RNA-Seq Analysis Pipeline for SRA Accessions

This document outlines the complete RNA-Seq analysis pipeline used for processing SRA accessions. Each section details the tool, its purpose, and the exact commands (with comments for clarity and reproducibility).

Table of Contents

---

1. Data Conversion

Tools:
- **SRA Toolkit** (`prefetch`, `fastq-dump`)

Purpose:
Convert the SRA files to FASTQ format.

```
# Convert the SRA files to FASTQ format
fastq-dump --skip-technical --read-filter pass SRR11412215 --outdir Data
fastq-dump --skip-technical --read-filter pass SRR11412216 --outdir Data
fastq-dump --skip-technical --read-filter pass SRR11412230 --outdir Data
fastq-dump --skip-technical --read-filter pass SRR11412229 --outdir Data
```

---

2. Quality Control (QC) of Raw Reads

Tool:
- **FastQC**

Purpose:
Assess the quality of the raw FASTQ files.

```
fastqc Data/SRR11412215_pass.fastq Data/SRR11412216_pass.fastq
Data/SRR11412229_pass.fastq Data/SRR11412230_pass.fastq -o Analysis
```

---

3. Read Trimming

Tool:
- **Trimmomatic**

Purpose:
Remove low-quality bases and adapter sequences from the reads.

```
trimmomatic SE -threads 4 -phred33 Data/SRR11412215_pass.fastq
Data/controlled_trimmed.fastq LEADING:3 TRAILING:10 SLIDINGWINDOW:4:15
MINLEN:36
trimmomatic SE -threads 4 -phred33 Data/SRR11412216_pass.fastq
Data/controlled1_trimmed.fastq LEADING:3 TRAILING:10 SLIDINGWINDOW:4:15
MINLEN:36
trimmomatic SE -threads 4 -phred33 Data/SRR11412229_pass.fastq
Data/infected1_trimmed.fastq LEADING:3 TRAILING:10 SLIDINGWINDOW:4:15
MINLEN:36
trimmomatic SE -threads 4 -phred33 Data/SRR11412230_pass.fastq
Data/infected_trimmed.fastq LEADING:3 TRAILING:10 SLIDINGWINDOW:4:15
MINLEN:36
```

---

4. Read Alignment to the Reference Genome

Tools:
- **HISAT2**

Purpose:
Align the trimmed reads to the reference genome.

```
hisat2 -x Reference_Genome/grch38/genome -U Data/infected_trimmed.fastq -S
Alignment/infected.sam
hisat2 -x Reference_Genome/grch38/genome -U Data/infected1_trimmed.fastq -S
Alignment/infected1.sam
hisat2 -x Reference_Genome/grch38/genome -U Data/controlled_trimmed.fastq -S
Alignment/controlled.sam
hisat2 -x Reference_Genome/grch38/genome -U Data/controlled1_trimmed.fastq -S
Alignment/controlled1.sam
```

---

5. SAM-to-BAM Conversion, Sorting, and Indexing

Tool:
- **Samtools**

Purpose:
Convert, sort, and index BAM files.

```
samtools sort -o Alignment/infected.bam Alignment/infected.sam
samtools sort -o Alignment/controlled.bam Alignment/controlled.sam
samtools sort -o Alignment/infected1.bam Alignment/infected1.sam
samtools sort -o Alignment/controlled1.bam Alignment/controlled1.sam

samtools index Alignment/infected.bam
samtools index Alignment/controlled.bam
samtools index Alignment/infected1.bam
samtools index Alignment/controlled1.bam
```

---

6. Quantification

Tool:
- **featureCounts**

Purpose:
Count reads per gene from the sorted BAM files.

```
featureCounts -T 4 -a Reference_Genome/Homo_sapiens.GRCh38.106.gtf.gz -o
Counts/gene_counts.txt Alignment/controlled.bam Alignment/infected.bam
Alignment/infected1.bam Alignment/controlled1.bam
```

---

7. Differential Expression Analysis

Tool:
- **R** with **DESeq2**

Purpose:
Analyze gene count data to identify differentially expressed genes.

```
library("DESeq2")
counts <- read.table("Counts/gene_counts.txt", header = TRUE, row.names = 1)
sampleInfo <- data.frame(
  row.names = colnames(counts),
  condition = c("controlled", "infected", "controlled", "infected")
)
dds <- DESeqDataSetFromMatrix(countData = counts, colData = sampleInfo,
design = ~ condition)
dds <- DESeq(dds)
res <- results(dds)
write.csv(as.data.frame(res), file = "differential_expression_results.csv")
```

---

8. Visualization

Tool:
- R

Purpose:
Visualize the differential expression analysis results.

```
library("ggplot2")
library("ggrepel")

# MA Plot
plotMA(res, main="MA Plot", ylim=c(-2,2))

# PCA Plot (using variance stabilizing transformation)
vsd <- vst(dds, blind=FALSE)
plotPCA(vsd, intgroup="condition")

# Generate a volcano plot

# Compute -log10(p-value)
res$logPadj <- -log10(res$padj + 1e-10)

# Create a significance column
res$significance <- "Not Significant"
res$significance[which(res$pvalue < 0.05 & abs(res$log2FoldChange) >= 1)] <-
"Significant"

volcano <- ggplot(as.data.frame(res), aes(x = log2FoldChange, y = logPadj,
color = significance)) +
  geom_point(alpha = 0.8, size = 2) +
  # Add threshold lines for fold change and significance
  geom_vline(xintercept = c(-1, 1), linetype = "dotted", color = "black") +
  geom_hline(yintercept = -log10(0.05), linetype = "dotted", color = "black")
+
  theme_minimal() +
  scale_color_manual(values = c("Significant" = "orange", "Not Significant" =
"cyan")) +
  xlab("log2 Fold Change") +
  ylab("-log10(P-adj)") +
  ggtitle("Volcano Plot of Differential Expression") +
  theme(plot.title = element_text(hjust = 0.5),
        legend.title = element_blank())


top_genes <- head(as.data.frame(res[order(res$logP), ]), 10)
 volcano <- volcano +
   geom_text_repel(data = top_genes,
```

```
    aes(label = rownames(top_genes)),
    size = 3,
    box.padding = 0.3,
    point.padding = 0.2)
```

9. Pipeline Execution

To run the entire pipeline: 1. **Set Up Environment:**
Import the Conda environment with `conda env create -f environment/environment.yml` and activate it. 2. **Data Acquisition:**
Run `bash Pipeline/data.sh` to download necessary data and references. 3. **RNA-Seq Processing:**
Execute `bash Pipeline/RNA_Seq.sh` to process the data from FASTQ conversion through gene counting. 4. **DE Analysis & Visualization:**
Run the R scripts in the `Pipeline/` folder to perform differential expression analysis and generate plots.

This document serves as a detailed guide to help users understand and reproduce the RNA-Seq differential expression pipeline provided in this repository.