Prepared by Liya Simon

RNA-Seq Analysis Pipeline

This document outlines the steps and commands required to perform bulk RNA-seq analysis, from data preparation through differential expression analysis. The analysis includes quality control, trimming, alignment, transcript assembly, and differential expression analysis using the following tools:

- Data Download
- Quality control: FastQC
- Trimming low-quality bases and adapters: Trimmomatic
- Read alignment: HISAT2
- Transcript assembly and quantification: StringTie
- SAM/ BAM conversion
- Gff compare
- Differential expression analysis: Ballgown

Before starting the analysis we need a platform in our system, here we are doing it in linux terminal. Then create an environment using the following code.

```
Open Terminal - Cntrl+Alt+T

Go to Working Directory - cd Desktop/ngs/

conda create –name rnaseq

conda activate rnaseq
```

Data Download

In this pipeline, we use the following command to download the data(1).

```
wget ftp://ftp.ccb.jhu.edu/pub/RNAseq_protocol/chrX_data.tar.gz
```

```
tar -xzf chrX_data.tar.gz
```

Colour of codes : Red - data, Green - tools, Parameters - Black, output- Blue

This command downloads chrX_data.tar.gz, after downloading, you can extract the contents using tar -xzf chrX_data.tar.gz. This will uncompress and unpack the archive, making the files and folders within it accessible for further analysis.

The data folder contains gene annotations, the genome, sample FASTQ files, and HISAT2 index files required for this RNA-seq pipeline.

Data Preprocessing

1. Quality Control

The following command uses FastQC to assess the quality of raw RNA-Seq data before any downstream analysis. Fastqc is a tool used to assess the quality of raw sequencing data, output stored in qc_output/. So first we need to create an output folder using mkdir(linux command).

```
mkdir qc_output/.
```

```
fastqc chrX_data/samples/ERR188044_chrX_1.fastq.gz
chrX_data/samples/ERR188044_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188104_chrX_1.fastq.gz
chrX_data/samples/ERR188104_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188234_chrX_1.fastq.gz
chrX_data/samples/ERR188234_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188245_chrX_1.fastq.gz
chrX_data/samples/ERR188245_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188257_chrX_1.fastq.gz
chrX_data/samples/ERR188257_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188273_chrX_1.fastq.gz
chrX_data/samples/ERR188273_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188337_chrX_1.fastq.gz
chrX_data/samples/ERR188337_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188383_chrX_1.fastq.gz
chrX_data/samples/ERR188383_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188401_chrX_1.fastq.gz
chrX_data/samples/ERR188401_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188428_chrX_1.fastq.gz
chrX_data/samples/ERR188428_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR188454_chrX_1.fastq.gz
chrX_data/samples/ERR188454_chrX_2.fastq.gz -o qc_output/
```

```
fastqc chrX_data/samples/ERR204916_chrX_1.fastq.gz
chrX_data/samples/ERR204916_chrX_2.fastq.gz -o qc_output/
```

2. Trimming

This trimming process is based on the quality metrics obtained from tools like FastQC, ensuring only high-quality sequences are used for alignment. So if there are any quality issues of residues we can clean the data using the following code, which uses trimmomatic tool to do this.

(In this data it is not necessary to do trimming, so we can jump to the Assembly)

```
trimmomatic PE chrX_data/samples/ERR188044_chrX_1.fastq.gz
chrX_data/samples/ERR188044_chrX_2.fastq.gz \

paired/ERR188044_chrX_1_paired.fq.gz unpaired/ERR188044_chrX_1_unpaired.fq.gz \

paired/ERR188044_chrX_2_paired.fq.gz unpaired/ERR188044_chrX_2_unpaired.fq.gz \

ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3
SLIDINGWINDOW:4:15 MINLEN:36
```

Read Alignment

1. Build HISAT2 Index

The foremost step of alignment is to build an index of the reference genome using HISAT2 for efficient alignment.

```
hisat2-build Homo_sapiens.GRCh38.dna.primary_assembly.fa hisat_index
```

hisat2-build: Creates an index from the reference genome file.

hisat_index: The output index prefix for HISAT2.

2. Hisat2 alignments (explain the code)

The following command uses **HISAT2**, an alignment tool, to align paired-end RNA-seq reads to a reference genome. code to do it. In this step, we align the reads with the reference genome.

```
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1
chrX_data/samples/ERR188044_chrX_1.fastq.gz -2
chrX_data/samples/ERR188044_chrX_2.fastq.gz -S ERR188044_chrX.sam
```

1. Hista2- tool
2. -p : is the flag for threads.
3. 8 : means the program will utilize 8 CPU cores to speed up computation.
4. --dta: This option ensures the aligner (e.g., HISAT2) produces output that is suitable for downstream transcriptome analysis tools, such as

StringTie.Specifically, it reports alignments for reads across splice junctions and retains certain details in the SAM/BAM files required for transcript assembly or quantification.

5. -x : is followed by the path to the genome index built using the alignment tool (e.g., HISAT2-build).
6. -1, -2: stands for forward and reverse strands.
7. ERR188044_chrX_1.fastq.gz and ERR188044_chrX_2.fastq.gz : This command aligns paired-end RNA-Seq reads
8. -S : Store output in sam format
9. ERR188044_chrX.sam : output sam file

hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188104_chrX_1.fastq.gz -2 chrX_data/samples/ERR188104_chrX_2.fastq.gz -S ERR188104_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188234_chrX_1.fastq.gz -2 chrX_data/samples/ERR188234_chrX_2.fastq.gz -S ERR188234_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188245_chrX_1.fastq.gz -2 chrX_data/samples/ERR188245_chrX_2.fastq.gz -S ERR188245_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188257_chrX_1.fastq.gz -2 chrX_data/samples/ERR188257_chrX_2.fastq.gz -S ERR188257_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188273_chrX_1.fastq.gz -2 chrX_data/samples/ERR188273_chrX_2.fastq.gz -S ERR188273_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188337_chrX_1.fastq.gz -2 chrX_data/samples/ERR188337_chrX_2.fastq.gz -S ERR188337_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188383_chrX_1.fastq.gz -2 chrX_data/samples/ERR188383_chrX_2.fastq.gz -S ERR188383_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188401_chrX_1.fastq.gz -2 chrX_data/samples/ERR188401_chrX_2.fastq.gz -S ERR188401_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188428_chrX_1.fastq.gz -2

chrX_data/samples/ERR188428_chrX_2.fastq.gz -S ERR188428_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR188454_chrX_1.fastq.gz -2 chrX_data/samples/ERR188454_chrX_2.fastq.gz -S ERR188454_chrX.sam
hisat2 -p 8 --dta -x chrX_data/indexes/chrX_tran -1 chrX_data/samples/ERR204916_chrX_1.fastq.gz -2 chrX_data/samples/ERR204916_chrX_2.fastq.gz -S ERR204916_chrX.sam

3. SAM to BAM

The command samtools sort is an essential step in processing RNA-Seq data after alignment. In this process, sort the alignments in the SAM file and save the result as a compressed BAM file (binary format) using the following code.

```
samtools sort -@ 8 -o ERR188044_chrX.bam ERR188044_chrX.sam
```

1. samtools sort : command to sort an alignment file, SAMtools organizes alignment data in a specific order.
2. -@ 8 : Specifies the number of **threads** (8 in this case) to use for faster processing. This makes the sorting process quicker on machines with multiple CPU cores-o ERR188044_chrX.bam: output
3. ERR188044_chrX.sam: This is the input file containing the alignments in SAM format (from the HISAT2 step).

samtools sort -@ 8 -o ERR188104_chrX.bam ERR188104_chrX.sam
samtools sort -@ 8 -o ERR188234_chrX.bam ERR188234_chrX.sam
samtools sort -@ 8 -o ERR188245_chrX.bam ERR188245_chrX.sam
samtools sort -@ 8 -o ERR188257_chrX.bam ERR188257_chrX.sam
samtools sort -@ 8 -o ERR188273_chrX.bam ERR188273_chrX.sam
samtools sort -@ 8 -o ERR188337_chrX.bam ERR188337_chrX.sam
samtools sort -@ 8 -o ERR188383_chrX.bam ERR188383_chrX.sam
samtools sort -@ 8 -o ERR188401_chrX.bam ERR188401_chrX.sam

samtools sort -@ 8 -o ERR188428_chrX.bam ERR188428_chrX.sam
samtools sort -@ 8 -o ERR188454_chrX.bam ERR188454_chrX.sam
samtools sort -@ 8 -o ERR204916_chrX.bam ERR204916_chrX.sam

Transcript Assembly

3. Stringtie

The command is for running Stringtie, a popular tool used in RNA-Seq analysis to assemble transcripts, estimate their abundances, and generate transcript annotations. The output will be a 'gtf' file containing the assembled transcripts for the sample ERR188044.

```
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188044_chrX.gtf -l ERR188044 ERR188044_chrX.bam
```

1. Stringtie: The command to run the StringTie program.
2. -p 12: Specifies the number of CPU threads to use for processing. In this case, StringTie will use 12 threads to speed up the computation.
3. -G genes/chrX.gtf: The reference annotation file in GTF format (here, genes/chrX.gtf).
4. -o ERR188044_chrX.gtf: Specifies the output file where the assembled transcripts for this sample will be saved (in GTF format). Here, the output will be written to ERR188044_chrX.gtf.
5. -l ERR188044: Sets a prefix for the transcript IDs generated by StringTie. In this case, transcripts assembled from this sample will have IDs prefixed with ERR188044.
6. ERR188044_chrX.bam: This is the input file, a sorted BAM file containing the mapped RNA-Seq reads for this sample (ERR188044).

stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188104_chrX.gtf -l ERR188104 ERR188104_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188234_chrX.gtf -l ERR188234 ERR188234_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188245_chrX.gtf -l ERR188245

Colour of codes : Red - data, Green - tools, Parameters - Black, output- Blue

ERR188245_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188257_chrX.gtf -l ERR188257 ERR188257_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188273_chrX.gtf -l ERR188273 ERR188273_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188337_chrX.gtf -l ERR188337 ERR188337_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188383_chrX.gtf -l ERR188383 ERR188383_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188401_chrX.gtf -l ERR188401 ERR188401_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188428_chrX.gtf -l ERR188428 ERR188428_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR188454_chrX.gtf -l ERR188454 ERR188454_chrX.bam
stringtie -p 12 -G chrX_data/genes/chrX.gtf -o ERR204916_chrX.gtf -l ERR204916 ERR204916_chrX.bam

4. Merge transcripts from all samples.

```
stringtie --merge -p 12 -G chrX_data/genes/chrX.gtf -o stringtie_merged.gtf mergelist.txt
```

1. Stringtie: This is the command to run the StringTie program.
2. --merge: This option tells StringTie to merge multiple transcript assemblies into one unified GTF file.
3. -p 12: StringTie should use 12 CPU threads for faster processing.
4. -G genes/chrX.gtf: This is the reference annotation file (in GTF format).It is used to guide the merging process.
5. -o stringtie_merged.gtf: Specifies the output file where the merged transcripts will be written.
6. Mergelist.txt: This text file contains a list of GTF files to be merged.

5. Examine how the transcripts compare with the reference annotation

```
gffcompare –r chrX.gtf –G –o merged stringtie_merged.gtf
```

6. Estimate transcript abundances and create table counts for Ballgown:

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188044/ERR188044_chrX.gtf ERR188044_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188104/ERR188104_chrX.gtf ERR188104_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188104/ERR188104_chrX.gtf ERR188104_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188245/ERR188245_chrX.gtf ERR188245_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188257/ERR188257_chrX.gtf ERR188257_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188273/ERR188273_chrX.gtf ERR188273_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188337/ERR188337_chrX.gtf ERR188337_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188383/ERR188383_chrX.gtf ERR188383_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188401/ERR188401_chrX.gtf ERR188401_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188428/ERR188428_chrX.gtf ERR188428_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR188454/ERR188454_chrX.gtf ERR188454_chrX.bam
```

```
stringtie -e -B -p 12 -G stringtie_merged.gtf -o
ballgown/ERR204916/ERR204916_chrX.gtf ERR204916_chrX.bam
```

Data Quantification

Colour of codes : Red - data, Green - tools, Parameters - Black, output- Blue

REFERENCE:

1. https://davetang.org/muse/2017/10/25/getting-started-hisat-stringtie-ballgown/
2. https://bioinformatics-core-shared-training.github.io/RNAseq_September_2018/Supplementary_Materials/S1_Getting_raw_reads_from_SRA.html