A Project Report

On

**Pre-processing of NGS data**

BY

**SARTHAK SOMANI**

**2021B1A70793H**

Under the supervision of

**JAYAPRAKASH K S**

**SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF**

**BIO F266: STUDY PROJECT**



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)**

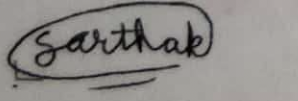
**HYDERABAD CAMPUS**

**(MAY 2023**)

**ACKNOWLEDGMENTS**

**I would like to express my special thanks to my mentor Dr. Jayaprakash K. S. for his time and efforts he provided throughout the project. Your useful advice and suggestions were really helpful to me during the project’s completion. In this aspect, I am eternally grateful to you.**

**I would like to acknowledge that this project was completed entirely by me.**



**Signature**

**Sarthak Somani**

****

**Birla Institute of Technology and Science-Pilani,**

**Hyderabad Campus**

**Certificate**

This is to certify that the project report entitled “**Pre-processing of NGS data”** submitted by Mr. SARTHAK SOMANI (ID No. 2021B1A70793H) in partial fulfillment of the requirements of the course BIO F266, Study Project Course, embodies the work done by him under my supervision and guidance.

**Date: (JAYAPRAKASH K S)**

BITS- Pilani, Hyderabad Campus

**ABSTRACT**

The project is about pre-processing of Next Generation Sequencing data. It is taken from an article already published in a journal. The article talks about designing of effective primers for 8 genes of influenza virus so that each gene present in different strains is expressed in equal quantity and error like amplication of gene from an empty droplets is also reduced. Two strains of influenza virus H3N2 and H1N1 are taken. First step involves learning the basics of NGS and UNIX commands from a website followed by quality control with help of FastQC software. Quality control involves interpretation of graphs and report generated from FastQC software. Second step is mapping of data. STAR aligner was used for this. GTF files were made from fasta files of reference genome using plastid package and converted into ensemble format using pygtftk package. Genome index was generated with help of STAR which was used for mapping paired end reads. Data in log files (output of aligner) for bulk sequencing was used to make graph for number of reads corresponding to a particular gene so as to check if each gene in different strains are produced in same quantity. Both steps are common for bulk sequencing and single cell sequencing. But in second step there are some changes in command for single cell sequencing. Data in matrix.mtx, features.tsv and barcodes.tsv was used to generate raw count matrix with help of scanpy package which was then used for generating a BC rank plot or knee plot. The raw count matrix was converted to normalized count matrix using square root function in scanpy package which was then used for generating various graphs like total number of UMI corresponding to a particular gene, number of BC corresponding to a particular number of UMI and number of cell corresponding to a particular strain.

**CONTENTS**

Title page………………………………………………………………………..1

Acknowledgements……………………………………………………………..2

Certificate…………………………………………………………….................3

Abstract………………………………………………………………................5

1. Basics of NGS ……………………………………………….…………........7

2. Quality Control ………………………………………………………………9

3.Mapping………………………………………………………………………12

………………………………………………………………………………….

Conclusion………………………………………………………………………17

References………………………………………………………………………18

Appendix………………………………………………………………………..19

# **Chapter 1 : Basics of NGS**

## NGS is a way to sequence DNA, RNA upto million base pairs in a very short interval of time i.e it has a high throughput. Our NGS data is generated with help of Illumina Sequencer in our case. Our datasets are derived from sequencing of RNA from H3 and H1 influenza strains. Two strains are not allowed to interact. There are two sets A&B. In each set a paired end read pair for bulk and single cell sequencing is present. A library is made with help of primers. The above process involves 2 cycles of addition of primers and then breaking the whole segment into small segments. Illumina Sequencer can sequence a read of maximum length of 300 bp.

## The small sequences are amplified using bridge amplification. Adapters are added to small sequence on both sides. There are fluorescent dNTPs which could generate a signal as process of base pairing is done. After data is generated we get a processed file in fastq format. This are the read files.

## 

## Fig.1 Workflow of processing of NGS data (https://bioinformaticsworkbook.org/)

## 

## Fig. 2 Structure of fastq file(https://bioinformaticsworkbook.org/)

## Fastq file has a four line format. First line contains SeqID, second line “+”, third line sequence, fourth line quality score. Quality score is calculated from the formula Q=-log10 P where P is probability of incorrect base pairing.

## Next step is the quality control with help of Fast QC software in which the quality of our read file is checked. If quality is not good then adapter trimming is done in which a certain length of reads are selected which have good quality. Next step is mapping with help of STAR aligner in which GTF files are generated so as to generate genome indexes and then finally SAM/BAM files as output of mapping step.After mapping, normalization is carried out so as to bring uniformity in number of UMI counts or gene expression with help of different functions. Next step is downstream analysis which depends on the purpose for which we want to use the data.

## There are two types of sequencing : bulk sequencing and single cell sequencing. In bulk sequencing there are no barcodes and UMIs whereas in single cell sequencing they are present. Barcodes help to identify a cell and unique molecular identifier(UMI) help to identify a particular gene. Single cell sequencing can be used to check different expression of a particular gene in different cell types.

# **Chapter 2 : Quality Control Step**

## In this step we provide fastq file as input to FastQC software which gives us a report as a output. This report includes various graphs, filename, filetype, encoding, total sequences, sequence length. Per base sequence quality graph gives us information about quality score of bases which is average for all bases in different read at that position. Per tile sequence quality graph show average quality score for each tile. The colors are on cold to hot scale. A graph with hotter colors indicate tile having worst quality scores. Per sequence quality scores graph show sequences of a particular quality against the no. of sequences having same quality. Per base sequence content graph shows proportion of each base in a file. A file having equal proportion of all bases is recommended for sequencing as it increases efficiency and accuracy. Per sequence N content graph shows N content across all bases. It presence indicate that we have sequences of shorter length than expected. Sequence length distribution graph shows number of sequences of a particular length against length. Sequence duplication levels graph show different degree of duplication of relative number of sequences. Overrepresented sequences graph shows sequences which are presented more than 0.1% of total reads. Adaptor content graph shows content of adaptor in our sequences which should be zero in ideal case. If it is present then adaptor trimming is done.

## 

Name of fastq file

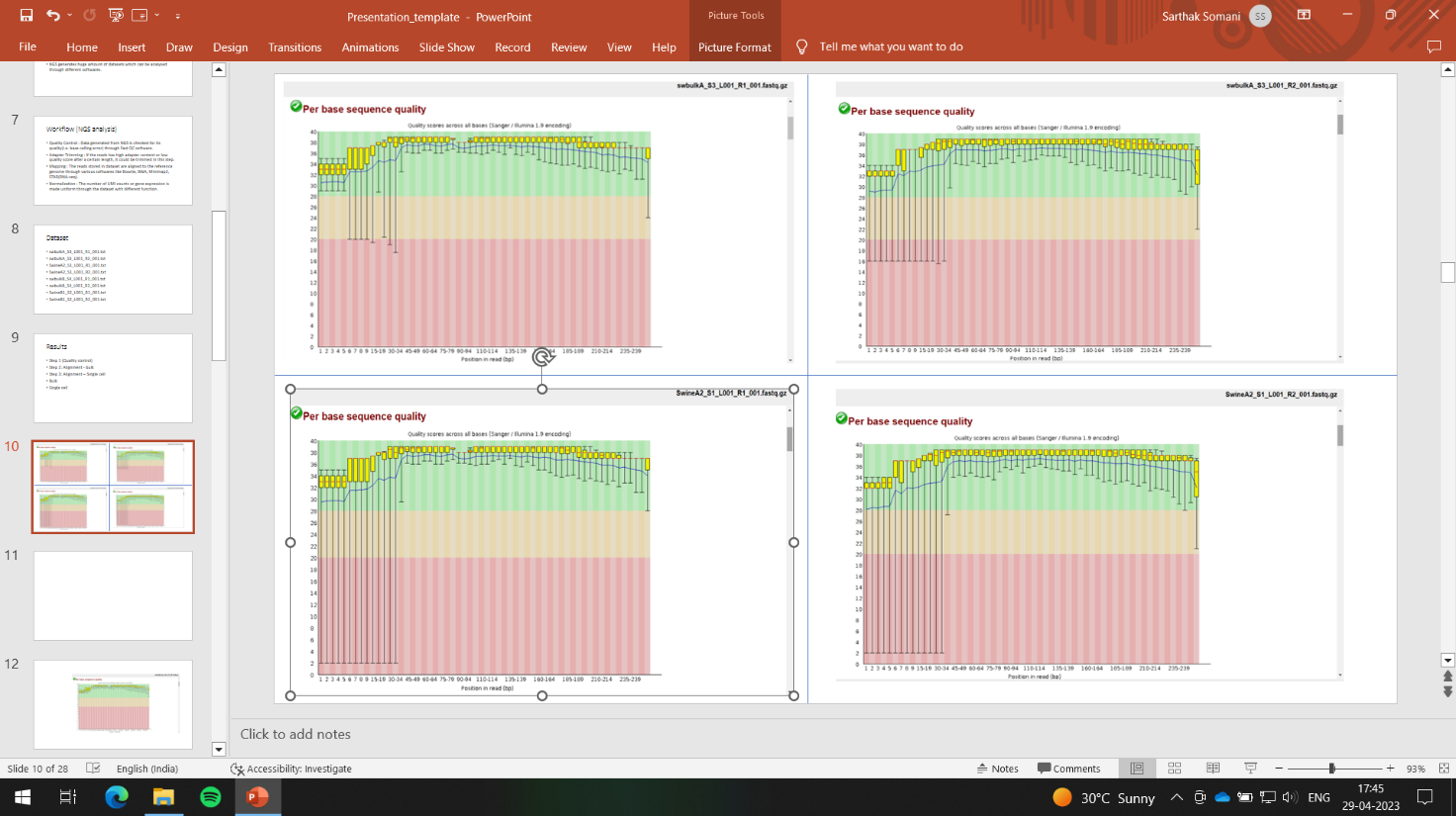
graph

title

Various parameters

## Fig. 3 Structure of QC report

Bulk sequencing



Single cell sequencing

Fig.4 QC reports for SET A

Bulk Sequencing

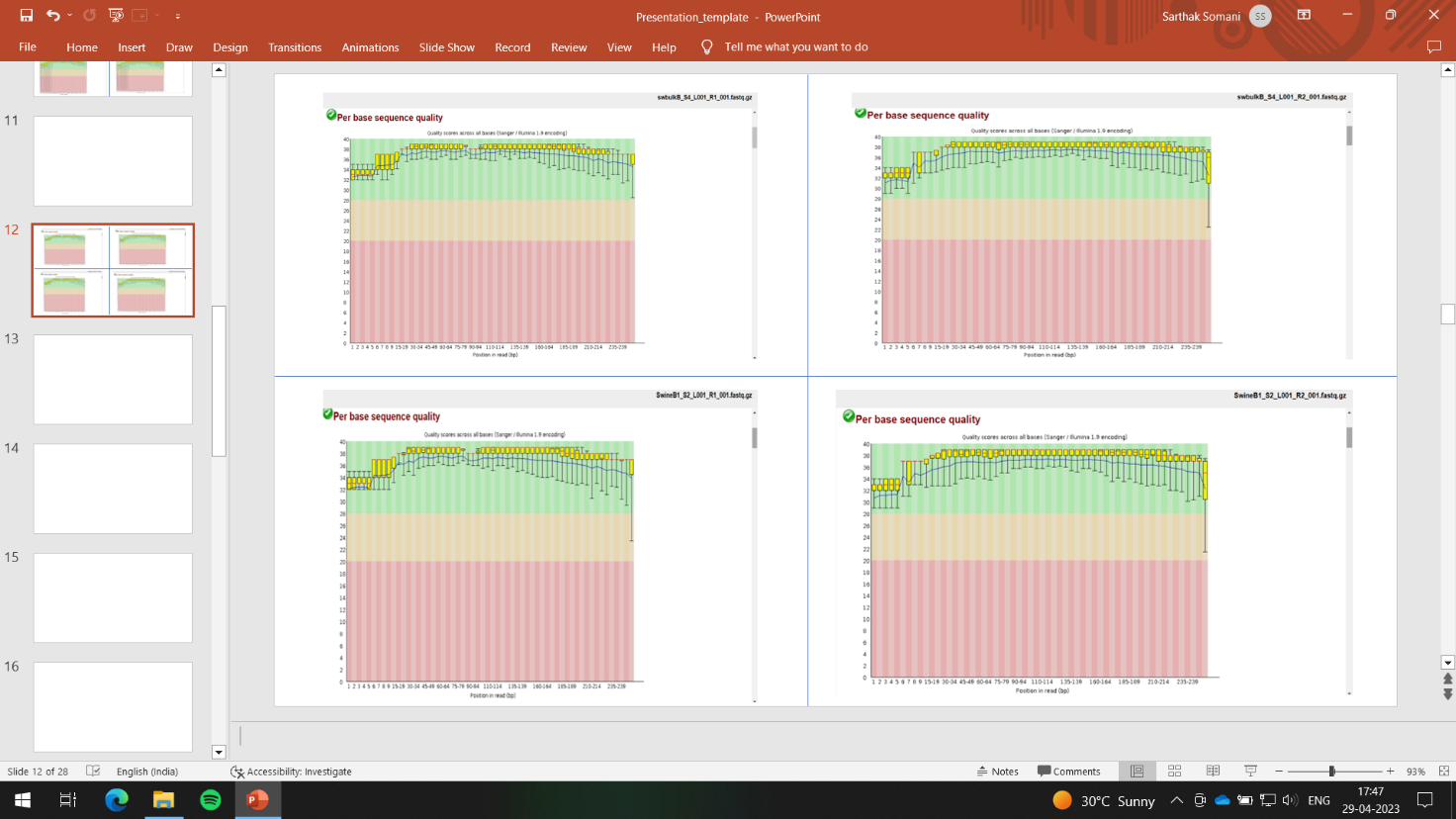


Fig.5 QC reports for SET B

Single cell sequencing

# **Chapter 3 : Mapping**

## Once quality control test is passed by data it is used for mapping. In this step reads are aligned to reference genome. STAR aligner is used. Reference genome in form of fasta file, GTF file are provided as input for generating genome indexes which are further used with fastq files to generate a BAM file in case of bulk sequencing. BAM file is a large file, therefore it is interpreted through featureCounts package. GTF file can be downloaded from EMBL or generated through packages like plastid and pygtftk. Log files result and graphs in case of bulk sequencing are shown in table below:

## 

## Fig.6 Log file results for Bulk sequencing of SET A & B

## 

## Fig.7 Number of reads corresponding to a particular gene for SET A

## 

## Fig.8 Number of reads corresponding to a particular gene for SET B

## In case of single cell sequencing, barcodes.tsv, features.tsv and matrix.mtx are obtained. Matrix.mtx is a file consisting of different matrixes in different layers. It can be opened through scanpy package. Raw count matrixes was extracted from it and BC rank plot was generated from this. This plot gives the idea of selection of barcodes having more than particular number of UMI.

## 

## Fig.9 BC rank plot for SET A

## 

## Fig.10 BC rank plot for SET B

## The raw count matrixes were converted into normalized count matrixes so as to bring uniformity in gene expression. Normalized count matrixes were generated using a square root function. Various graphs were generated using normalized count matrix.

## 

## Fig.11 Number of UMIs for a particular gene for SET A

## 

## Fig.12 Number of UMIs for a particular gene for SET B

## 

## Fig.13 Number of BC for a particular length of UMI

## 

## Fig.14 Number of cells of a particular strain

## 

# **Conclusion**

* Cells expressing more are less in number and vice versa. Hence the normalization process is required to get a uniform expression in all cells.
* There is homogeneity in expression of a particular gene in both strains which means our primers have similar efficiencies for a particular gene in both strains. This verifies the main aim of the experiment.
* There is presence of reassortment and both strain which tells us that there is some error in library preparation as reassortment was not allowed in this experiment. The error might be in droplet capturing part or library preparation part.
* It can be seen that maximum number of barcodes have 9 UMIs .
* From graphs we can conclude that our experiment was successful with some errors. Now this data can be taken for any downstream analysis

# **References**

## <https://bioinformaticsworkbook.org/>

## <https://github.com/ONeillMB1/>

## <https://github.com/alexdobin/STAR/blob/master/>

## [https://sampled.com/bulk-rna-sequencing-vs-single-cell-rna-sequencing/#:~:text=So%2C%20Bulk%20RNA%20sequencing%20(bulk,gene%20expression%20in%20individual%20cells](https://sampled.com/bulk-rna-sequencing-vs-single-cell-rna-sequencing/).

## <https://www.biorxiv.org/content/10.1101/770388v3.full>

# **Appendix**

## Code for generating genome indexes

## STAR --runMode genomeGenerate --runThread 4 --genomeDir /path to genome directory where index will be stored --genomeFastaFiles /path to reference genome in fasta format --genomeSAindexNbases 5 --sjdbGTFfile /path to GTF files

## Code for mapping of bulk sequencing

## STAR --runMode alignReads --genomeDir /path to genome directory where index is stored --readFilesIn /path to fastq files --outFilterScoreMinOverLread 0 --outFilterMatchNminOverLread 0 --outFilterMatchNmin 30 --outSAMtype BAM SortedByCoordinate --limitBAMsortRAM 1010005903

## Code for mapping single cell sequencing

STAR --runMode alignReads --genomeDir /path to genome directory where index is stored --readFilesIn /path to fastq files --runThreadN 16 --soloType CB\_UMI\_Complex --soloCBposition 0\_0\_0\_15 0\_20\_0\_35 0\_40\_0\_55 0\_60\_0\_75 --soloUMIposition 0\_94\_0\_103 --soloCBwhitelist /path to BC combination text file --soloCBmatchWLtype Exact --soloCellFilter None --outFilterScoreMinOverLread 0 --outFilterMatchNminOverLread 0 --outFilterMatchNmin 30 --soloStrand Reverse --outSAMtype BAM SortedByCoordinate --outSAMattributes NH HI nM AS CR UR CB UB GX GN sM --limitBAMsortRAM 1028683128

Code for generating raw count matrix, normalization

import scanpy as sc

import pandas as pd

import numpy as np

// for generating raw count matrix

adata = sc.read\_10x\_mtx("/path to matrix.mtx,barcode.tsv,features.tsv files",var\_names='gene\_ids')

test = pd.DataFrame(adata.to\_df())

test1.to\_csv("/path to raw count matrix")

// for normalization

adata.layers["raw"] = adata.X.copy()

adata.layers["sqrt\_norm"] = np.sqrt(

sc.pp.normalize\_total(adata, inplace=False)["X"]

)

test = pd.DataFrame(adata.to\_df(layer="sqrt\_norm"))

test.to\_csv("/path to normalized count matrix")