Genome QC and Assembly Lab:

Introduction:

Generating large amounts of genetic data quickly is now possible thanks to the ever-improving sequencing technology. The data may still contain inaccuracies, though, as each sequencing platform has unique drawbacks. Data from the Illumina platform may have "noise," adapters must be taken into account, and if primers run out before sequencing is finished, the data may be disorganized. Sequence read quality control (QC), which ensures the validity of data utilized for a growing volume of genomic research, must be carried out with care for sequencing systems to counteract such errors.

The process of QC includes visualization and cleaning of the raw sequence data. The purpose of cleaning is to get rid of adapter sequences, ranges of sequences that fall below a quality score threshold, mean per-base score read, and extremely short sequences. In this lab, we will use the SRA toolkit for downloading data, Trimmomatic for data cleaning, and FastQC for data visualization. Data samples include the following:

1. ERR008613 (a set of paired-end Illumina sequence reads from ends of 200bp **E. coli** fragments)
    - ERR008613_1.fastq
    - ERR008613_2.fastq
2. ERR022075 (a set of paired-end Illumina sequence reads from ends of 600bp **E. coli** fragments)
    - ERR022075_1.fastq
    - ERR022075_2.fastq
3. sets of PacBio CCS and CLR reads for E. coli
    - PacBio_10kb_CLR.fastq
    - PacBio_2kb_CCS_500bp.fastq

First, we will utilize fastq-dump to access runs from the SRA website and will visualize the data before trimming reads. Then we will use 3 different parameters for each dataset and visualize the data again. Before and after visualizations that are generated will be used for comparisons between and across the various trimming commands.

Methods:

To perform the lab, there are several steps to be followed:

1. Using **fastq-dump** to access runs from the SRA website
2. Checking the fastq file quality using **FastQC**
3. Trimming the reads using **Trimmomatic** followed by checking the trimmed reads using FastQC
4. Assembling the reads using **SPAdes**
5. Running **QUAST** using the reference genome to get a graphical summary of the assembly quality

Step 1: Using fastq-dump to access runs from the SRA website

We are using Linux Bash for Windows 11 Shell. The data was available on the student cluster which can be accessed by every student. The files were downloaded on the local machine and the FastQC software was used after this. These sequences were originally downloaded using the "**fastq-dumps**" which is a tool for downloading sequencing reads from NCBI's Sequence Read Archive (SRA) as FASTQ files.

Code:

```
#!/bin/bash
#SBATCH --job-name=getsras
#SBATCH --partition=Centaurus
#SBATCH --time=01:00:00

module load hdf5/1.10.7
module load sra-tools

fastq-dump --split-3 ERR022075
fastq-dump --split-3 ERR008613
```

Step 2: Checking the fastq file quality using FastQC:

The fastq files were downloaded on the local machine and were run on the FastQC software. There were 6 files mentioned above that we checked here, 4 of which were Illumina files and 2 were PacBio files. The PacBio files were comparatively larger than the Illumina files. There are 10 parameters that were taken into consideration for checking the quality.

Step 3: Trimming the reads using Trimmomatic followed by checking the trimmed reads using FastQC.

This is a crucial step for cleaning the reads. In this step, we remove the adapters and low-quality bases (leading or trailing), scan the read with a sliding window, cut the average quality per base drops below a cap and drop reads below the minimum length mentioned. It is used for both kinds of reads for the Illumina data, for paired ends and single ends. In total, we get 10 files. Trimmomatic is a java executable (*.jar) file.

Code:

```
#!/bin/bash
#SBATCH --job-name=trimmomatic
#SBATCH --partition=Centaurus
#SBATCH --time=01:00:00

module load trimmomatic

java -jar $TRIM/trimmomatic-0.39.jar PE ERR022075_1.fastq ERR022075_2.fastq
ERR022075_1_paired.fq ERR022075_1_unpaired.fq ERR022075_2_paired.fq
ERR022075_2_unpaired.fq ILLUMINACLIP:$TRIM/adapters/TruSeq3-PE.fa:2:30:10
SLIDINGWINDOW:15:20 MINLEN:70
```

java -jar $TRIM/trimmomatic-0.39.jar PE ERR008613_1.fastq ERR008613_2.fastq ERR008613_1_paired.fq ERR008613_1_unpaired.fq ERR008613_2_paired.fq ERR008613_2_unpaired.fq ILLUMINACLIP:$TRIM/adapters/TruSeq3-PE.fa:2:30:10 SLIDINGWINDOW:15:20 MINLEN:70

java -jar $TRIM/trimmomatic-0.39.jar SE -phred33 PacBio_10kb_CLR.fastq PacBio_10kb_CLR_trim-out.fq SLIDINGWINDOW:100:6 MINLEN:70

java -jar $TRIM/trimmomatic-0.39.jar SE -phred33 PacBio_2kb_CCS_500bp.fastq PacBio_2kb_CCS_500bp_trim-out.fq SLIDINGWINDOW:100:19 MINLEN:70

Here, we decide on sliding window size and minimum length based on the outputs from the FastQC. Followed by this, the trimmed files are downloaded again and checked using FastQC for comparison.

After trimming, we can see how the reads look in the FastQC again to check the difference between the trimmed and raw reads. We can also see this based on the file size, as it gets trimmed it is lesser in size.

Step 4: Assembling the reads using SPAdes

Spades is a genome assembly algorithm. There are multiple ways used to combine the available reads. e.g. Illumina only, 2 Illumina libraries combined, Illumina Pac Bio, and Illumina with both Pac Bio. This is the most time-consuming process in the whole assignment. The spades require an anaconda3 module to be loaded before loading the spades module. The most important outputs here are contigs.fasta and scaffolds.fasta. We will be using the contigs.fasta file for QUAST further

Code:

```
#!/bin/bash
#SBATCH --job-name=spades
#SBATCH --partition=Centaurus
#SBATCH --time=10:00:00
#SBATCH --mem=64GB
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16

module load anaconda3
module load spades/3.14.1

spades.py -m 64 -1 ERR008613_1_paired.fq -2 ERR008613_2_paired.fq -s ERR008613_1_unpaired.fq -s ERR008613_2_unpaired.fq -o illumina_8613_only
spades.py -m 64 --pe1-1 ERR008613_1_paired.fq --pe1-2 ERR008613_2_paired.fq --pe2-1 ERR022075_1_paired.fq --pe2-2 ERR022075_2_paired.fq -o illumina_ComLib
spades.py -m 64 --pe1-1 ERR008613_1_paired.fq --pe1-2 ERR008613_2_paired.fq --pe2-1 ERR022075_1_paired.fq --pe2-2 ERR022075_2_paired.fq -s PacBio_2kb_CCS_500bp_trim-out.fq -o illumina_pac
```

spades.py -m 64 --pe1-1 ERR008613_1_paired.fq --pe1-2 ERR008613_2_paired.fq --pe2-1 ERR022075_1_paired.fq --pe2-2 ERR022075_2_paired.fq -s PacBio_2kb_CCS_500bp_trim-out.fq --pacbio PacBio_10kb_CLR_trim-out.fq -o illumina_pac_both

Here, pe means paired-ends, pe1-1 is the forward file for the first read, pe1-2 is the reverse file for the first read. Similarly, pe2-1 and pe2-2 are forward and reverse files for the second read respectively.

Step 5: Running QUAST using the reference genome to get a graphical summary of the assembly quality

After running the SPAdes, we want to check if the quality of the Assembly is good or not. Hence, we compare it with the reference genome. Here, we need to have the reference genome FASTA file and reference annotation GFF3 file.

Code:

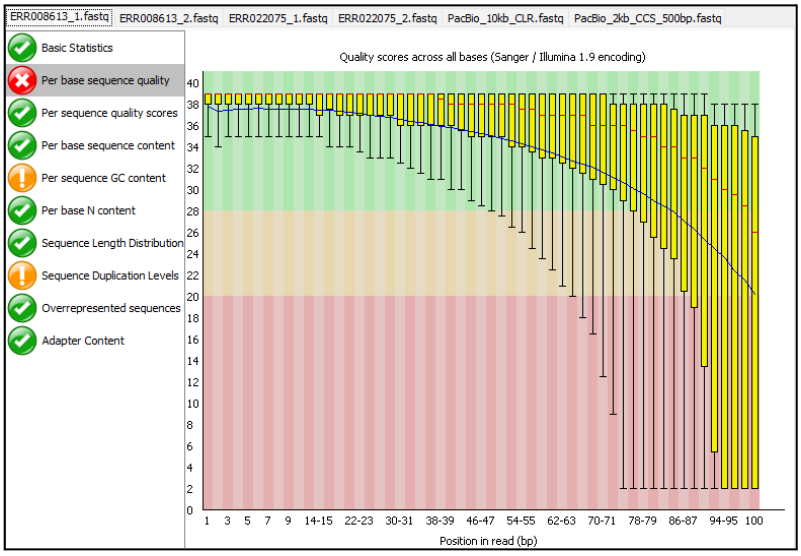
```
#!/bin/bash
#SBATCH --job-name=quast
#SBATCH --partition=Centaurus
#SBATCH --time=00:30:00

module load quast

quast.py -o illumina_8613_only_quast -r refseq.fasta --features refseq1.gff3 -t 16 --est-ref-size 4641652 illumina_8613_only/contigs.fasta
quast.py -o illumina_ComLib_quast -r refseq.fasta --features refseq1.gff3 -t 16 --est-ref-size 4641652 illumina_ComLib/contigs.fasta
quast.py -o illumina_pac_quast -r refseq.fasta --features refseq1.gff3 -t 16 --est-ref-size 4641652 illumina_pac/contigs.fasta
quast.py -o illumina_pac_both_quast -r refseq.fasta --features refseq1.gff3 -t 16 --est-ref-size 4641652 illumina_pac_both/contigs.fasta
```

Results:

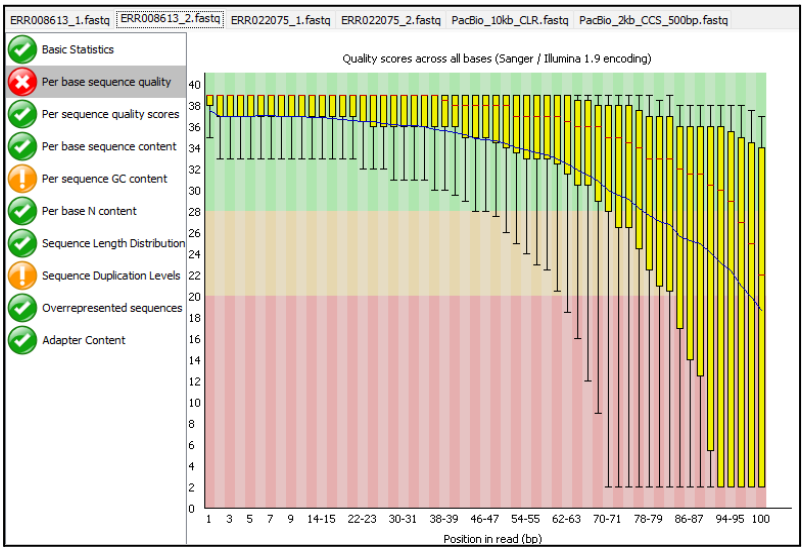
1. FastQC Results before trimming (raw reads):

The results showed that majorly with all the sequences, the per base sequence quality was very poor.

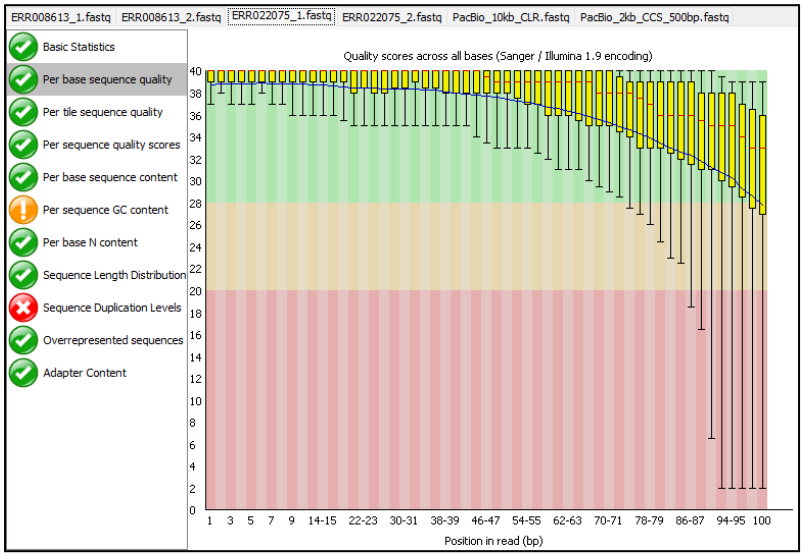
(A)


ERR008613_1.fastq
ERR008613_2.fastq
ERR022075_1.fastq
ERR022075_2.fastq
PacBio_10kb_CLR.fastq
PacBio_2kb_CCS_500bp.fastq
Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

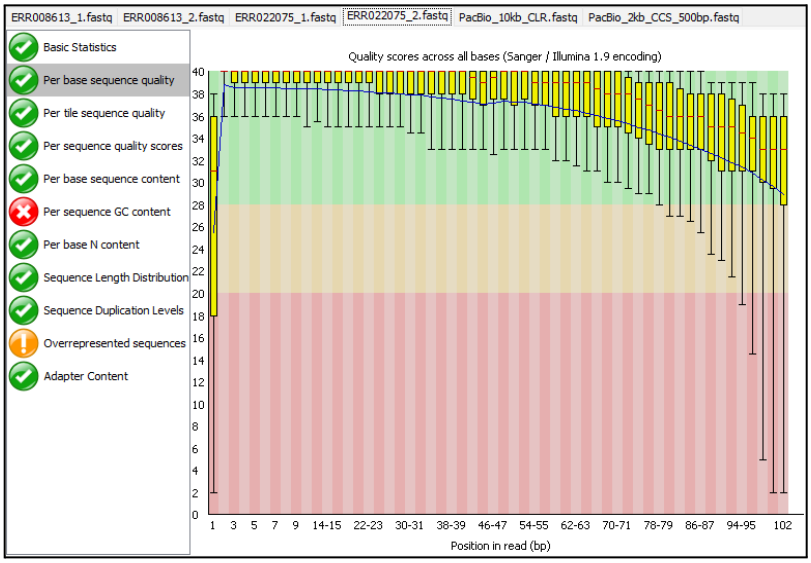
(B)


ERR008613_1.fastq
ERR008613_2.fastq
ERR022075_1.fastq
ERR022075_2.fastq
PacBio_10kb_CLR.fastq
PacBio_2kb_CCS_500bp.fastq
Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

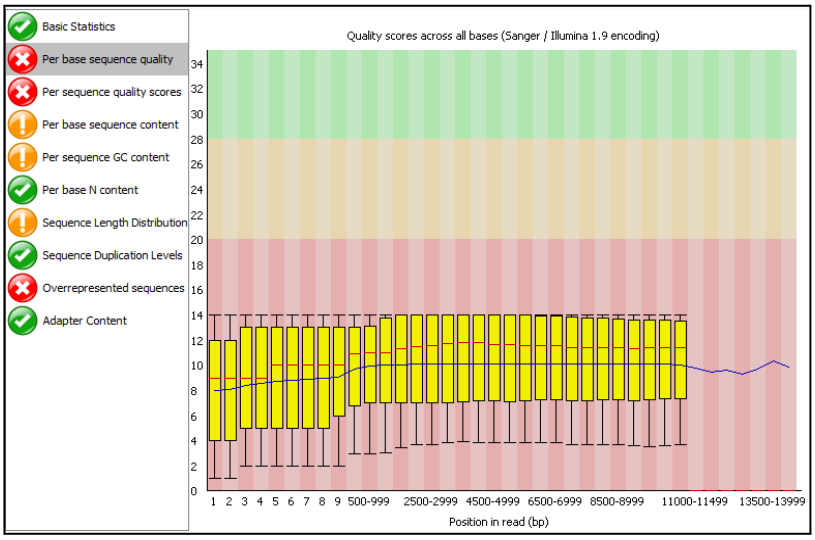
(C)


ERR008613_1.fastq
ERR008613_2.fastq
ERR022075_1.fastq
ERR022075_2.fastq
PacBio_10kb_CLR.fastq
PacBio_2kb_CCS_500bp.fastq
Basic Statistics
Per base sequence quality
Per tile sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

(D)


ERR008613_1.fastq
ERR008613_2.fastq
ERR022075_1.fastq
ERR022075_2.fastq
PacBio_10kb_CLR.fastq
PacBio_2kb_CCS_500bp.fastq
Basic Statistics
Per base sequence quality
Per tile sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

(E)


Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

(F)

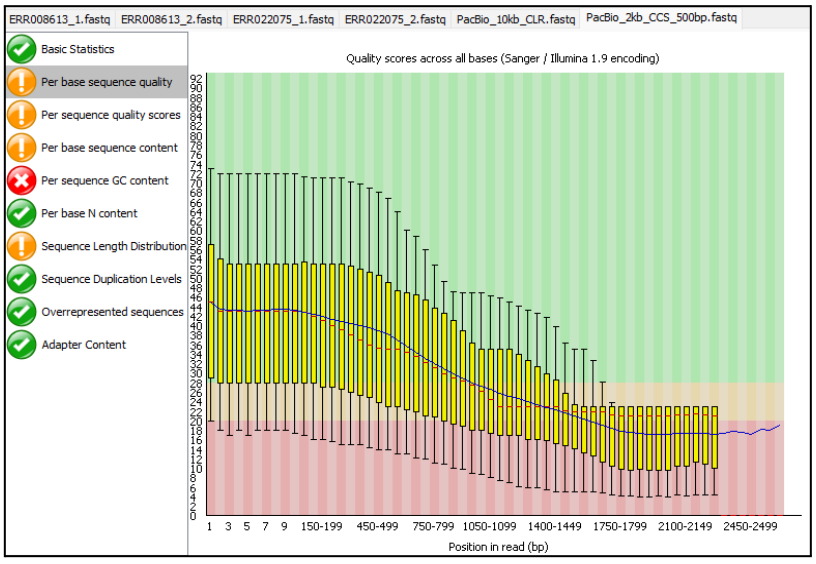
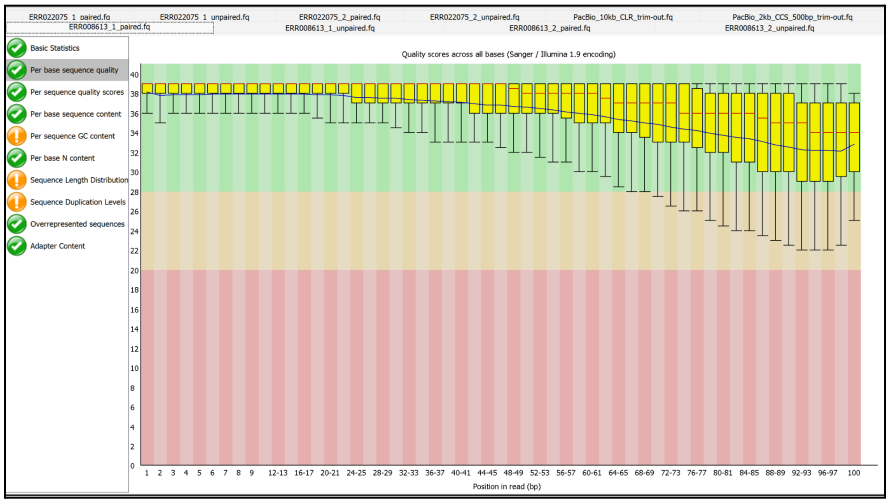

ERR008613_1.fastq
ERR008613_2.fastq
ERR022075_1.fastq
ERR022075_2.fastq
PacBio_10kb_CLR.fastq
PacBio_2kb_CCS_500bp.fastq
Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

Fig 1: (A) ERR008613_1.fastq, (B) ERR008613_2.fastq, (C) ERR022075_1.fastq, (D) ERR022075_2.fastq, (E) PacBio_10kb_CLR.fastq, (F) PacBio_2kb_CCS_500bp.fastq

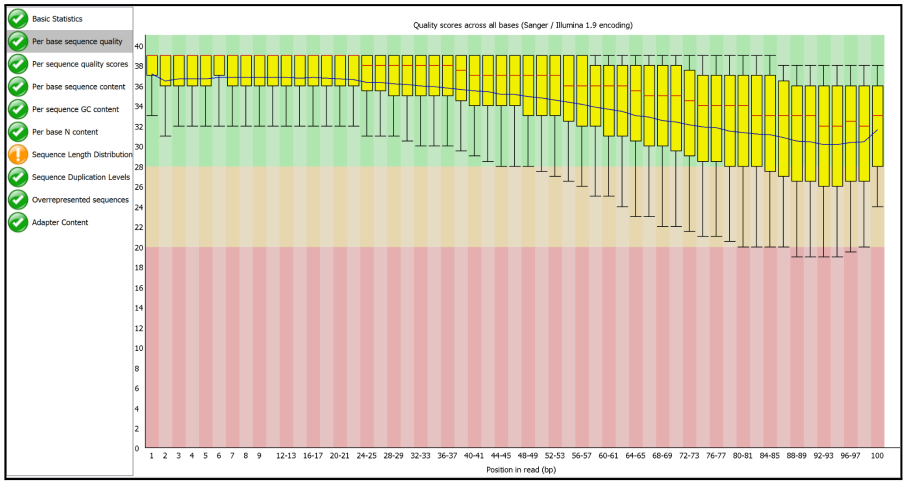
2. FastQC Results after using Trimmomatic:

We can see here that most of the files show an improvement in the per base sequence quality. The pictures are attached below:

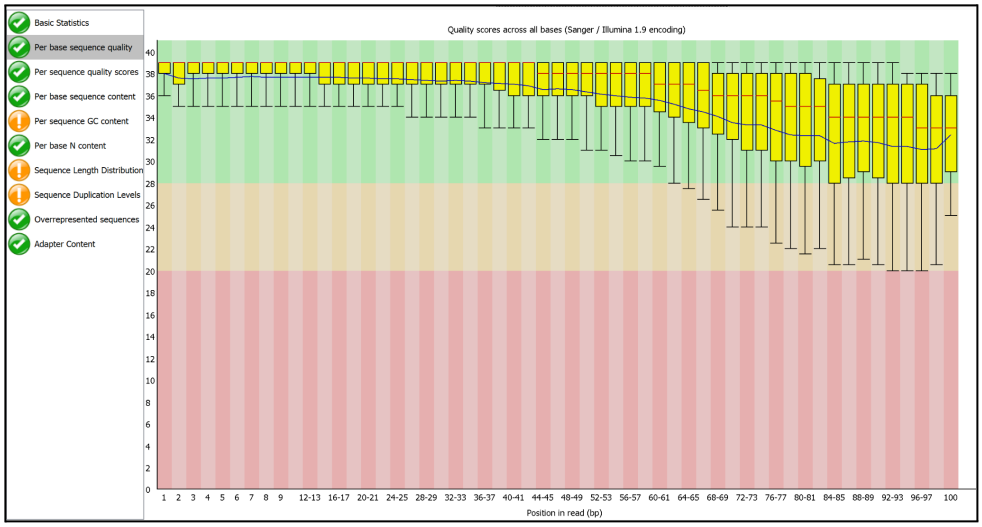
(A)



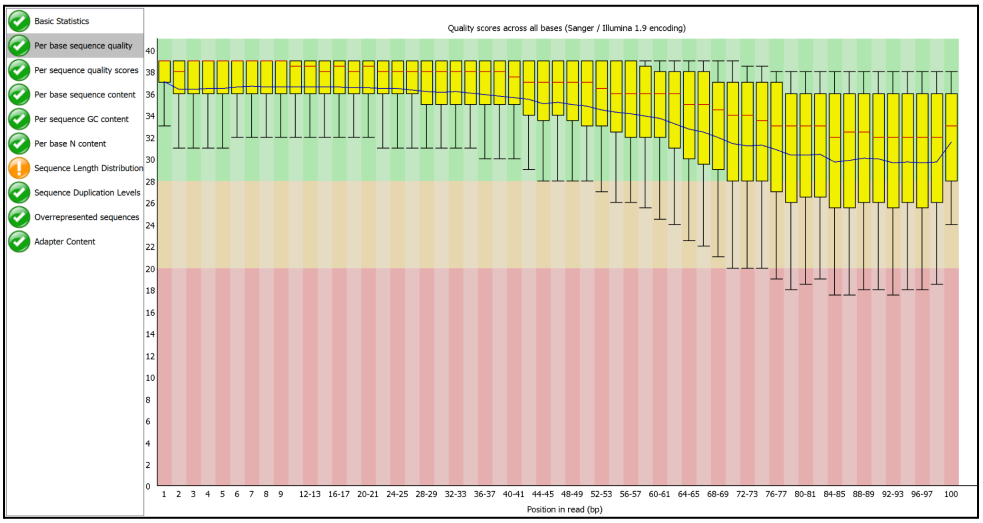
(B)



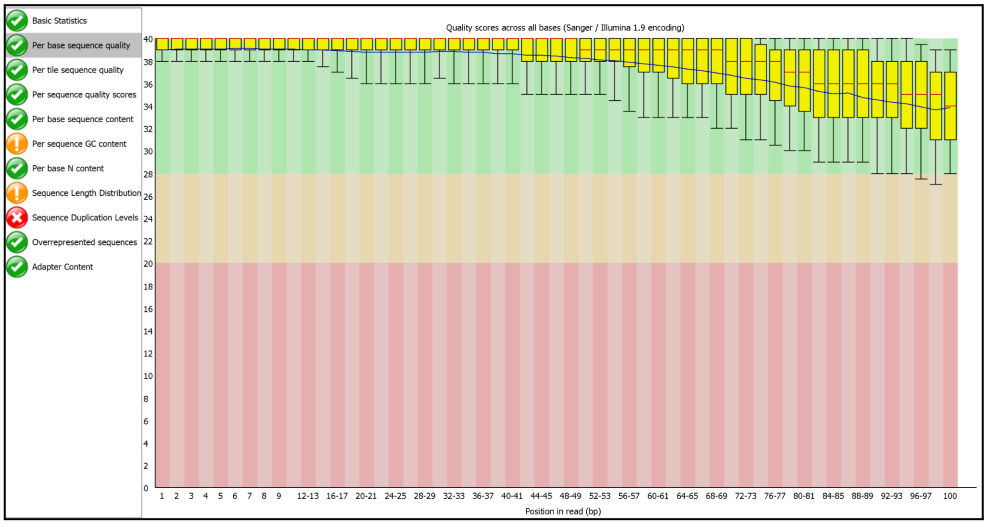
(C)


Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

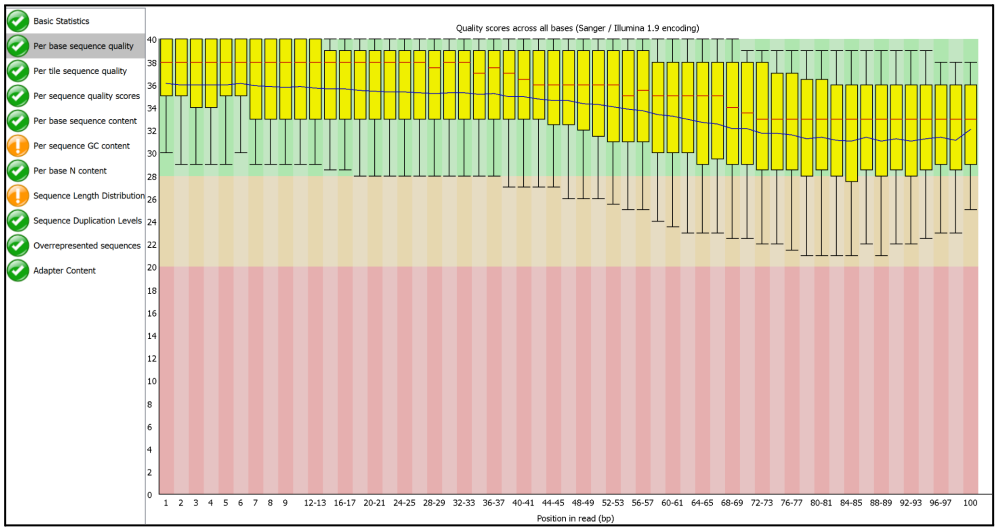
(D)


Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

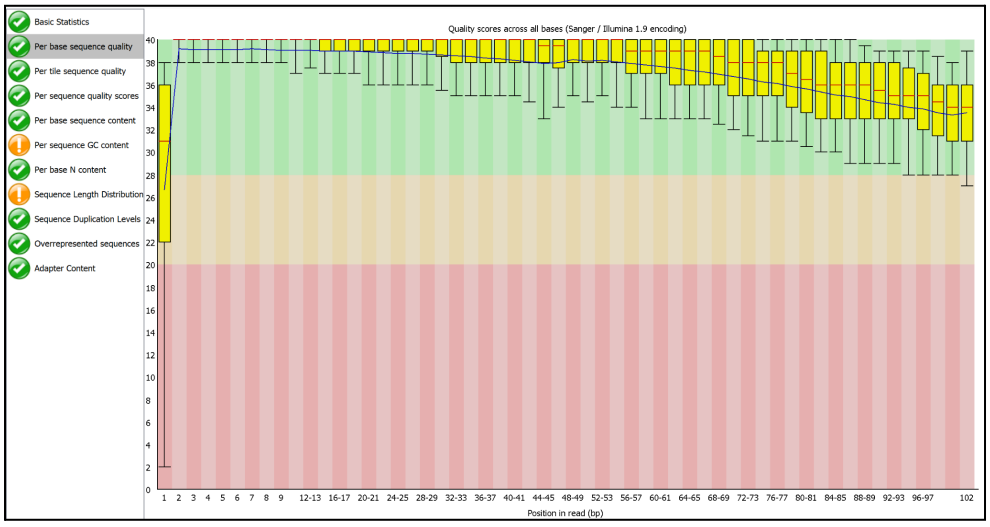
(E)


Basic Statistics
Per base sequence quality
Per tile sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

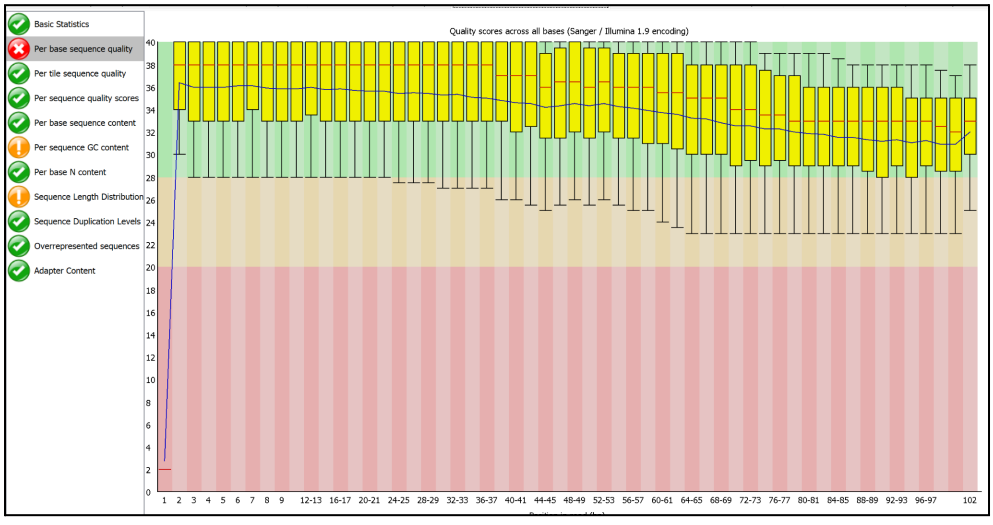
(F)


Basic Statistics
Per base sequence quality
Per tile sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

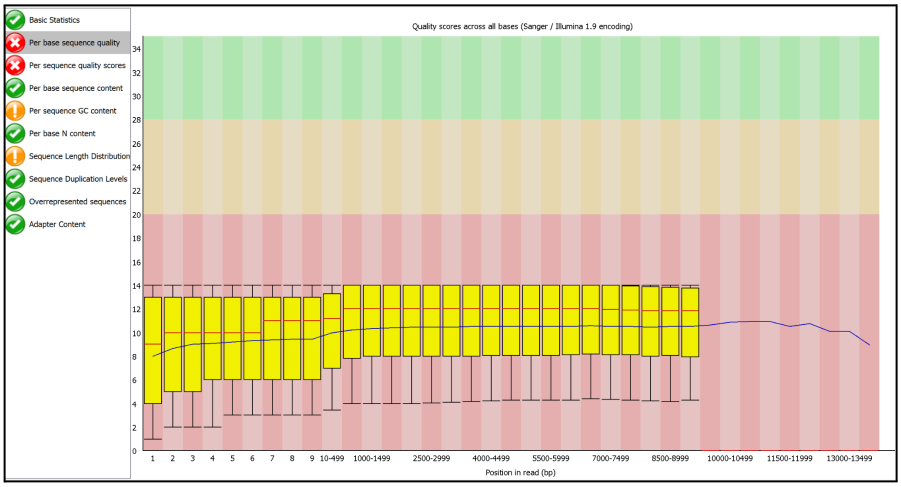
(G)


Basic Statistics
Per base sequence quality
Per tile sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Position in read (bp)

(H)


Basic Statistics
Per base sequence quality
Per tile sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Quality scores across all bases (Sanger / Illumina 1.9 encoding)

(I)



(J)

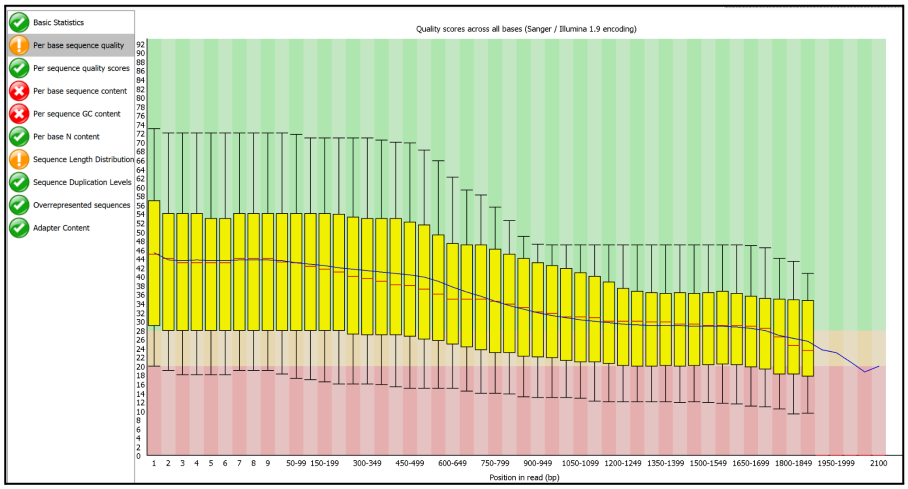


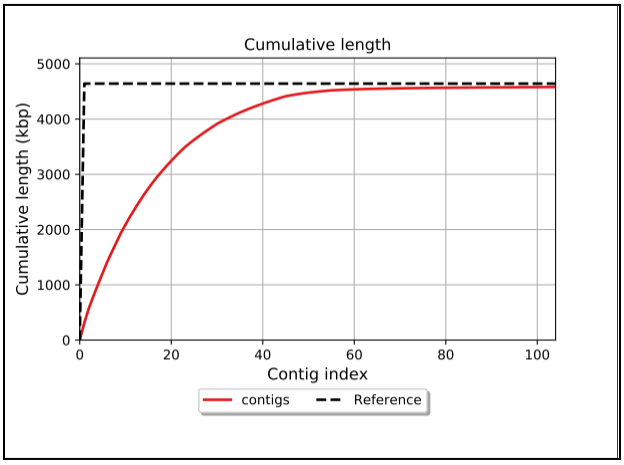
Fig 2:
(A) ERR008613_1_paired.fq, (B) ERR008613_1_unpaired.fq, (C) ERR008613_2_paired.fq, (D) ERR008613_2_unpaired.fq, (E) ERR022075_1_paired.fq, (F) ERR022075_1_unpaired.fq, (G) ERR022075_2_paired.fq, (H) ERR022075_2_unpaired.fq, (I) PacBio_10kb_CLR_trim-out.fq, (J) PacBio_2kb_CCS_500bp_trim-out.fq

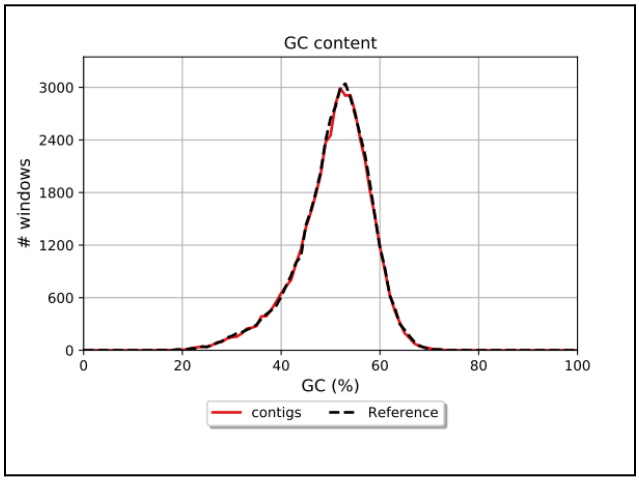
3. Output folder of SPAdes:

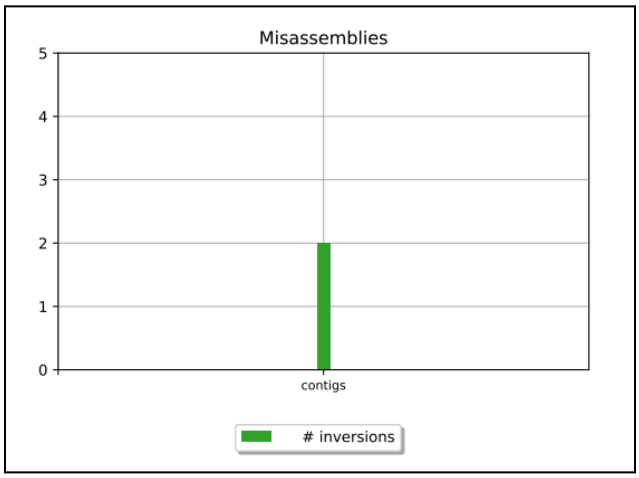
```
assembly_graph_after_simplification.gfa
assembly_graph.fastg
assembly_graph_with_scaffolds.gfa
before_rr.fasta
contigs.fasta
contigs.paths
corrected
dataset.info
input_dataset.yaml
K21
K33
K55
misc
params.txt
pipeline_state
run_spades.sh
run_spades.yaml
scaffolds.fasta
scaffolds.paths
spades.log
tmp
warnings.log
```

Fig 3: Output folder from Illumina only for spades.

4. QUAST Results:







Genome statistics	contigs
Genome fraction (%)	98.474
Duplication ratio	1.002
# genomic features	9190 + 166 part
Largest alignment	312 088
Total aligned length	4 578 576
NGA50	125 933
LGA50	13
Misassemblies	
# misassemblies	2
Misassembled contigs length	202 148
Mismatches	
# mismatches per 100 kbp	0.92
# indels per 100 kbp	16.26
# N's per 100 kbp	0
Statistics without reference	
# contigs	104
Largest contig	312 088
Total length	4 580 516
Total length (>= 1000 bp)	4 562 759
Total length (>= 10000 bp)	4 477 912
Total length (>= 50000 bp)	3 913 512

Extended report

Fig 4: (A) Cumulative length (B) GC content (C) Misassemblies (D) Genome statistics

Discussion:

For the E.coli genome, the quality of the data shaped into a relatively uniform acceptable Q-range. Using trimmomatic solved the per base sequence quality issue, but for some of the PacBio reads, it was still low. Many modifications were made to the reference code given to fit the scenario.

Challenges faced while performing the assignment:

- To decide the parameters to perform each of the slurm scripts for all the steps.
- Analyzing the output files and comparing them after each and every step
- Transferring large data files from remote to local machine
- Understanding the breakdown of the codes

- Understanding time provided and if the script goes wrong, waiting for those hours to try to sort it again after it has finished running.
- Missing small details like names of the files or the locations.