```r
set.seed(123)
# 1. Install and Load Required Packages ----
# This block installs and loads the necessary libraries.
# It first checks if the packages are installed, and if not, installs them.

required_packages <- c("NOISeq", "pheatmap", "ggplot2", "RColorBrewer", "cowplot")
for (pkg in required_packages) {
  if (!requireNamespace(pkg, quietly = TRUE)) {
    install.packages(pkg)
  }
  library(pkg, character.only = TRUE)
}

# 2. Load RNA-seq Data ----
# Reads the RNA-seq data from a local file, selecting only the count matrix columns.

rna_seq_data <- read.table("C:/Users/talkt/Dropbox/PC/Downloads/GSE159717_rnaseq_deseq_5dpi_counts_raw.tsv",
                           header = TRUE, row.names = 1, sep = "\t")
count_matrix <- rna_seq_data[, 6:ncol(rna_seq_data)]
sample_names <- colnames(count_matrix)

# 3. Define Experimental Conditions ----
# Creates a 'coldata' data frame with sample names and corresponding conditions (Rem, SARS, mock).

conditions <- ifelse(grepl("Rem", sample_names), "Rem",
                     ifelse(grepl("SARS", sample_names), "SARS", "mock"))
coldata <- data.frame(sample = sample_names, condition = factor(conditions))
rownames(coldata) <- sample_names
```

```r
# 4. Create NOISeq Data Object ----

# Prepares the RNA-seq data for analysis with NOISeq.


noidata <- readData(data = count_matrix, factors = coldata)

print(noidata)


# 5. Differential Expression Analysis (SARS vs Mock) ----

# Runs the NOISeqbio function for SARS vs Mock comparison and writes the results to a CSV file.


noi_SARS_vs_Mock <- noiseqbio(noidata, k = 0.5, norm = "tmm", factor = "condition", conditions = c("SARS", "mock"))

write.csv(noi_SARS_vs_Mock@results[[1]], "sig_SARS_vs_Mock_NOISeq5.csv", row.names = TRUE)


# 6. Differential Expression Analysis (Rem vs Mock) ----

# Runs the NOISeqbio function for Rem vs Mock comparison and writes the results to a CSV file.


noi_Rem_vs_Mock <- noiseqbio(noidata, k = 0.5, norm = "tmm", factor = "condition", conditions = c("Rem", "mock"))

write.csv(noi_Rem_vs_Mock@results[[1]], "sig_Rem_vs_Mock_NOISeq5.csv", row.names = TRUE)


# 7. Volcano Plot Creation ----

# Create Volcano plots for SARS vs Mock and Rem vs Mock comparisons.


volcano_data_sars <- noi_SARS_vs_Mock@results[[1]]

volcano_data_rem <- noi_Rem_vs_Mock@results[[1]]


# Add significance criteria based on probability values.

volcano_data_sars$significant <- ifelse(volcano_data_sars$prob >= 0.95, "Significant", "Not Significant")
```

```
volcano_data_rem$significant <- ifelse(volcano_data_rem$prob >= 0.95, "Significant", "Not Significant")

# Define colors for significant and non-significant points.
colors <- c("Not Significant" = "grey", "Significant" = "red")

# Create Volcano plot for SARS vs Mock.
volcano_plot_sars <- ggplot(volcano_data_sars, aes(x = log2FC, y = -log10(prob), color = significant)) +
  geom_point(alpha = 0.6, na.rm = TRUE) +
  scale_color_manual(values = colors) +
  labs(title = "Volcano Plot: SARS vs Mock", x = "Log2 Fold Change", y = "-log10 Probability") +
  theme_minimal()

# Create Volcano plot for Rem vs Mock.
volcano_plot_rem <- ggplot(volcano_data_rem, aes(x = log2FC, y = -log10(prob), color = significant)) +
  geom_point(alpha = 0.6, na.rm = TRUE) +
  scale_color_manual(values = colors) +
  labs(title = "Volcano Plot: Rem vs Mock", x = "Log2 Fold Change", y = "-log10 Probability") +
  theme_minimal()

# Combine the two volcano plots side by side using cowplot.
combined_volcano <- plot_grid(volcano_plot_sars, volcano_plot_rem, labels = c("A", "B"), ncol = 2)
print(combined_volcano)

# 8. Principal Component Analysis (PCA) ----
# Performs PCA to visualize data variability across conditions (SARS, Rem, Mock).

pca_results <- dat(noidata, type = "PCA")
pca_scores <- as.data.frame(pca_results@dat$result$scores)
```

```
colnames(pca_scores) <- c("PC1", "PC2")

rownames(pca_scores) <- coldata$sample

pca_scores$sample <- rownames(pca_scores)


# Merge PCA Scores with Conditions.

pca_scores <- merge(pca_scores, coldata, by = "sample", all.x = TRUE)


# Calculate Explained Variance for PCA components.

explained_var <- pca_results@dat$result$var.exp[, 1] * 100

pc1_label <- paste0("PC1: ", round(explained_var[1], 1), "% variance")

pc2_label <- paste0("PC2: ", round(explained_var[2], 1), "% variance")


# Plot the PCA results.

ggplot(pca_scores, aes(x = PC1, y = PC2, color = condition)) +

 geom_point(size = 5, alpha = 0.8) +

 labs(title = "PCA across SARS, Rem, and Mock", x = pc1_label, y = pc2_label) +

 theme_minimal(base_size = 15) +

 scale_color_brewer(palette = "Set1", name = "Group")


# 9. Heatmap of Correlation Matrix ----

# Computes the correlation matrix of samples and generates a heatmap.


cor_matrix <- cor(count_matrix)

sample_labels <- colnames(count_matrix)


pheatmap(cor_matrix,

        color = colorRampPalette(c("blue", "white", "red"))(100),

        clustering_distance_rows = "euclidean",

        clustering_distance_cols = "euclidean",

        clustering_method = "complete",

        labels_row = sample_labels,
        labels_col = sample_labels,
        legend_breaks = seq(0.99, 1, by = 0.0025),
        legend_labels = round(seq(0.99, 1, by = 0.0025), 3),
        main = "Correlation Heatmap of RNA-seq Samples",
        fontsize = 10, fontsize_row = 8, fontsize_col = 8,
        display_numbers = TRUE, number_format = "%.2f", border_color = NA)

# 10. MA Plot Creation ----
# MA plots display mean expression vs log2 fold change for SARS vs Mock and Rem vs Mock.

# Extract data for SARS vs Mock (before shrinkage).
ma_data_before_shrink <- as.data.frame(noi_SARS_vs_Mock@results[[1]])

# Create MA Plot (Before Shrinkage).
ggplot(ma_data_before_shrink, aes(x = SARS_mean, y = log2FC)) +
  geom_point(alpha = 0.4) +
  labs(title = "MA Plot: SARS vs Mock (Before Shrinkage)", x = "Mean of Normalized Counts (SARS)", y = "Log2 Fold Change") +
  theme_minimal() +
  scale_y_continuous(limits = c(-2, 2))

# Manual Shrinkage of log2 fold change for SARS vs Mock.
ma_data_after_shrink <- ma_data_before_shrink
ma_data_after_shrink$log2FC_shrunk <- ma_data_before_shrink$log2FC * 0.5

# Create MA Plot (After Shrinkage).
ggplot(ma_data_after_shrink, aes(x = SARS_mean, y = log2FC_shrunk)) +
  geom_point(alpha = 0.4) +
  labs(title = "MA Plot: SARS vs Mock (After Shrinkage)", x = "Mean of Normalized Counts (SARS)", y = "Log2 Fold Change (Shrunk)") +
  theme_minimal() +
  scale_y_continuous(limits = c(-2, 2))
```

```
# Extract data for Rem vs Mock (before shrinkage) and generate similar plots.
ma_data_rem_before_shrink <- as.data.frame(noi_Rem_vs_Mock@results[[1]])

ggplot(ma_data_rem_before_shrink, aes(x = Rem_mean, y = log2FC)) +
  geom_point(alpha = 0.4) +
  labs(title = "MA Plot: Rem vs Mock (Before Shrinkage)", x = "Mean of Normalized Counts (Rem)", y = "Log2 Fold Change") +
  theme_minimal() +
  scale_y_continuous(limits = c(-2, 2))

# Manually Apply Shrinkage to Rem vs Mock log2 fold change.
ma_data_rem_after_shrink <- ma_data_rem_before_shrink
ma_data_rem_after_shrink$log2FC_shrunk <- ma_data_rem_after_shrink$log
# 10. MA Plot Creation (continued) ----

# Manually Apply Shrinkage to Rem vs Mock log2 fold change.
ma_data_rem_after_shrink <- ma_data_rem_before_shrink
ma_data_rem_after_shrink$log2FC_shrunk <- ma_data_rem_before_shrink$log2FC * 0.5

# Create MA Plot (After Shrinkage) for Rem vs Mock.
ggplot(ma_data_rem_after_shrink, aes(x = Rem_mean, y = log2FC_shrunk)) +
  geom_point(alpha = 0.4) +
  labs(title = "MA Plot: Rem vs Mock (After Shrinkage)",
       x = "Mean of Normalized Counts (Rem)",
       y = "Log2 Fold Change (Shrunk)") +
  theme_minimal() +
  scale_y_continuous(limits = c(-2, 2))

# 11. Differential Expression Analysis: Rem vs SARS ----
# This section runs the differential expression analysis for the Rem vs SARS comparison.
```

```
noi_Rem_vs_SARS <- noiseqbio(noidata, k = 0.5, norm = "tmm", factor = "condition", conditions
= c("Rem", "SARS"))


# Extract the results and save them to a CSV file.

rem_vs_sars_results <- noi_Rem_vs_SARS@results[[1]]

write.csv(rem_vs_sars_results, "rem_vs_sars_NOISeq.csv", row.names = TRUE)


# Create the MA plot for Rem vs SARS comparison (Before Shrinkage).

ma_data_before_rem_vs_sars <- as.data.frame(rem_vs_sars_results)


ggplot(ma_data_before_rem_vs_sars, aes(x = Rem_mean, y = log2FC)) +

 geom_point(alpha = 0.4, na.rm = TRUE) +

 labs(title = "MA Plot: Rem vs SARS (Before Shrinkage)",

    x = "Mean of Normalized Counts (Rem)",

    y = "Log2 Fold Change") +

 theme_minimal()


# Manually Apply Shrinkage to log2 fold change for Rem vs SARS.

ma_data_after_rem_vs_sars <- ma_data_before_rem_vs_sars

ma_data_after_rem_vs_sars$log2FC_shrunk <- ma_data_before_rem_vs_sars$log2FC * 0.5


# Create the MA Plot for Rem vs SARS (After Shrinkage).

ggplot(ma_data_after_rem_vs_sars, aes(x = Rem_mean, y = log2FC_shrunk)) +

 geom_point(alpha = 0.4, na.rm = TRUE) +

 labs(title = "MA Plot: Rem vs SARS (After Shrinkage)",

    x = "Mean of Normalized Counts (Rem)",

    y = "Log2 Fold Change (Shrunk)") +

 theme_minimal()


# 12. Summary of Differential Expression Results ----
# This section summarizes the number of significantly differentially expressed genes for each comparison.

# Subset SARS vs Mock results for significant genes (prob >= 0.95, log2FC > 0.6).
filtered_sars_mock <- subset(noi_SARS_vs_Mock@results[[1]],
                             prob >= 0.95 & abs(log2FC) > 0.6)
num_filtered_sars_mock <- nrow(filtered_sars_mock)
cat("Number of filtered genes for SARS vs Mock:", num_filtered_sars_mock, "\n")

# Subset Rem vs Mock results for significant genes (prob >= 0.95, log2FC > 0.6).
filtered_rem_mock <- subset(noi_Rem_vs_Mock@results[[1]],
                            prob >= 0.95 & abs(log2FC) > 0.6)
num_filtered_rem_mock <- nrow(filtered_rem_mock)
cat("Number of filtered genes for Rem vs Mock:", num_filtered_rem_mock, "\n")

# Subset Rem vs SARS results for significant genes (prob >= 0.95, log2FC > 0.6).
filtered_rem_vs_sars <- subset(noi_Rem_vs_SARS@results[[1]],
                               prob >= 0.95 & abs(log2FC) > 0.6)
num_filtered_rem_vs_sars <- nrow(filtered_rem_vs_sars)
cat("Number of filtered genes for Rem vs SARS:", num_filtered_rem_vs_sars, "\n")
```

```
> sessionInfo()
R version 4.4.1 (2024-06-14 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 22631)

Matrix products: default


locale:
[1] LC_COLLATE=English_India.utf8  LC_CTYPE=English_India.utf8
```

```
[3] LC_MONETARY=English_India.utf8 LC_NUMERIC=C
[5] LC_TIME=English_India.utf8

time zone: Australia/Brisbane
tzcode source: internal

attached base packages:
[1] splines   stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
[1] cowplot_1.1.3      RColorBrewer_1.1-3  ggplot2_3.5.1
[4] pheatmap_1.0.12    NOISeq_2.48.0       Matrix_1.7-0
[7] Biobase_2.64.0     BiocGenerics_0.50.0

loaded via a namespace (and not attached):
 [1] vctrs_0.6.5     cli_3.6.3       rlang_1.1.4     generics_0.1.3
 [5] labeling_0.4.3  glue_1.8.0      colorspace_2.1-1 scales_1.3.0
 [9] fansi_1.0.6     grid_4.4.1      munsell_0.5.1   tibble_3.2.1
[13] lifecycle_1.0.4  compiler_4.4.1  dplyr_1.1.4     pkgconfig_2.0.3
[17] farver_2.1.2    lattice_0.22-6  R6_2.5.1        tidyselect_1.2.1
[21] utf8_1.2.4      pillar_1.9.0    magrittr_2.0.3  withr_3.0.1
[25] gtable_0.3.5
>
```