RNA-seq Pipeline Analysis Using GSE61490 Dataset to Uncover Transcriptomes of Monocytes in Chronic Periodontitis Inflammatory disease

Azari I. Bantan
BESE394: Setting Bioinformatics Pipelines
Instructors: David Gomez-Cabrero, Jesper N Tegner, Vincenzo Lagani & Robert Lehmann

Abstract

This study employs RNA sequencing (RNA-seq) technology to comprehensively analyze the transcriptome of peripheral blood monocytes (PBMs) in individuals with Chronic Periodontitis (CP; n=5) as compared to healthy individuals (n=5), aiming to elucidate the molecular underpinnings of this chronic inflammatory disorder. The primary objective of this report is to provide a detailed and comprehensive analysis pipeline for RNA-seq data, elucidating the step-by-step process involved in the examination of transcriptomic information, using the respective study dataset.

Keywords: Chronic Periodontitis (CP), Monocytes, Pipeline, RNA Sequencing, Transcriptomes.

RNA-seq Pipeline Analysis Using GSE61490 Dataset to Uncover Transcriptomes of Monocytes in Chronic Periodontitis (CP) Inflammatory disease

Chronic Periodontitis (CP) is a prevalent inflammatory condition, characterized by the dysregulation of host immune responses where monocytes have shown to play a major role in CP pathogenesis [1]. Investigating the monocyte transcriptomic profile in the context of CP provides a unique opportunity to uncover novel functional genes and pathways associated with the disease. The step-by-step analysis of the RNA-seq pipeline in this context is specifically centered around the subsequent stages following the acquisition of raw count data, subsequent to prior pre-processing steps, such as: experimental design, library preparation, sequencing, annotation, mapping, and quantification.

Data availability

Dataset retrieved from Liu, Y.-Z. et al. (2016) study deposited to Gene Expression Omnibus (GEO) under the accession number of GSE61490 [1].

Methodology

The RNA-seq analysis pipeline is executed using R version 4.3.2, employing various Bioconductor packages such as DESeq2, NOISeq, and clusterProfiler. For visualization purposes, ggplot2 is utilized. A comprehensive breakdown of the analysis process is provided (including the R script code), illustrating detailed steps at each respective stage [2-6].

1) Loading the dataset

Dataset retrieved from GEO using the following URL path:

```
urld <
"https://www.ncbi.nlm.nih.gov/geo/download/?format=file&type=rnaseq_counts"
path <- paste(urld, "acc=GSE61490",
"file=GSE61490_raw_counts_GRCh38.p13_NCBI.tsv.gz", sep="&")
GSE61490_count <- as.matrix(data.table::fread(path, header=T,
colClasses="integer"), rownames=1)
```

From the above, only the following metadata were extracted:

```
library(GEOquery)
gds <- getGEO("GSE61490")
Meta_GSE61490 <- pData(gds$GSE61490_series_matrix.txt.gz@phenoData)
Meta_GSE61490 <- Meta_GSE61490 <- Meta_GSE61490[,c("source_name_ch1","cell
type:ch1","disease:ch1")]
```

Table 1

GSE61490 Metadata

	source_name_ch1	cell type:ch1	disease:ch1
GSM1506096	Monocytes	Peripheral blood monocytes	Chronic periodontitis
GSM1506097	Monocytes	Peripheral blood monocytes	Chronic periodontitis
GSM1506098	Monocytes	Peripheral blood monocytes	Chronic periodontitis
GSM1506099	Monocytes	Peripheral blood monocytes	Chronic periodontitis
GSM1506100	Monocytes	Peripheral blood monocytes	Chronic periodontitis
GSM1506101	Monocytes	Peripheral blood monocytes	Health control
GSM1506102	Monocytes	Peripheral blood monocytes	Health control
GSM1506103	Monocytes	Peripheral blood monocytes	Health control
GSM1506104	Monocytes	Peripheral blood monocytes	Health control
GSM1506105	Monocytes	Peripheral blood monocytes	Health control

Expected gene expression is modelled by a group factor in the column ‘disease:ch1’, which will be used in fitting the design model later in the analysis. The disease status names are changed from ‘Chronic periodontitis’ and ‘Health control’ to ‘disease’ and ‘control’, respectively, for simplicity purposes.

```
Status <- Meta_GSE61490[,c("disease:ch1")]
Factors_GSE61490 <- data.frame(Status)
rownames(Factors_GSE61490) <- rownames(Meta_GSE61490)
Factors_GSE61490[Factors_GSE61490=="Chronic periodontitis"] <- "disease"
Factors_GSE61490[Factors_GSE61490=="Health control"] <- "control"
Factors_GSE61490$Status <- factor(Factors_GSE61490$Status)
```

2) Exploring potential technology biases using the count data

Various technical factors, such as transcript length, GC content, PCR artifacts, uneven coverage of transcript reads, off-target transcript contamination, and significant variations in transcript distributions, can disrupt the linear relationship between transcript abundance and the count of mapped reads at a gene locus. This can be explored using DESeq Bioconductor package using the biological metadata of the mapped genes exported from Ensembl BioMart: https://www.ensembl.org.

```
annotgene <- read.csv("GSE61490_mart_export.txt",sep="\t",header = T)
annotgene <-
annotgene[!duplicated(annotgene$NCBI.gene..formerly.Entrezgene..ID),]
rownames(annotgene) <- annotgene$NCBI.gene..formerly.Entrezgene..ID
```

Additional filtering is applied to remove i.e., duplicated annotated genes and keeping 1:22, X and Y chromosomes.

```
annotgene <- annotgene[annotgene$Chromosome %in% c(as.character(1:22)
,"X","Y"),]
sum(rownames(GSE61490_count) %in% rownames(annotgene))
annotgene_filt <- annotgene[!duplicated(rownames(annotgene)),]


# filtering the count-matrix accordingly
GSE61490_count_filt <- GSE61490_count[rownames(GSE61490_count) %in%
rownames(annotgene_filt),]
annotgene_filt$GeneID <- rownames(annotgene_filt)
annotgene_ord <- annotgene_filt[rownames(GSE61490_count_filt),]
 # sum(rownames(annotgene_ord)==rownames(GSE61490_count_filt))
```

Now, a QC report can be produced for general inspection of count data using QCreport() function from NOISEq package after creating a NOISeq object using the biological data of each gene in our dataset, retrieved from BioMart (i.e., GC content, biotype, chromosome start and end), as well as the count-matrix with specifying the factors (‘Status’) [4].

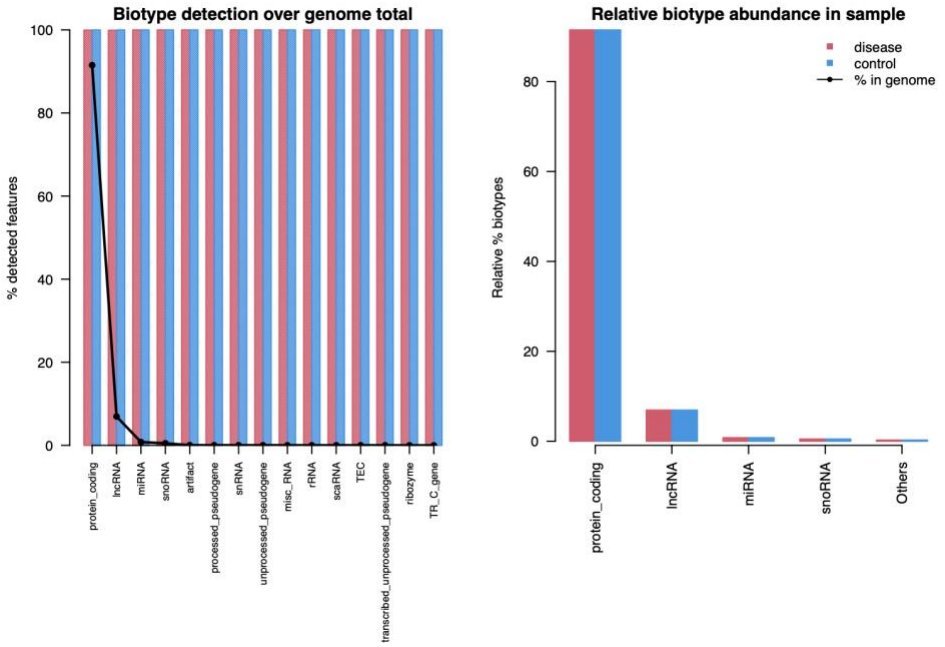
The QC report is attached as a supplementary file ‘QCreport_GSE61490.pdf’. The chosen parameter for the argument “factor” is ‘`factor = "Status"`, hence, the exploratory plots compare two experimental conditions (samples are aggregated into two groups) instead of comparing two individual samples.

```
# Preparing inputs for readData()
lengthuse <- abs(annotgene_ord$Gene.end..bp.-annotgene_ord$Gene.start..bp.);
names(lengthuse) <- rownames(annotgene_ord)
gc <- annotgene_ord$Gene...GC.content; names(gc) <- rownames(annotgene_ord)
biotype <-annotgene_ord$Gene.type; names(biotype) <- rownames(annotgene_ord)
chromosome <-
annotgene_ord[,c("Chromosome.scaffold.name","Gene.start..bp.","Gene.end..bp.")]

# Creating a NOISeq object
data_NOISEQ <- readData(data = GSE61490_count_filt,
                        length=lengthuse,
                        gc=gc,
                        biotype= biotype,
                        chromosome =
annotgene_ord[,c("Chromosome.scaffold.name","Gene.start..bp.","Gene.end..bp.")],
                        factors = Factors_GSE61490)

# Generating QC report
QCreport(data_NOISEQ, file ="QCreport_GSE61490.pdf", factor = "Status", norm =
FALSE)
```

Example: The following exploratory plot show the distribution of counts within each of the biotypes across the two conditions. As expected, protein-coding has the highest abundance in both conditions since oligo-dT selection was performed primarily targeting protein-coding mRNA [1], which typically contains polyA tails. This means that the abundance of protein-coding transcripts will be selectively increased in the samples.

Figure 1

Biotype detection and relative abundance across the samples of two group conditions.



3) Creating a DESeq2 object

DESeqDataSetFromMatrix() is a function from the DESeq2 package used to create a DESeqDataSet object, a specific data structure for performing subsequent analysis such as differential expression. Here, we also specify the experimental design formula. In this case, the variable 'Status' in the 'Factors_GSE61490' metadata is the main factor of interest for the differential expression analysis, which has two levels: 'disease' and 'control'. Further discussion on this will be elaborated in the 'Differential Expression Analysis' section.

```
library(DESeq2)
GSE61490_DESeq2 <- DESeqDataSetFromMatrix(countData = GSE61490_count,
                                          colData = Factors_GSE61490,
                                          design = ~ Status)
```

4) Normalization

Before normalization is performed on the data, we need to consider reducing the noise from genes with no or very low expression which can maximize the sensitivity of differentially expression analysis (ref: doi: 10.1038/nmeth.3885). Here, we assume that a gene is 'expressed' in a given sample if it has a count of => 10 in at least 2 samples. This will result in filtering out 22,828 genes (see: "> table(keep)").

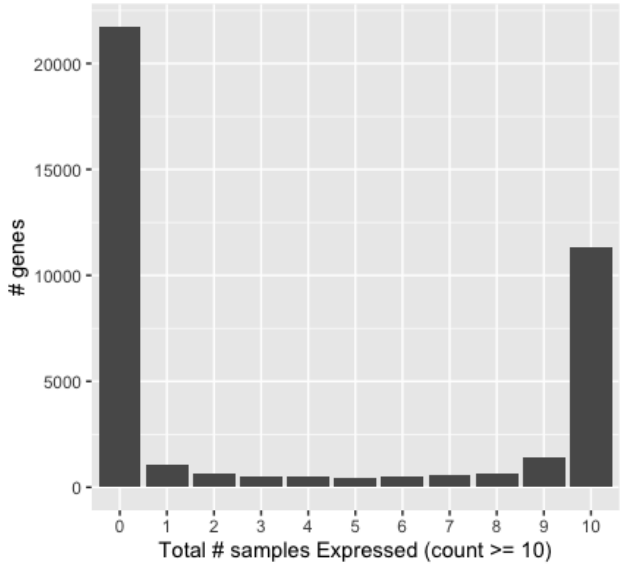
```
smallestGroupSize <- 2
keep <- rowSums(counts(GSE61490_DESeq2) >= 10) >= smallestGroupSize
table(keep)
GSE61490_DESeq2_f <- GSE61490_DESeq2[keep,]

> table(keep)
keep
FALSE  TRUE
22828 16548

GSE61490_DESeq2_f <- GSE61490_DESeq2[keep,]
```

Using the following code, we can generate a plot (figure 2) to inspect how many genes are 'expressed' (y-axis) in at least an x number of samples (x-axis). We can see genes are either not expressed in any samples or 'the rest of majority' are expressed in all of the samples.

```
Library(ggplot2)

is_expressed <- assay(GSE61490_DESeq2) >= 10
df <- data.frame(Expressed = rowSums(is_expressed))
df$Expressed_factor <- factor(df$Expressed, levels = 0:10)
ggplot(df, aes(x = Expressed_factor)) +
  geom_bar() +
  labs(x = "Total # samples Expressed (count >= 10)", y = "# genes")
```

Figure 2

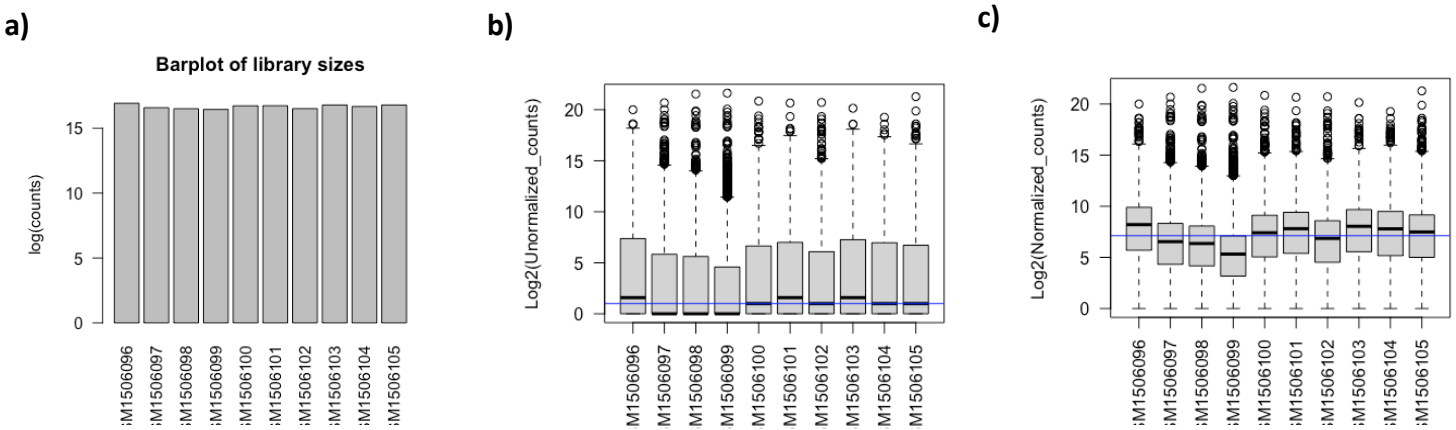
Number of genes accounted as 'expressed' based on the assumption that a gene is expressed if it has a count of => 10, in a total of X number of samples defined by the X-axis.



For normalization, DESeq() function is used. The normalization approach employed by DESeq2 helps control for systematic biases introduced by variations in library size, although the library size in this dataset seems to be evenly distributed across samples (figure 3a). DESeq2 estimates for a 'size factor normalization' to account for differences in library size or sequencing depth across samples. DESeq2 first estimates a size factor for each sample which can be used by the next step of 'scaling the count data'. See figure 3b-c, which shows the log of counts before and after normalization.

Figure 3

Inspection of counts for a) library size, before b) and c) after normalization.



```
### Normalization
GSE61490_dds <- DESeq(GSE61490_DESeq2_f)

# inspection of Normalized data
librarySizes <- log(colSums(GSE61490_count))
barplot(librarySizes,
        names=names(librarySizes),
        las=2,
        main="Barplot of library sizes",
        ylab = "log(colSums(GSE61490_count)")
abline(h=20e6, lty=2)

# Get log2 counts per million
logcounts <- log2(GSE61490_count + 1)
boxplot(logcounts,
        xlab="",
        ylab="Log2(Unormalized_counts)",
        las=2)
# Adding a blue horizontal line that corresponds to the median logCount
abline(h=median(as.matrix(logcounts)), col="blue")

logcounts <- log2(assay(GSE61490_dds) + 1)
boxplot(logcounts,
        xlab="",
        ylab="Log2(Normalized_counts)",
        las=2)
# Adding a blue horizontal line that corresponds to the median logCount
abline(h=median(as.matrix(logcounts)), col="blue")
```

5) Principle Component Analysis (PCA)

PCA helps in reducing the dimensionality of the data while retaining the most important information, thus can be used for exploration of technical/biological variation. Figure 4, show the principal Component 1 (PC1) and principal component 2 (PC2). The plot reveals a clear separation between control and disease. Notably, PC1 explains 43% of the total variance, emphasizing its significance in discriminating between the two groups. One outlier might be indicated in the disease group (bottom-right), potentially indicating unique biological characteristics. Additional inspection of clustering of samples based on gene expression can be explored by the calculation of pairwise correlation values across the samples based on the transformed gene expression (i.e, using VST), followed by illustration in a i.e., heatmap (Figure 5). It demonstrates the degree of similarity of samples in their gene expression patterns. Control samples show higher similarity, whereas the disease samples seem to be more variable.

Note: DESeq2 provides other options such as variance stabilizing transformation (VST), which can be useful for exploratory data analysis and visualization (hence, applied here).

```
## PCA
vsd <- vst(GSE61490_dds,blind=TRUE)
plotPCA(vsd,intgroup=c("Status"))
```

Figure 4

Principle component analysis (PCA) of samples from healthy (n=5) and disease (n-5).

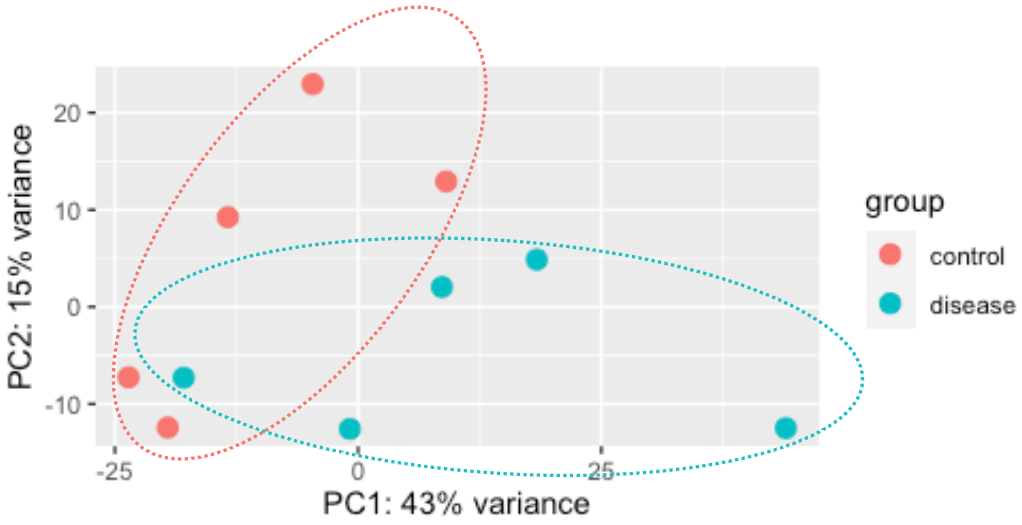
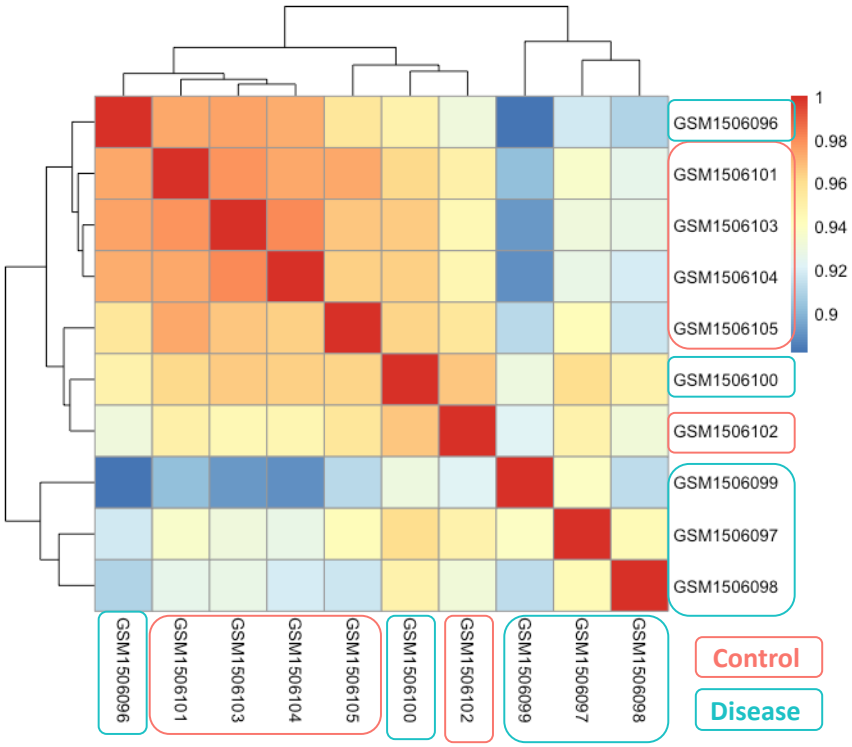


Figure 5

Heatmap of correlation values between samples using heatmap.



6) Differential Expression Analysis

Recall the design model that was specified in creating the DESeq2 object using the following function: `DESeqDataSetFromMatrix()`. The formula is used to model the relationship between the gene expression and the experimental conditions ('Status'). In this case, the design formula is

$$\text{genes} \sim \beta_0 + Status$$

For differential expression, DESeq2 uses a negative binomial distribution-based model, taking into account the inherent variability and overdispersion. To perform this, the function `DESeq()` is utilized. It also performs hypothesis testing to identify differentially expressed genes. The primary test used by default is the Wald test, but likelihood ratio tests (LRT) and score tests can also be performed. This can be chosen as an augment in `DESeq(dds, test= c("Wald", "LRT", "Score")`. In this analysis, Wald test is used (default). It tests whether the coefficients for each variable in the model (e.g., different experimental conditions) are significantly different from zero, whereas LRT compares the fit of the full model against other models.

Note: The contrast of differential expression analysis is disease vs control.

```
GSE61490_dds <- DESeq(GSE61490_DESeq2_f)
> resultsNames(GSE61490_dds)
[1] "Intercept"                 "Status_disease_vs_control"
```

A dispersion plot (figure 6a) is a diagnostic plot which visualizes the relationship between the estimated dispersions and mean expression of genes in RNA-seq, reflecting the variability in expression for each gene. Here, the dispersion values are relatively constant across the range of mean expression levels, indicating homogeneity of dispersion. Als consideration is to stabilize for low counts and high variability

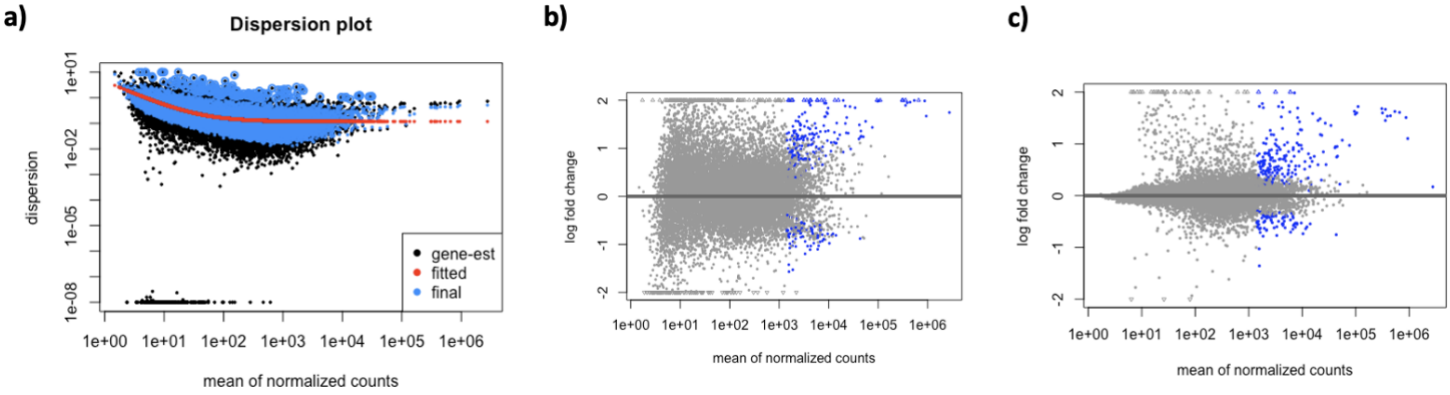
`lfcShrink()` function from the DESeq2 package, is used for shrinkage estimation of log2 fold changes (LFCs), which helps to stabilize the LFC estimates, especially when dealing with genes with low counts or high levels of variability. In fact, Shrinkage methods help prevent overfitting, especially when dealing with a small number of samples (which is the case in our dataset). DESeq2 uses various shrinkage methods, including the empirical Bayes shrinkage implemented by the apeglm package. Empirical Bayes methods are well-suited for RNA-seq data due to their ability to adapt to the data distribution. `lfcShrink()` is applied, and the effect of stabilization is observed before (Figure 6b) and after (Figure 6c).

Note: The stabilized log2FoldChange is used in subsequent analysis/visualization.

```
# defining the contrast and output the results
GSE61490_res <- results(GSE61490_dds, contrast = c("Status", "disease",
"control"))

# Plotting the dispersion estimates
plotDispEsts(GSE61490_dds, main="Dispersion plot")

#### Shrinkage
# before
plotMA(GSE61490_res, ylim=c(-2,2))
# after
plotMA(lfcShrink(GSE61490_dds,coef=c("Status_disease_vs_control")),
ylim=c(-2,2))

resLFC <- lfcShrink(GSE61490_dds, res =GSE61490_res,
coef="Status_disease_vs_control", type="apeglm")
```

Figure 6

Representation of mean expression with respect to a) estimated dispersion b) logFC before and c) after stabilization using resLFC() function.



The results of differentially expressed (DE) genes are represented in the volcano plot (Figure 7), where each dot represents a single gene. Significant genes (p.adj < 0.05) are colored in green. The stabilized LFC estimates show more DE genes in disease. Top 20 significant genes are selected and shown in a heatmap (Figure 8). Two clusters are observed, where a set of genes are downregulated in disease, but upregulated in control, also a set of genes which show the opposite.

```
## volcano plot
ggplot(as.data.frame(resLFC), aes(x = log2FoldChange, y = -log10(padj)))
+
  geom_point(aes(color = ifelse(padj < 0.05, "red", "black")), alpha =
0.6) +
  geom_hline(yintercept = -log10(0.05), linetype = "dashed", color =
"blue") +
  labs(title = "Volcano Plot for Differentially Expressed Genes",
       x = "Log2 Fold Change",
       y = "-log10(Adjusted p-value)")

## identifying top genes by sorting them by p-value.
resSort <- resLFC[order(resLFC$padj),]
head(resSort, 20)
top20_DE <- resSort[1:20,]
library("org.Hs.eg.db")
top20_DE$GeneSymbol <- mapIds(org.Hs.eg.db, keys = rownames(top20_DE),
keytype = "ENTREZID", column = "SYMBOL")

### heatmap
library("pheatmap")
vsd <- assay(vst(GSE61490_dds))[rownames(top20_DE),]
rownames(vsd) <- top20_DE$GeneSymbol
Z <- t(scale(t(vsd)))
pheatmap(Z, cluster_rows=T, show_rownames=T,
         cluster_cols=FALSE, annotation_col=Factors_GSE61490)
```

Figure 7

Volcano plots for differential expressed (DE) genes in disease compared to control. a) shows DE genes before stabilizing LFC estimates and b) after.

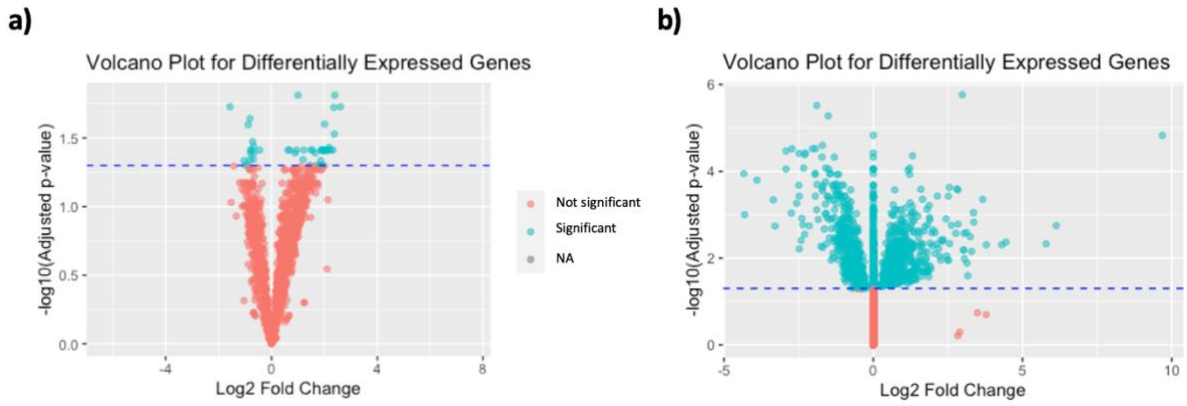
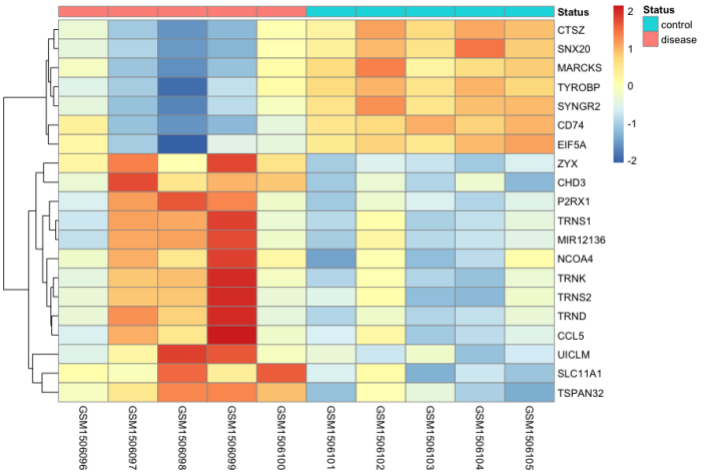


Figure 8

Heatmap of top 20 differentially expressed genes (p.adj < 0.05).



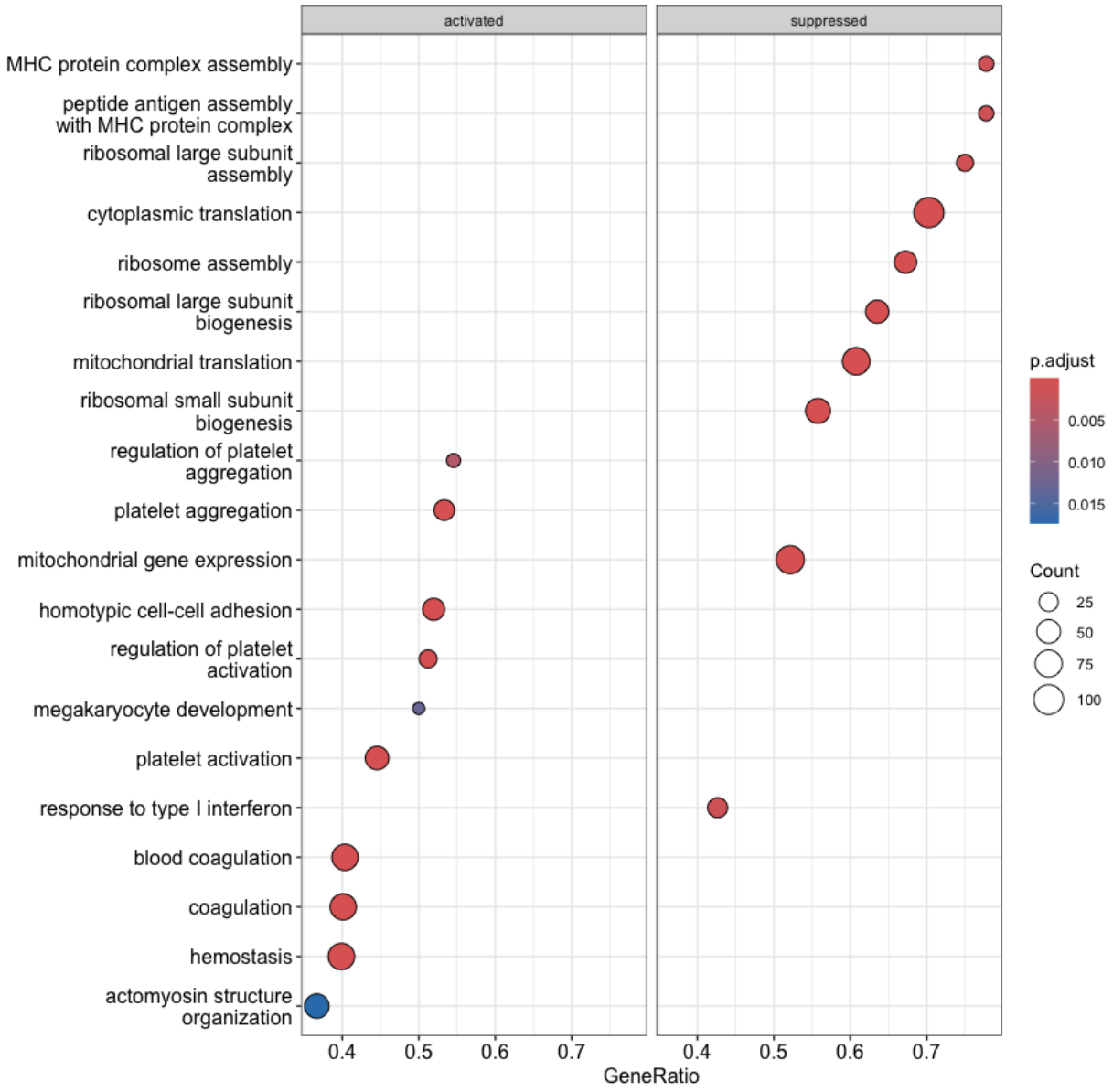
7) Functional Annotation using Gene Set Enrichment Analysis (GSEA)

To understand the biological function of the DE genes, GSEA can be performed. GSEA assesses whether a set of a genes are randomly distributed throughout the ranked list of genes (i.e., based on fold-change) or whether they tend to cluster toward the top or bottom. Hence, it looks at enrichment patterns that might indicate biological relevance. The enriched pathways are summarized in Figure 9. Such analysis is conducted using clusterProfiler packages, and it primarily relies on various databases and annotations, including Gene Ontology (GO) Database, Kyoto Encyclopedia of Genes and Genomes (KEGG), DOSE, MSigDB.

```
### GSEA
library(clusterProfiler)
library(enrichplot)
organism = "org.Hs.eg.db"
library(organism, character.only = TRUE)

GSE61490_res_genes <- GSE61490_res$log2FoldChange
names(GSE61490_res_genes) <- rownames(GSE61490_res)
gene_list<-na.omit(GSE61490_res_genes)
gene_list = sort(gene_list, decreasing = TRUE)
gse <- gseGO(geneList=gene_list,
             ont ="BP",
             keyType = "ENTREZID",
             minGSSize = 3,
             maxGSSize = 200,
             pvalueCutoff = 0.05,
             verbose = TRUE,
             OrgDb = organism,
             pAdjustMethod = "BH")
# dotplot
require(DOSE)
dotplot(gse, showCategory=10, split=".sign") + facet_grid(.~.sign)
```

Figure 9

Dot plot illustrating Gene Set Enrichment Analysis (GSEA) results obtained with gseGO()



References

1. Liu, Y.-Z. et al. (2016). 'RNA-sequencing study of peripheral blood monocytes in chronic periodontitis'. *Gene*. 581(2), pp. 152-160. doi:10.1016/j.gene.2016.01.036
2. R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/
3. Love MI, Huber W, Anders S (2014). "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2." *Genome Biology*, 15, 550. doi:10.1186/s13059-014-0550-8
4. Tarazona S, Furio-Tari P, Turra D, Pietro AD, Nueda MJ, Ferrer A, Conesa A (2015). "Data quality aware analysis of differential expression in RNA-seq with NOISeqR/Bioc package." *Nucleic Acids Research*, 43(21)
5. Wu T, Hu E, Xu S, Chen M, Guo P, Dai Z, Feng T, Zhou L, Tang W, Zhan L, Fu x, Liu S, Bo X, Yu G (2021). "clusterProfiler 4.0: A universal enrichment tool for interpreting omics data." *The Innovation*, 2(3), 100141. doi:10.1016/j.xinn.2021.100141
6. Wickham H (2016). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.