DESeq2 and GSEA with clusterProfiler

Irin Ann Paul

2024-09-26

Importing libraries

```
suppressPackageStartupMessages({
    library(RColorBrewer)
    library(pheatmap)
    library(ggplot2)
    library(ggrepel)
    library(stringr)
    library(tidyverse)
    library(magrittr)
    library(dplyr)
    library(cowplot)
    library(GenomicFeatures)
    library(DESeq2)
    library(DEGreport)
    library(apeglm)
    library(tximport)
    library(biomaRt)
    library(clusterProfiler)
    library(pathview)
    library(enrichplot)
    library(msigdbr)
    library(DOSE)
    library(org.Hs.eg.db)
})
```

Loading annotations

Using `biomaRt` to get gene annotations from Ensembl. These annotations will be essential later on for mapping transcript IDs to gene symbols.

```
# Setting up biomaRt to query Ensembl's human dataset for gene annotations
human_mart <- biomaRt::useMart("ENSEMBL_MART_ENSEMBL", dataset = "hsapiens_gene_ensembl", host = "https
```

Getting annotation entries and creating annotation file:

```
# Querying for transcript IDs, gene IDs, descriptions, and gene names
transcript_to_gene <- getBM(attributes = c(
        "ensembl_transcript_id",
        "transcript_version",
        "ensembl_gene_id",
        "description",
        "external_gene_name"),
    mart = human_mart)

# Combining transcript ID and version into a single identifier
transcript_to_gene$ensembl_transcript_id <- paste(transcript_to_gene$ensembl_transcript_id, transcript_
```

Organizing the annotations into a more usable format:

```
# Renaming and selecting relevant columns for the annotation dataframe
t2g <- dplyr::rename(
    transcript_to_gene,
    target_id = ensembl_transcript_id,
    ens_gene = ensembl_gene_id,
    ext_gene = external_gene_name
    )[,c(
        "target_id",
        "ens_gene",
        "description",
        "ext_gene")]

knitr::kable(head(t2g))
```

target_id	ens_gene	description	ext_gene
ENST00000387314.1	ENSG00000210049	mitochondrially encoded tRNA-Phe (UUU/C) [Source:HGNC Symbol;Acc:HGNC:7481]	MT-TF
ENST00000389680.2	ENSG00000211459	mitochondrially encoded 12S rRNA [Source:HGNC Symbol;Acc:HGNC:7470]	MT-RNR1
ENST00000387342.1	ENSG00000210077	mitochondrially encoded tRNA-Val (GUN) [Source:HGNC Symbol;Acc:HGNC:7500]	MT-TV
ENST00000387347.2	ENSG00000210082	mitochondrially encoded 16S rRNA [Source:HGNC Symbol;Acc:HGNC:7471]	MT-RNR2
ENST00000386347.1	ENSG00000209082	mitochondrially encoded tRNA-Leu (UUA/G) 1 [Source:HGNC Symbol;Acc:HGNC:7490]	MT-TL1
ENST00000361390.2	ENSG00000198888	mitochondrially encoded NADH:ubiquinone oxidoreductase core subunit 1 [Source:HGNC Symbol;Acc:HGNC:7455]	MT-ND1

Differential Expression Analysis with DESeq2

Loading example data for BMAT analysis:

```
data_bmat <- readRDS("data_bmat.Rds")
s2c_bmat <- readRDS("s2c_bmat.Rds")
```

```
# DESeq2 dataset object with count data, sample information, and experimental design
dds_bmat <- DESeqDataSetFromMatrix(
    countData = data_bmat,
    colData = s2c_bmat,
    design = ~replicate + co_cultured)
```

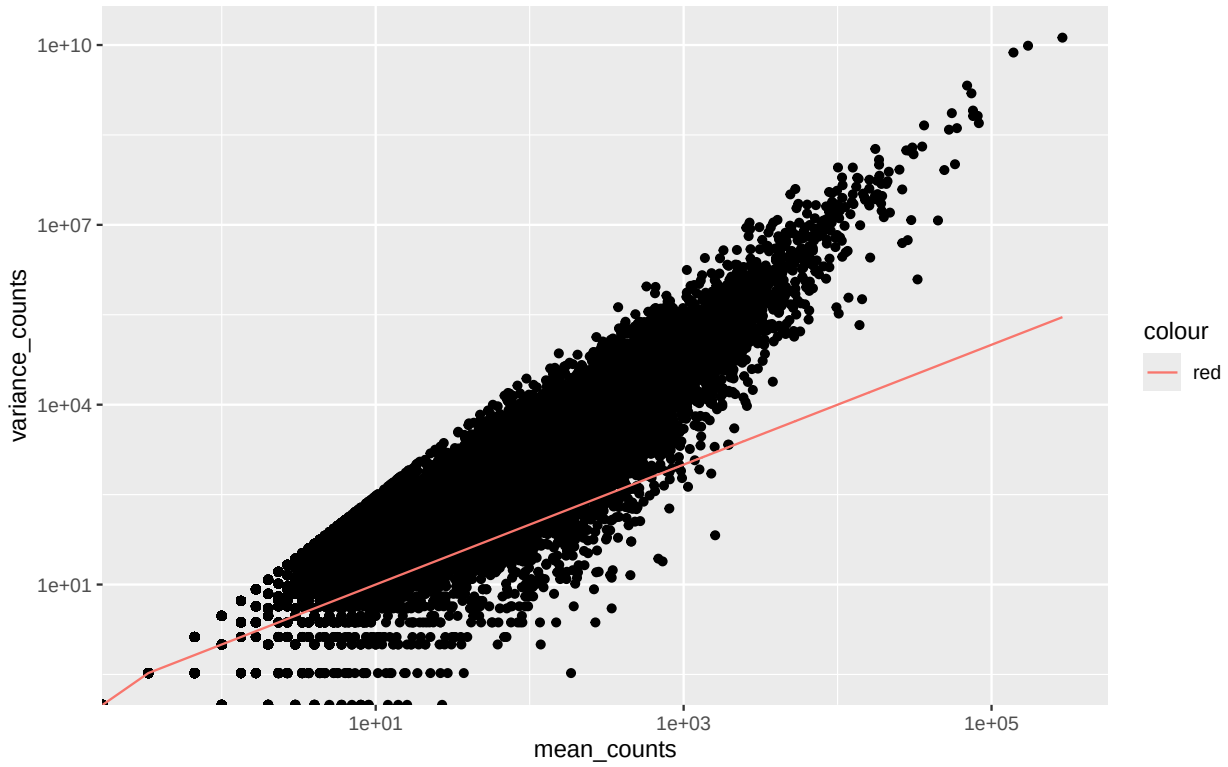
Preparing the DESeq2 DataSet

```
dds_bmat <- DESeq(dds_bmat)
```

Running differential expression

Estimates gene-wise dispersions

```
# Visualizing the relationship between the mean and variance for all genes in the dataset
mean_counts <- apply(data_bmat[,c(2,4,6)], 1, mean)
variance_counts <- apply(data_bmat[,c(2,4,6)], 1, var)
df <- data.frame(mean_counts, variance_counts)
ggplot(df)+
    geom_point(aes(x=mean_counts, y=variance_counts)) +
    geom_line(aes(x=mean_counts, y=mean_counts, color="red")) +
    scale_y_log10() +
    scale_x_log10()
```



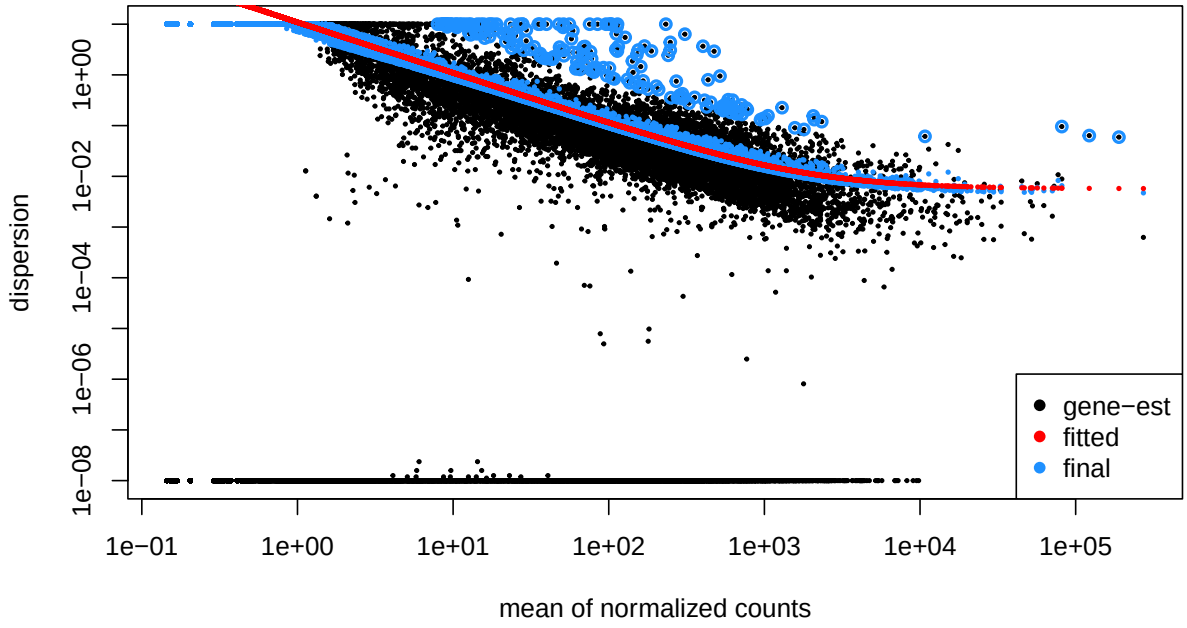
The above plot shows the variance of each gene as a function of its mean expression. DESeq2 uses this information to account for variance when identifying differentially expressed genes. As expected, the variance increases with mean expression.

Instead of using variance as the measure of variation in the data (since variance correlates with gene expression level), it uses a measure of variation called dispersion, which accounts for a gene's variance and mean expression level.

For genes with moderate to high count values, the square root of dispersion will be equal to the coefficient of variation `(Var /mu )`. So 0.01 dispersion means 10% variation around the mean expected across biological replicates. The dispersion estimates for genes with the same mean will differ only based on their variance. Therefore, the dispersion estimates reflect the variance in gene expression for a given mean value (given as black dots in image below).

In the plot below, each black dot represents a gene, and the red line shows the fitted dispersion trend. Genes with higher counts tend to have lower dispersions, as expected in RNA-seq data.

```
# Plotting the dispersion estimates for each gene. This shows how dispersion is estimated as a function
plotDispEsts(dds_bmat)
```




```
# BMAT
dds_bmat <- estimateSizeFactors(dds_bmat)
dds_bmat <- estimateDispersions(dds_bmat)
```

```
## found already estimated dispersions, replacing these

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates
```

```
dds_bmat <- nbinomWaldTest(dds_bmat)
```

PCA Analysis

```
# Performing variance-stabilizing transformation (VST) on the data
vsd_bmat <- vst(dds_bmat, blind=FALSE)

# Plotting the first two principal components before batch correction
pca_bmat <- plotPCA(vsd_bmat, intgroup=c("co_cultured", "replicate"), returnData=TRUE)
percentVar <- round(100 * attr(pca_bmat, "percentVar"))
pca_bmat_before <- ggplot(pca_bmat, aes(PC1, PC2, color=co_cultured)) +
```

```
    geom_point(size=2) +
    xlab(paste0("PC1: ",percentVar[1],"% variance")) +
    ylab(paste0("PC2: ",percentVar[2],"% variance")) +
    coord_fixed()

# Removing batch effect from the transformed data
assay(vsd_bmat) <- limma::removeBatchEffect(assay(vsd_bmat), vsd_bmat$replicate)

# Plotting PCA again after batch correction
pca_bmat <- plotPCA(vsd_bmat, intgroup=c("co_cultured", "replicate"), returnData=TRUE)
percentVar <- round(100 * attr(pca_bmat, "percentVar"))
pca_bmat_after <- ggplot(pca_bmat, aes(PC1, PC2, color=co_cultured)) +
    geom_point(size=2) +
    xlab(paste0("PC1: ",percentVar[1],"% variance")) +
    ylab(paste0("PC2: ",percentVar[2],"% variance")) +
    coord_fixed()

# Plotting PCA before and after batch correction
title <-ggdraw() +
    draw_label(
        "BMAT alone vs co_cultured",
        fontface = 'bold',
        x = 0,
        hjust = 0
    )
plot_grid(title, pca_bmat_before, pca_bmat_after, ncol = 1, rel_heights = c(0.1, 1, 1.2))
```

Before and After Batch Correction

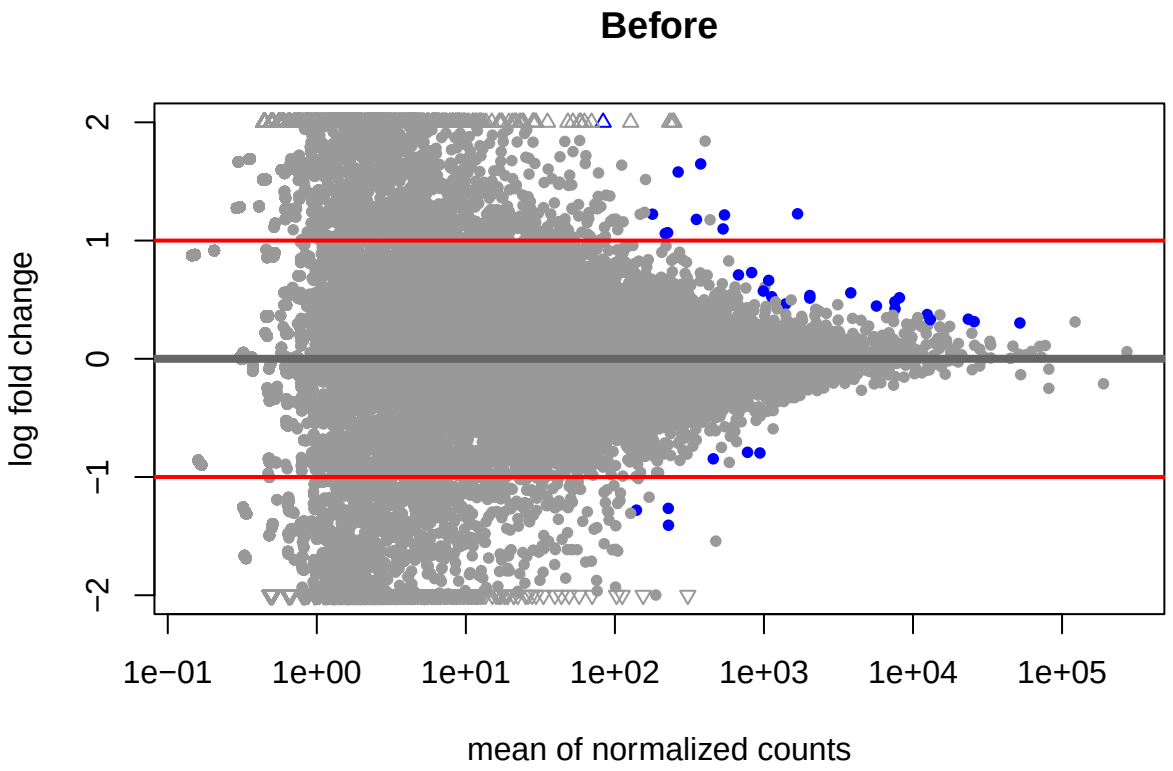
BMAT alone vs co_cultured

Before batch correction: PCA shows the variation between samples due to the experimental conditions (co-cultured vs. alone). After batch correction: Batch effects (due to replicate differences) have been corrected, revealing more accurate clustering of samples based on the primary experimental condition.

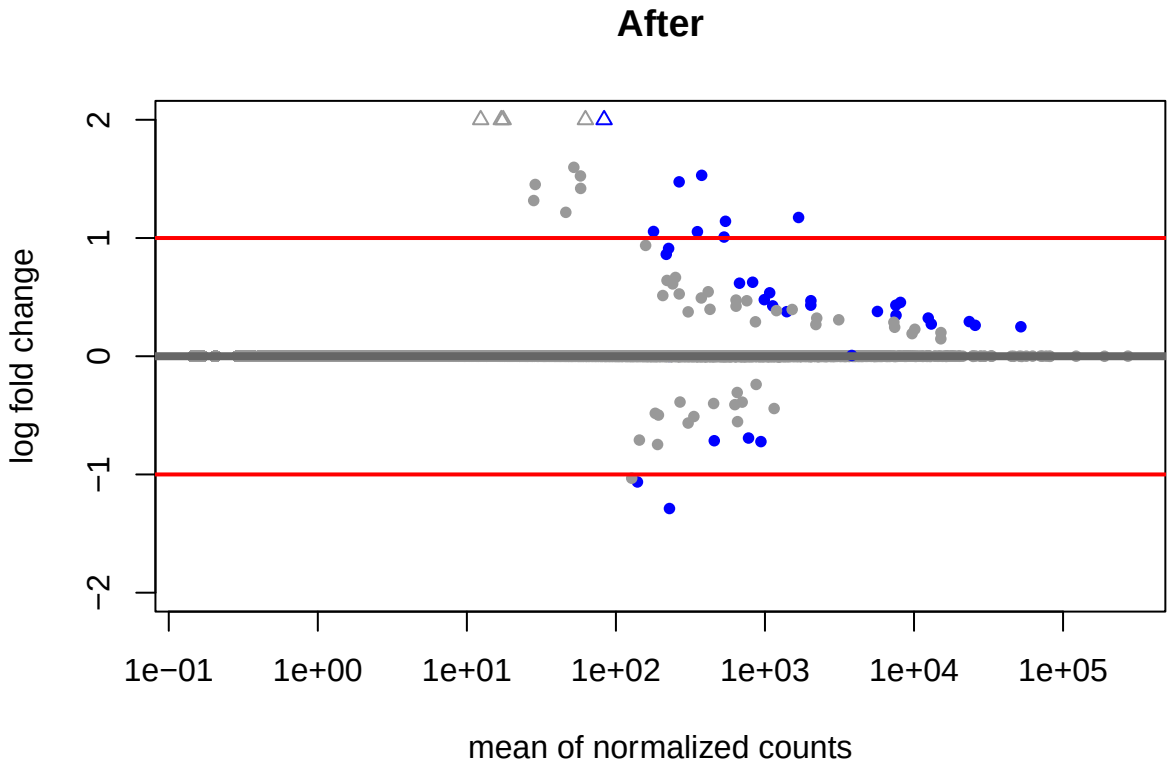
Log2 Fold Change Shrinkage

Log fold change shrinkage helps mitigate the effect of outliers and genes with low expression, resulting in more reliable estimates of differential expression.

```
# Setting contrast to test differential expression between co-cultured vs alone cells
contrast <- c("co_cultured", "TRUE", "FALSE")

#bmat
# Calculating the differential expression results before applying log2 fold change shrinkage
res_bmat <- results(dds_bmat, contrast = contrast, alpha = 0.1)

# Shrinking the log2 fold change estimates to avoid high variance for low-count genes
res_bmat_before <- res_bmat
res_bmat <- lfcShrink(
    dds_bmat,
    res = res_bmat,
    coef = "co_cultured_TRUE_vs_FALSE",
    type = "apeglm")

# Plot the before and after
ma_bmat_before <- plotMA(res_bmat_before, ylim=c(-2,2), cex=.8, main = 'Before') +
    abline(h=c(-1,1), col='red', lwd=2)
```



```
ma_bmat_after <- plotMA(res_bmat, ylim=c(-2,2), cex=.8, main = 'After') +
    abline(h=c(-1,1), col='red', lwd=2)
```



Summarizing results

The results seen here are the genes that are significantly DE without applying a threshold on fold-change

```
summary(res_bmat, alpha = 0.1)
```

```
##
## out of 25240 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 28, 0.11%
## LFC < 0 (down)     : 6, 0.024%
## outliers [1]       : 0, 0%
## low counts [2]     : 15773, 62%
## (mean count < 74)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
# Wrangling for downstream analysis
res_bmat_tb <- res_bmat %>%
    data.frame() %>%
```

```
    rownames_to_column(var="gene") %>%
    as_tibble()
```

Full DE genes To get the full list of DE genes that adhere to a fold change threshold, some additional data wrangling is performed. A log2 fold change threshold of 1 and a padj value of less than 0.1 for significance are used here. The resulting gene symbols and Entrez IDs are also retrieved for downstream analysis.

```
# Filtering DE genes with a fold change threshold and significant adjusted p-value
DE_bmat <- res_bmat_tb %>%
    filter(padj < 0.1 & abs(log2FoldChange) > 1)

# Adding gene symbols and Entrez IDs
DE_bmat$symbol <- mapIds(
    org.Hs.eg.db,
    keys=DE_bmat$gene,
    column="SYMBOL",
    keytype="ENSEMBL",
    multiVals="first")

DE_bmat$entrez <- mapIds(
    org.Hs.eg.db,
    keys=DE_bmat$gene,
    column="ENTREZID",
    keytype="ENSEMBL",
    multiVals="first")

write.csv(DE_bmat, "DE_BMAT.csv", row.names = T)
knitr::kable(head(DE_bmat))
```

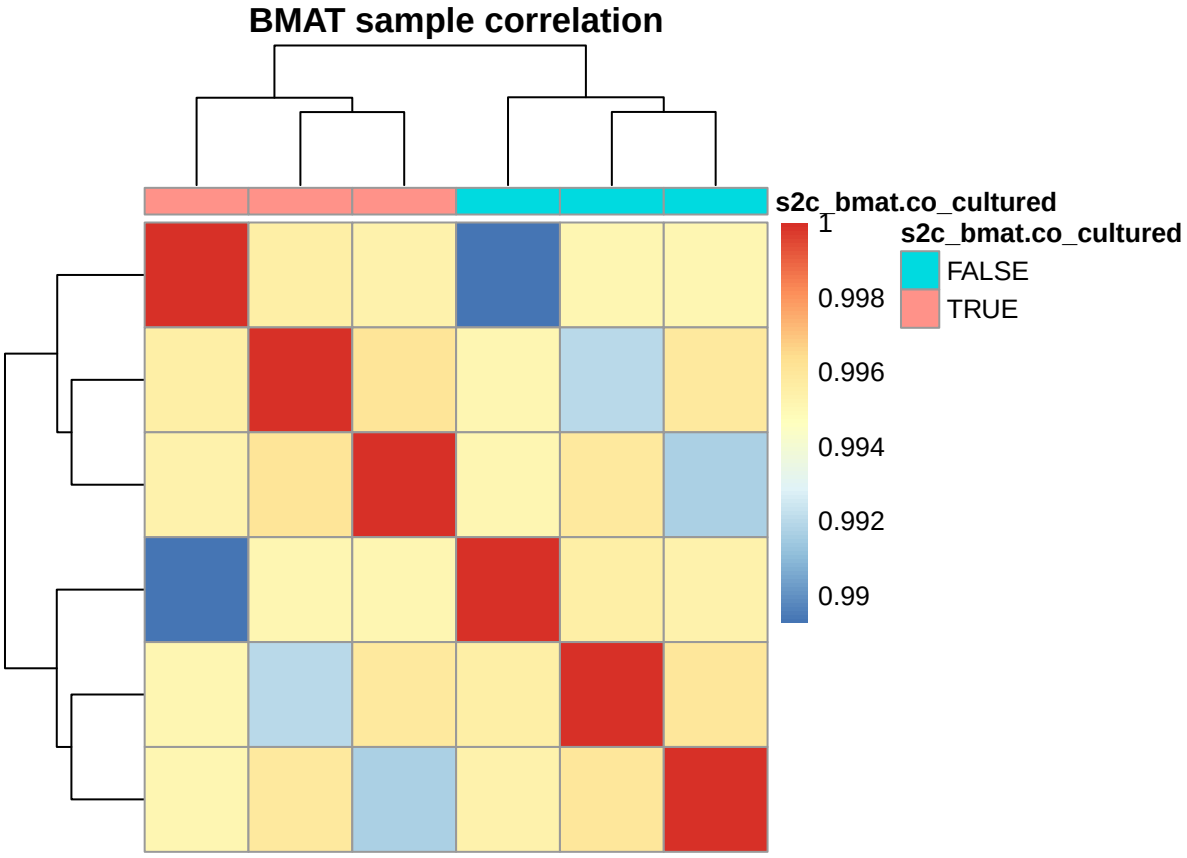
gene	baseMean	log2FoldChange	lfcSE	pvalue	padj	symbol	entrez
ENSG00000099998	352.1375	1.053354	0.2742588	3.6e-06	0.0031566	GGT5	2687
ENSG00000101868	139.4418	-1.063507	0.3803569	1.5e-04	0.0525856	POLA1	5422
ENSG00000124333	83.3271	7.171929	2.0807018	1.5e-06	0.0015911	VAMP7	6845
ENSG00000134900	543.3568	1.141291	0.2158531	0.0e+00	0.0000204	TPP2	7174
ENSG00000142949	531.7403	1.008092	0.2243275	2.0e-07	0.0003645	PTPRF	5792
ENSG00000164742	178.7295	1.055091	0.3252670	3.9e-05	0.0217231	ADCY1	107

Sanity checks

Below, sample correlation heatmaps are plotted. This allows to check whether the samples cluster as expected, with similar samples (e.g., co-cultured vs alone) clustering together. This ensures that the data is reliable for downstream analysis.

```
# Creating a correlation matrix based on the variance-stabilized data
vst_mat_bmat <- assay(vsd_bmat)
vst_cor_bmat <- cor(vst_mat_bmat)

# Adding sample annotations for heatmap
bmat_annotation_heatmap <- data.frame(row.names=colnames(vst_mat_bmat), s2c_bmat$co_cultured)
```

```
# Plotting the correlation heatmap
pheatmap(vst_cor_bmat,
         annotation_col = bmat_annotation_heatmap,
         main = "BMAT sample correlation",
         show_rownames = F,
         show_colnames = F)
```



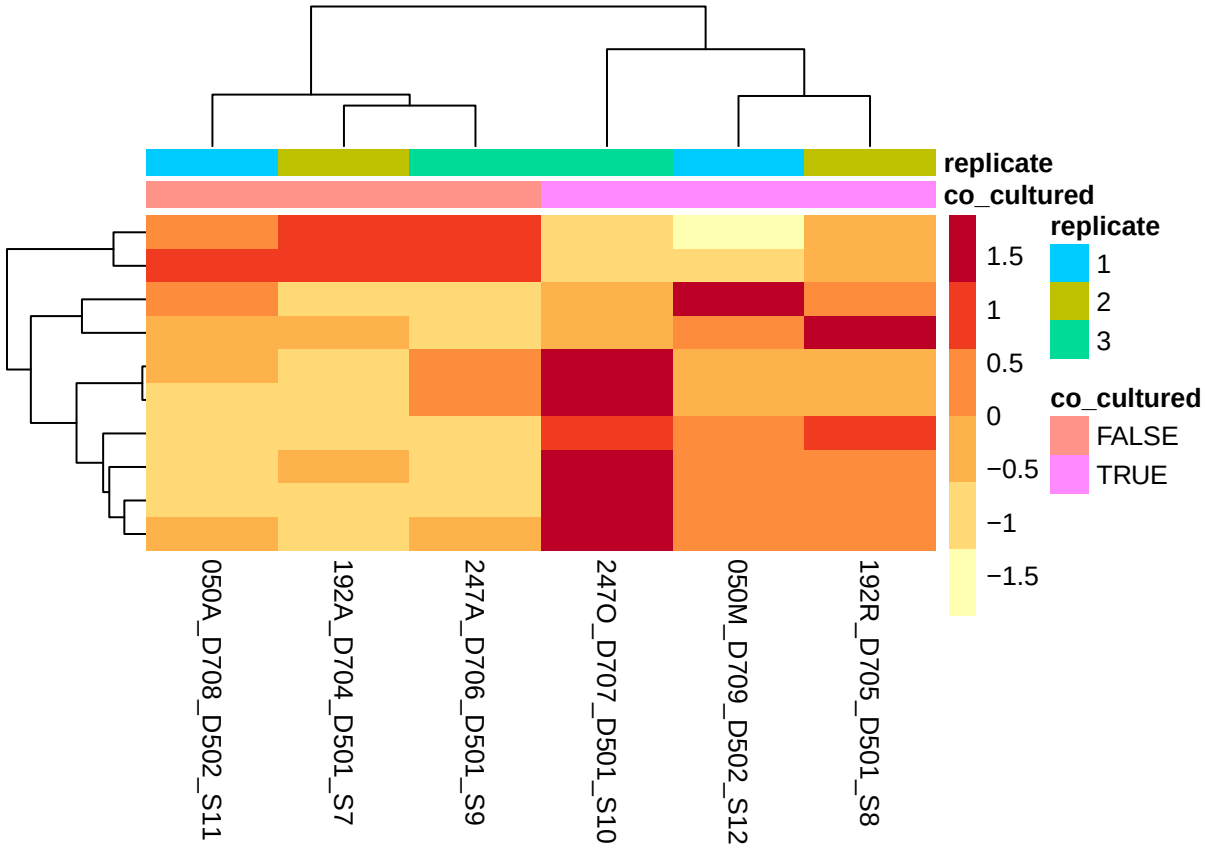
Another annotation file:

```
grch38annot <- transcript_to_gene %>%
    dplyr::select(ensembl_gene_id, external_gene_name) %>%
    dplyr::distinct()
```

pheatmap Below, normalized read counts across samples are compared and similar samples are clustered together

```
normalized_counts_bmat <- counts(dds_bmat, normalized=T) %>%
    data.frame() %>%
    rownames_to_column(var="gene")

names(normalized_counts_bmat) <- sub("X", "", names(normalized_counts_bmat))

normalized_counts_bmat <- merge(normalized_counts_bmat, grch38annot, by.x="gene", by.y="ensembl_gene_id
    as_tibble()
```

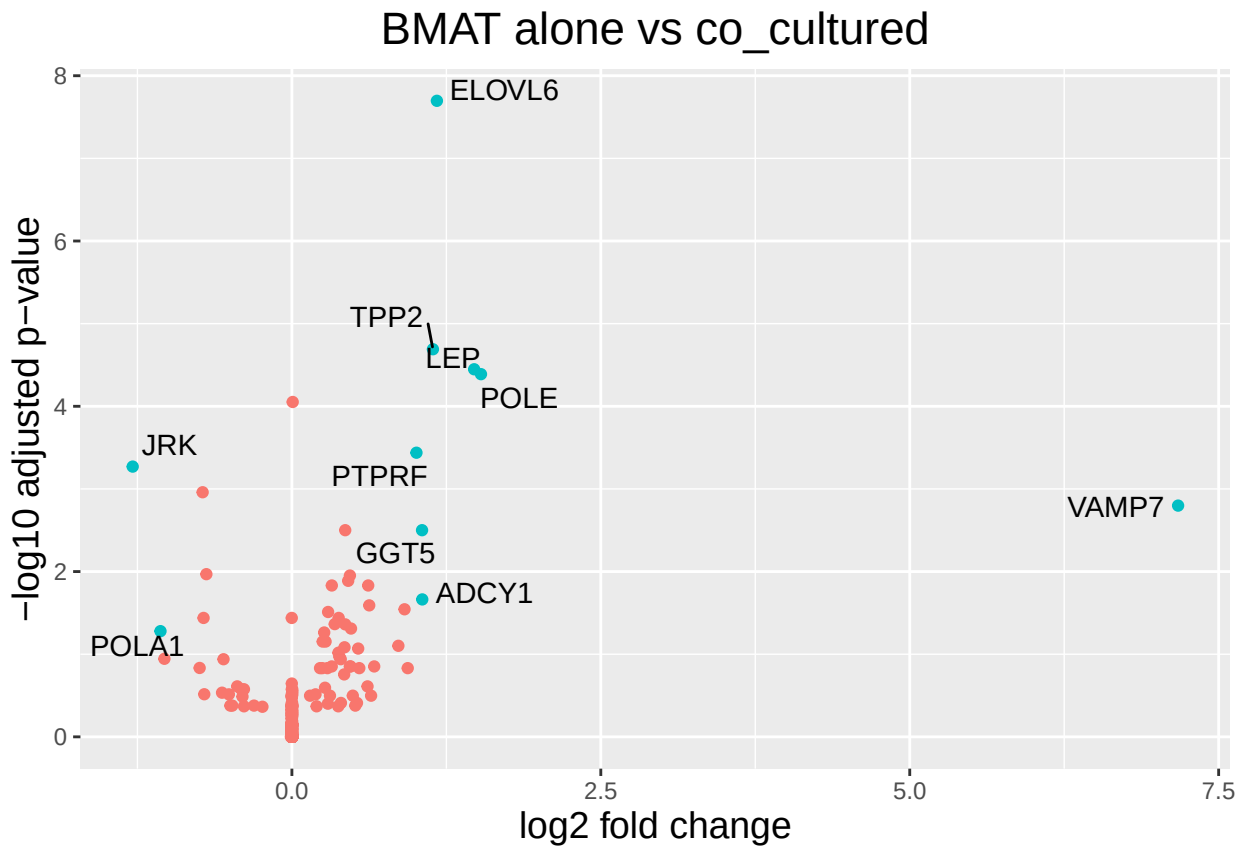
```
norm_sig_bmat <- normalized_counts_bmat %>% filter(gene %in% DE_bmat$gene)


heat_colors <- brewer.pal(6, "YlOrRd")
pheatmap(norm_sig_bmat[,2:7],
         color = heat_colors,
         cluster_rows = T,
         show_rownames = F,
         annotation = s2c_bmat,
         border_color = NA,
         fontsize = 10,
         scale = "row",
         fontsize_row = 10,
         height = 20,
         drop_levels = T)
```



Volcano plot We can use volcano plots to visually see the spread of DE genes, either up or down regulated. The color of the points denotes significance and the label is applied to the genes that met p-value and `log2foldchange` thresholds.

```
# Preparing data for volcano plot
res_bmat_tb <- na.omit(res_bmat_tb %>%
    mutate(threshold_OE = padj < 0.1 & abs(log2FoldChange) >= 1))
```

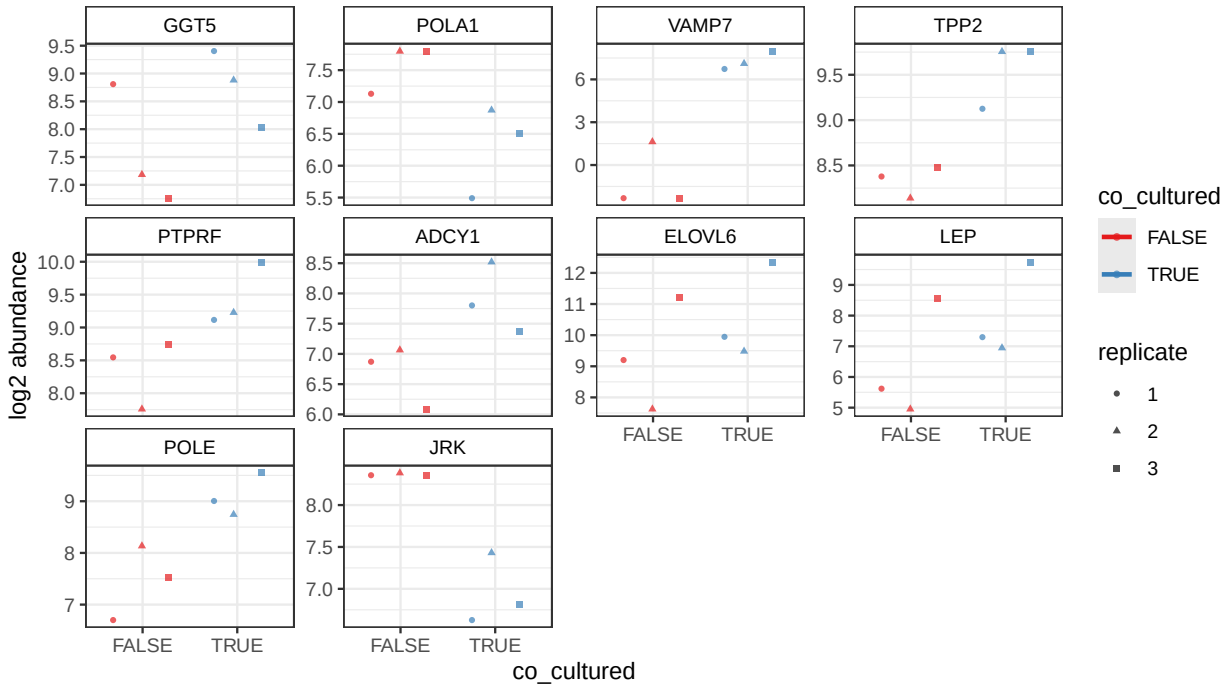
```
## Add all the gene symbols as a column from the grch38 table using bind_cols()
res_bmat_tb <- bind_cols(res_bmat_tb, symbol=grch38annot$external_gene_name[match(res_bmat_tb$gene, grch
res_bmat_tb <- res_bmat_tb %>% mutate(genelabels = "")
res_bmat_tb <- res_bmat_tb %>% arrange(padj)

# Labeling significant genes
res_bmat_tb$genelabels[res_bmat_tb$threshold_OE == T] <- as.character(res_bmat_tb$symbol[res_bmat_tb$thr

# Plotting the volcano plot
ggplot(res_bmat_tb, aes(x = log2FoldChange, y = -log10(padj))) +
    geom_point(aes(colour = threshold_OE)) +
    geom_text_repel(aes(label = genelabels)) +
    ggtitle("BMAT alone vs co_cultured") +
    xlab("log2 fold change") +
    ylab("-log10 adjusted p-value") +
    theme(legend.position = "none",
          plot.title = element_text(size = rel(1.5), hjust = 0.5),
          axis.title = element_text(size = rel(1.25)))
```



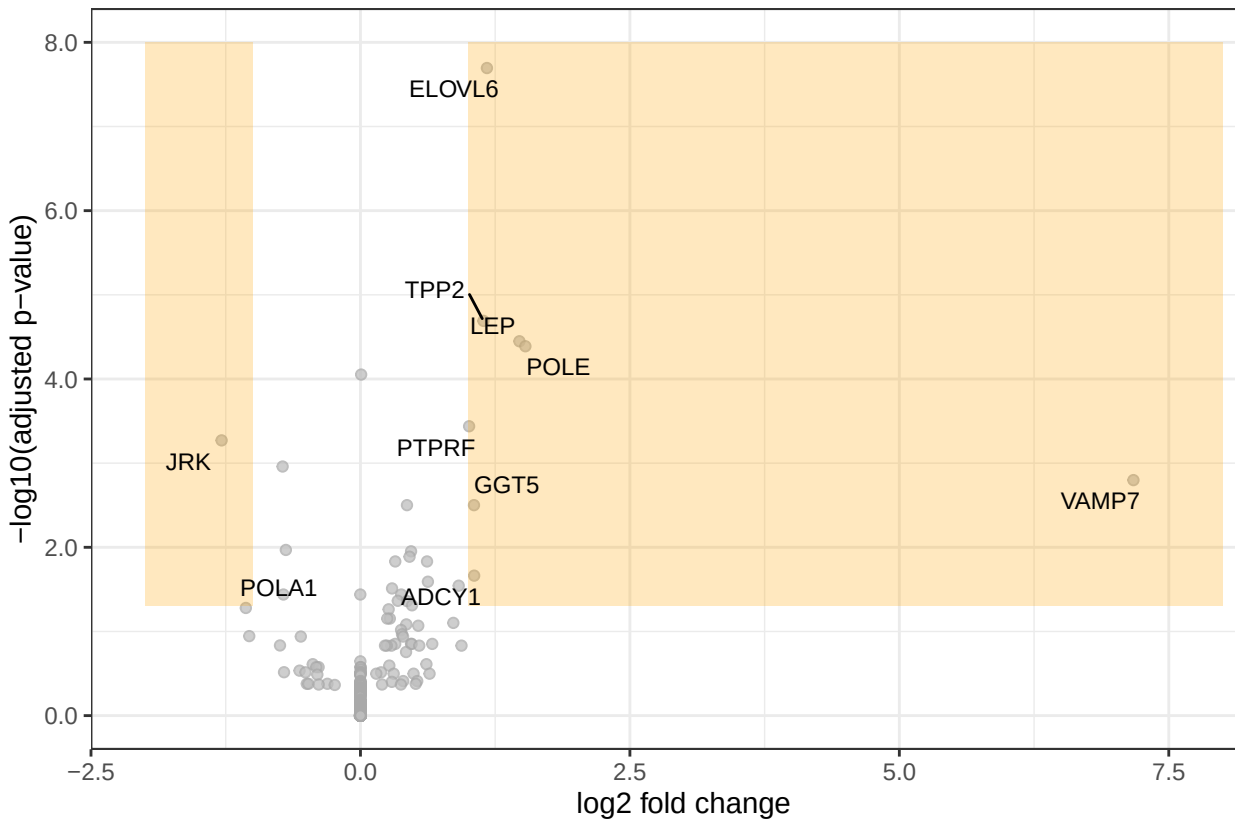
Using degPlot:

```
dds_bmat_deg <- dds_bmat

rowData(dds_bmat_deg)$symbol <- mapIds(
    org.Hs.eg.db,
    keys=rownames(dds_bmat_deg),
```

```
    column="SYMBOL",
    keytype="ENSEMBL",
    multiVals="first")

rowData(dds_bmat_deg)$gene <- rownames(dds_bmat_deg)

degPlot(
    dds=dds_bmat_deg,
    res=DE_bmat,
    n = nrows(DE_bmat),
    genes = DE_bmat$gene,
    xs = "co_cultured",
    group = "co_cultured",
    ann = c("gene", "symbol"),
    batch = "replicate")
```




```
# res_bmat_tb$genelabels[1:20] <- as.character(res_bmat_tb$symbol[1:20])
degVolcano(data.frame(res_bmat_tb[,c("log2FoldChange","padj")]),
           plot_text=data.frame(res_bmat_tb[res_bmat_tb$threshold_OE == T,c("log2FoldChange","padj","ge
```



Gene Set Enrichment Analysis

Because of the small number of DE genes that have been identified, programmatic analysis of the DE genes is somewhat constrained and results will be minimum.

GSEA with MSigDB Hallmark Pathways

GSEA allows identification of pathways that are significantly enriched among upregulated or downregulated genes. This analysis helps us understand the biological processes impacted by the differential gene expression.

```
original_gene_list_bmat <- DE_bmat$log2FoldChange
names(original_gene_list_bmat) <- DE_bmat$gene
gene_list_bmat <- na.omit(original_gene_list_bmat)
gene_list_bmat <- sort(original_gene_list_bmat, decreasing = T)

gse_bmat_enrich <- enrichGO(names(gene_list_bmat),
              ont ="ALL",
              keyType = "ENSEMBL",
              minGSSize = 3,
              maxGSSize = 800,
              pvalueCutoff = 1,
              OrgDb = org.Hs.eg.db,
              pAdjustMethod = "hochberg")
```

```
gse_bmat <- gseGO(geneList=gene_list_bmat,
              ont ="ALL",
              keyType = "ENSEMBL",
              minGSSize = 3,
              maxGSSize = 800,
              pvalueCutoff = 1,
              verbose = TRUE,
              OrgDb = org.Hs.eg.db,
              pAdjustMethod = "hochberg")
```

```
## preparing geneSet collections...

## GSEA analysis...

## leading edge analysis...

## done...
```

```
gse_bmat_pairwise <- pairwise_termsim(gse_bmat)

# Converting some names from Ensembl to Entrez, which is used by KEGG
original_gene_list_bmat_entrez <- original_gene_list_bmat
names(original_gene_list_bmat_entrez) <- DE_bmat$entrez
gene_list_bmat_entrez <- na.omit(original_gene_list_bmat_entrez)
gene_list_bmat_entrez <- sort(original_gene_list_bmat_entrez, decreasing = T)
gse_bmat_do <- gseDO(gene_list_bmat_entrez, pvalueCutoff = 1)
```

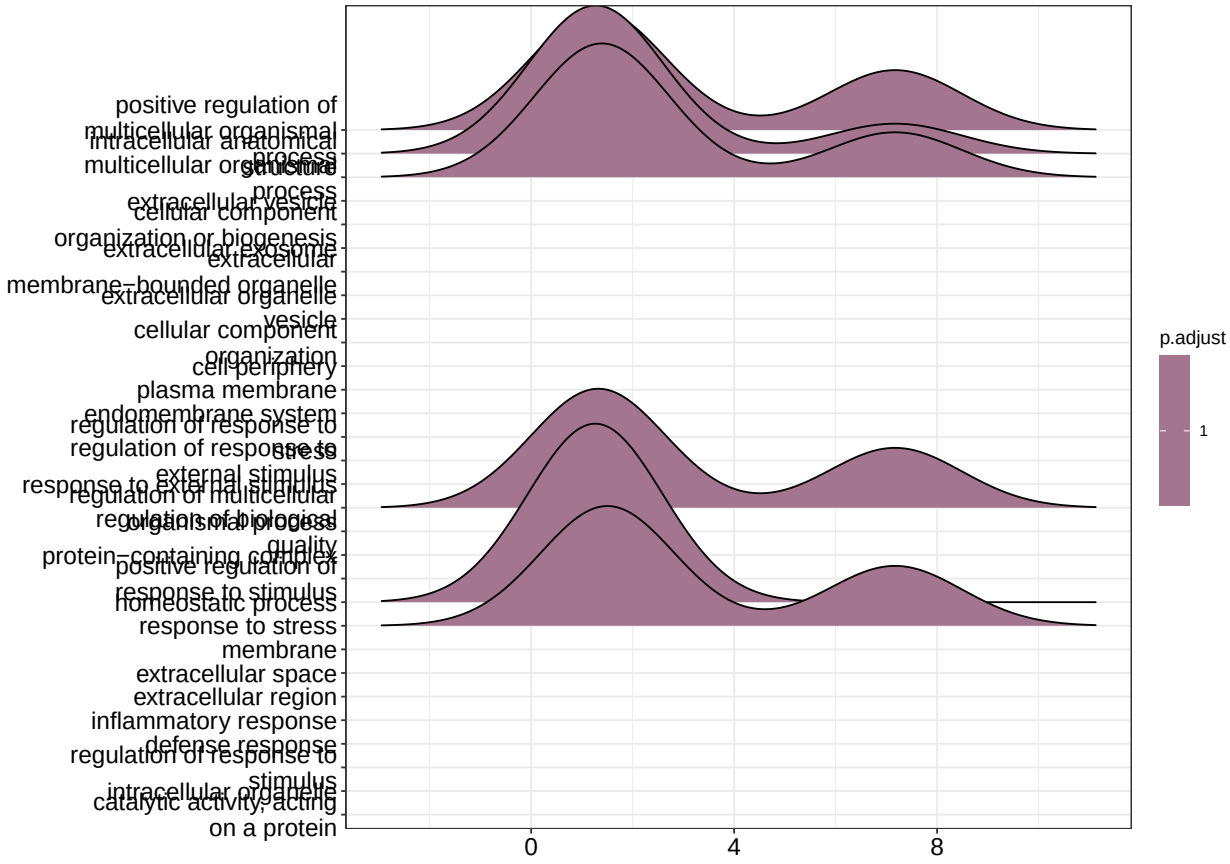
```
## preparing geneSet collections...

## GSEA analysis...

## no term enriched under specific pvalueCutoff...
```

Visualization of GSEA Results

Pathway Visualization with Ridge Plot

```
# Ridgeplot of GSEA results
ridgeplot(gse_bmat) + ggtitle("Enriched Pathways (GSEA)" + theme(axis.text.y = element_text(angle = 45,
                                                                 axis.text.x = element_text(size = 6)))
```

```
## Picking joint bandwidth of 1.32
```



'ridgeplot' shows the distribution of enrichment scores across the top enriched pathways. Pathways at the top are significantly enriched.

KEGG Pathway Enrichment

```
# Running KEGG enrichment analysis
kk_bmat<- enrichKEGG(names(gene_list_bmat_entrez),
                organism     = "hsa",
                minGSSize    = 3,
                maxGSSize    = 800,
                pvalueCutoff = 1,
                pAdjustMethod = "BH",
                keyType       = "kegg")
```

```
## Reading KEGG annotation online: "https://rest.kegg.jp/link/hsa/pathway"...
```

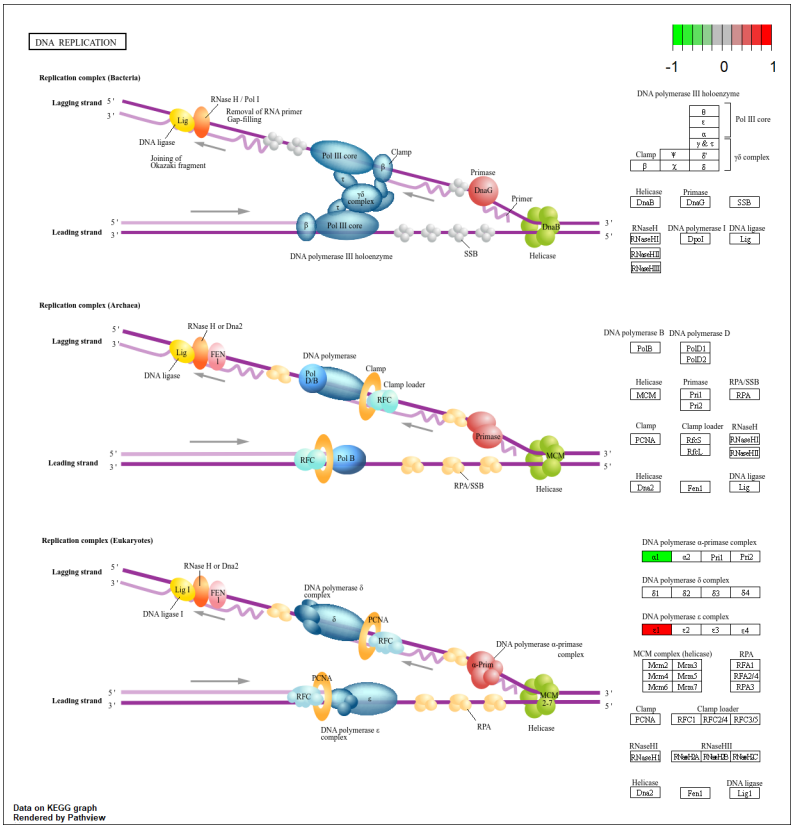
```
## Reading KEGG annotation online: "https://rest.kegg.jp/list/pathway/hsa"...
```

```
# Visualizing KEGG results with a dotplot
dotplot(kk_bmat, showCategory=10)
```

KEGG pathway analysis reveals potential biological pathways that are enriched in the DE gene set. The dotplot highlights the top enriched pathways, providing insights into the biological context of the gene expression changes.

Pathway Visulaization

```
pathview(gene_list_bmat_entrez, pathway.id = "hsa03030", species = "hsa")
knitr::include_graphics("hsa03030.pathview.png")
```



The above pathway map shows the genes from the dataset that are involved in the DNA replication pathway. Genes that are up or down regulated are color-coded. By examining this pathway, we can understand how the biological processes related to DNA replication might be altered in the condition being studied.

Up-regulated genes are shown in red or similar warm tones. Down-regulated genes appear in blue or cool tones.