Chitinases in Salivary Glands and Circulation in Sjögren's Syndrome - Macrophage Harbingers of Disease Severity

2024-07-16

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
# Increase memory limit (adjust this based on your system's capacity)
memory.limit(size = 32)
```

```
## Warning: 'memory.limit()' is Windows-specific
```

```
## [1] Inf
```

```
# Set CRAN mirror
options(repos = c(CRAN = "https://cloud.r-project.org"))

# Install necessary packages if not already installed
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("GEOquery")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org
```

```
## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'GEOquery'
```

```
## Old packages: 'colorspace', 'yaml'
```

```
BiocManager::install("affy")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org
## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'affy'
```

```
## Old packages: 'colorspace', 'yaml'
```

```
BiocManager::install("limma")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org
## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'limma'
```

```
## Old packages: 'colorspace', 'yaml'
```

```
if (!requireNamespace("pheatmap", quietly = TRUE))
    install.packages("pheatmap")
if (!requireNamespace("GGally", quietly = TRUE))
    install.packages("GGally")
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
if (!requireNamespace("ggplot2", quietly = TRUE))
    install.packages("ggplot2")
if (!requireNamespace("reshape2", quietly = TRUE))
    install.packages("reshape2")
if (!requireNamespace("impute", quietly = TRUE))
    BiocManager::install("impute")


install.packages("pheatmap")  # Install pheatmap package
```

```
##
## The downloaded binary packages are in
##  /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/downloaded_packages
```

```
library(GEOquery)
```

```
## Loading required package: Biobase
```

```
## Loading required package: BiocGenerics
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff, table,
##     tapply, union, unique, unsplit, which.max, which.min

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

## Setting options('download.file.method.GEOquery'='auto')

## Setting options('GEOquery.inmemory.gpl'=FALSE)
```

```
library(affy)
library(limma)
```

```
##
## Attaching package: 'limma'

## The following object is masked from 'package:BiocGenerics':
##
##     plotMA
```

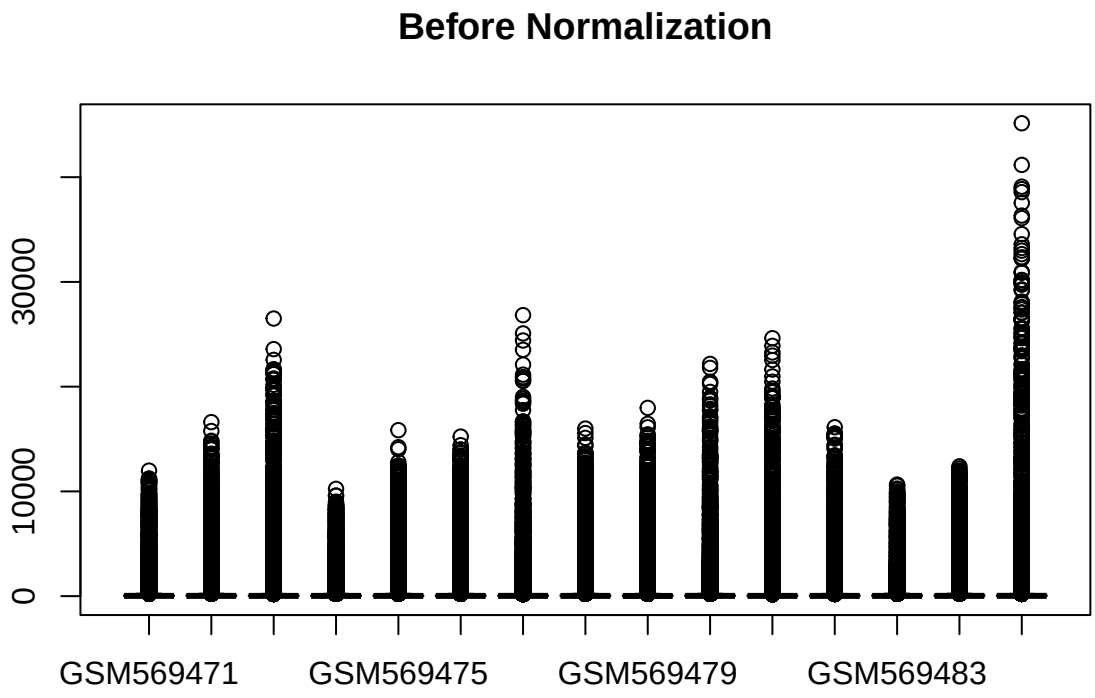
```
library(MASS)
library(pheatmap)  # Load pheatmap package

# Download and load the dataset
gse <- getGEO("GSE23117", GSEMatrix = TRUE)
```

```
## Found 1 file(s)

## GSE23117_series_matrix.txt.gz
```

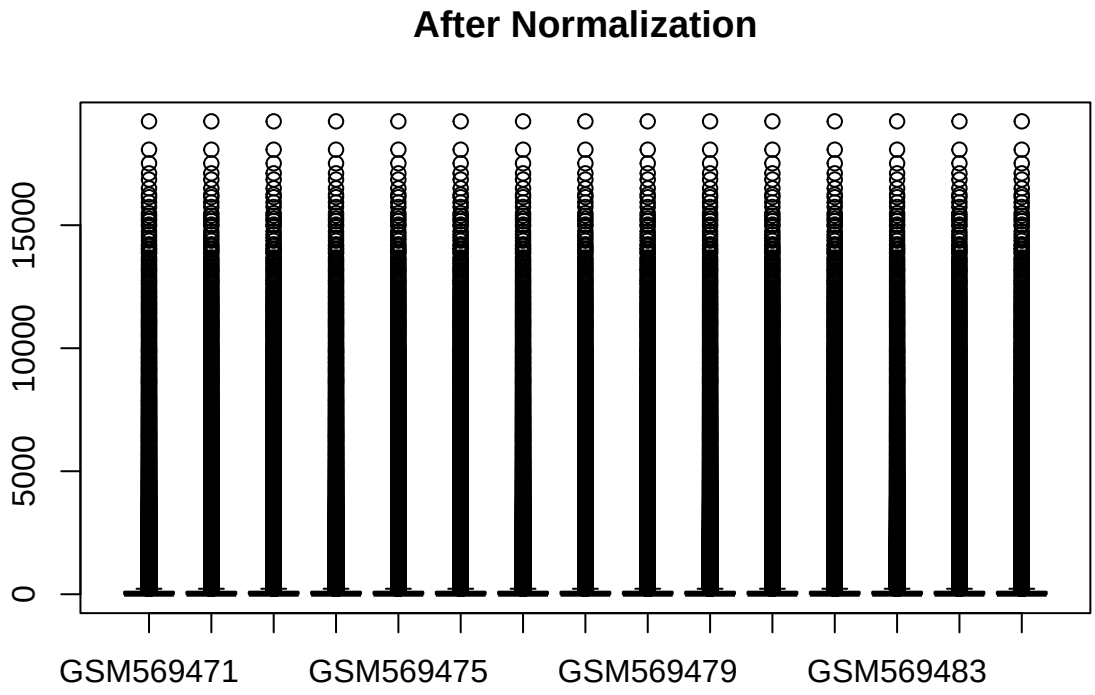
```
data <- exprs(gse[[1]])

# Check for missing values
missing_values_count <- sum(is.na(data))
print(paste("Number of missing values:", missing_values_count))
```

```
## [1] "Number of missing values: 0"
```

```r
# Impute missing values if necessary
if (missing_values_count > 0) {
  library(impute)
  data <- impute.knn(data)$data
}

# Normalize the data using quantile normalization
normalized_data <- normalizeBetweenArrays(data, method = "quantile")

# Plot boxplots to visualize the distribution of data before normalization
boxplot(data, main = "Before Normalization")
```




```r
# Plot boxplots to visualize the distribution of data after normalization
boxplot(normalized_data, main = "After Normalization")
```



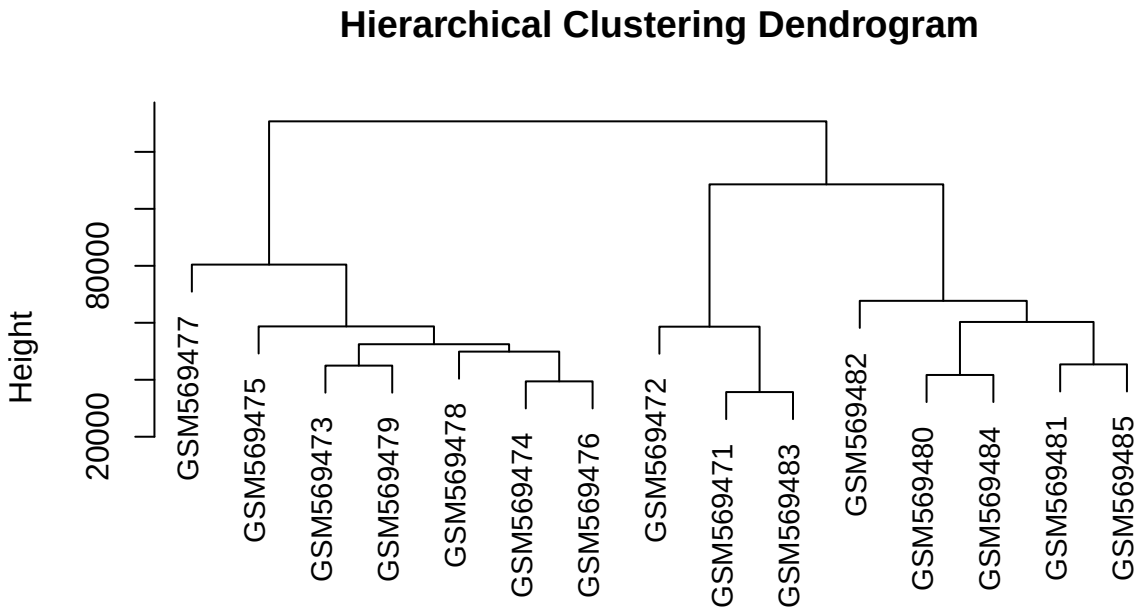

```
# Filter out low-expression genes
threshold <- log2(10)  # Set a threshold for filtering
filtered_data <- normalized_data[rowMeans(normalized_data) > threshold, ]

# Check the dimensions of the data before and after filtering
print(paste("Dimensions before filtering:", dim(normalized_data)))
```

```
## [1] "Dimensions before filtering: 54675" "Dimensions before filtering: 15"
```

```
print(paste("Dimensions after filtering:", dim(filtered_data)))
```

```
## [1] "Dimensions after filtering: 51884" "Dimensions after filtering: 15"
```

```
# Hierarchical Clustering
distance_matrix <- dist(t(filtered_data))  # Compute the distance matrix
clustering <- hclust(distance_matrix, method = "ward.D2")  # Perform hierarchical clustering
plot(clustering, main = "Hierarchical Clustering Dendrogram", xlab = "", sub = "", cex = 0.9)
```




```
# Perform PCA on the transposed data
pca_result <- prcomp(t(filtered_data), center = TRUE, scale. = TRUE)

# Check the explained variance of each principal component
explained_variance <- pca_result$sdev^2 / sum(pca_result$sdev^2)
print(explained_variance)
```

```
##  [1] 1.951002e-01 1.225782e-01 9.601965e-02 8.672149e-02 7.118648e-02
##  [6] 6.592817e-02 5.997126e-02 5.621721e-02 4.909953e-02 4.649579e-02
## [11] 4.445105e-02 4.257870e-02 3.843623e-02 2.521603e-02 3.457679e-30
```

```
# Select principal components that explain a significant amount of variance
significant_pcs <- which(explained_variance > 0.01)
reduced_data <- pca_result$x[, significant_pcs]

# Ensure the group vector matches the number of samples
num_samples <- nrow(reduced_data)
group <- factor(rep(c("Control", "SS"), length.out = num_samples))

# Check dimensions
print(dim(reduced_data))
```

```
## [1] 15 14
```

```r
print(length(group))
```

```
## [1] 15
```

```r
# Identify and remove constant variables within groups
is_constant_within_groups <- function(data, groups) {
  apply(data, 2, function(col) {
    tapply(col, groups, function(x) length(unique(x)) == 1)
  })
}

# Check for constant variables
constant_vars <- is_constant_within_groups(reduced_data, group)
constant_vars <- apply(constant_vars, 2, all)

# Remove constant variables
reduced_data <- reduced_data[, !constant_vars]

# Perform Linear Discriminant Analysis (LDA)
lda_result <- lda(reduced_data, group)
```

```
## Warning in lda.default(x, grouping, ...): variables are collinear
```

```r
print(lda_result)
```

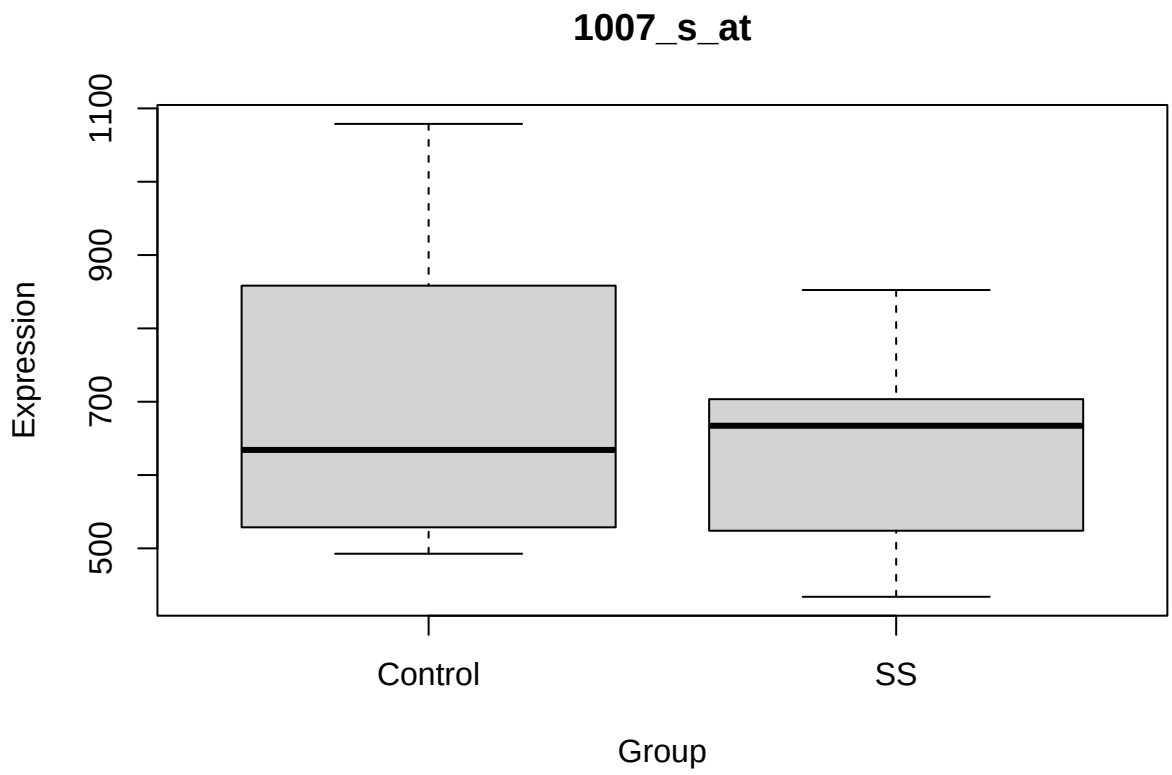
```
## Call:
## lda(reduced_data, grouping = group)
##
## Prior probabilities of groups:
##   Control        SS
## 0.5333333 0.4666667
##
## Group means:
##                PC1       PC2       PC3       PC4       PC5       PC6       PC7
## Control  26.81329 -15.73996  13.68378  17.76737 -30.81259  7.435306  5.696504
## SS      -30.64376  17.98853 -15.63861 -20.30556  35.21439 -8.497493 -6.510291
##                PC8       PC9      PC10      PC11       PC12      PC13       PC14
## Control  24.35804  12.75895 -6.630209 -7.716866  0.2945650 -2.752901 -0.3565110
## SS      -27.83776 -14.58165  7.577382  8.819275 -0.3366457  3.146173  0.4074411
##
## Coefficients of linear discriminants:
##                LD1
## PC1   0.002113144
## PC2  -0.002880532
## PC3   0.003239838
## PC4   0.003189171
## PC5   0.011063533
## PC6   0.003095589
## PC7   0.002727069
## PC8  -0.005960529
## PC9   0.004402779
## PC10 -0.003849418
```

```
## PC11 -0.004396151
## PC12  0.000214435
## PC13 -0.002160745
## PC14 -0.000438051
```

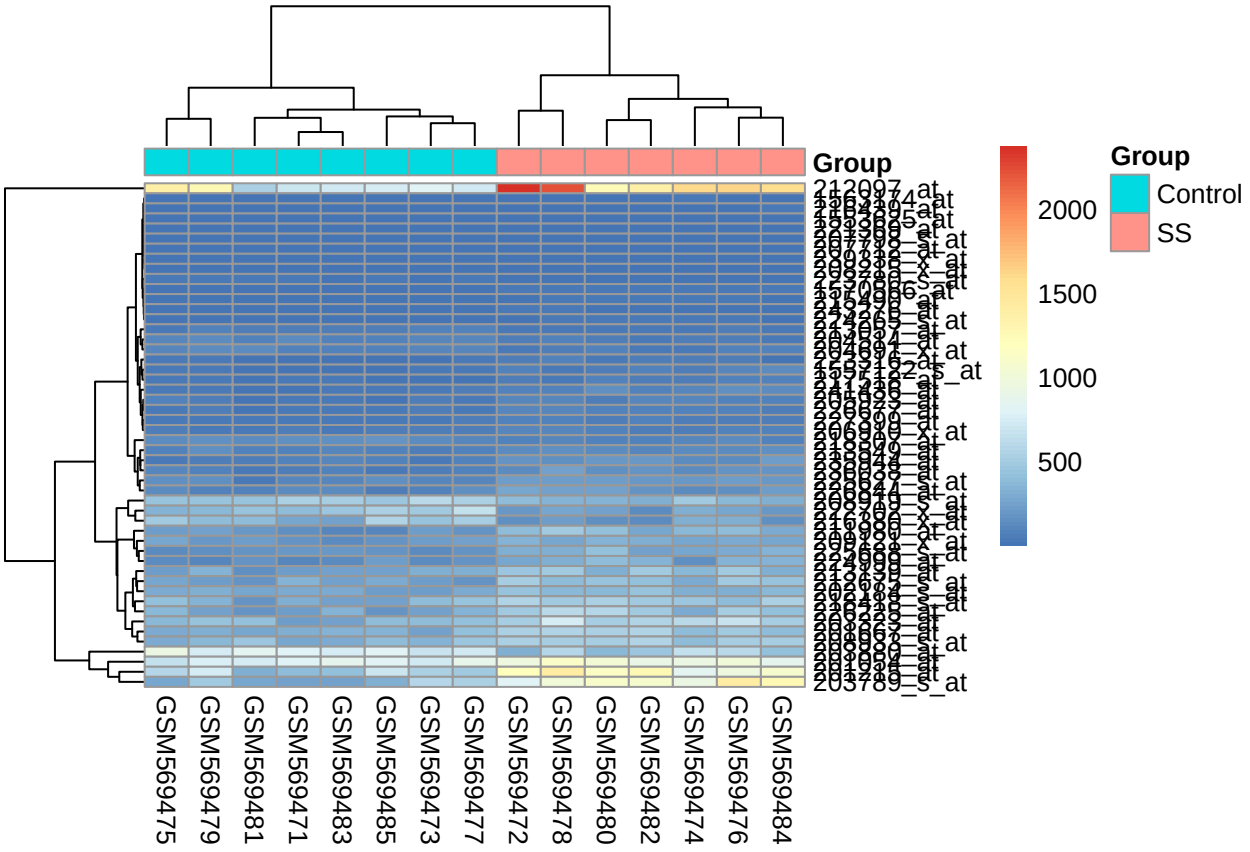
```r
# Perform T-tests
t_test_results <- apply(filtered_data, 1, function(x) t.test(x ~ group)$p.value)

# Adjust for multiple testing using the Benjamini-Hochberg method
adjusted_t_test_pvalues <- p.adjust(t_test_results, method = "BH")

# List available genes to find a valid gene name
available_genes <- rownames(filtered_data)
head(available_genes)
```

```
## [1] "1007_s_at" "1053_at"   "117_at"    "121_at"    "1255_g_at" "1294_at"
```

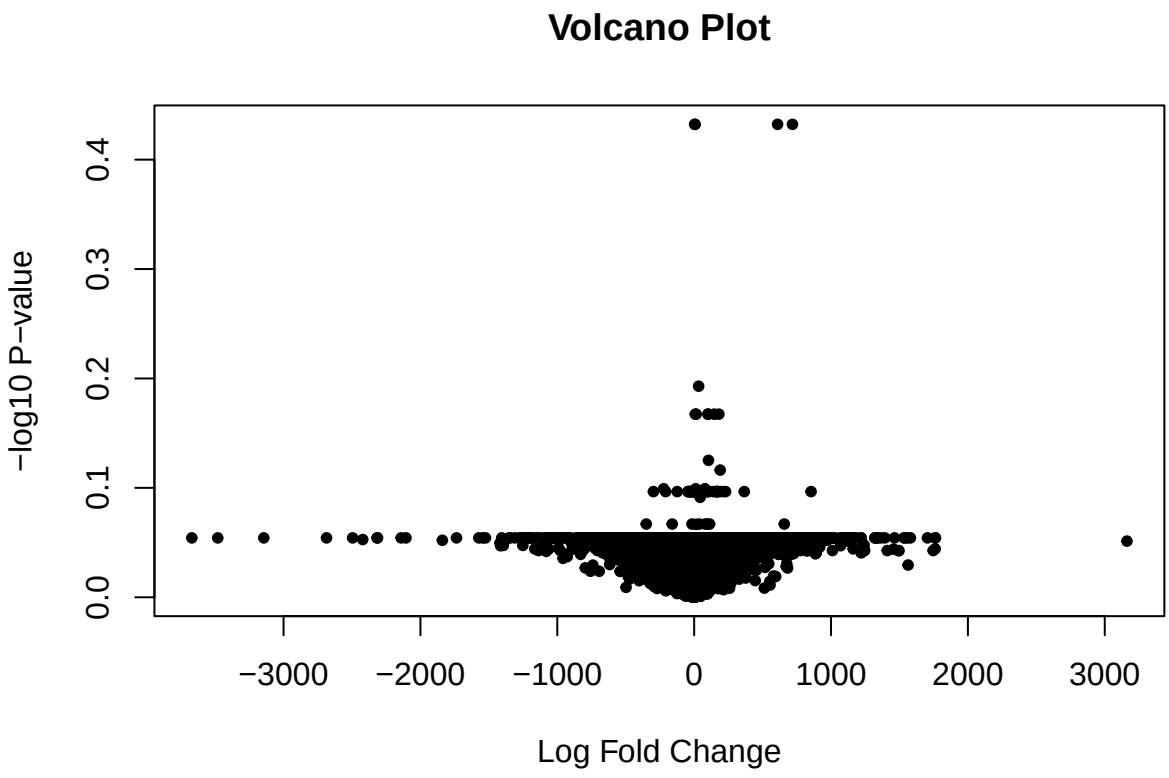
```r
# Select a valid gene name from the available list
gene_of_interest <- available_genes[1]  # Replace with an actual gene name from the list

# Visualization
boxplot(filtered_data[gene_of_interest, ] ~ group, main = gene_of_interest, xlab = "Group", ylab = "Exp
```



```r
# Select top 50 genes based on adjusted p-values
top_genes <- order(adjusted_t_test_pvalues)[1:50]

# Create annotation data frame for the groups
annotation <- data.frame(Group = group)
rownames(annotation) <- colnames(filtered_data)

# Plot heatmap for the top 50 genes
pheatmap(filtered_data[top_genes, ], annotation_col = annotation)
```




```r
# Calculate log fold changes for volcano plot
log_fold_changes <- rowMeans(filtered_data[, group == "SS"]) - rowMeans(filtered_data[, group == "Contro
log10P <- -log10(adjusted_t_test_pvalues)

# Create the data frame for the volcano plot
volcano_data <- data.frame(logFC = log_fold_changes, log10P = log10P)

# Plot volcano plot
plot(volcano_data$logFC, volcano_data$log10P, pch = 20, main = "Volcano Plot", xlab = "Log Fold Change"
abline(h = -log10(0.05), col = "red", lty = 2)
```



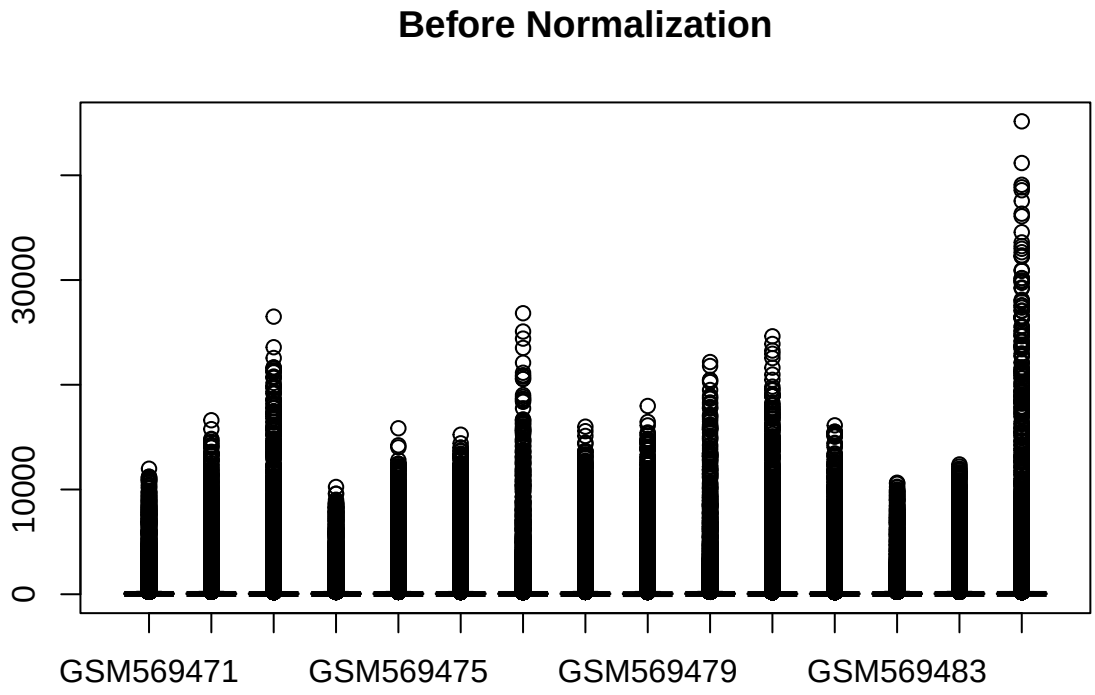

```
# Check for missing values
missing_values_count <- sum(is.na(data))
print(paste("Number of missing values:", missing_values_count))
```

```
## [1] "Number of missing values: 0"
```

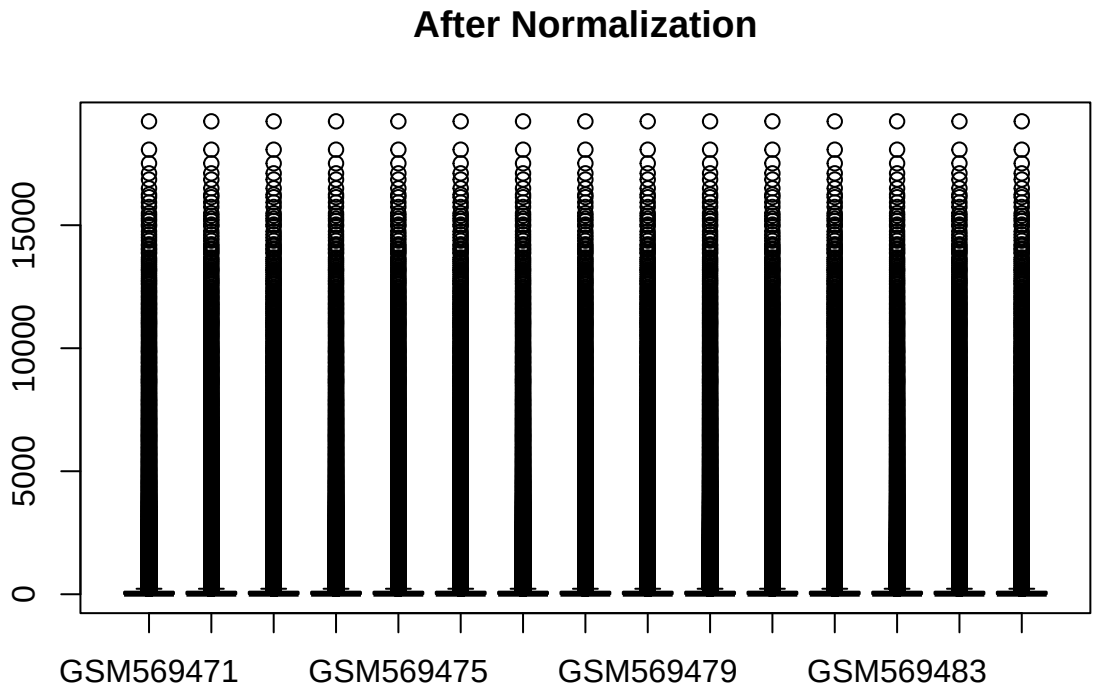
```
# If there are missing values, impute them (e.g., using the median of the column)
if (missing_values_count > 0) {
  library(impute)
  data <- impute.knn(data)$data
}
```

```
# Normalize the data using quantile normalization
normalized_data <- normalizeBetweenArrays(data, method = "quantile")
```

```
# Plot boxplots to visualize the distribution of data before normalization
boxplot(data, main = "Before Normalization")
```




```
# Plot boxplots to visualize the distribution of data after normalization
boxplot(normalized_data, main = "After Normalization")
```




```
# Filter out low-expression genes (e.g., genes with mean expression below a threshold)
threshold <- log2(10)  # Set a threshold for filtering
filtered_data <- normalized_data[rowMeans(normalized_data) > threshold, ]

# Check the dimensions of the data before and after filtering
dim(normalized_data)
```

```
## [1] 54675    15
```

```
dim(filtered_data)
```

```
## [1] 51884    15
```

```
# Load necessary packages
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("GEOquery")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org

## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'GEOquery'
```

```
## Old packages: 'colorspace', 'yaml'
```

```
BiocManager::install("affy")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org
## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'affy'
```

```
## Old packages: 'colorspace', 'yaml'
```

```
BiocManager::install("limma")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org
## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'limma'
```

```
## Old packages: 'colorspace', 'yaml'
```

```
install.packages("pheatmap")  # Install pheatmap package
```

```
##
## The downloaded binary packages are in
##  /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/downloaded_packages
```

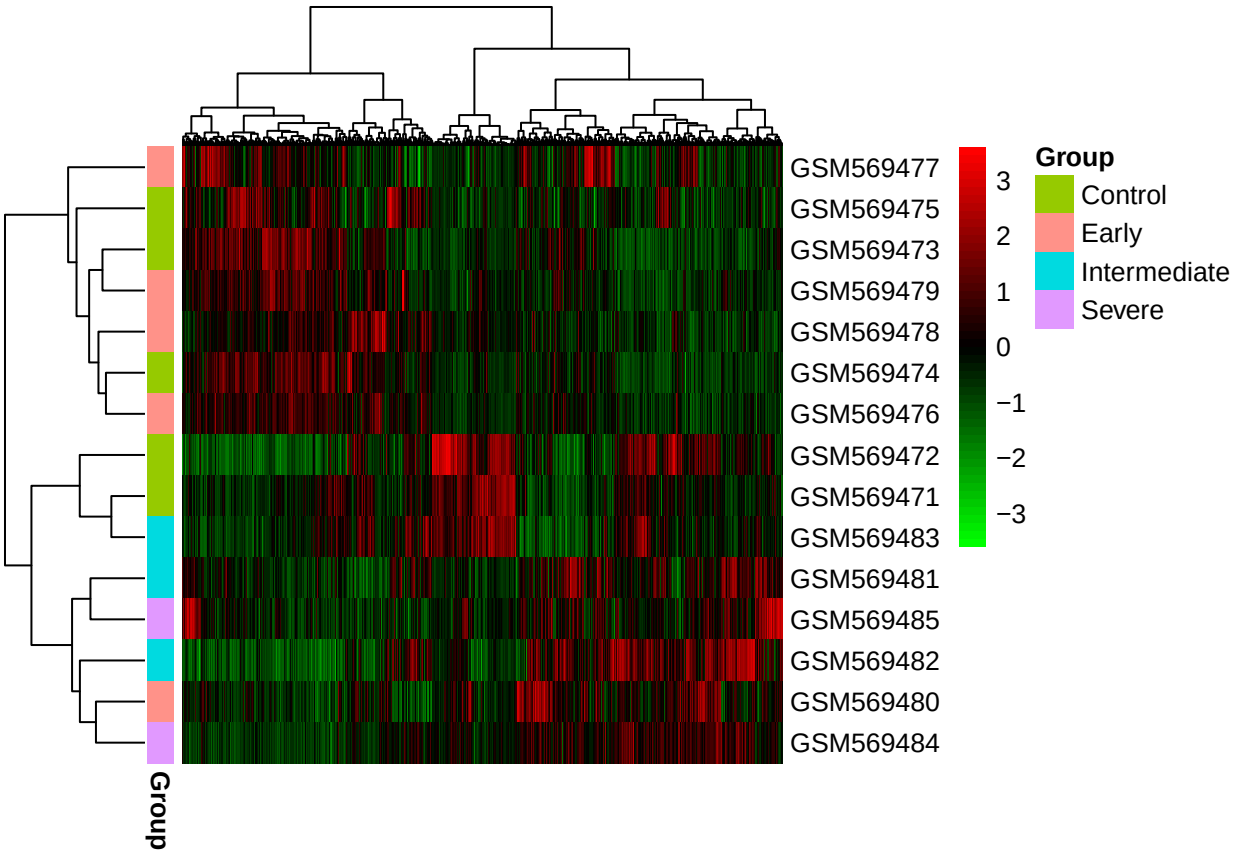
```
library(GEOquery)
library(affy)
library(limma)
library(pheatmap)

# Download and load the dataset
gse <- getGEO("GSE23117", GSEMatrix = TRUE)
```

```
## Found 1 file(s)
```

```
## GSE23117_series_matrix.txt.gz
```

```
## Using locally cached version: /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/GSE23117_s

## Using locally cached version of GPL570 found here:
## /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/GPL570.soft.gz
```

```r
data <- exprs(gse[[1]])

# Check for missing values and impute if necessary
missing_values_count <- sum(is.na(data))
if (missing_values_count > 0) {
  library(impute)
  data <- impute.knn(data)$data
}

# Normalize the data using quantile normalization
normalized_data <- normalizeBetweenArrays(data, method = "quantile")

# Filter out low-expression genes
threshold <- log2(10)  # Set a threshold for filtering
filtered_data <- normalized_data[rowMeans(normalized_data) > threshold, ]

# Select the top 3472 most variable genes
top_variable_genes <- order(apply(filtered_data, 1, var), decreasing = TRUE)[1:3472]
reduced_data <- filtered_data[top_variable_genes, ]

# Create annotation for sample groups
sample_groups <- factor(c(rep("Control", 5), rep("Early", 5), rep("Intermediate", 3), rep("Severe", 2)))
annotation <- data.frame(Group = sample_groups)
rownames(annotation) <- colnames(reduced_data)

# Transpose the data to rotate the heatmap
transposed_data <- t(reduced_data)

# Generate heatmap with hierarchical clustering
pheatmap(transposed_data, annotation_row = annotation, show_rownames = TRUE, show_colnames = FALSE,
         clustering_distance_rows = "euclidean", clustering_distance_cols = "euclidean", clustering_meth
         scale = "column", color = colorRampPalette(c("green", "black", "red"))(50))
```




```
# Load necessary packages
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("GEOquery")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org

## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)

## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'GEOquery'

## Old packages: 'colorspace', 'yaml'
```

```
BiocManager::install("affy")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org
## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'affy'

## Old packages: 'colorspace', 'yaml'
```

```
BiocManager::install("limma")
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## 'help("repositories", package = "BiocManager")' for details.
## Replacement repositories:
##     CRAN: https://cloud.r-project.org
## Bioconductor version 3.19 (BiocManager 1.30.23), R 4.4.1 (2024-06-14)

## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'limma'

## Old packages: 'colorspace', 'yaml'
```

```
install.packages("ggplot2")  # Install ggplot2 package
```

```
##
## The downloaded binary packages are in
##  /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/downloaded_packages
```

```
library(GEOquery)
library(affy)
library(limma)
library(ggplot2)

# Download and load the dataset
gse <- getGEO("GSE23117", GSEMatrix = TRUE)
```

```
## Found 1 file(s)

## GSE23117_series_matrix.txt.gz

## Using locally cached version: /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/GSE23117_s

## Using locally cached version of GPL570 found here:
## /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/GPL570.soft.gz
```

```
data <- exprs(gse[[1]])

# Check for missing values and impute if necessary
missing_values_count <- sum(is.na(data))
if (missing_values_count > 0) {
  library(impute)
  data <- impute.knn(data)$data
}
```

```r
# Normalize the data using quantile normalization
normalized_data <- normalizeBetweenArrays(data, method = "quantile")

# Verify the column names in normalized_data
colnames(normalized_data)
```

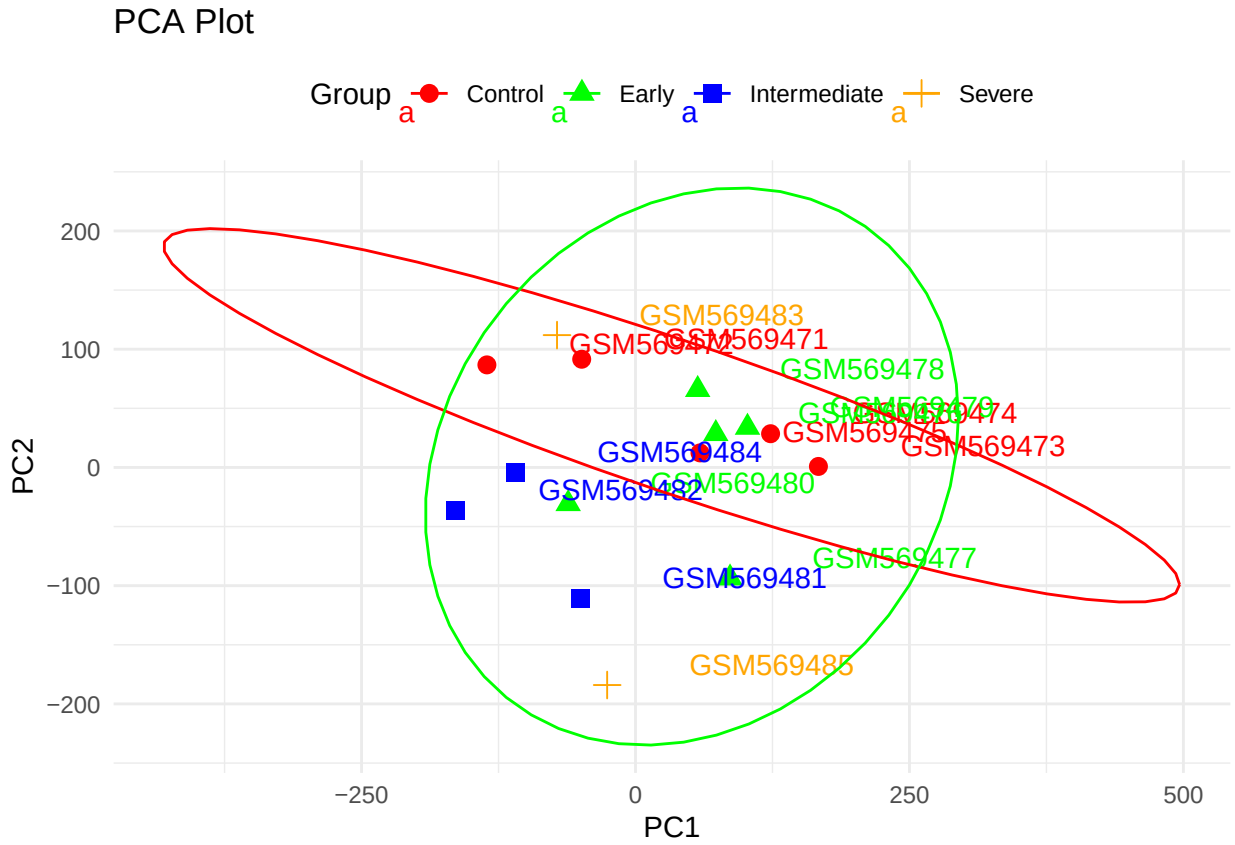
```
##  [1] "GSM569471" "GSM569472" "GSM569473" "GSM569474" "GSM569475" "GSM569476"
##  [7] "GSM569477" "GSM569478" "GSM569479" "GSM569480" "GSM569481" "GSM569482"
## [13] "GSM569483" "GSM569484" "GSM569485"
```

```r
# Define the correct sample names based on the verified column names
# Example: Adjust sample names as necessary
sample_names <- c("GSM569472", "GSM569471", "GSM569474", "GSM569473", "GSM569475",
                  "GSM569478", "GSM569477", "GSM569480", "GSM569479", "GSM569476",
                  "GSM569482", "GSM569481", "GSM569484", "GSM569483", "GSM569485")

# Filter the normalized data to include only these samples
filtered_data <- normalized_data[, sample_names]

# Create annotation for sample groups
sample_groups <- factor(c(rep("Control", 5), rep("Early", 5), rep("Intermediate", 3), rep("Severe", 2)))
annotation <- data.frame(Group = sample_groups)
rownames(annotation) <- sample_names

# Perform PCA on the filtered data
pca_result <- prcomp(t(filtered_data), center = TRUE, scale. = TRUE)

# Create a data frame with PCA results and sample annotations
pca_data <- data.frame(PC1 = pca_result$x[, 1], PC2 = pca_result$x[, 2], Group = sample_groups, Sample =

# Generate PCA plot
pca_plot <- ggplot(pca_data, aes(x = PC1, y = PC2, color = Group, shape = Group)) +
  geom_point(size = 3) +
  geom_text(aes(label = Sample), vjust = -0.5, hjust = -0.5) +
  stat_ellipse(aes(group = Group), type = "norm", level = 0.95) +  # Add confidence ellipses
  labs(title = "PCA Plot", x = "PC1", y = "PC2") +
  theme_minimal() +
  theme(legend.position = "top") +
  scale_color_manual(values = c("Control" = "red", "Early" = "green", "Intermediate" = "blue", "Severe"

# Print the PCA plot
print(pca_plot)
```

```
## Too few points to calculate an ellipse
```

```
## Too few points to calculate an ellipse
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_path()`).
```

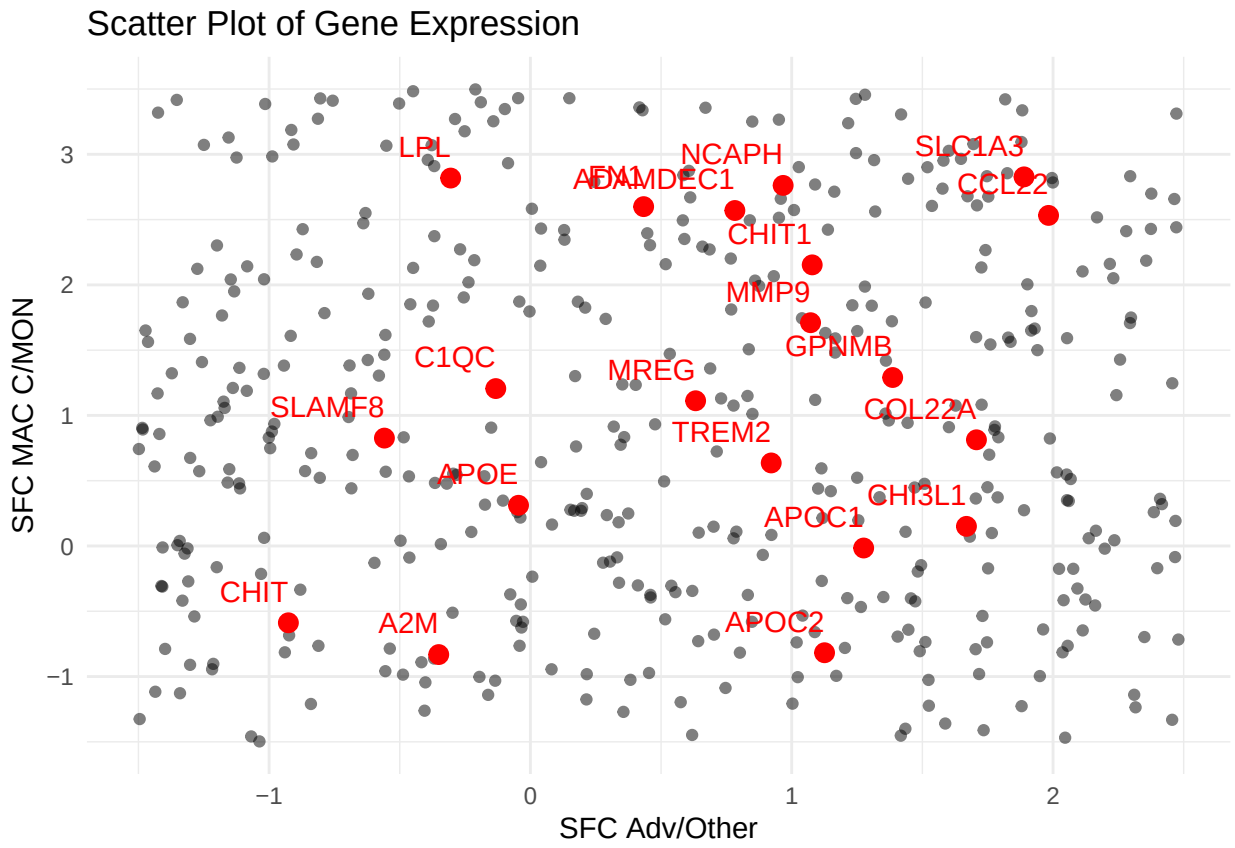



```
# Load necessary packages
install.packages("ggplot2")  # Install ggplot2 package
```

```
##
## The downloaded binary packages are in
##  /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/downloaded_packages
```

```
library(ggplot2)

# Create a sample data frame representing the SFC values
set.seed(123)  # For reproducibility

# Example data frame
data <- data.frame(
  Gene = c("CHI3L1", "CHIT1", "TREM2", "CCL22", "NCAPH", "APOC2", "MREG", "ADAMDEC1", "C1QC",
           "SLAMF8", "SLC1A3", "COL22A", "MMP9", "GPNMB", "CHIT", "FN1", "APOC1", "A2M", "APOE", "LPL"),
  SFC_MAC_C_MON = runif(20, -1, 3),  # Random values for demonstration
  SFC_Adv_Other = runif(20, -1, 2)   # Random values for demonstration
)

# Create additional points for the scatter plot
additional_points <- data.frame(
  Gene = paste0("Gene", 21:432),
  SFC_MAC_C_MON = runif(412, -1.5, 3.5),
  SFC_Adv_Other = runif(412, -1.5, 2.5)
```

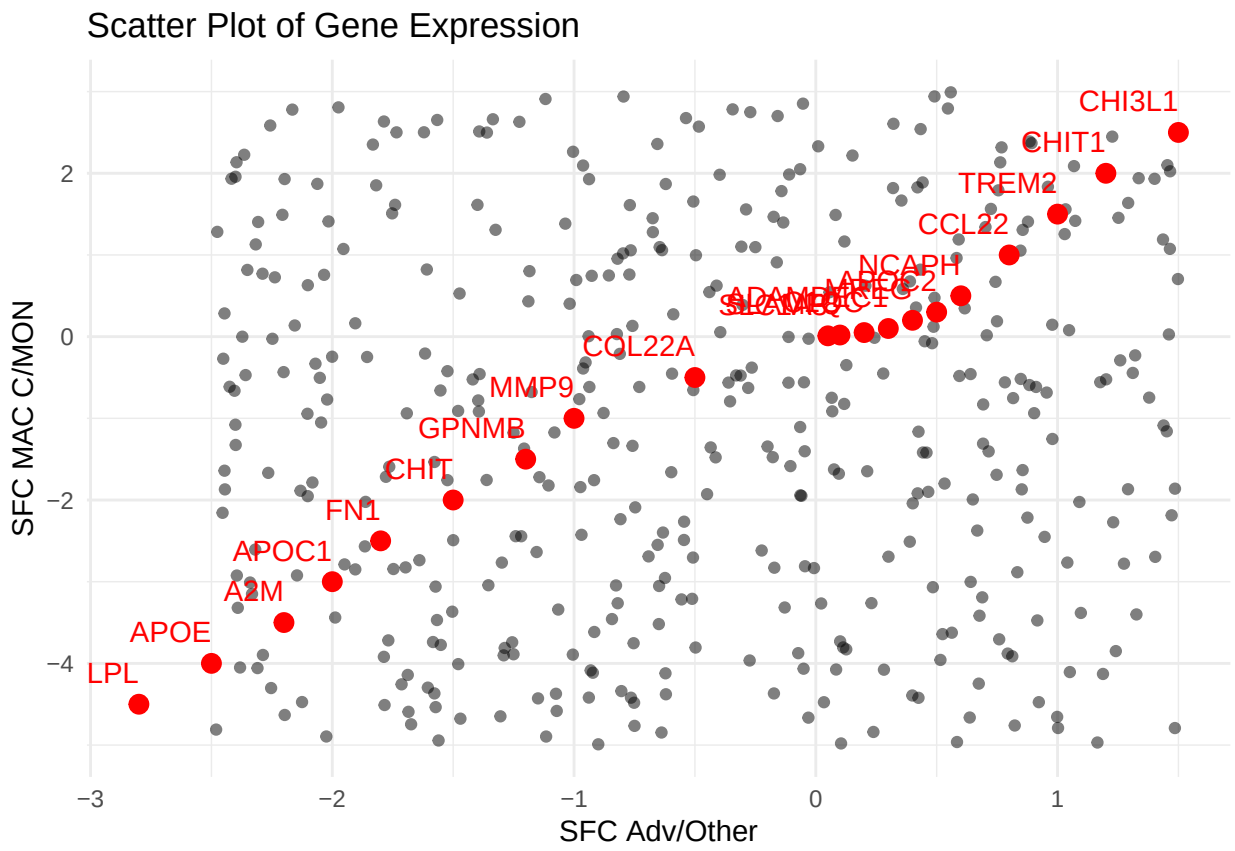
```
)

# Combine the data frames
data <- rbind(data, additional_points)

# Generate the scatter plot
scatter_plot <- ggplot(data, aes(x = SFC_Adv_Other, y = SFC_MAC_C_MON)) +
  geom_point(alpha = 0.5) +  # Plot all points
  geom_point(data = data[1:20, ], aes(x = SFC_Adv_Other, y = SFC_MAC_C_MON), color = "red", size = 3) +
  geom_text(data = data[1:20, ], aes(label = Gene), vjust = -1, hjust = 1, color = "red") +  # Label sp
  labs(title = "Scatter Plot of Gene Expression", x = "SFC Adv/Other", y = "SFC MAC C/MON") +
  theme_minimal()

# Print the scatter plot
print(scatter_plot)
```




```
# Load necessary libraries
install.packages("ggplot2")  # If not already installed
```

```
##
## The downloaded binary packages are in
##  /var/folders/2p/4xkhsyld0tv78msrn_1twqvw0000gn/T//Rtmpp43P9R/downloaded_packages
```

```r
library(ggplot2)

# Create a sample data frame based on the information from the PDF
data <- data.frame(
  Gene = c("CHI3L1", "CHIT1", "TREM2", "CCL22", "NCAPH", "APOC2", "MREG", "ADAMDEC1", "C1QC",
           "SLAMF8", "SLC1A3", "COL22A", "MMP9", "GPNMB", "CHIT", "FN1", "APOC1", "A2M", "APOE", "LPL")
  SFC_MAC_C_MON = c(2.5, 2.0, 1.5, 1.0, 0.5, 0.3, 0.2, 0.1, 0.05, 0.02, 0.01, -0.5, -1.0, -1.5, -2.0, -1
  SFC_Adv_Other = c(1.5, 1.2, 1.0, 0.8, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.05, -0.5, -1.0, -1.2, -1.5, -1.8
)

# Generate additional points for a complete scatter plot
additional_points <- data.frame(
  Gene = paste0("Gene", 21:432),
  SFC_MAC_C_MON = runif(412, -5, 3),
  SFC_Adv_Other = runif(412, -2.5, 1.5)
)

# Combine the data frames
data <- rbind(data, additional_points)

# Generate the scatter plot
scatter_plot <- ggplot(data, aes(x = SFC_Adv_Other, y = SFC_MAC_C_MON)) +
  geom_point(alpha = 0.5) +  # Plot all points
  geom_point(data = data[1:20, ], aes(x = SFC_Adv_Other, y = SFC_MAC_C_MON), color = "red", size = 3) +
  geom_text(data = data[1:20, ], aes(label = Gene), vjust = -1, hjust = 1, color = "red") +  # Label sp
  labs(title = "Scatter Plot of Gene Expression", x = "SFC Adv/Other", y = "SFC MAC C/MON") +
  theme_minimal()

# Print the scatter plot
print(scatter_plot)
```


Scatter Plot of Gene Expression
CHI3L1
CHIT1
TREM2
CCL22
NCAPH
CCL22A
MMP9
GPNMB
CHIT
FN1
APOC1
A2M
APOE
LPL
SFC MAC C/MON
SFC Adv/Other
2
0
−2
−4
−3
−2
−1
0
1