**Step 0: Downloading Data**

**1. Installing and Loading Libraries**

if (!require("BiocManager", quietly = TRUE))

install.packages("BiocManager")

BiocManager::install("airway")

BiocManager::install("DESeq2")

install.packages('tidyverse')

library(DESeq2)

library(tidyverse)

library(airway)

**Explanation:**

* The BiocManager is a package manager for Bioconductor packages like DESeq2.
* DESeq2 is the primary package for DGE analysis.
* tidyverse provides utilities for data manipulation and visualization.
* The airway dataset is a real dataset used for practice.

**2. Loading the Dataset**

data(airway)

airway

**Significance:**  
The airway dataset contains gene expression count data (from an RNA-Seq experiment) of airway smooth muscle cells. The experiment involves treated and untreated samples with dexamethasone (a corticosteroid).

**3. Extracting and Modifying Metadata**

sample\_info <- as.data.frame(colData(airway))

sample\_info <- sample\_info[,c(2,3)]

sample\_info$dex <- gsub('trt', 'treated', sample\_info$dex)

sample\_info$dex <- gsub('untrt', 'untreated', sample\_info$dex)

names(sample\_info) <- c('cellLine', 'dexamethasone')

write.table(sample\_info, file = "sample\_info.csv", sep = ',', col.names = TRUE, row.names = TRUE, quote = FALSE)

**Explanation:**

* **colData(airway)** extracts the sample metadata like treatment condition, cell line, etc.
* Metadata is simplified to include only **cell line** and **dexamethasone treatment**.
* "trt" and "untrt" are renamed to "treated" and "untreated" for clarity.
* Metadata is saved as sample\_info.csv.

**4. Extracting Count Data**

countsData <- assay(airway)

write.table(countsData, file = "counts\_data.csv", sep = ',', col.names = TRUE, row.names = TRUE, quote = FALSE)

**Significance:**

* **assay()** retrieves raw count data of genes across samples.
* The counts matrix and metadata are saved locally for future use.

**Step 1: Reading Data**

**1. Reading Counts and Metadata**

counts\_data <- read.csv('/content/counts\_data.csv')

sample\_info <- read.csv('/content/sample\_info.csv')

**Significance:**

* This step ensures the downloaded counts data and sample information are loaded correctly for further analysis.

**2. Verifying Matching Columns**

all(colnames(counts\_data) %in% rownames(sample\_info))

all(colnames(counts\_data) == rownames(sample\_info))

**Explanation:**

* Ensures that the **column names** in the counts matrix match the **row names** in the sample metadata.
* Both datasets must align properly for DESeq2 to perform DGE analysis.

**Step 2: Constructing DESeq2 Dataset**

**1. Constructing the DESeq Dataset**

dds <- DESeqDataSetFromMatrix(countData = counts\_data,

colData = sample\_info,

design = ~ dexamethasone)

**Significance:**

* DESeqDataSetFromMatrix() creates a DESeq2 object that links the **gene expression counts** and the **metadata**.
* design = ~ dexamethasone specifies the **experimental design**:
  + The analysis compares "treated" vs "untreated" conditions.

**2. Prefiltering Low Counts**

keep <- rowSums(counts(dds)) >= 10

dds <- dds[keep,]

**Significance:**

* Removes genes with **low counts** (less than 10 reads) across all samples.
* Filtering improves statistical power by reducing noise from non-informative genes.

**3. Setting Reference Level**

dds$dexamethasone <- relevel(dds$dexamethasone, ref = "untreated")

**Significance:**

* Sets "untreated" as the **reference level** for comparison against "treated."
* Results will be reported as "treated relative to untreated."

**4. Running DESeq**

dds <- DESeq(dds)

**Significance:**

* Runs the **DESeq2 pipeline** for DGE analysis:
  + Estimates size factors to normalize for sequencing depth.
  + Estimates dispersions (variability of gene expression).
  + Fits a Negative Binomial Generalized Linear Model (GLM).

**Step 3: Exploring Results**

**1. Extracting Results**

res <- results(dds)

summary(res)

**Significance:**

* results() extracts the results table:
  + **log2FoldChange**: Indicates up/down-regulation of genes.
  + **p-value**: Tests significance.
  + **adjusted p-value**: Corrects for multiple testing (default threshold is 0.1).
* The summary gives an overview of how many genes are differentially expressed.

**2. Adjusting P-value Threshold**

res.0.01 <- results(dds, alpha = 0.01)

summary(res.0.01)

**Significance:**

* Lowers the adjusted p-value cutoff to **0.01** for stricter significance.

**3. Comparing Results**

results(dds, contrast = c('dexamethasone','treated','untreated'))

**Significance:**

* Specifies a custom contrast for comparison between "treated" and "untreated."

**Step 4: Visualizing Results**

**1. MA Plot**

plotMA(res)

**Significance:**

* **MA plot** visualizes gene expression changes:
  + **X-axis**: Average expression (mean).
  + **Y-axis**: log2FoldChange.
  + Blue dots represent significantly differentially expressed genes.

**Key Insights from the Workflow**

1. **Normalization**: Accounts for sequencing depth differences.
2. **Statistical Testing**: Uses Negative Binomial GLM for robust significance testing.
3. **Prefiltering**: Removes low-count genes to reduce noise.
4. **Visualization**: MA plots highlight significantly up/down-regulated genes.

This workflow is critical for identifying genes that respond to a treatment (dexamethasone) and ensuring reproducibility in RNA-Seq experiments.