumina_R2.fastq.gz

Working on F22_illumina_R2.fastq.gz...

Metrics for F22_illumina_R2.fastq.gz:

Total bases:    291,401,774
# reads:        1,161,675
Longest:        300
Shortest:       35
Average:        251
Median:         299
N50:            281
N75:            251
N90:            210

[biF22_14@Mozart F22_BIOL550_LA]$
```

iii. - If required (and possible), filter accordingly.

▪ Applying Nanofilt to Nanopore Dataset

In order to improve the average quality of the set of reads, the Nanofilt command was applied, experimenting with different quality scores such that the quality did in fact improve, but not at the expense of filtering out all of the reads. Initially, we started with a basic quality score of 10 then opted for 30 and then 17. It was ultimately decided that 10 was the best as it did not filter too many results, and produced an assembly of expected size in the "assembly" step. After analyzing the FastQC result of the Nanopore dataset, we realized that the Per Base Sequence quality was in red. In order to improve the Per Base Sequence Quality, we used headcrop as 20.

NanoFilt command was used with the following parameters: -q 10, -q 30, and -q 17 in order to identify the differences when the quality score was changed.
An initial FastQC report indicated that the nanopore results were poor, especially towards the beginning of the sequence.

```
→ $ gunzip -c F22_nanopore.fastq.gz | NanoFilt -q 10 -l 500 --headcrop 20 | gzip > trimmed_F22_nanopore.fastq.gz
```

```
→ $ fastqc -t 4 trimmed_F22_nanopore.fastq.gz
```

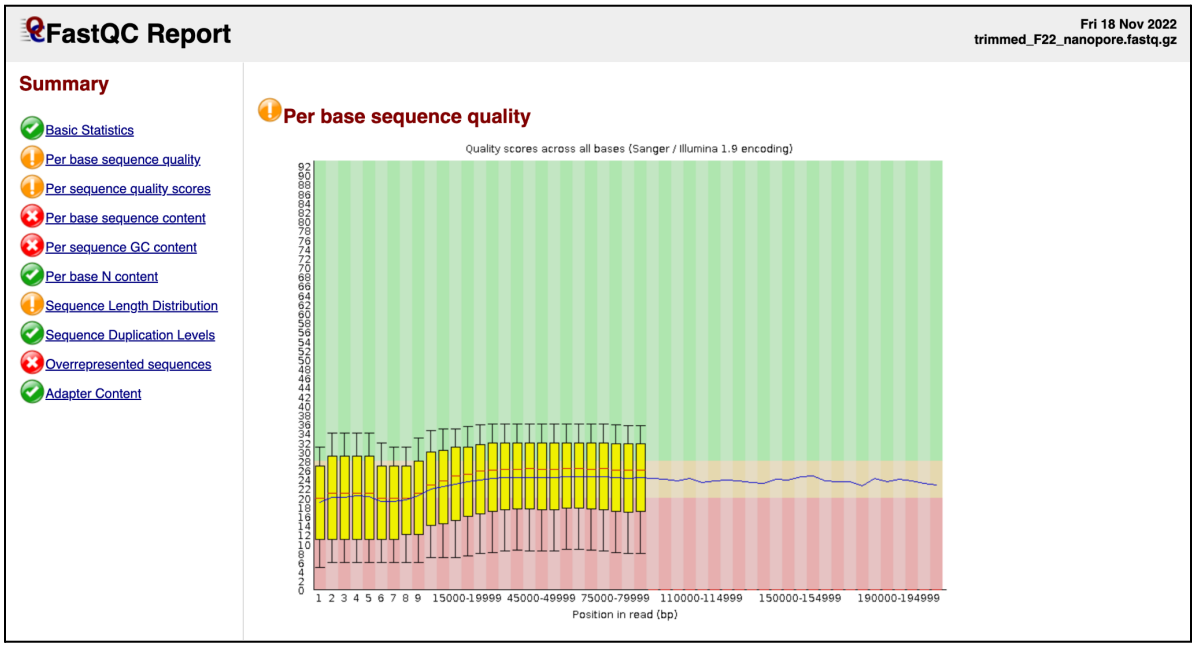


Figure 4. FastQC report results for Nanopore after NanoFilt -q 10 applied

```
→ $ gunzip -c F22_nanopore.fastq.gz | NanoFilt -q 30 -l 500 --headcrop 20 | gzip > trimmed_F22_nanopore_2.fastq.gz
```

After that, we again generated a FASTQC report of our trimmed dataset to check if the quality of sequence improved.

```
→ $ fastqc -t 4 trimmed_F22_nanopore_2.fastq.gz
```

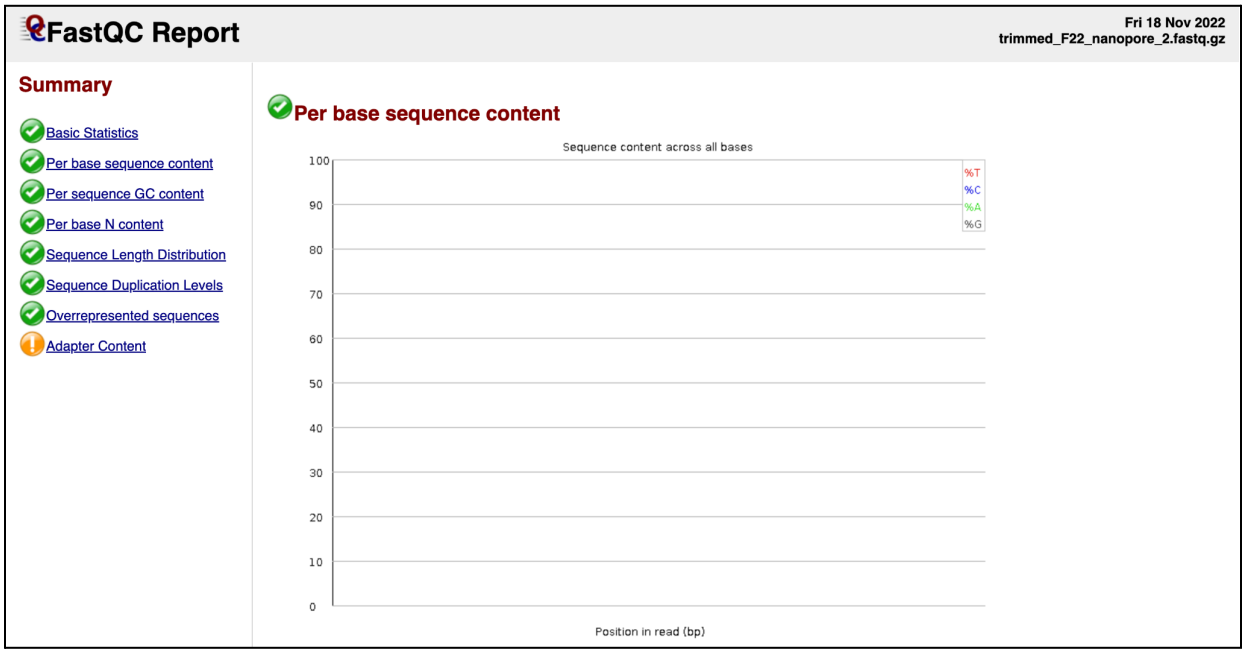


Figure 5. FastQC report results for Nanopore after NanoFilt -q 30 applied

```
→ $gunzip -c F22_nanopore.fastq.gz | NanoFilt -q 17 -l 500 --headcrop 20 | gzip >
trimmed_F22_nanopore_3.fastq.gz
```

```
→ $ fastqc -t 4 trimmed_F22_nanopore_3.fastq.gz
```

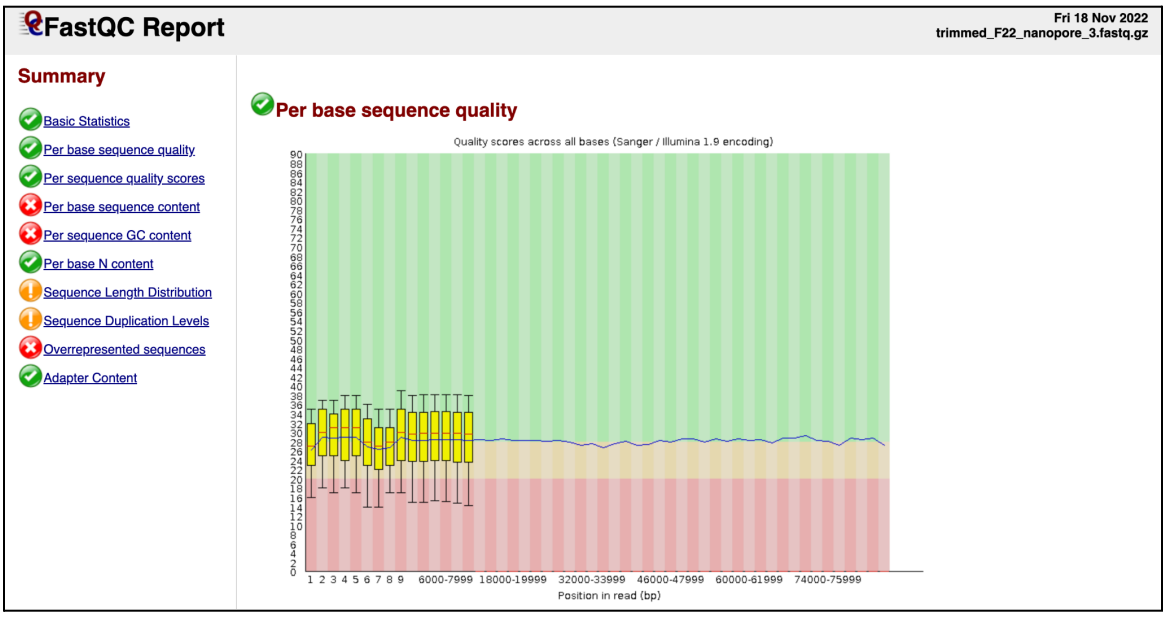


Figure 6. FastQC report results for Nanopore after NanoFilt -q 17 applied

When applying NanoFilt to the Nanopore datasets, each quality score provided a different set of results. In *figure 4*, the quality score was set at 10 and the majority of the poor quality sequences were removed from the beginning. As for *figure 5,* it displayed no results from the dataset and implied that the quality score of 30 was too high, and should not be used. As for the quality score of 17, we can see in *figure 6* that the dataset portrayed a set of good results, but most of the contents of the sequence were eliminated from the beginning, even dismissing more than what was needed. In the end, we decided to select the quality score of 10 as the best nanopore dataset undergoing Nanofilt.

For future use, we will be using “trimmed_F22_nanopore.fastq.gz” containing the quality score of 10

▪ **Applying Fastp to Illumina R1 and R2 Datasets**

Fastp was applied to the Illumina reads in a similar manner to Nanofilt in order to improve the average quality of the reads. For Illumina datasets, a good quality score is 30. So, we used a quality score to filter Illumina files using Fastp.

The fastp command was used with the following options:

fastp -i -o for the fastq files; fastqc reports were generated for the output file to ensure quality was acceptable.

```
→ $ fastp -w 10 -i F22_illumina_R1.fastq.gz -I F22_illumina_R2.fastq.gz -o
F22_illumina_R1_output.fastq.gz -O F22_illumina_R2_output.fastq.gz -M 30 -r -l 100
```

Output:

```
[biF22_14@Mozart F22_BIOL550_LA]$ fastp -w 10 -i F22_illumina_R1.fastq.gz -I F22_illumina_R2.fastq.gz -o F22_illumina_R1_output.fastq.gz -O F22_illumina_R2_ou
tput.fastq.gz -M 30 -r -l 100
Read1 before filtering:
total reads: 1161675
total bases: 290576132
Q20 bases: 282074467(97.0742%)
Q30 bases: 267294522(91.9878%)

Read2 before filtering:
total reads: 1161675
total bases: 291401774
Q20 bases: 264647970(90.8189%)
Q30 bases: 239848822(82.3086%)

Read1 after filtering:
total reads: 757434
total bases: 178268170
Q20 bases: 178174991(99.9477%)
Q30 bases: 177338667(99.4786%)

Read2 after filtering:
total reads: 757434
total bases: 158013846
Q20 bases: 157828640(99.8828%)
Q30 bases: 156780349(99.2194%)

Filtering result:
reads passed filter: 1514868
reads failed due to low quality: 6
reads failed due to too many N: 0
reads failed due to too short: 808476
reads with adapter trimmed: 12320
bases trimmed due to adapters: 768030

Duplication rate: 0.451176%

Insert size peak (evaluated by paired-end reads): 251

JSON report: fastp.json
HTML report: fastp.html
```

After that, we again generated a FastQC report of our trimmed dataset to check if the quality of sequence improved.

```
→ $ fastqc -t 4 F22_illumina_R1_output.fastq.gz
→ $ fastqc -t 4 F22_illumina_R2_output.fastq.gz
```

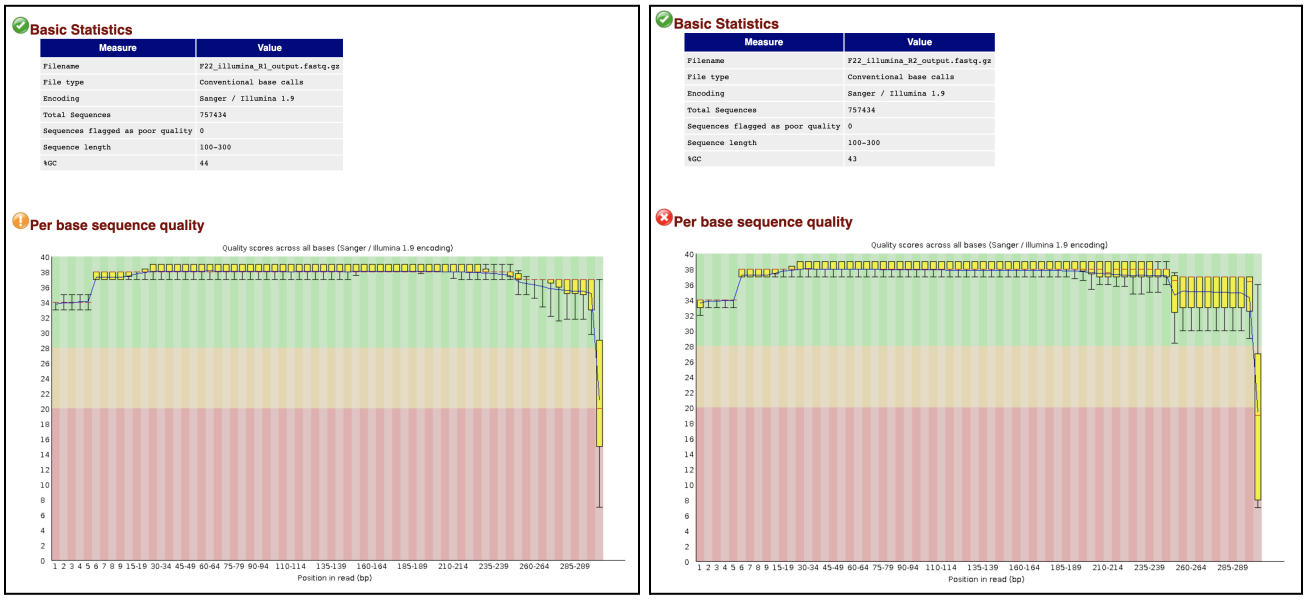


Figure 7. FastQC report results for Illumina R1 and R2 after Fastp applied

When running the Fastp command on the Illumina R1 and Illumina R2 files displayed acceptable results by trimming PE files in order to detect and remove any adapter sequences. When comparing *figure 1* and *figure 3* to *figure 7*, we can see that the trimming mostly occurred at the end of the sequence. We can hypothesize that Fastp removed any low quality bases from the 5' to 3' reading frame and corrected any mismatched base pairs in overlapped regions of the paired end reads found within the middle- towards the end area.

Part 2. Genome Assembly

Parts 2, 3, and 4 describe our process of assembling the reads and determining the optimal assembly to use for downstream analysis. To do this, different assembly methods were applied to the filtered read sets and produced assemblies were compared to each other using QUAST. SPAdes, Shasta, Fly and Unicycler assembly methods were utilized. Produced assemblies were further filtered using BLAST in order to determine which one was most congruent with being a *Staphylococcus*. It was ultimately determined that Unicycler produced the best assembly as it matched the expected size of 2.8 million, and QUAST output parameters of contig number and N50 value were the most optimized.

iv - Assemble each dataset using the screen command

▪ SPAdes - Illumina Assembly

Spades.py was executed using the following parameters

```
→ $ screen -S spades
```

```
→ $ spades.py -k 21,33,55,77 –careful –pe1-1 F22_illumina_R1_output.fastq.gz –pe2-2
F22_illumina_R2_output.fastq.gz -t 4 -o spades
```

▪ Shasta - Nanopore Assembly

"gunzip" command was used to extract the gzip file, followed by the shasta command with the following

→ $ gunzip -c trimmed_F22_nanopore.fastq.gz | shasta --input tnan3.fasta --assemblyDirectory SHASTA2 --config Nanopore-May2022 --Reads.minReadLength 10000

▪ **Flye - Nanopore Assembly**

→ $ flye -t 4 –nano-raw trimmed_F22_nanopore_3.fastq.gz -o flye_trimmed_F22_nanopore_3.fastq.gz
(nanopore file that has quality 17) - 43K in assembly.fasta

Output:

```
        Total length:   43167
        Fragments:      3
        Fragments N50:  15808
        Largest frg:    17270
        Scaffolds:      0
        Mean coverage:  3

[2022-11-21 18:53:17] INFO: Final assembly: /home/F22_T01/F22_BIOL550_LA/flye_trimmed_F22_nanopore_3.fastq.gz/assembly.fasta
```

Here, we can see that after using the nanopore dataset which was filtered with quality score 17, the length of our flye assembly file is too small. The expected genome size is 2.8M where we are only getting 43K with this dataset. This means that we may have overfiltered our dataset. So, we tried executing Flye with our unfiltered nanopore dataset to check for the results.

→ $ flye -t 4 –nano-raw F22_nanopore.fastq.gz -o flye_F22_nanopore.fastq.qz
Nanofilt -q 10 with 9.9M in assembly.fasta

```
[2022-11-21 20:07:04] INFO: Correcting bubbles
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
[2022-11-21 20:12:02] INFO: >>>STAGE: finalize
[2022-11-21 20:12:02] INFO: Assembly statistics:

        Total length:   10179866
        Fragments:      9
        Fragments N50:  2750242
        Largest frg:    4735760
        Scaffolds:      0
        Mean coverage:  99

[2022-11-21 20:12:02] INFO: Final assembly: /home/F22_T01/F22_BIOL550_LA/flye_F22_nanopore.fastq.gz/assembly.fasta
[biF22_09@Mozart F22_BIOL550_LA]$
```

After checking the above result of the original nanopore dataset, we decided to use the nanopore dataset which is filtered on a quality score of 10.

→ $ flye -t 4 --nano-raw F22_BIOL550_LA/trimmed_F22_nanopore.fastq.gz -o
→ $ flye_trimmed_F22_nanopore.fastq.gz

```
[biF22_14@Mozart F22_T01]$ ls
BLAST  contigs_files  F22_BIOL550_LA  flye_output  flye_trimmed_F22_nanopore.fastq.gz  Johanna  read_len_plot.py  test_kajol
[biF22_14@Mozart F22_T01]$ cd flye_trimmed_F22_nanopore.fastq.gz
[biF22_14@Mozart flye_trimmed_F22_nanopore.fastq.gz]$ ls
00-assembly  10-consensus  20-repeat  30-contigger  40-polishing  assembly.fasta  assembly_graph.gfa  assembly_graph.gv  assembly_info.txt  flye.log  params.json
[biF22_14@Mozart flye_trimmed_F22_nanopore.fastq.gz]$ ls -lh
total 20M
drwxr-xr-x. 1 biF22_14 biF22_14   88 Nov 27 18:15 00-assembly
drwxr-xr-x. 1 biF22_14 biF22_14   88 Nov 27 18:29 10-consensus
drwxr-xr-x. 1 biF22_14 biF22_14  232 Nov 27 18:37 20-repeat
drwxr-xr-x. 1 biF22_14 biF22_14  224 Nov 27 18:37 30-contigger
drwxr-xr-x. 1 biF22_14 biF22_14  298 Nov 27 18:53 40-polishing
-rw-r--r--. 1 biF22_14 biF22_14 9.9M Nov 27 18:53 assembly.fasta
-rw-r--r--. 1 biF22_14 biF22_14 9.7M Nov 27 18:53 assembly_graph.gfa
-rw-r--r--. 1 biF22_14 biF22_14 1.5K Nov 27 18:53 assembly_graph.gv
-rw-r--r--. 1 biF22_14 biF22_14  336 Nov 27 18:53 assembly_info.txt
-rw-r--r--. 1 biF22_14 biF22_14  72K Nov 27 18:53 flye.log
-rw-r--r--. 1 biF22_14 biF22_14   92 Nov 27 18:53 params.json
[biF22_14@Mozart flye_trimmed_F22_nanopore.fastq.gz]$
```

Now, we can see that the assembly size is 9.9M for the nanopore dataset (assembly.fasta file). The genome size seems to be fine now.

▪ Unicycler - Illumina & Nanopore Hybrid Assembly

```
→ $ unicycler -t 8 -1 F22_illumina_R1_output.fastq.gz -2 F22_illumina_R2_output.fastq.gz
--pilon_path /opt/pilon/pilon-1.22.jar -l trimmed_F22_nanopore.fastq.gz -o
hybrid_assembly_nov28 --spades_path /opt/SPAdes-3.13.0-Linux/bin/spades.py
```

```
Polishing assembly with Pilon (2022-11-28 16:11:16)
    Unicycler now conducts multiple rounds of Pilon in an attempt to repair any remaining small-scale errors with the assembly.

Aligning reads to find appropriate insert size range...
Insert size 1st percentile:  118
Insert size 99th percentile: 821

Pilon polish round 1
Total number of changes: 119

Pilon polish round 2
Total number of changes: 26

Pilon polish round 3
Total number of changes: 3

Pilon polish round 4
No Pilon changes

Pilon polish round 5
Total number of changes: 1

Pilon polish round 6
No Pilon changes

Saving /home/F22_T01/F22_BIOL550_LA/hybrid_assembly_nov28/006_polished.gfa


Rotating completed replicons (2022-11-28 16:47:57)
    Any completed circular contigs (i.e. single contigs which have one link connecting end to start) can have their start position changed without altering the sequence. For consistency, Unicycler now
searches for a starting gene (dnaA or repA) in each such contig, and if one is found, the contig is rotated to start with that gene on the forward strand.

Segment   Length      Depth    Starting gene     Position    Strand    Identity   Coverage
      1   4,736,463    1.00x   UniRef90_Q8XBZ3   4,535,954   reverse      99.6%     100.0%
      2   2,751,675    1.49x   UniRef90_B5WZP1     663,166   reverse      98.0%     100.0%
      5       6,200    1.42x   none found
      6       5,472   79.42x   none found

Saving /home/F22_T01/F22_BIOL550_LA/hybrid_assembly_nov28/007_rotated.gfa


Assembly complete (2022-11-28 16:52:07)
Saving /home/F22_T01/F22_BIOL550_LA/hybrid_assembly_nov28/assembly.gfa
Saving /home/F22_T01/F22_BIOL550_LA/hybrid_assembly_nov28/assembly.fasta

[biF22_14@Mozart F22_BIOL550_LA]$ cd F22_BIOL550_LA
```

In the genome assembly, each assembly was run under different conditions while using 'screen'. The SPAdes assembly only allowed Illumina R1 and Illumina R2 files. Shasta and Flye assembly can only be used with the Nanopore dataset. Lastly, Unicycler assembly can be applied to both Illumina and Nanopore data sets.

Part 3. Contamination Removal of Genome Assembly and Final QUAST Analysis

▪ SPAdes - Illumina Assembly

A BLAST was applied to the SPAdes-produced assembly via the "runTaxonomizedBLAST.pl" and the following output was produced, (head and tail shown below)

Output:

```
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    195122  240478  99.101  45380  81515  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    47301   90975   99.206  43720  78769  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    280609  325023  98.145  44463  77453  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    117452  154157  99.521  36711  66809  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    161903  195415  97.728  33579  57690  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    324959  357733  98.201  32794  57230  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    93073   117244  99.313  24173  43716  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    417033  436264  99.241  19232  34703  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    255092  273623  97.127  18549  31272  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
NODE_1_length_436264_cov_34.891232      gi|88193823|ref|NC_007795.1|    406661  417045  98.547  10393  18347  0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
```

```
NODE_1962_length_81_cov_85.000000       gi|88193823|ref|NC_007795.1|    1       81      100.000 81      150     1.28e-33        93061   Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
NODE_1963_length_81_cov_51.750000       gi|1447699251|ref|NC_002695.2|  1       81      100.000 81      150     1.28e-33        386585  Escherichia coli O157:H7 str. Sakai     Bacteria        enterobacteria
NODE_1964_length_81_cov_23.000000       gi|556503834|ref|NC_000913.3|   1       81      100.000 81      150     1.28e-33        511145  Escherichia coli str. K-12 substr. MG1655       Bacteria        enterobacteria
NODE_1965_length_81_cov_22.750000       gi|985817611|ref|NZ_AP014857.1| 1       81      100.000 81      150     1.28e-33        208962  Escherichia albertii    Bacteria        enterobacteria
NODE_1966_length_80_cov_27.000000       gi|1369006130|ref|NZ_CP027770.1|        1       80      100.000 80      148     4.50e-33        46127   Staphylococcus felis    Bacteria        firmicutes
NODE_1968_length_79_cov_89.000000       gi|88193823|ref|NC_007795.1|    4       79      100.000 76      141     7.38e-31        93061   Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
NODE_1969_length_79_cov_84.000000       gi|88193823|ref|NC_007795.1|    1       79      100.000 79      147     1.59e-32        93061   Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
NODE_1971_length_78_cov_63.000000       gi|88193823|ref|NC_007795.1|    1       78      100.000 78      145     5.59e-32        93061   Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
NODE_1972_length_78_cov_9.000000        gi|1369006130|ref|NZ_CP027770.1|        1       78      100.000 78      145     5.59e-32        46127   Staphylococcus felis    Bacteria        firmicutes
NODE_1973_length_78_cov_7.000000        gi|1369006130|ref|NZ_CP027770.1|        1       78      100.000 78      145     5.59e-32        46127   Staphylococcus felis    Bacteria        firmicutes
```

"parseTaxonomizedBLAST.pl" was applied to this output to produce the following fasta file, which was 4645182 base pairs long

▪ **Shasta - Nanopore Assembly**

A BLAST was applied to the Shasta-produced assembly via the runTaxonomizedBLAST.pl and the following output was produced, (head and tail shown below)

```
→ perl runTaxonomizedBLST.pl -t 4 -q SHASTA_ASSEMBLY.fasta -d
../REPGENOMES/ref_prok_rep_genomes
```

Output:

```
0       gi|88193823|ref|NC_007795.1|    1691527 1788485 99.457  97020   1.762e+05       0.0     93061       Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    1318728 1388757 99.275  70104   1.266e+05       0.0     93061       Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    1922318 1990390 99.465  68097   1.237e+05       0.0     93061       Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    1626151 1688991 99.380  62891   1.139e+05       0.0     93061       Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    317995  373487  99.258  55540   1.002e+05       0.0     93061       Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    935731  991069  99.305  55387   1.001e+05       0.0     93061       Staphylococcus aureus subsp. aureus NCTC 8325   Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    1107746 1159681 99.382  51981   94158   0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    2521500 2570752 99.645  49277   89993   0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    1448490 1497330 99.529  48869   88935   0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
0       gi|88193823|ref|NC_007795.1|    1251970 1300553 99.523  48597   88435   0.0     93061   Staphylococcus aureus subsp. aureus NCTC 8325       Bacteria        firmicutes
```

```
4       gi|31983523|ref|NC_004851.1|    110034  110110  96.104  77      126     5.75e-23        198214  Shigella flexneri 2a str. 301   Bacteria        enterobacteria
4       gi|431792069|ref|NC_019903.1|   120677  120786  99.091  110     198     1.20e-44        871963  Desulfitobacterium dichloroeliminans LMG P-21439        Bacteria        firmicutes
4       gi|1231405978|ref|NZ_NQBD01000075.1|    65096   65194   100.000 99      183     3.36e-40        624     Shigella sonnei Bacteria        enterobacteria
4       gi|1899485803|ref|NZ_NKHP01000008.1|    65855   66033   82.123  179     154     2.64e-31        333964  Xenorhabdus indica      Bacteria        enterobacteria
4       gi|1222082909|ref|NZ_FMBC01000087.1|    10077   10137   96.721  61      102     9.68e-16        1005667 Kosakonia oryziphila    Bacteria        enterobacteria
4       gi|1231405995|ref|NZ_NQBD01000092.1|    75545   75599   94.545  55      84.2    3.51e-10        624     Shigella sonnei Bacteria        enterobacteria
6       gi|1889771154|ref|NZ_AP022188.1|        1       1428    99.442  1433    2595    0.0     651     Aeromonas media Bacteria        g-proteobacteria
8       gi|1369006130|ref|NZ_CP027770.1|        1       9390    99.079  9447    16886   0.0     46127   Staphylococcus felis    Bacteria        firmicutes
10      gi|1369006130|ref|NZ_CP027770.1|        1       4495    99.490  4506    8172    0.0     46127   Staphylococcus felis    Bacteria        firmicutes
12      gi|1369006130|ref|NZ_CP027770.1|        1       639     99.844  639     1173    0.0     46127   Staphylococcus felis    Bacteria        firmicutes
```

"parseTaxonomizedBLAST.pl" was applied to this output to produce the following fasta file, which was 2811790 basepairs long

```
→ $ perl parseTaxonomizedBLST.pl -b SHASTA_ASSEMBLY.fasta.blastn.6 -f
SHASTA_ASSEMBLY.fasta -n "Staphylococcus" -c sscinames -o Staph_spades.fasta
```

▪ **Flye - Nanopore Assembly**

```
→ $ perl runTaxonomizedBLAST.pl -t 4 -q assembly.fasta -db REPGENOMES/ref_prok_rep_genomes
-o BLAST_assembly_fasta_flye
```

```
→ $ perl parseTaxonomizedBLAST.pl -v -b BLAST_assembly_fasta_flye/assembly.fasta.blastn.6
-f BLAST_assembly_fasta_flye/assembly.fasta -n "Staphylococcus aureus" Staphylococcus
aureus -c sscinames -o S_aureus.fasta
```

Output:

```
[biF22_09@Mozart BLAST]$ perl parseTaxonomizedBLAST.pl -v -b BLAST_assembly_fasta_flye/assembly.fasta.blastn.6 -f BLAST_assembly_fasta_flye/assembly.fasta -n "Staphylococcus aureus" Staphylococcus aureus -c sscinames -o S_aureus.fasta
Best hit for contig_1 = Escherichia coli str. K-12 substr. MG1655
Best hit for contig_2 = Shewanella bicestrii
Best hit for contig_3 = Staphylococcus felis
Match found for Staphylococcus: contig_3 gi|1369006130|ref|NZ_CP027770.1| 1 428318 99.562 429880 7.818e+05 0.0 46127 Staphylococcus felis Bacteria firmicutes
Best hit for contig_4 = Staphylococcus felis
Match found for Staphylococcus: contig_4 gi|1369006130|ref|NZ_CP027770.1| 175807 2000711 99.611 1831195 3.36e+06 0.0 46127 Staphylococcus felis Bacteria firmicutes
Best hit for contig_5 = Staphylococcus felis
Match found for Staphylococcus: contig_5 gi|1369006130|ref|NZ_CP027770.1| 1 10444 99.599 10475 19077 0.0 46127 Staphylococcus felis Bacteria firmicutes
Best hit for contig_6 = Staphylococcus felis
Match found for Staphylococcus: contig_6 gi|1369006130|ref|NZ_CP027770.1| 1 44371 99.737 44463 81358 0.0 46127 Staphylococcus felis Bacteria firmicutes
Best hit for contig_7 = Klebsiella sp. WCHK1090001
Best hit for contig_8 = Staphylococcus aureus subsp. aureus NCTC 8325
Match found for Staphylococcus: contig_8 gi|88193823|ref|NC_007795.1| 733207 830144 99.445 97016 1.761e+05 0.0 93061 Staphylococcus aureus subsp. aureus NCTC 8325 Bacteria firmicutes
Best hit for contig_9 = Shigella sonnei
[biF22_09@Mozart BLAST]$
```

▪ **Unicycler - Illumina & Nanopore Hybrid Assembly**

```
→ $ runTaxonomizedBLAST.pl -t 8 -p blastn -a megablast -db ref_prok_rep_genomes -q
../contigs_files/assembly_uni.fasta -e 1e-30 -c 1 -o Blast_results_un
```

Output:

```
[biF22_14@Mozart BLAST]$ runTaxonomizedBLAST.pl -t 8 -p blastn -a megablast -db ref_prok_rep_genomes -q ../contigs_files/assembly_uni.fasta -e 1e-30 -c 1 -o Blast_results_uni
Running blastn on ../contigs_files/assembly_uni.fasta against ref_prok_rep_genomes using 8 threads. This might take a while...
[biF22_14@Mozart BLAST]$ ls
assembly.fasta  assembly.fasta.blastn.6  BLAST_assembly_fasta_flye  Blast_results_uni  JS_B  parseTaxonomizedBLAST.pl  REPGENOMES  runTaxonomizedBLAST.pl
[biF22_14@Mozart BLAST]$ cd Blast_results_uni
[biF22_14@Mozart Blast_results_uni]$ ls
assembly_uni.blastn.6
[biF22_14@Mozart Blast_results_uni]$ cd ..
```

```
→ $ parseTaxonomizedBLAST.pl -v -b Blast_results_uni/assembly_uni.blastn.6 -f
../contigs_files/assembly_uni.fasta -n "Staphylococcus aureus" -c sscinames -o
S_aureus_uni.fasta
```

Output:

```
[biF22_14@Mozart BLAST]$ parseTaxonomizedBLAST.pl -v -b Blast_results_uni/assembly_uni.blastn.6 -f ../contigs_files/assembly_uni.fasta -n "Staphylococcus aureus" -c sscinames -o S_aureus_uni.fasta
Best hit for 1 = Escherichia coli str. K-12 substr. MG1655
Best hit for 2 = Staphylococcus aureus subsp. aureus NCTC 8325
Match found for Staphylococcus aureus: 2 gi|88193823|ref|NC_007795.1| 607234 704237 99.513 97017 1.765e+05 0.0 93061 Staphylococcus aureus subsp. aureus NCTC 8325 Bacteria firmicutes
Best hit for 3 = Shigella sonnei
Best hit for 4 = Staphylococcus intermedius NCTC 11048
Best hit for 5 = Proteus cibarius
Best hit for 6 = Aeromonas media
Best hit for 7 = Corynebacterium urealyticum
[biF22_14@Mozart BLAST]$ ls
BLAST_assembly_fasta_flye  Blast_results_uni  contigs.list  JS_B  parseTaxonomizedBLAST.pl  REPGENOMES  runTaxonomizedBLAST.pl  S_aureus_uni.fasta  Staph_SHASTA.fasta
[biF22_14@Mozart BLAST]$
```

Part 4. Genome Assembly Comparisons

vi - use Quast to compare assemblies before and after filtering and determine best assembly for downstream analysis; determine whether the size is appropriate; determine optimal sequencing approach

▪ QUAST Results for Pre-Filtered Assemblies

```
→ $ quast.py --min-contig 500 *.fasta -o quast_assemblies_4
```

Assembly.fasta → Flye
Assembly_uni.fasta → Unicycler
Contigs.fasta → Spades
SHASTA_assembly.fasta →Shasta

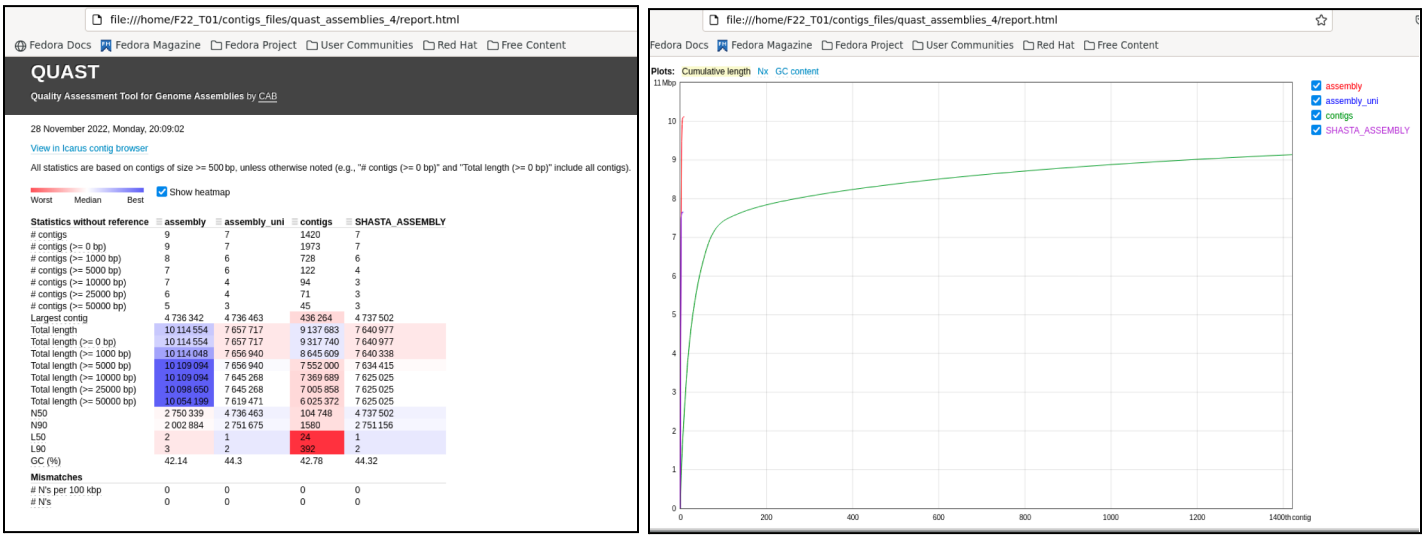
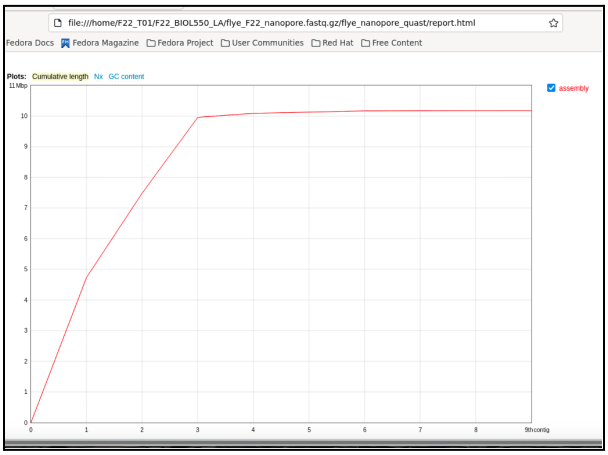


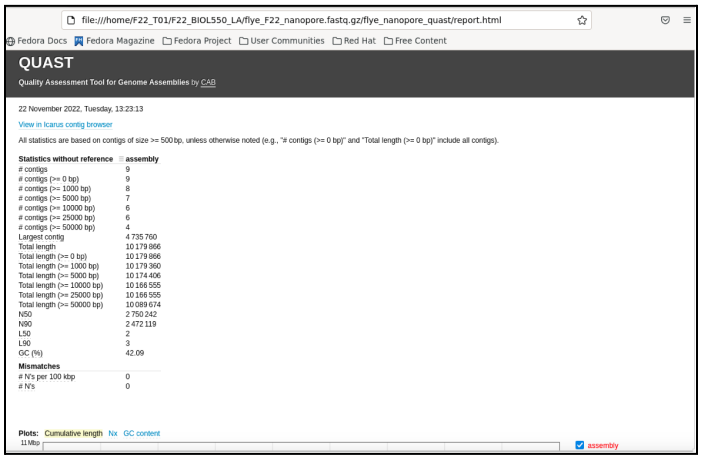
Figure 8. Quast report results for Flye, Unicycler, Spades, and Shasta assemblies after BLAST

▪ **QUAST Results for Processed Assemblies —-**

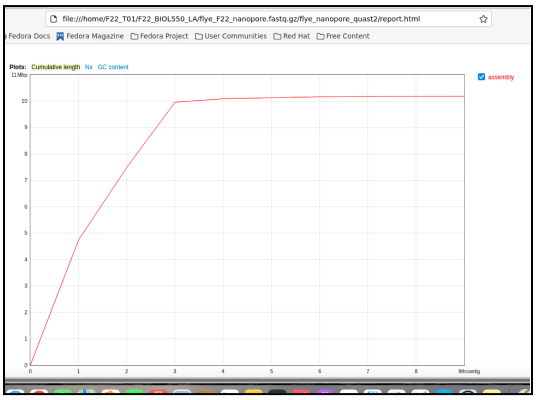
➔ **Flye vs. Quast (using the nanopore - unfiltered file)**

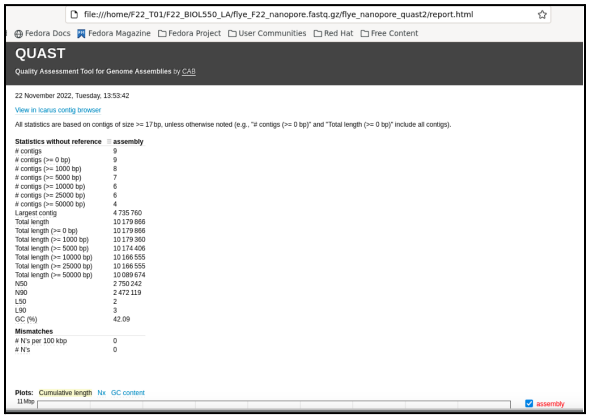
→ `$ quast.py --min-contig 500 assembly.fasta  -o flye_nanopore_quast`





→ `$ quast.py --min-contig 17 assembly.fasta  -o flye_nanopore_quast2`





In this section, we just wanted to do a test run on using Quast with the Flye assembly. We were wondering to see if the "min contig" differed when it was either restricted to 500 or 17 when being applied to any of the assemblies. In fact, we identified that 500 was set as the default when running Quast and using 17 did not display any significant differences in the results. In other words, "min contig" was set as 500 when running the rest of the assemblies: Shasta, spades, and Unicycler.

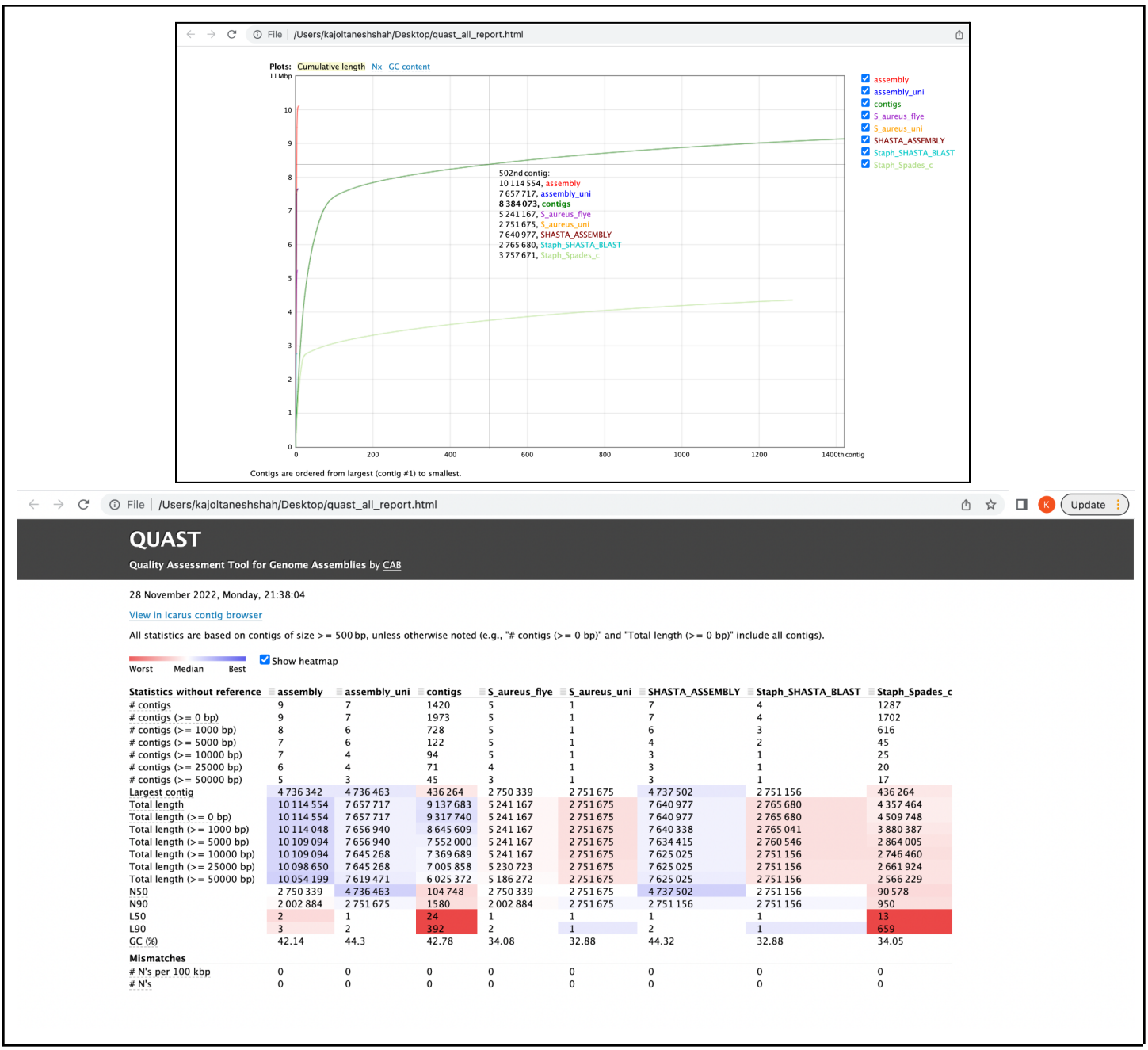
Running Quast on all the assemblies (before and after blast)

```
$ quast.py --min-contig 500 *.fasta -o quast_all_assemblies
```

Output:

```
[biF22_14@Mozart contigs_files]$ quast.py --min-contig 500 *.fasta -o quast_all_assemblies
/opt/quast-5.1/quast.py --min-contig 500 assembly.fasta assembly_uni.fasta contigs.fasta S_aureus_flye.fasta S_aureus_uni.fasta SHASTA_ASSEMBLY.fasta Staph_SHASTA_BLAST.fasta Staph_Sp
_all_assemblies

Version: 5.1.0rc1

System information:
  OS: Linux-5.15.16-100.fc34.x86_64-x86_64-with-glibc2.33 (linux_64)
  Python version: 3.9.9
  CPUs number: 64

Started: 2022-11-28 21:37:48

Logging to /home/F22_T01/contigs_files/quast_all_assemblies/quast.log
NOTICE: Output directory already exists and looks like a QUAST output dir. Existing results can be reused (e.g. previously generated alignments)!
NOTICE: Maximum number of threads is set to 16 (use --threads option to set it manually)

CWD: /home/F22_T01/contigs_files
Main parameters:
  MODE: default, threads: 16, min contig length: 500, min alignment length: 65, min alignment IDY: 95.0, \
  ambiguity: one, threshold for extensive misassembly size: 1000

Contigs:
  Pre-processing...
  1  assembly.fasta ==> assembly
  2  assembly_uni.fasta ==> assembly_uni
  3  contigs.fasta ==> contigs
  4  S_aureus_flye.fasta ==> S_aureus_flye
  5  S_aureus_uni.fasta ==> S_aureus_uni
  6  SHASTA_ASSEMBLY.fasta ==> SHASTA_ASSEMBLY
  7  Staph_SHASTA_BLAST.fasta ==> Staph_SHASTA_BLAST
  8  Staph_Spades_c.fasta ==> Staph_Spades_c

2022-11-28 21:37:49
Running Basic statistics processor...
  Contig files:
    1  assembly
    2  assembly_uni
    3  contigs
    4  S_aureus_flye
    5  S_aureus_uni
    6  SHASTA_ASSEMBLY
    7  Staph_SHASTA_BLAST
    8  Staph_Spades_c
  Calculating N50 and L50...
    1  assembly, N50 = 2750339, L50 = 2, Total length = 10114554, GC % = 42.14, # N's per 100 kbp =  0.00
    2  assembly_uni, N50 = 4736463, L50 = 1, Total length = 7657717, GC % = 44.30, # N's per 100 kbp =  0.00
    3  contigs, N50 = 104748, L50 = 24, Total length = 9137683, GC % = 42.78, # N's per 100 kbp =  0.00
    4  S_aureus_flye, N50 = 2750339, L50 = 1, Total length = 5241167, GC % = 34.08, # N's per 100 kbp =  0.00
    5  S_aureus_uni, N50 = 2751675, L50 = 1, Total length = 2751675, GC % = 32.88, # N's per 100 kbp =  0.00
    6  SHASTA_ASSEMBLY, N50 = 4737502, L50 = 1, Total length = 7640977, GC % = 44.32, # N's per 100 kbp =  0.00
    7  Staph_SHASTA_BLAST, N50 = 2751156, L50 = 1, Total length = 2765680, GC % = 32.88, # N's per 100 kbp =  0.00
    8  Staph_Spades_c, N50 = 90578, L50 = 13, Total length = 4357464, GC % = 34.05, # N's per 100 kbp =  0.00
  Drawing Nx plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/Nx_plot.pdf
  Drawing cumulative plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/cumulative_plot.pdf
  Drawing GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/GC_content_plot.pdf
  Drawing assembly GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/assembly_GC_content_plot.pdf
  Drawing assembly_uni GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/assembly_uni_GC_content_plot.pdf
  Drawing contigs GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/contigs_GC_content_plot.pdf
  Drawing S_aureus_flye GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/S_aureus_flye_GC_content_plot.pdf
  Drawing S_aureus_uni GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/S_aureus_uni_GC_content_plot.pdf
  Drawing SHASTA_ASSEMBLY GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/SHASTA_ASSEMBLY_GC_content_plot.pdf
  Drawing Staph_SHASTA_BLAST GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/Staph_SHASTA_BLAST_GC_content_plot.pdf
  Drawing Staph_Spades_c GC content plot...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/Staph_Spades_c_GC_content_plot.pdf
  Drawing Coverage histogram (bin size: 1x)...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/coverage_histogram.pdf
  Drawing contigs coverage histogram (bin size: 1x)...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/contigs_coverage_histogram.pdf
  Drawing Staph_Spades_c coverage histogram (bin size: 5x)...
    saved to /home/F22_T01/contigs_files/quast_all_assemblies/basic_stats/Staph_Spades_c_coverage_histogram.pdf
Done.

NOTICE: Genes are not predicted by default. Use --gene-finding or --glimmer option to enable it.

2022-11-28 21:37:59
Creating large visual summaries...
This may take a while: press Ctrl-C to skip this step..
  1 of 2: Creating PDF with all tables and plots...
  2 of 2: Creating Icarus viewers...
Done

2022-11-28 21:38:04
RESULTS:
  Text versions of total report are saved to /home/F22_T01/contigs_files/quast_all_assemblies/report.txt, report.tsv, and report.tex
  Text versions of transposed total report are saved to /home/F22_T01/contigs_files/quast_all_assemblies/transposed_report.txt, transposed_report.tsv, and transp
  HTML version (interactive tables and plots) is saved to /home/F22_T01/contigs_files/quast_all_assemblies/report.html
  PDF version (tables and plots) is saved to /home/F22_T01/contigs_files/quast_all_assemblies/report.pdf
  Icarus (contig browser) is saved to /home/F22_T01/contigs_files/quast_all_assemblies/icarus.html
  Log is saved to /home/F22_T01/contigs_files/quast_all_assemblies/quast.log

Finished: 2022-11-28 21:38:04
Elapsed time: 0:00:16.285005
NOTICEs: 3; WARNINGs: 0; non-fatal ERRORs: 0

Thank you for using QUAST!
[biF22_14@Mozart contigs_files]$ firefox quast_all_assemblies/report.html
```

Output:



Statistics without reference	assembly	assembly_uni	contigs	S_aureus_flye	S_aureus_uni	SHASTA_ASSEMBLY	Staph_SHASTA_BLAST	Staph_Spades_c
# contigs	9	7	1420	5	1	7	4	1287
# contigs (>= 0 bp)	9	7	1973	5	1	7	4	1702
# contigs (>= 1000 bp)	8	6	728	5	1	6	3	616
# contigs (>= 5000 bp)	7	6	122	5	1	4	2	45
# contigs (>= 10000 bp)	7	4	94	5	1	3	1	25
# contigs (>= 25000 bp)	6	4	71	4	1	3	1	20
# contigs (>= 50000 bp)	5	3	45	3	1	3	1	17
Largest contig	4 736 342	4 736 463	436 264	2 750 339	2 751 675	4 737 502	2 751 156	436 264
Total length	10 114 554	7 657 717	9 137 683	5 241 167	2 751 675	7 640 977	2 765 680	4 357 464
Total length (>= 0 bp)	10 114 554	7 657 717	9 317 740	5 241 167	2 751 675	7 640 977	2 765 680	4 509 748
Total length (>= 1000 bp)	10 114 048	7 656 940	8 645 609	5 241 167	2 751 675	7 640 338	2 765 041	3 880 387
Total length (>= 5000 bp)	10 109 094	7 656 940	7 552 000	5 241 167	2 751 675	7 634 415	2 760 546	2 864 005
Total length (>= 10000 bp)	10 109 094	7 645 268	7 369 689	5 241 167	2 751 675	7 625 025	2 751 156	2 746 460
Total length (>= 25000 bp)	10 098 650	7 645 268	7 005 858	5 230 723	2 751 675	7 625 025	2 751 156	2 661 924
Total length (>= 50000 bp)	10 054 199	7 619 471	6 025 372	5 186 272	2 751 675	7 625 025	2 751 156	2 566 229
N50	2 750 339	4 736 463	104 748	2 750 339	2 751 675	4 737 502	2 751 156	90 578
N90	2 002 884	2 751 675	1580	2 002 884	2 751 675	2 751 156	2 751 156	950
L50	2	1	24	1	1	1	1	13
L90	3	2	392	2	1	2	1	659
GC (%)	42.14	44.3	42.78	34.08	32.88	44.32	32.88	34.05
Mismatches								
# N's per 100 kbp	0	0	0	0	0	0	0	0
# N's	0	0	0	0	0	0	0	0

Figure 9. Quast report results for Flye, Unicycler, Spades, and Shasta assemblies before and after BLAST

```
→ $ quast.py --min-contig 500 *.fasta -o quast_assemblies_blast
```

Contains only the new files - where contamination has been removed

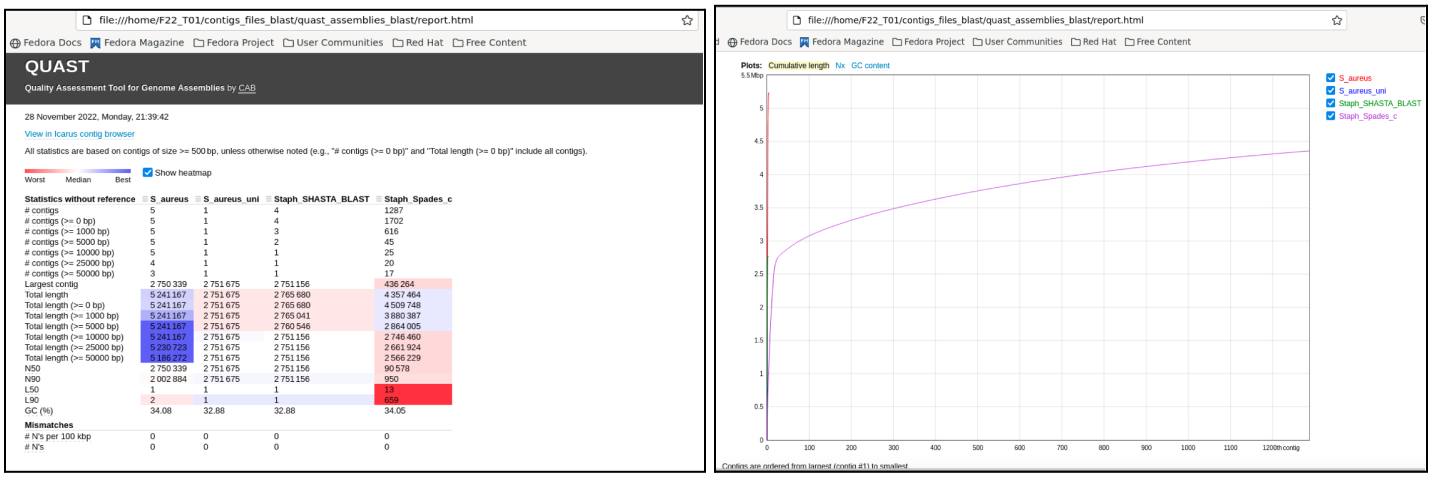


Figure 10. Quast report results for Flye, Unicycler, Spades, and Shasta assemblies after BLAST

We determined that the best assembly for downstream analysis was the assembly generated by ***Unicycler.*** When analyzed with QUAST, it was found to contain a lower number of contigs and higher value for N50 (especially compared to the Shasta output), a sharper upward curve. Additionally, it produced a final product that matched the expected size: We expected a genome of 2.8 million base pairs and the final product size was 2751675.

Part 5. Genome Annotation

After the highest quality assembly was selected (the one produced by Unicycler), it was annotated using PROKKA and DFAST in order to determine gene and coding regions.

vii - Use PROKKA (with GenBank compliance) and DFAST to annotate chosen assembly, using F22 as the locus-tag prefix and autoincrement values by 10

▪ PROKKA Annotation

The prokka command was applied on a separate screen to the chosen assembly using the following code:

```
→ $ prokka --addgenes --compliant --genus Staphylococcus --species aureus --gcode 11
--gram pos --locustag F22 --increment 10 --cpus 4 ./S_aureus_uni.fasta
```

Output:

```
[biF22_09@Mozart PROKKA_12012022]$ less PROKKA_12012022.gbk

LOCUS       F22_1               2751675 bp    DNA     linear       01-DEC-2022
DEFINITION  Staphylococcus aureus strain strain.
ACCESSION
VERSION
KEYWORDS    .
SOURCE      Staphylococcus aureus
  ORGANISM  Staphylococcus aureus
            Unclassified.
COMMENT     Annotated using prokka 1.14.6 from
            https://github.com/tseemann/prokka.
FEATURES             Location/Qualifiers
     source          1..2751675
                     /organism="Staphylococcus aureus"
                     /mol_type="genomic DNA"
                     /strain="strain"
     gene            1..1362
                     /locus_tag="F22_00010"
     CDS             1..1362
                     /locus_tag="F22_00010"
                     /inference="ab initio prediction:Prodigal:002006"
                     /codon_start=1
                     /transl_table=11
                     /product="hypothetical protein"
                     /protein_id="Prokka:F22_00010"
                     /translation="MGRKNGKTTTISGVANYAVSQDGENGAEIHLLANVMKQARILFD
                     ESKAMIKASPKLRENFRPLRDEIHYDATISKIMPQASDSDKLDGLNTHMGIFDEIHEF
                     KDYKLISVIKNSRAARLQPLLIYITTAGYQLDGPLVDMVEAGRDTLDQIIEDERTFYY
                     LASLDDDDDINDSSNWIKANPNLGVSIDLDEMKEEWEKAKRTPAERGDFITKRFNIFA
                     NNDEMSFIDYPTLQKNNEIISLDELEGRPCTIGYDLSETEDFTAACATFALDNGKVAV
                     LTHSWIPKHKVEYSNEKIPYREWEEDGLLTIQDKPYIDYQDVLNWIIKMNEHYVVEKI
                     TYDRANAFKLNQELKNYGFETEETRQGALTLSPALKDLKEMFLDGKIIFNNNPLMKWY
                     INNVQLKLDRNGNWLPSKQSRYRKIDGFAAFLNTYTDIMNKVVSDSGEGNIEFISIKD
                     IMR"
     gene            1367..2605
                     /locus_tag="F22_00020"
     CDS             1367..2605
                     /locus_tag="F22_00020"
                     /inference="ab initio prediction:Prodigal:002006"
                     /codon_start=1
                     /transl_table=11
                     /product="hypothetical protein"
                     /protein_id="Prokka:F22_00020"
                     /translation="MNVIAKENIVTRIKKKLIDNWIDQSASKLYDFSPWKNKSFWGVI
                     NNTLETNETIFSAITKLSNSMASLPLKMYEDYKVVNTEVSDLLTVSPNNSLSSFDFIN
                     QIETIRNEKGNAYVLIERDIYHQPSKLFLLNPDVVEMLIENQSRELYYSIHAATGNKL
                     IVHNMDMLHFKHIVASNMVQGISPIDVLKNTTDFDNAVRTFNLTEMQKPDSFMLKYGS
                     NVSTEKRQQVLEDFKQYYEENGGILFQEPGVEIEPLPKKYVSEDIVASENLTRERVAN
                     VFQLPSIFLNARSNTNFAKNEELNRFYLQHTLLPIVKQYEEEFNRKLLTKTDREKNRY
                     FKFNVKSYLRADSATQAEVYFKAVRSGYYTINDIREWEDLPPVEGGEKPLISGDLYPI
                     DTPLELRKSLKGGDKNVKES"
```

▪ DFAST Annotation

→ $ `dfast --locus_tag_prefix F22 --step 10 --cpu 4 -g S_aureus_uni.fasta -o DFAST`

Output:

```
LOCUS       sequence1           2751675 bp    DNA     linear   BCT 01-DEC-2022
DEFINITION  .
ACCESSION   sequence1
VERSION     sequence1
KEYWORDS    .
SOURCE
  ORGANISM
            .
FEATURES             Location/Qualifiers
     source          1..2751675
                     /mol_type="genomic DNA"
                     /organism=""
                     /note="sequence1"
     CDS             1..1362
                     /product="terminase"
                     /inference="COORDINATES:ab initio
                     prediction:MetaGeneAnnotator"
                     /inference="similar to AA sequence:RefSeq:WP_010965199.1"
                     /transl_table=11
                     /codon_start=1
                     /translation="MGRKNGKTTTISGVANYAVSQDGENGAEIHLLANVMKQARILFDE
                     SKAMIKASPKLRENFRPLRDEIHYDATISKIMPQASDSDKLDGLNTHMGIFDEIHEFKD
                     YKLISVIKNSRAARLQPLLIYITTAGYQLDGPLVDMVEAGRDTLDQIIEDERTFYYLAS
                     LDDDDDINDSSNWIKANPNLGVSIDLDEMKEEWEKAKRTPAERGDFITKRFNIFANNDE
                     MSFIDYPTLQKNNEIISLDELEGRPCTIGYDLSETEDFTAACATFALDNGKVAVLTHSW
                     IPKHKVEYSNEKIPYREWEEDGLLTIQDKPYIDYQDVLNWIIKMNEHYVVEKITYDRAN
                     AFKLNQELKNYGFETEETRQGALTLSPALKDLKEMFLDGKIIFNNNPLMKWYINNVQLK
                     LDRNGNWLPSKQSRYRKIDGFAAFLNTYTDIMNKVVSDSGEGNIEFISIKDIMR"
                     /locus_tag="F22_00010"
                     /note="WP_010965199.1 terminase (Clostridium acetobutylicum
                     ATCC 824) [pid:50.4%, q_cov:99.1%, s_cov:75.7%,
                     Eval:2.1e-128]"
                     /note="COG:COG4626:YmfN Phage terminase-like protein, large
                     subunit, contains N-terminal HTH domain [Category:X,
                     Aligned:1-431, Eval:6.1e-124, score:368.6]"
                     /note="MGA_1"
     CDS             1367..2605
                     /product="phage portal protein"
                     /inference="COORDINATES:ab initio
                     prediction:MetaGeneAnnotator"
                     /inference="similar to AA sequence:RefSeq:WP_011475945.1"
                     /transl_table=11
                     /codon_start=1
                     /translation="MNVIAKENIVTRIKKKLIDNWIDQSASKLYDFSPWKNKSFWGVIN
                     NTLETNETIFSAITKLSNSMASLPLKMYEDYKVVNTEVSDLLTVSPNNSLSSFDFINQI
                     ETIRNEKGNAYVLIERDIYHQPSKLFLLNPDVVEMLIENQSRELYYSIHAATGNKLIVH
                     NMDMLHFKHIVASNMVQGISPIDVLKNTTDFDNAVRTFNLTEMQKPDSFMLKYGSNVST
                     EKRQQVLEDFKQYYEENGGILFQEPGVEIEPLPKKYVSEDIVASENLTRERVANVFQLP
                     SIFLNARSNTNFAKNEELNRFYLQHTLLPIVKQYEEEFNRKLLTKTDREKNRYFKFNVK
:
```

viii - Compare predicted proteins produced by PROKKA and DFAST and confirm congruency

▪ Comparing PROKKA and DFAST Predicted Proteins

grep command was applied to the ".gbk" files produced by PROKKA AND DFAST using the following command

```
→ $ grep -c '/locus_tag=' dfast_genome.gbk  prokka_genome.gbk
```

Output:

```
[biF22_17@Mozart johanna]$ ls
dfast_genome.gbk  prokka_genome.gbk
[biF22_17@Mozart johanna]$ grep -c '/locus_tag=' dfast_genome.gbk  prokka_genome.gbk
dfast_genome.gbk:2634
prokka_genome.gbk:5206
[biF22_17@Mozart johanna]$
```

```
  GNU nano 5.8  DFAST/statistics.txt
Total Sequence Length (bp)      2751675
Number of Sequences     1
Longest Sequences (bp)  2751675
N50 (bp)        2751675
Gap Ratio (%)   0.000000
GCcontent (%)   32.9
Number of CDSs  2554
Average Protein Length  299.8
Coding Ratio (%)        83.5
Number of rRNAs 19
Number of tRNAs 60
Number of CRISPRs       2
```

```
  PROKKA_12012022/PROKKA_12012022.txt
organism: Staphylococcus aureus strain
contigs: 1
bases: 2751675
CDS: 2523
CRISPR: 1
gene: 2603
rRNA: 19
tRNA: 60
tmRNA: 1
```

The number of hits for PROKKA is approximately double because PROKKA adds "locus_tag" twice for each protein. A direct comparison is 2634 hits for DFAST and 2603 for PROKKA which is similar enough to assume that the results agree with each other. The statistics files generated by each program shown above also indicate they agree in terms of rRNA and tRNA as well (19 rRna and 60 tRNA).

Part 6. Contamination Removal of Sequencing Dataset

Now that a final assembly has been generated it can be applied to the initial read sets in order to more thoroughly filter out the irrelevant reads and keep ones pertaining to *Staphylococcus aureus*.

ix - Filter out the sequencing datasets to keep only the reads from *Staphylococcus aureus*. Use your best assembly as reference. Use get_SNPs.pl (--rmo + --bam), minimap2 and samtools bam2fq for this purpose. Compare the number of reads before/after with FASTQC. Compare the *.fastq.gz sizes with du -sh too.

▪ get_SNP.pl & samtools bam2fq

get_SNP.pl script and the "samtools bam2fq" command was applied to the illumina and nanopore reads in order to remove unwanted contaminations and keep only reads from *Staphylococcus aureus*.

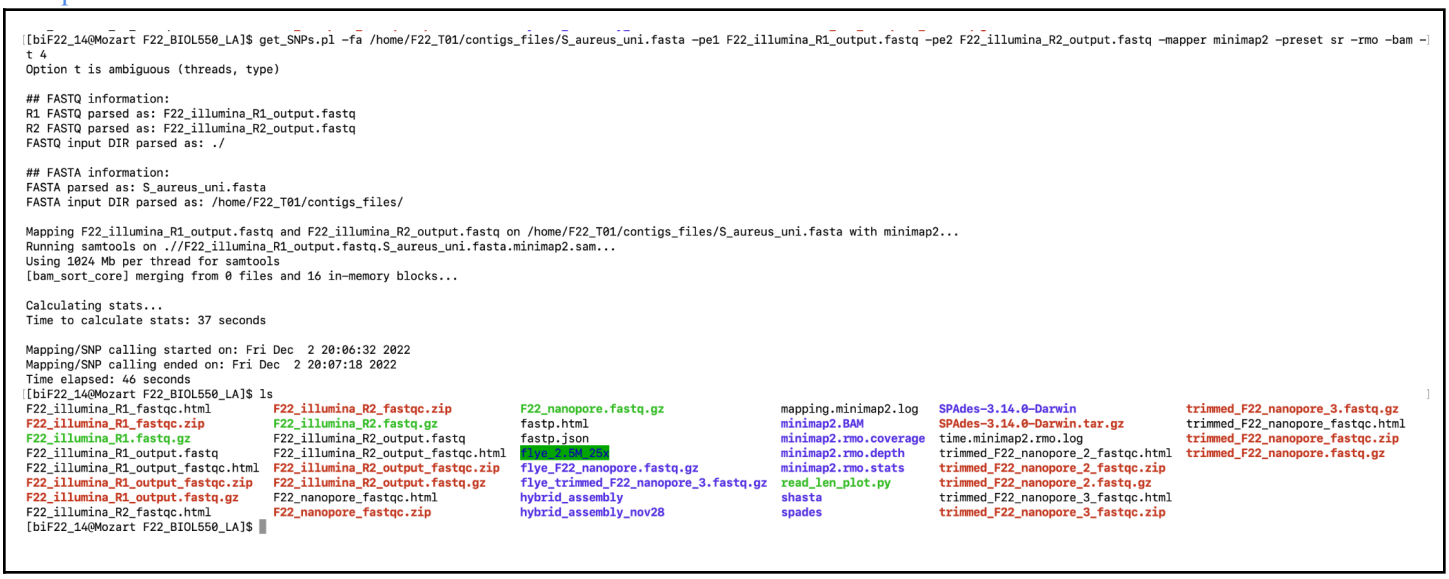
The "get_SNP" command utilizes minimap2 as the mapper in order to do this. In the get_SNP.pl script, the -preset command for illumina is short reads [sr], and nanopore is long reads [map-ont].

In addition to the "get_SNP.pl" script, the "samtools bam2fq" is used to parse the generated bam files.

illumina data + get_SNP

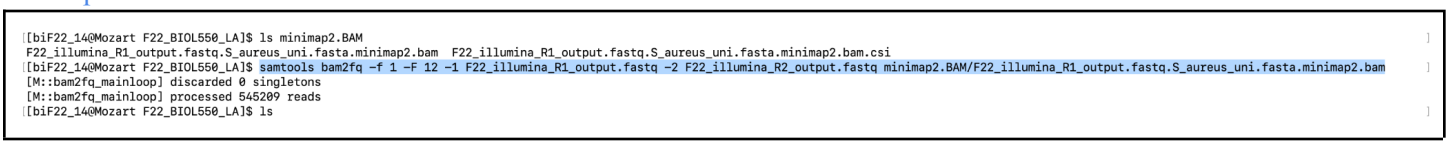
```
→ $ gunzip -k F22_illumina_R1_output.fastq.gz
$ gunzip -k F22_illumina_R2_output.fastq.gz

→ $ get_SNPs.pl -fa /home/F22_T01/contigs_files/S_aureus_uni.fasta -pe1
F22_illumina_R1_output.fastq -pe2 F22_illumina_R2_output.fastq -mapper minimap2 -preset sr
-rmo -bam -t 4
```

#Output:




```
→ $ samtools bam2fq -f 1 -F 12 -1 F22_illumina_R1_output.fastq -2
F22_illumina_R2_output.fastq
minimap2.BAM/F22_illumina_R1_output.fastq.S_aureus_uni.fasta.minimap2.bam
```

Output:




```
→ $ get_SNPs.pl -fq F22_BIOL550_LA/trimmed_F22_nanopore.fastq -fa
/home/F22_T01/contigs_files/S_aureus_uni.fasta -pe1
F22_BIOL550_LA/F22_illumina_R1_output.fastq -pe2
F22_BIOL550_LA/F22_illumina_R2_output.fastq -mapper minimap2 -preset sr -rmo -bam -t 4

→ $ gzip -c map_R1.fastq > map_R1.fastq.gz
→ $ gzip -c map_R2_output.fastq > map_R2.fastq.gz
```

Output:

```
[[biF22_14@Mozart cont_rem_last_step]$ gzip -c map_R1.fastq > map_R1.fastq.gz
[[biF22_14@Mozart cont_rem_last_step]$ gzip -c map_R2_output.fastq > map_R2.fastq.gz
[[biF22_14@Mozart cont_rem_last_step]$ ls
cr_nano               map_R1.fastq.gz      minimap2.BAM           minimap2.rmo.stats
mapping.minimap2.log  map_R2.fastq.gz      minimap2.rmo.coverage  time.minimap2.rmo.log
map_R1.fastq          map_R2_output.fastq  minimap2.rmo.depth
[biF22_14@Mozart cont_rem_last_step]$
```

Nanopore data + get_SNP

→ $ get_SNPs.pl -fq ../../F22_BIOL550_LA/trimmed_F22_nanopore.fastq -fa ../../contigs_files/S_aureus_uni.fasta -mapper minimap2 -preset map-ont -rmo -bam -t 4

Output:

```
[[biF22_14@Mozart cont_rem_last_step]$ mkdir cr_nano
[[biF22_14@Mozart cont_rem_last_step]$ cd cr_nano
[[biF22_14@Mozart cr_nano]$ get_SNPs.pl -fq ../../F22_BIOL550_LA/trimmed_F22_nanopore.fastq -fa ../../contigs_files/S_aureus_uni.
fasta -mapper minimap2 -preset map-ont -rmo -bam -t 4
Option t is ambiguous (threads, type)

## FASTQ information:
FASTQ parsed as: trimmed_F22_nanopore.fastq
FASTQ input DIR parsed as: ../../F22_BIOL550_LA/

## FASTA information:
FASTA parsed as: S_aureus_uni.fasta
FASTA input DIR parsed as: ../../contigs_files/

Mapping ../../F22_BIOL550_LA/trimmed_F22_nanopore.fastq on ../../contigs_files/S_aureus_uni.fasta with minimap2...
Running samtools on .//trimmed_F22_nanopore.fastq.S_aureus_uni.fasta.minimap2.sam...
Using 1024 Mb per thread for samtools
[bam_sort_core] merging from 0 files and 16 in-memory blocks...

Calculating stats...
Time to calculate stats: 113 seconds

Mapping/SNP calling started on: Mon Dec  5 19:03:16 2022
Mapping/SNP calling ended on: Mon Dec  5 19:05:17 2022
Time elapsed: 121 seconds
[biF22_14@Mozart cr_nano]$ ls
mapping.minimap2.log  minimap2.BAM  minimap2.rmo.coverage  minimap2.rmo.depth  minimap2.rmo.stats  time.minimap2.rmo.log
[biF22_14@Mozart cr_nano]$
```

→ $ samtools bam2fq -F 4 trimmed_F22_nanopore.fastq.S_aureus_uni.fasta.minimap2.bam > mapped_nano.fastq

Output:

```
[biF22_14@Mozart cr_nano]$ samtools bam2fq -F 4 minimap2.BAM/trimmed_F22_nanopore.fastq.S_aureus_uni.fasta.minimap2.bam > mapped_nano.fastq
[M::bam2fq_mainloop] discarded 0 singletons
[M::bam2fq_mainloop] processed 18143 reads
[biF22_14@Mozart cr_nano]$
```

→ $ gzip -c mapped_nano.fastq > mapped_nano.fastq.gz

Output:

```
[[biF22_14@Mozart cr_nano]$ gzip -c mapped_nano.fastq > mapped_nano.fastq.gz
[[biF22_14@Mozart cr_nano]$ ls
 mapped_nano.fastq      minimap2.BAM            minimap2.rmo.stats
 mapped_nano.fastq.gz   minimap2.rmo.coverage   nano_output.fastq
 mapping.minimap2.log   minimap2.rmo.depth      time.minimap2.rmo.log
 [biF22_14@Mozart cr_nano]$
```

▪ FASTQC Before and After Contamination Removal

Fastqc command was again applied to retrieve analysis for the illumina and nanopore reads after the contamination removal step:

```
→ $ fastqc -t 4 map_R1.fastq
```

Illumina FastQC

Post-Contamination-Removal FastQC:

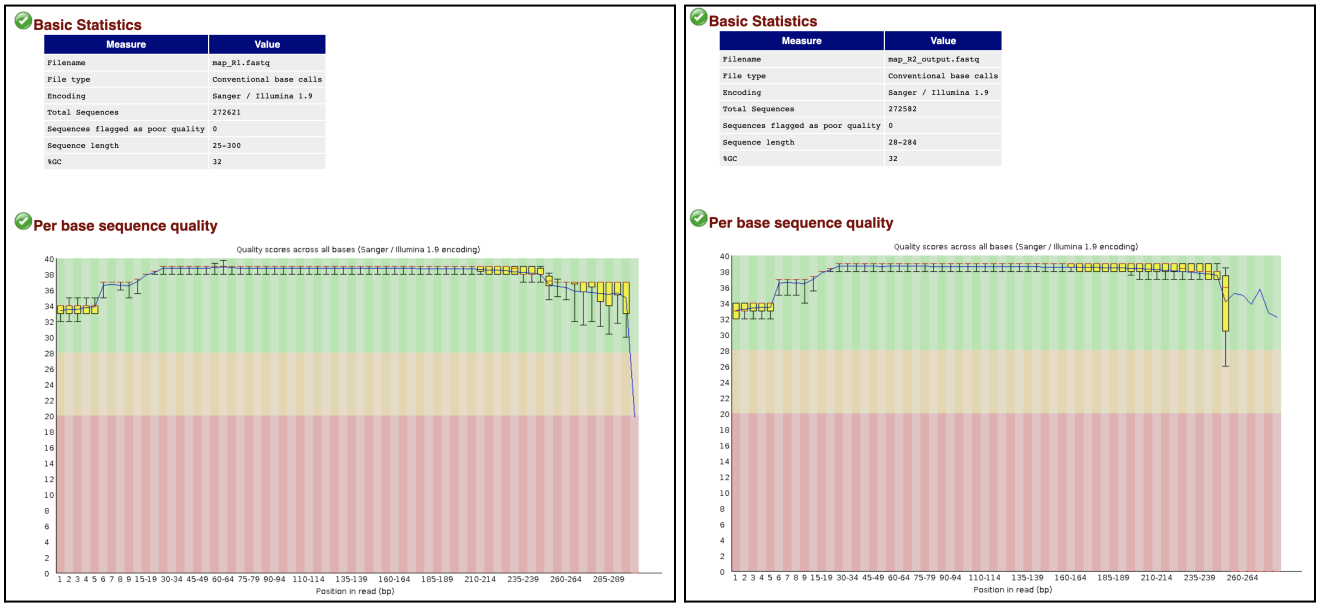


Figure 11. FastQC reports of map_R1 and map_R2 Illumina

Output was compared to both the raw reads and the previous filtration method of Fastp to determine if one method was optimal. A visual comparison of the FastQC outputs above and below indicates the contamination removal clearly removes all the poorest reads (in the “red” category) to a higher extent than Fastp did and it appears that the quality around the edges is a lot cleaner. Curating reads for analysis via get_SNPs seems to produce a cleaner outcome.

Raw Reads:

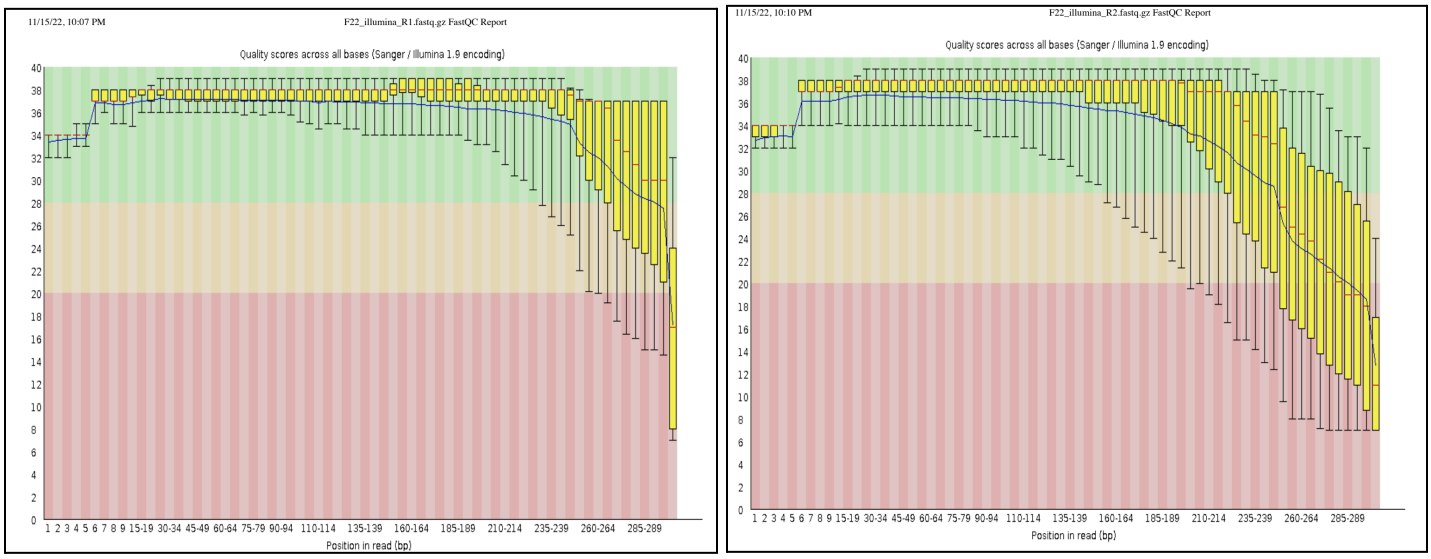


Figure 12. FastQC report of R1 and R2 before Fastp applied

Fastp Filtered Illumina Reads:

Figure 13. FastQC reports of R1 and R1 Illumina after Fastp applied

Nanopore FastQC

```
→ $ fastqc -t 4 mapped_nano.fastq.gz
```

Output:

```
[biF22_14@Mozart cr_nano]$ fastqc -t 4 mapped_nano.fastq.gz
Started analysis of mapped_nano.fastq.gz
Approx 5% complete for mapped_nano.fastq.gz
Approx 10% complete for mapped_nano.fastq.gz
Approx 15% complete for mapped_nano.fastq.gz
Approx 20% complete for mapped_nano.fastq.gz
Approx 30% complete for mapped_nano.fastq.gz
Approx 35% complete for mapped_nano.fastq.gz
Approx 45% complete for mapped_nano.fastq.gz
Approx 50% complete for mapped_nano.fastq.gz
Approx 55% complete for mapped_nano.fastq.gz
Approx 65% complete for mapped_nano.fastq.gz
Approx 75% complete for mapped_nano.fastq.gz
Approx 85% complete for mapped_nano.fastq.gz
Approx 90% complete for mapped_nano.fastq.gz
Approx 95% complete for mapped_nano.fastq.gz
Analysis complete for mapped_nano.fastq.gz
[biF22_14@Mozart cr_nano]$
```

Post-Contamination-Removal Step FastQC Results for Nanopore Reads:

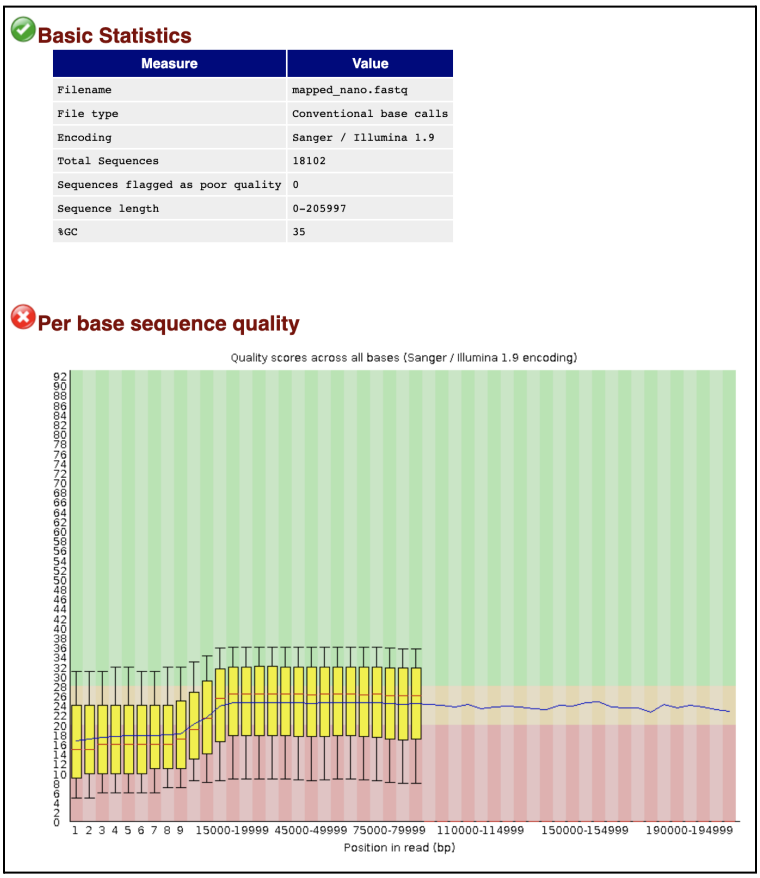


Figure 14. FastQC report of Nanopore after contamination removal

Raw and Nanofilt output is below. Mapped output compared to raw reads is clearly improved as the average per base sequence quality doesn't go below about 15. When compared to the initial cleanup method using Nanofilt, it's very

apparent that the total sequences are reduced (18102 vs 97719). It is assumed that this indicates the mapping processes are more thorough than Nanofilt, though it is difficult to confirm based on the FastQC report alone. The average "per base" quality appears to be similar for the mapped vs. Nanofilt cleaned set of reads.

Raw Nanopore Reads & Nanopore Reads with Nanofilt Applied FastQC Results:

"F22_nanopore.fastq.gz" are the initial reads, and "trimmed_F22_nanopore.fastq.gz" are reads with applied Nanofilt.

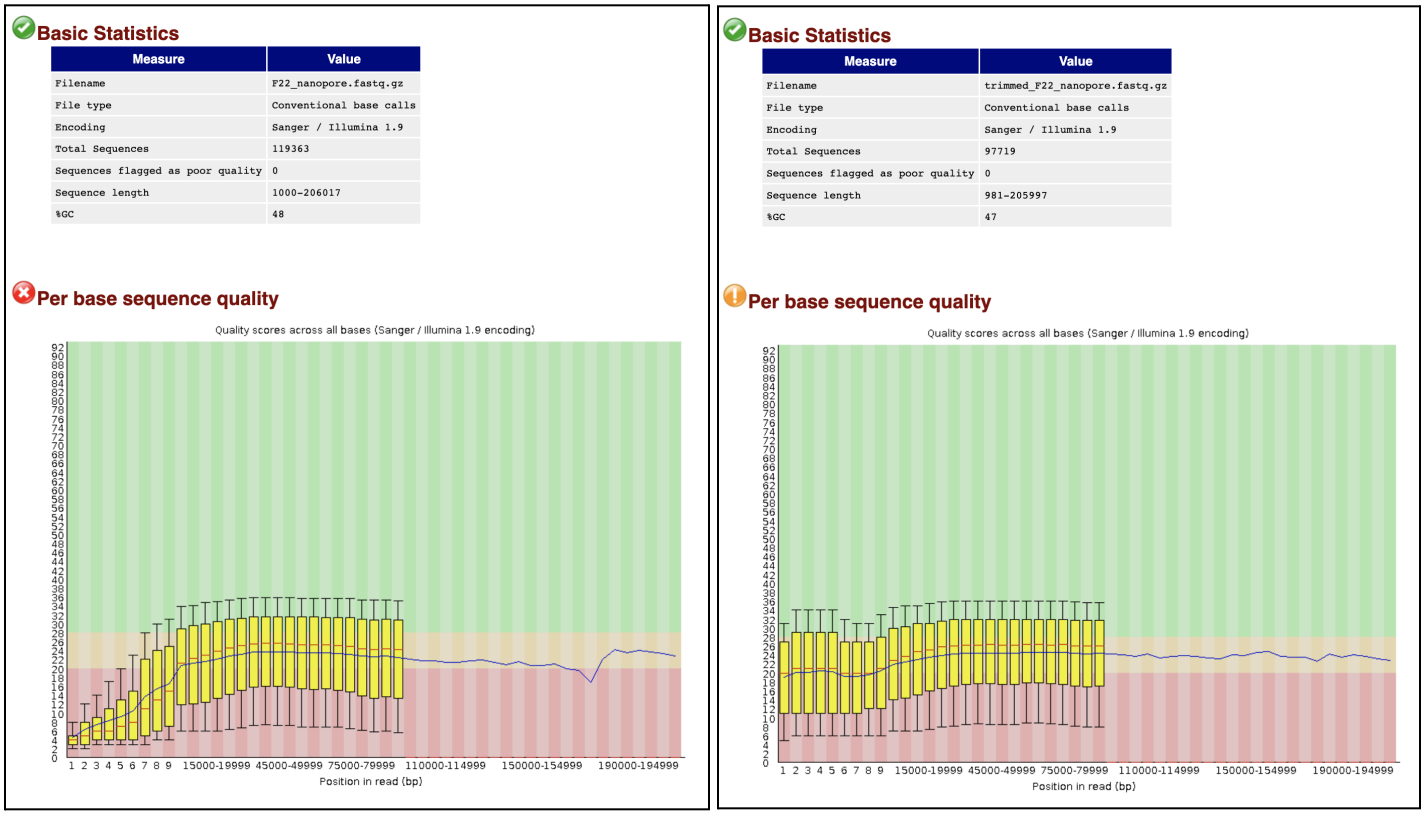


Figure 15. FastQC report of Nanopore before and after Nanofilt applied

▪ fastq.gz Size Comparison using "du -sh" command

The du -sh command stands for disk usage and it prints the number of bytes (-sh argument indicates "summarize" & "human readable")

Illumina + du-sh

Looking at the outputs from the FastQC files before and after, the contamination removal step appears to have reduced R1 by 51 (184 - 133) bytes and R2 by 82 bytes (211 - 129).

#Post contamination removal files

```
$ du -sh map_R1.fastq.gz
$ du -sh map_R2.fastq.g
```

#Illumina files trimmed with fastp

```
$ du -sh ../F22_BIOL550_LA/F22_illumina_R1_output.fastq.gz
```

```
$ du -sh ../F22_BIOL550_LA/F22_illumina_R2_output.fastq.gz
```

#Original Illumina files

```
$ du -sh ../F22_BIOL550_LA/F22_illumina_R1.fastq.gz
$ du -sh ../F22_BIOL550_LA/F22_illumina_R2.fastq.gz
```

For the Illumina datasets R1 and R2, the contamination removal step appears to have reduced the total size greatly, with the initial read sets being 184M and 211M and the final being 14M and 18M. The size of trimmed R1 and R2 by using Fastp are 97M and 95M.

Output:

```
[biF22_14@Mozart cont_rem_last_step]$ du -sh map_R1.fastq.gz
14M     map_R1.fastq.gz
[biF22_14@Mozart cont_rem_last_step]$ du -sh map_R2.fastq.gz
18M     map_R2.fastq.gz
[biF22_14@Mozart cont_rem_last_step]$ du -sh ../F22_BIOL550_LA/F22_illumina_R1_output.fastq.gz
97M     ../F22_BIOL550_LA/F22_illumina_R1_output.fastq.gz
[biF22_14@Mozart cont_rem_last_step]$ du -sh ../F22_BIOL550_LA/F22_illumina_R2_output.fastq.gz
95M     ../F22_BIOL550_LA/F22_illumina_R2_output.fastq.gz
[biF22_14@Mozart cont_rem_last_step]$ du -sh ../F22_BIOL550_LA/F22_illumina_R1.fastq.gz
184M    ../F22_BIOL550_LA/F22_illumina_R1.fastq.gz
[biF22_14@Mozart cont_rem_last_step]$ du -sh ../F22_BIOL550_LA/F22_illumina_R2.fastq.gz
211M    ../F22_BIOL550_LA/F22_illumina_R2.fastq.gz
[biF22_14@Mozart cont_rem_last_step]$
```

Nanopore + du-sh

For the nanopore the contamination removal step appears to have reduced the total size greatly, with the initial read set being 1.1G and the final being 146M. The size of filtered nanopore dataset by using nanofilt is 885M. The larger initial file size for nanopore data was assumed to be due to the fact that nanopore produces long reads, compared to illumina, which produces short reads.

#Post contamination removal file
→ `$ du -sh mapped_nano.fastq.gz`

#Nanopore dataset trimmed using Nanofilt
→ `$ du -sh ../../F22_BIOL550_LA/trimmed_F22_nanopore.fastq`

#Original nanopore dataset
→ `$ du -sh ../../F22_BIOL550_LA/F22_nanopore.fastq.gz`

Output:

```
[biF22_14@Mozart cr_nano]$ du -sh mapped_nano.fastq.gz
146M    mapped_nano.fastq.gz
[biF22_14@Mozart cr_nano]$ du -sh ../../F22_BIOL550_LA/trimmed_F22_nanopore.fastq.gz
885M    ../../F22_BIOL550_LA/trimmed_F22_nanopore.fastq.gz
[biF22_14@Mozart cr_nano]$ du -sh ../../F22_BIOL550_LA/F22_nanopore.fastq.gz
1.1G    ../../F22_BIOL550_LA/F22_nanopore.fastq.gz
[biF22_14@Mozart cr_nano]$
```

The output files of this step can be found at the below path:-
/home/F22_T01/final_step_output_files

map_R1.fastq.gz - Illumina R1 final file
map_R2.fastq.gz - Illumina R2 final file
Mapped_nano.fastq.gz - Nanopore final file
DFAST - folder contains dfast output file
PROKKA_12012022 - folder contains prokka output file

—----------Thank you----------