RNAseq Analysis E. coli strain MG1655 RNAseq on MG1655 reference and LF82 SD105 on LF82 reference

Hernan Lorenzi

12/06/2023

```
pacman::p_load(AnnotationDbi,pheatmap,EnhancedVolcano,ggpubr,DESeq2,stringr,biomaRt,tidyverse,pcaExplor
```

Load libraries

```
# # Load auxyliary functions
# source(file = "../results_MG1655/01_aux_rnaseq_functions.R")
#
# # Load enrichment functions
# source(file = "../results_MG1655/02_Gene_enrichment_functions.R")
```

```
#
# --- function for PCA plots ---
#
plot_PCA = function(object, color_by="condition",
                    shape_by = 19, ntop=500, size = 3,
                    returnData=FALSE, pcs = c(1,2))
{
  # Check variables are present in object
  intgroup = c(color_by)
  if (shape_by != 19){intgroup <- c(intgroup, shape_by)}
  if (!all(intgroup %in% names(colData(object)))) {
    stop("the argument 'intgroup' should specify columns of colData(dds)")
  }

  # calculate the variance for each gene
  rv <- rowVars(assay(object))

  # select the ntop genes by variance
  select <- order(rv, decreasing=TRUE)[seq_len(min(ntop, length(rv)))]

  # perform a PCA on the data in assay(x) for the selected genes
  pca <- prcomp(t(assay(object)[select,]))

  # the contribution to the total variance for each component
  percentVar <- pca$sdev^2 / sum( pca$sdev^2 )


  intgroup.df <- as.data.frame(colData(object)[, intgroup, drop=FALSE])
  # add the intgroup factors together to create a new grouping factor
  group <- if (length(intgroup) > 1) {
    factor(apply( intgroup.df, 1, paste, collapse=":"))
  } else {
    colData(object)[[intgroup]]
  }

  # assembly the data for the plot
  d <- data.frame(PC1=pca$x[,pcs[1]], PC2=pca$x[,pcs[2]], group=group, intgroup.df, name=colnames(objec
  colnames(d)[1] <- paste0("PC",pcs[1])
  colnames(d)[2] <- paste0("PC",pcs[2])

  if (returnData) {
    attr(d, "percentVar") <- percentVar[1:2]
    return(d)
  }

  ggplot(data=d, aes_string(x=colnames(d)[1], y=colnames(d)[2], color=color_by, shape=shape_by)) +
    geom_point(size=size) +
    scale_color_lancet() +
    xlab(paste0("PC",pcs[1],": ",round(percentVar[pcs[1]] * 100, digits = 2),"% variance")) + # fixed
    ylab(paste0("PC",pcs[2],": ",round(percentVar[pcs[2]] * 100, digits = 2),"% variance")) + # fixed
    coord_fixed(ratio = (max(d[,1])-min(d[,1]))/(max(d[,2])-min(d[,2])))
}
```

Define functions

```
all.star <- read.delim2("./data/read_counts_mg_lf82.txt",
                        sep = "\t",
                        header = TRUE,
                        row.names = 1,
                        comment.char = c("#") )

format_star <- function(star_file){
  names(star_file) <- names(star_file) %>%
    str_remove_all(pattern = "results.03map_reads.|.Aligned.sortedByCoord.out.bam")
  return(star_file[3:ncol(star_file)])
}

# Format star counts file
all <- format_star(star_file = all.star)

# Make sure read counts are numeric and rounded to 0 decimals
all.tmp <- as.data.frame(lapply(all, function(x){ round(as.numeric(x), digits = 0)} ))
rownames(all.tmp) <- rownames(all)
all <- all.tmp

#Remove all zero rows
#all <- remove_all_zero_rows(all, min_total_count = 0)

column_names <- names(all) %>% sort()
```

```
all <- dplyr::select(all, column_names)
```

Load read counts data MG1655

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(column_names)
##
##   # Now:
##   data %>% select(all_of(column_names))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
# Load metadata

metadata <- data.frame(row.names = column_names,
                       genotype = c(rep("WT", 3),rep("Mutant",3),rep("Mutant", 3),rep("WT",3)),
                       strain = c(rep("LF82",6),rep("MG1655",6)))


# Add total read counts and sample id columns to metadata
metadata$read_counts <- colSums(all)

# Add "Sample_name" as column in metadata
metadata$sample_name <- rownames(metadata)

# Add column combining genotype and treatment
metadata$group <- paste(metadata$strain,metadata$genotype, sep = "_")
```

Make metadata table from 'all'

```
# Using annotation version GRCm39 (current)
#all.tpm <- normalize_by_TPM(counts.df = all,
#                             gene_length = dplyr::select(all.star, c("Length")))
```

Normalize data to TPMs to run some comparative analysis across samples

Analysis of expression data using DESeq2

```
# Convert metadata to factors
for (variable in c("genotype", "strain","sample_name","group")){
  metadata[,variable] <- as.factor(str_replace_all(metadata[,variable], pattern = " ", replacement = "_
}
```

Analysis of Dataset

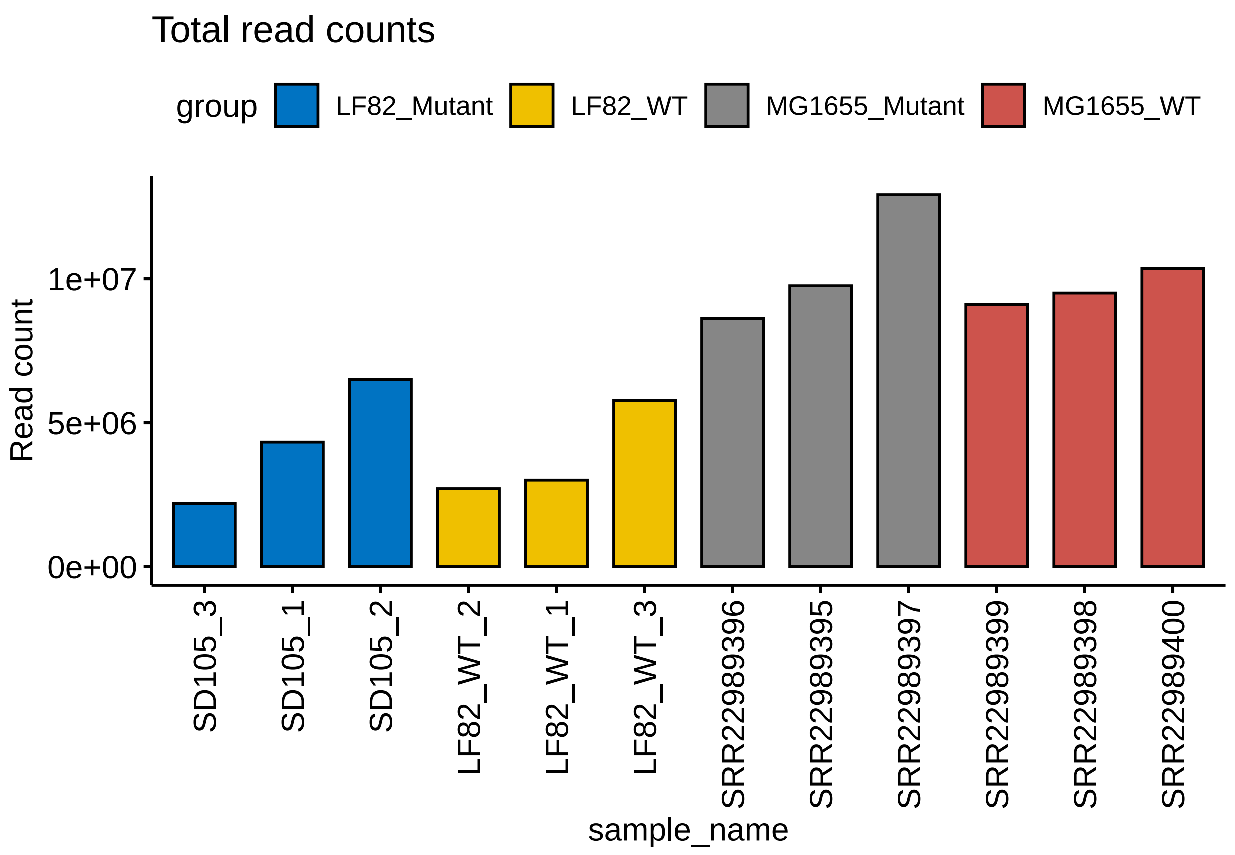
```r
# Generate DESeq2 object for NS and ST condition ONLY. We could potentially add Read_counts as either a

dir.create(path = "./Plots", showWarnings = FALSE)

# Create DESeq object
dds.all <- DESeqDataSetFromMatrix(countData = all,
                                  colData = metadata,
                                  design = ~ strain + genotype)
```

```
## converting counts to integer mode
```

```r
# Plot total reads per sample using barchar
p <- ggbarplot(data = metadata,
          x = "sample_name",
          y = "read_counts",
          x.text.angle = 90,
          fill = "group",
          title = "Total read counts",
          ylab = "Read count",
          sort.by.groups = TRUE,
          palette = "jco",
          sort.val = "asc")
ggsave2("Plots/barplot_read_counts.pdf", plot = p)
```

```
## Saving 6.5 x 4.5 in image
```

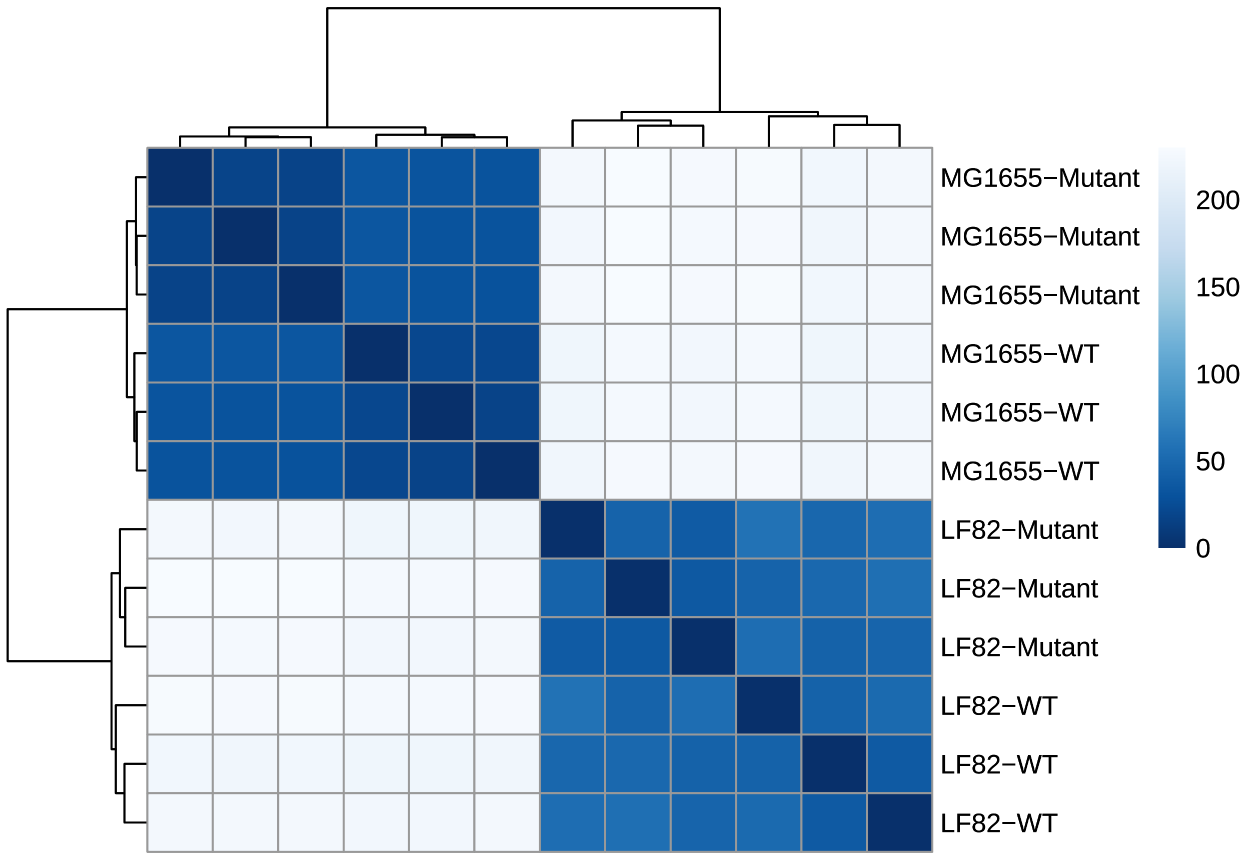
```r
print(p)
```



```
# Normalize counts
vsd.one <- vst(dds.all, blind=FALSE)
rlog.one <- rlog(dds.all, blind=FALSE)

# Keep genes with at least 20 reads total across samples
#keep <- rowSums(counts(dds.all)) >= 20
#dds.all <- dds.all[keep,]

# Calculate distances between samples
sampleDists <- dist(t(assay(vsd.one)))

# Plot inter-sample distances
old.par <- par(no.readonly=T)

sampleDistMatrix <- as.matrix(sampleDists)
rownames(sampleDistMatrix) <- paste(rlog.one$strain, rlog.one$genotype, sep="-")
colnames(sampleDistMatrix) <- NULL
colors <- colorRampPalette( rev(brewer.pal(9, "Blues")) )(255)
p.pheatmap <- pheatmap(sampleDistMatrix,
         clustering_distance_rows=sampleDists,
         clustering_distance_cols=sampleDists,
         col=colors)

ggsave2(filename = "unsupervised_clustering_rnaseq_profile_20plus_reads.pdf", plot = p.pheatmap, path =
```

```
## Saving 6.5 x 4.5 in image
```

```r
print(p.pheatmap)
```



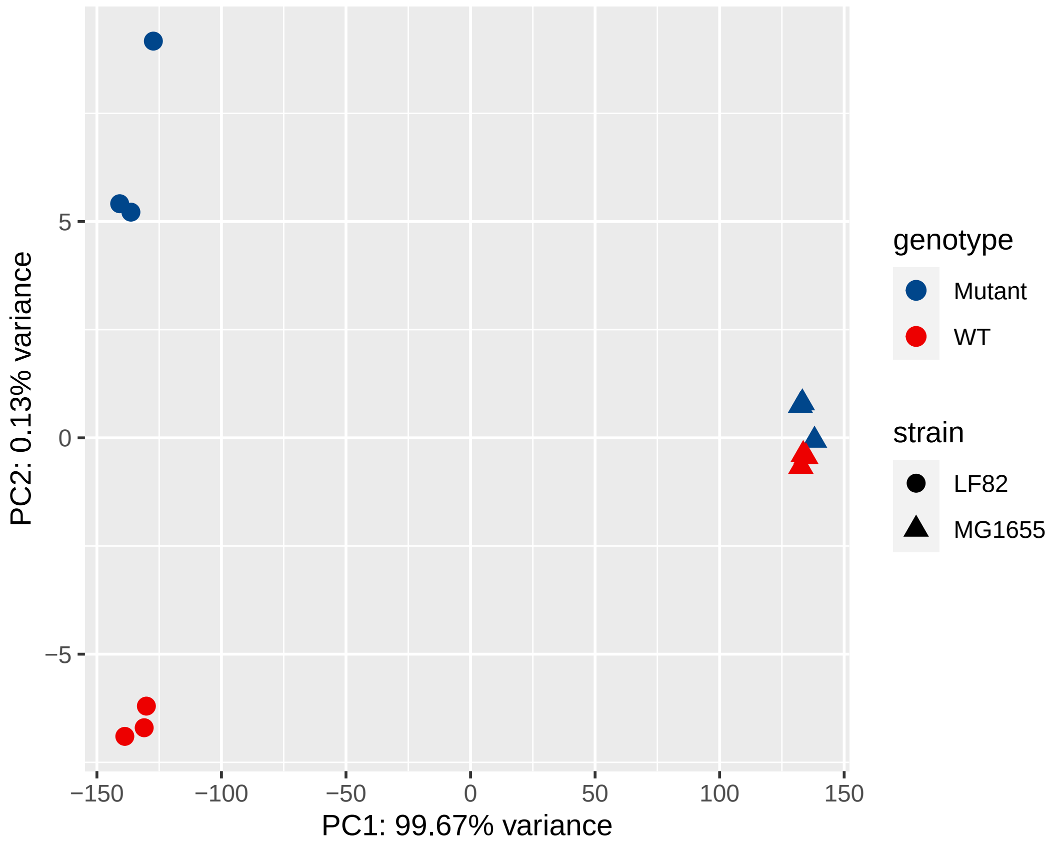
```r
dds_res <- list()

dds_res <- dds.all #[ , dds.all$Tissue=="all_data"]

rlog_res <- list()
rlog_res <- rlog(dds_res, blind=FALSE)

# Filter out genes that are expressed in LF82/MG1655 strain but not the other
keep <- apply(assay(rlog_res), 1, function(x){sum(x>0) > 6})
rlog_res.all_strains <- rlog_res[keep,]

# PCA
rlog.one <- rlog_res

# PC1 - PC2
principal_components <- c(1,2)
pca_12.p <- plot_PCA(object = rlog.one,color_by = "genotype", shape_by = "strain", returnData = FALSE ,
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
# PC2 - PC3
principal_components <- c(2,3)
pca_23.p <- plot_PCA(object = rlog.one,color_by = "genotype", shape_by = "strain", returnData = FALSE ,p

# PC1 - PC3
principal_components <- c(1,3)
pca_13.p <- plot_PCA(object = rlog.one,color_by = "genotype", shape_by = "strain", returnData = FALSE ,p

ggsave(paste0("Plots/pca_PC12_Group.pdf"), plot = pca_12.p)
```

```
## Saving 6.5 x 4.5 in image
```

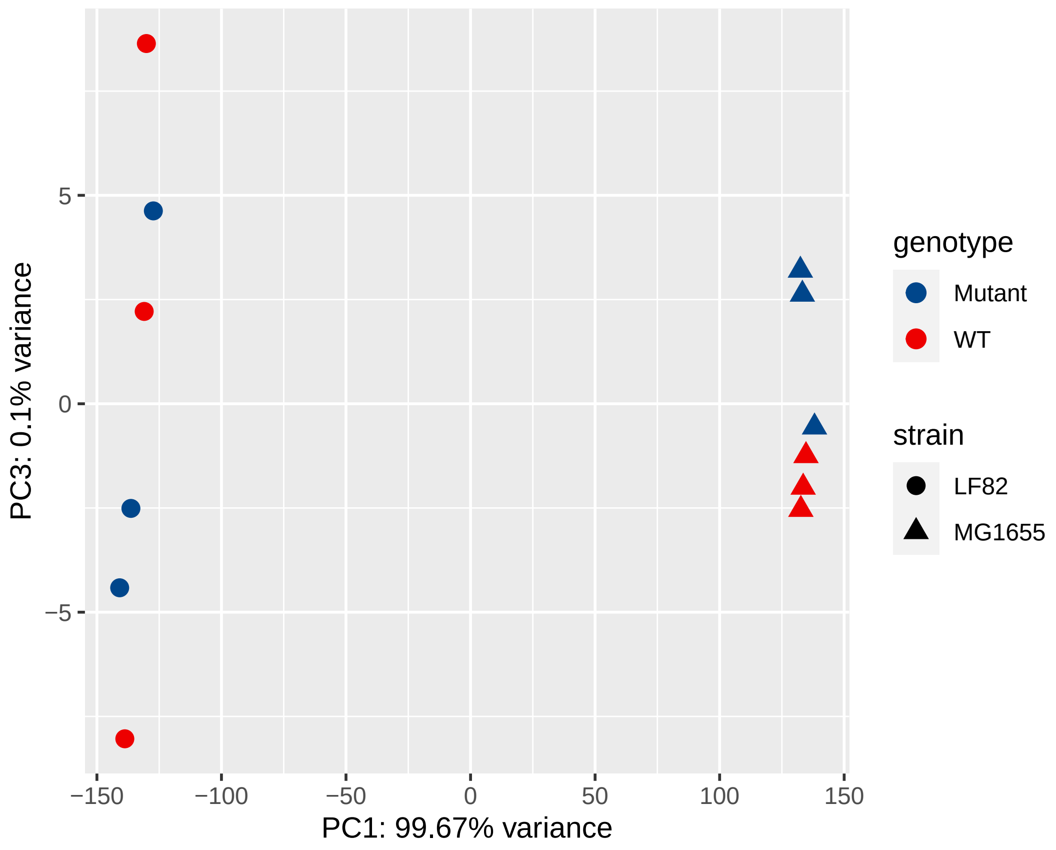
```
print(pca_12.p)
```



```
ggsave(paste0("Plots/pca_PC13_Group.pdf"), plot = pca_13.p)
```

```
## Saving 6.5 x 4.5 in image
```

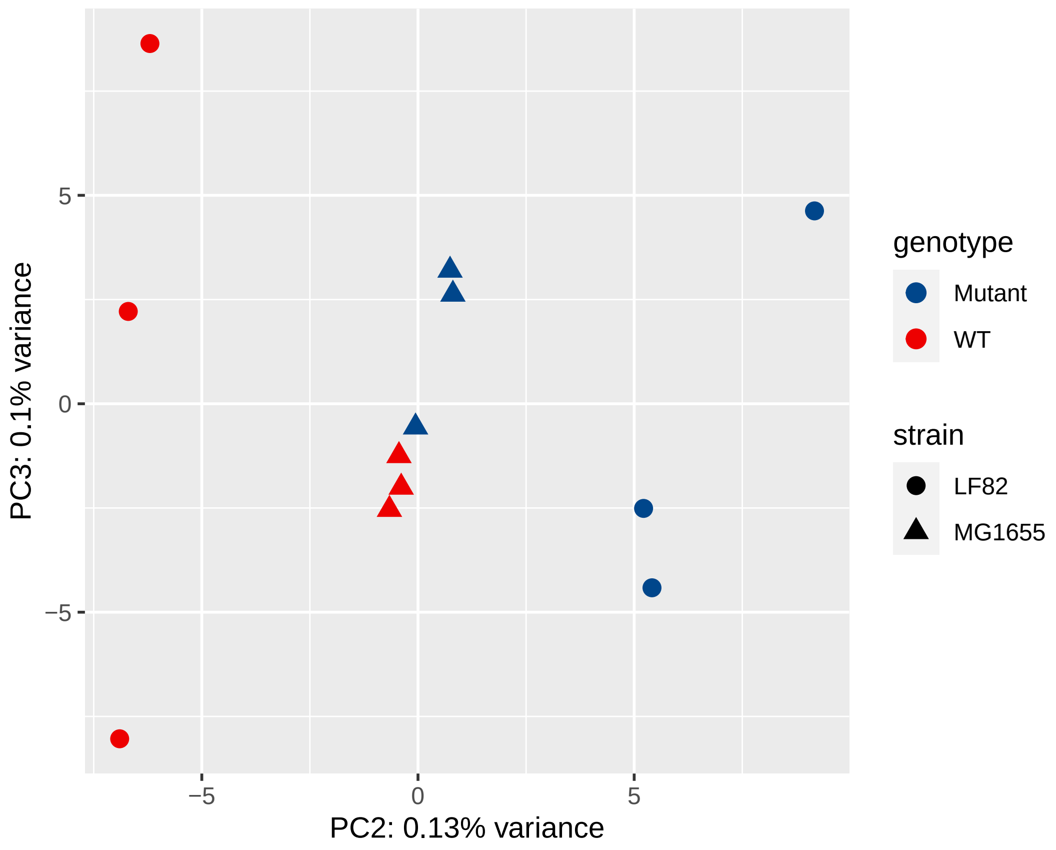
```
print(pca_13.p)
```



```
ggsave(paste0("Plots/pca_PC23_Group.pdf"), plot = pca_23.p)
```

```
## Saving 6.5 x 4.5 in image
```

```
print(pca_23.p)
```



PCA analysis shows that samples separate by genotype and treatment.

resultsNames(dds)

```
# Keep genes with at least 10 reads total across samples
keep <- rowSums(counts(dds_res)) >= 20
dds_res <- dds_res[keep,]
all <- all[keep,]
all.star <- all.star[keep,]
```

Filtering out poorly-expressed genes (less than 20 reads across all samples)

```
ensembl_to_symbol <- read.delim(file = "./data/gene_names mg_lf82.txt", col.names = c("Ensembl_ID","gene

# Save sorted files as a list
DE_results <- list()
geneids.DE <- list()

# Define function for processing and saving result tables
sort_and_write_res_table <- function(result_table, file_name){
  dir.create(path = "./DE", showWarnings = FALSE)
  # Sort genes by (padj)
  result_table_sorted <- result_table[order(result_table$padj, decreasing = FALSE),]
  # Add gene symbols
  gene_list <- rownames(result_table_sorted)
  symbol_list <- ensembl_to_symbol$gene_name[match(gene_list, ensembl_to_symbol$Ensembl_ID)]
  df <-as.data.frame(cbind(result_table_sorted, Gene_name = symbol_list))

  # Write sorted table to file
  write.table(df, file = paste0("./DE/",file_name,".txt"),
            sep = "\t", col.names=NA)
  return(df)
}
```

```
# Calculate DE for all_data samples
#design(dds.rnaseA) <- ~Treatment # Removid Read.depth from formula given that all samples are Read.dep

design(dds_res) <- ~group
dds_res$group <- relevel(dds_res$group, "LF82_Mutant")
dds_res <- DESeq(dds_res)
```

Using groups instead of interactions

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
my_contrasts <- resultsNames(dds_res)
res_WT_vs_Mut_LF82 <- lfcShrink(dds_res, contrast=c("group", "LF82_WT", "LF82_Mutant"), type = "ashr",
```

```
## using 'ashr' for LFC shrinkage. If used in published research, please cite:
##     Stephens, M. (2016) False discovery rates: a new deal. Biostatistics, 18:2.
##     https://doi.org/10.1093/biostatistics/kxw041
```

```
res_WT_vs_Mut_LF82$LF82_IDs <- all.star$LF82_IDs

dds_res$group <- relevel(dds_res$group, "MG1655_WT")
dds_res <- DESeq(dds_res)
```

```
## using pre-existing size factors

## estimating dispersions

## found already estimated dispersions, replacing these

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
my_contrasts <- resultsNames(dds_res)

res_Mut_vs_WT_MG1655.no_shrink <- results(dds_res,
                                  contrast=c("group", "MG1655_Mutant", "MG1655_WT"))

res_Mut_vs_WT_MG1655 <- lfcShrink(dds_res,
                                  contrast=c("group", "MG1655_Mutant", "MG1655_WT"),
                                  type = "ashr" )
```

```
## using 'ashr' for LFC shrinkage. If used in published research, please cite:
##     Stephens, M. (2016) False discovery rates: a new deal. Biostatistics, 18:2.
##     https://doi.org/10.1093/biostatistics/kxw041
```

```
res_Mut_vs_WT_MG1655$MG1655 <- all.star$MG1655_IDs

summary(res_WT_vs_Mut_LF82, alpha = 0.05)
```

```
##
## out of 5236 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 326, 6.2%
## LFC < 0 (down)     : 413, 7.9%
## outliers [1]       : 9, 0.17%
## low counts [2]     : 1117, 21%
## (mean count < 8)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_Mut_vs_WT_MG1655, alpha = 0.05)
```

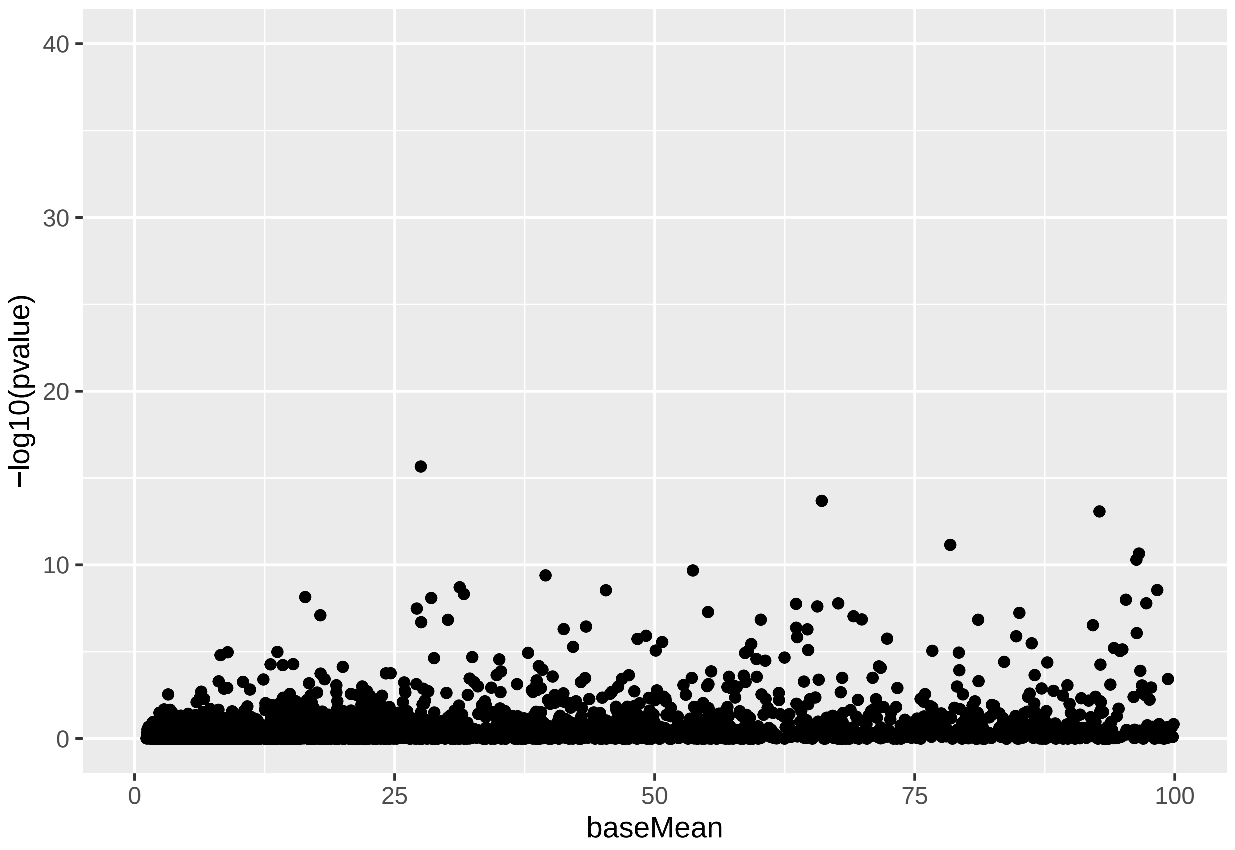
```
##
## out of 5236 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 347, 6.6%
## LFC < 0 (down)     : 281, 5.4%
## outliers [1]       : 9, 0.17%
## low counts [2]     : 1117, 21%
## (mean count < 8)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
# Sort results by Log2FC
res_WT_vs_Mut_LF82_sorted <- sort_and_write_res_table(result_table = res_WT_vs_Mut_LF82, file_name = pa
res_Mut_vs_WT_MG1655_sorted <- sort_and_write_res_table(result_table = res_Mut_vs_WT_MG1655, file_name =
```

```

p1 <- ggplot(as.data.frame(res_Mut_vs_WT_MG1655.no_shrink), aes(x=baseMean, y=-log10(pvalue))) + geom_p

print(p1)
```

MA plots

```
## Warning: Removed 1889 rows containing missing values (`geom_point()`).
```



```
table_counts_normalized <- counts(dds_res, normalized=TRUE)
write.table(x = as.data.frame(table_counts_normalized), file = "read_counts_deseq2_normalized.txt", sep
```

```

# yifL = lptM
# ybgD b0719 has less than 20 reads across all samples
genes_of_interest <- c("metE", "yobH", "lptM", "yigA", "xerC", "yigB", "ampC", "yciY", "shiA", "acrZ",
```

Genes of interest

```

volcano_plot_with_ids <- function(res.tmp, log_scale = FALSE, gene_list){
  vp <- EnhancedVolcano(res.tmp,
                        lab = res.tmp$Gene_name,
                        x = 'log2FoldChange',
                        y = 'padj',
                        pCutoff = 0.05,
                        FCcutoff = 1,
                        pointSize = 1,
                        colAlpha = 4/5,
                        labSize = 3,  # Controls labels size
                        labCol = "black",
                        title = '',
                        titleLabSize = 10,
                        subtitle = '', # add subtitle here
                        subtitleLabSize = 10,
                        legendPosition = 'right',
```

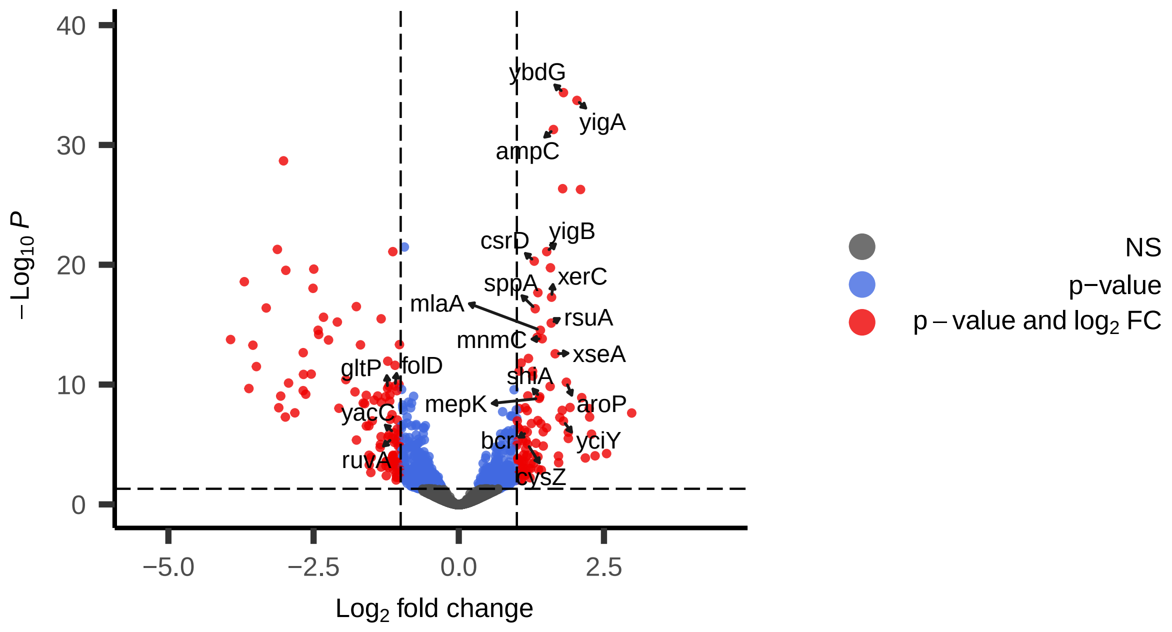
```r
                       legendLabSize = 10,
                       legendIconSize = 4.0,
                       axisLabSize = 10,
                       drawConnectors = TRUE,
                       selectLab = gene_list, # vector of gene symbols to label on volcanoplot
                       boxedLabels = FALSE,
                       gridlines.major = FALSE,
                       gridlines.minor = FALSE,
                       hlineCol = "gray", vlineCol = "gray"
  )

  #theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())

  if (log_scale){
    vp <- vp + scale_x_log10()
  }


  return(vp)
}


vp1 <- volcano_plot_with_ids(res.tmp = res_WT_vs_Mut_LF82_sorted,
                              log_scale = FALSE,
                              gene_list = genes_of_interest)

ggsave(filename = paste0("./Plots/WT_vs_Mut_LF82_VolcanoPlot.pdf"),
       plot = vp1, width = 6, height = 6)

vp2 <- volcano_plot_with_ids(res.tmp = res_Mut_vs_WT_MG1655_sorted,
                              log_scale = FALSE,
                              gene_list =genes_of_interest
                              )

ggsave(filename = paste0("./Plots/Mut_vs_WT_MG1655_VolcanoPlot.pdf"),
       plot = vp2, width = 6, height = 6)
```

Generate volcano plots

```
## Warning: ggrepel: 7 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```r
print(vp1)
```

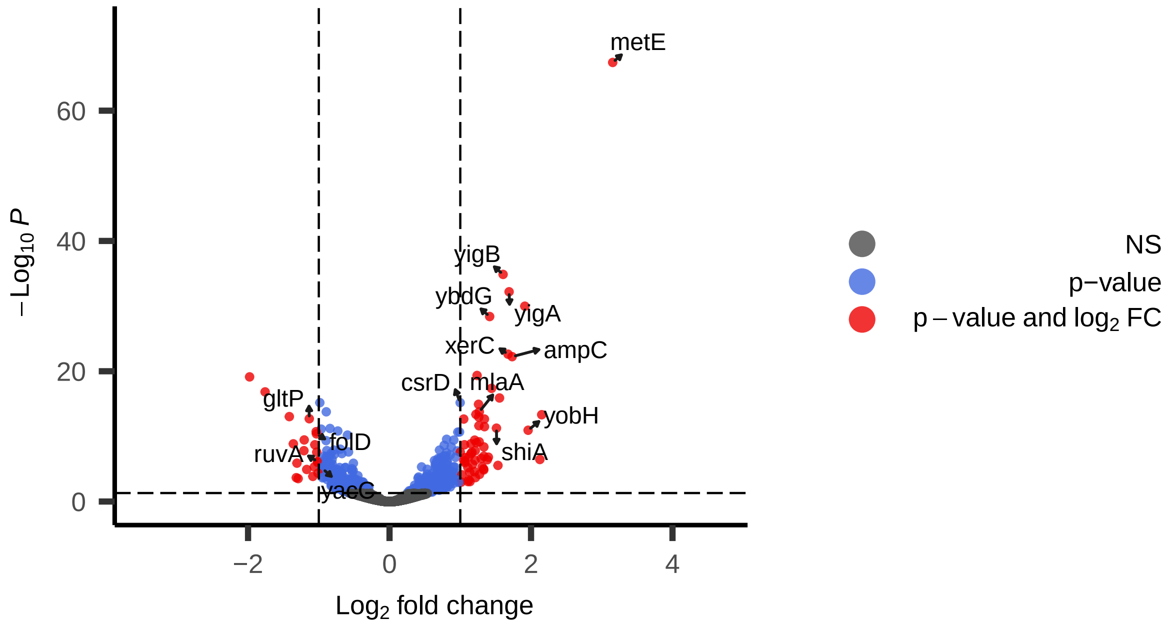
```
## Warning: ggrepel: 5 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



total = 5236 variables

```r
print(vp2)
```

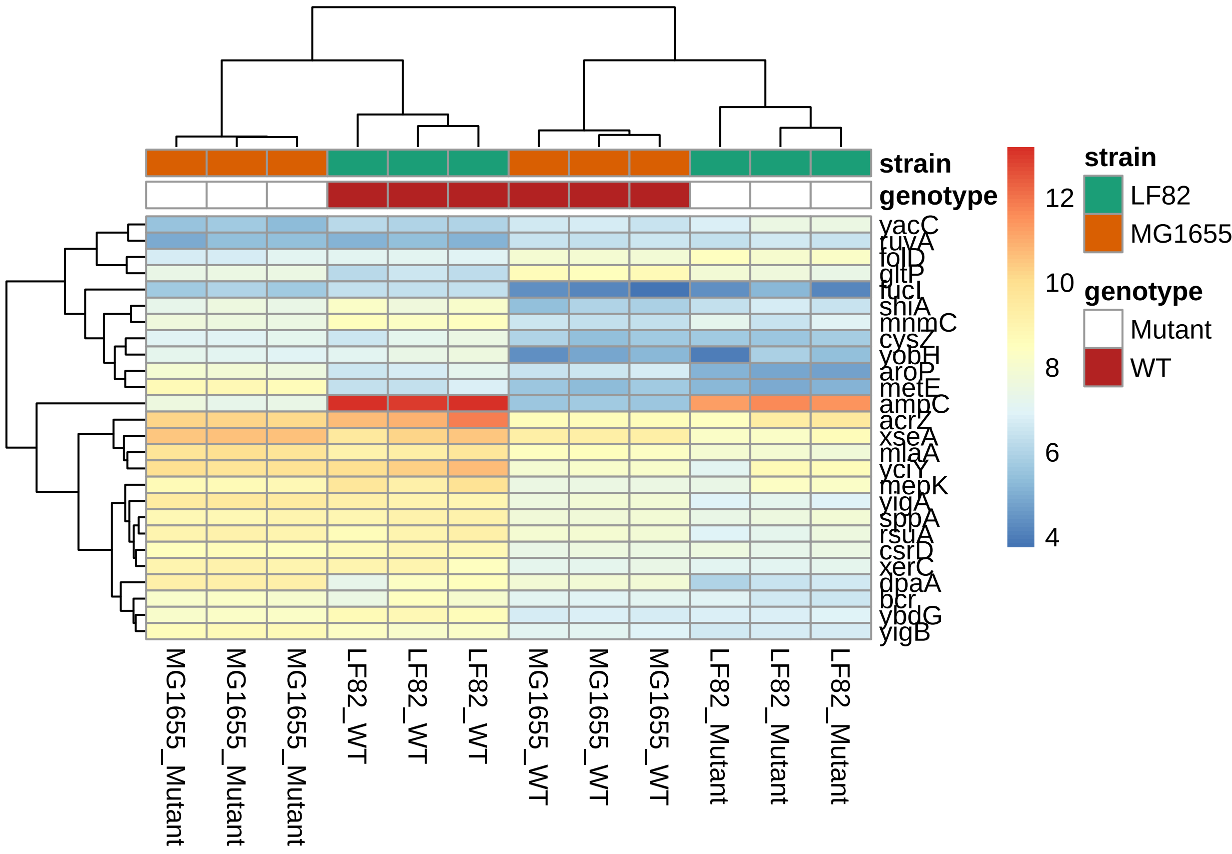
```
## Warning: ggrepel: 12 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



total = 5236 variables

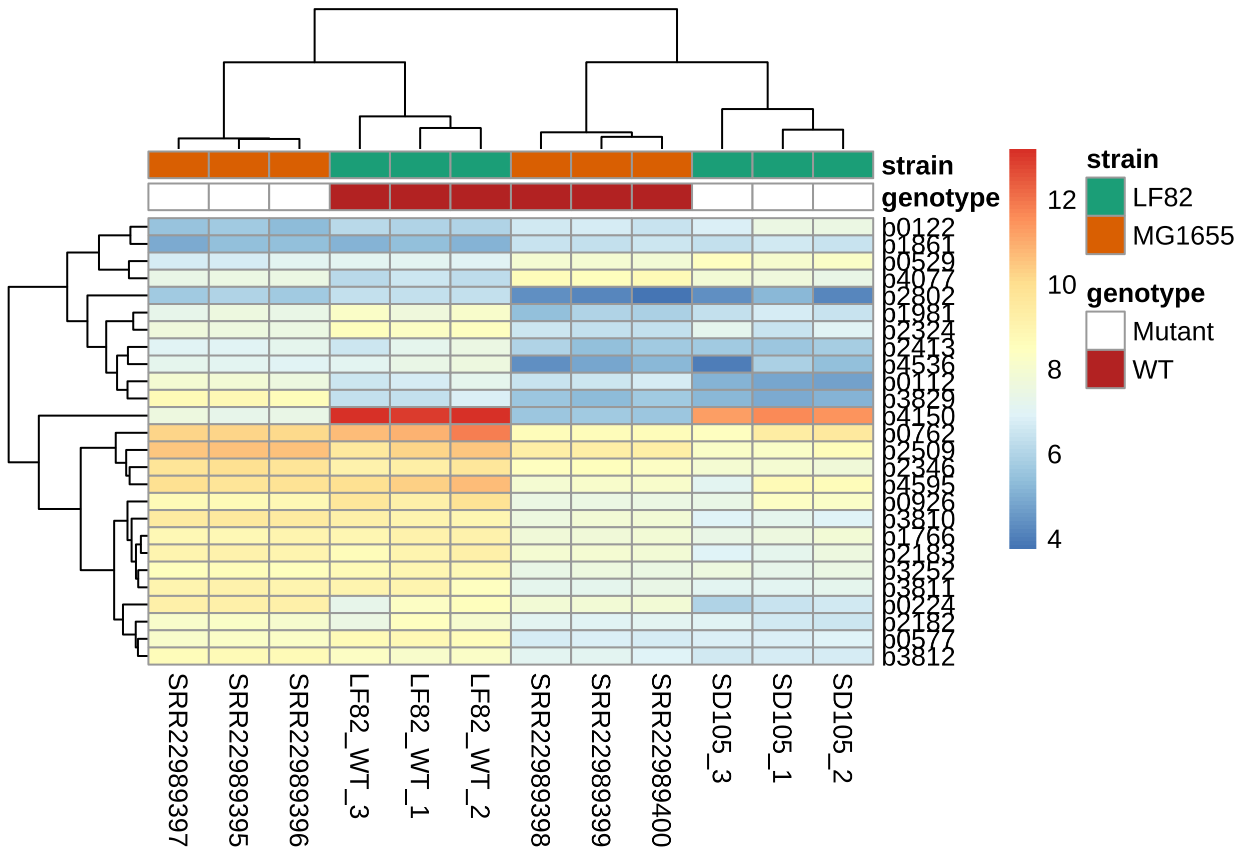
```r
#genes_of_interest.ensmbl <- rownames(head(assay(rlog.one)))
#ensembl_to_symbol.bkp <- ensembl_to_symbol
ensembl_to_symbol <- subset(ensembl_to_symbol, Ensembl_ID %in% rownames(assay(rlog.one)))
```

```
genes_of_interest.names<- subset(ensembl_to_symbol, gene_name %in% genes_of_interest)


# Specify colors
ann_colors = list(
    genotype = c(Mutant = "white", WT = "firebrick"),
    strain = c(LF82 = "#1B9E77", MG1655 = "#D95F02"))

annot_col <- as.data.frame(dplyr::select(metadata,c("genotype","strain")))

p1.heatmap <- pheatmap(assay(rlog.one)[genes_of_interest.names$Ensembl_ID,],
          cluster_rows=T,
          show_rownames=TRUE,
          cluster_cols=T,
          annotation_col = annot_col,
         labels_row = genes_of_interest.names$gene_name,
         labels_col = metadata$group,
         annotation_colors = ann_colors)
```



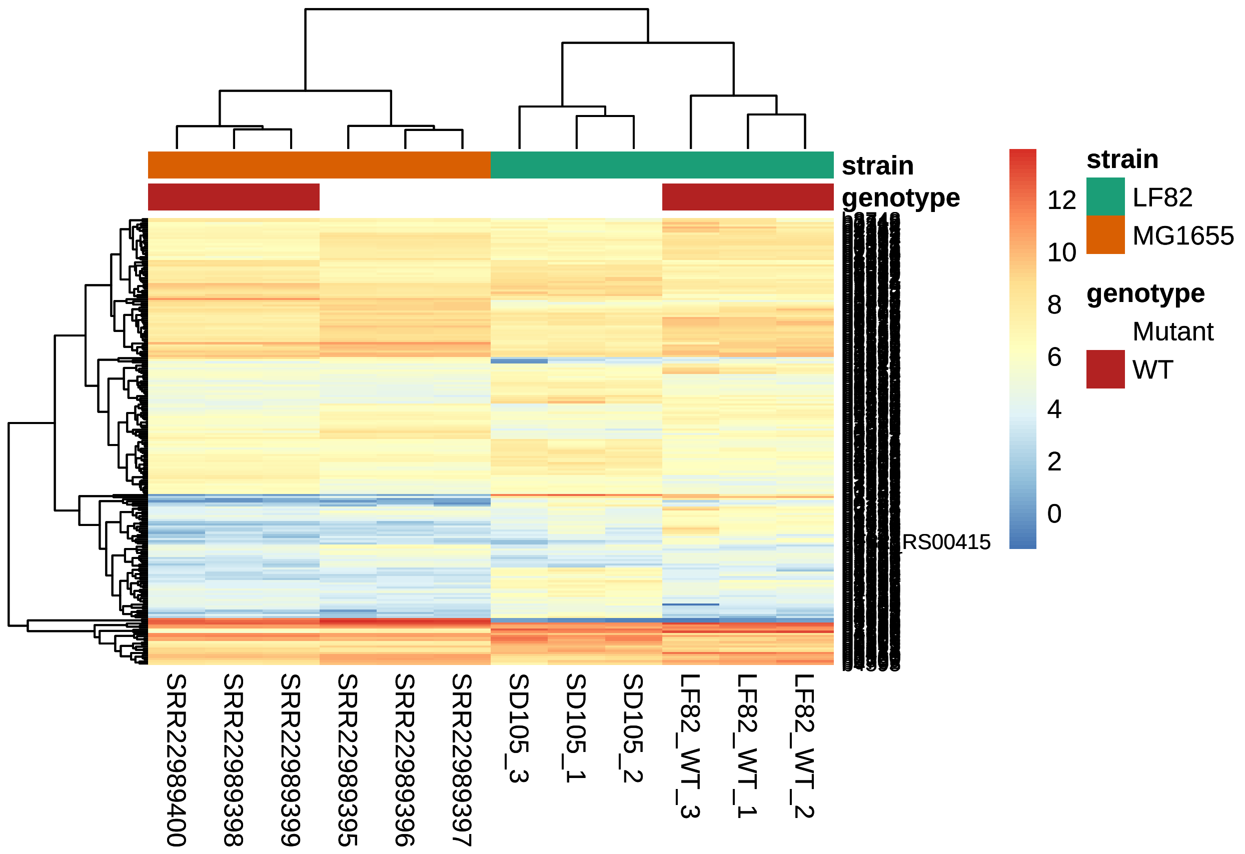
Plot heatmaps

```
p2.heatmap <- pheatmap(assay(rlog.one)[genes_of_interest.names$Ensembl_ID,],
          cluster_rows=T,
          show_rownames=TRUE,
          cluster_cols=T,
          annotation_col = annot_col,
          annotation_colors = ann_colors)
```



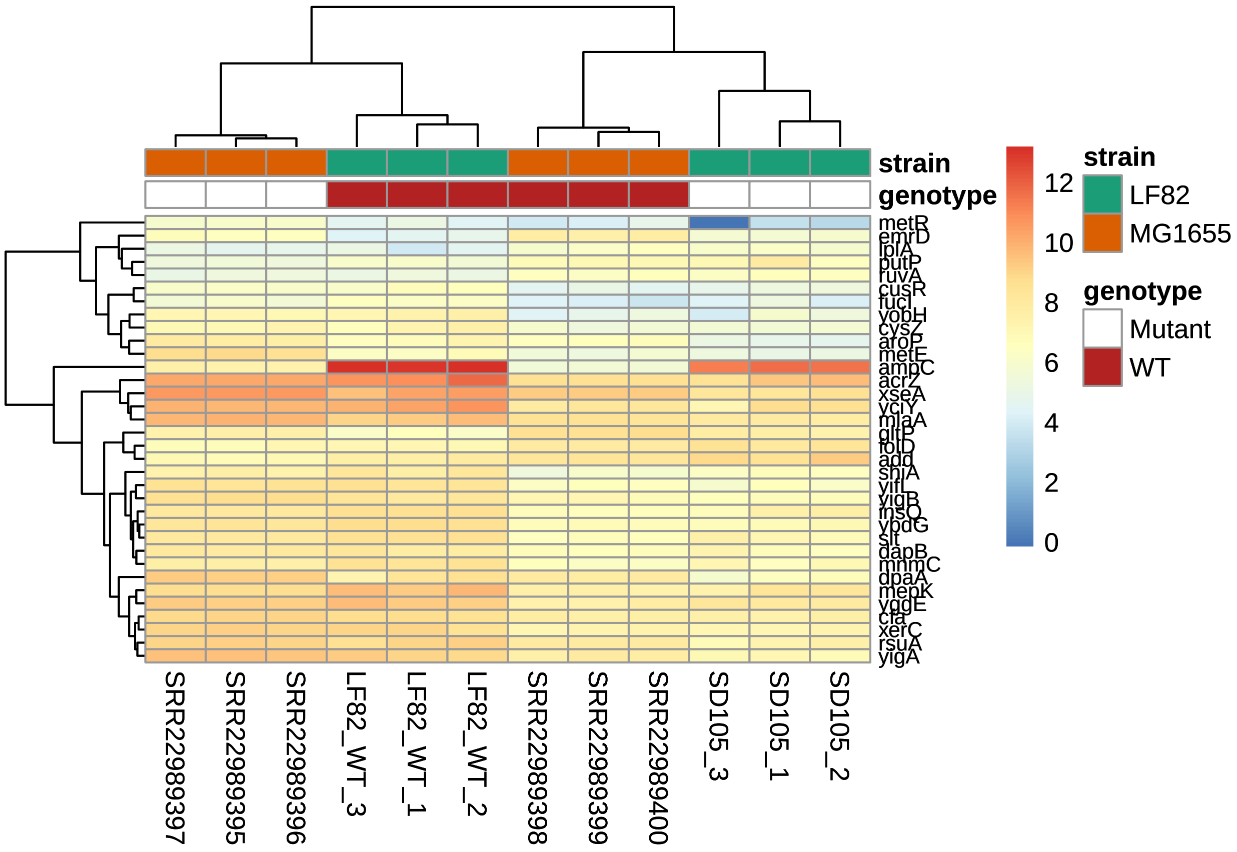
```
ggsave2(filename = "mut_vs_wt_heatmap.pdf", plot = p1.heatmap, path = "./Plots", width = 7, height = 5)
```

```
res_WT_vs_Mut_LF82.sig <- subset(res_WT_vs_Mut_LF82, padj <= 0.05 & abs(log2FoldChange) >= 1)
res_Mut_vs_WT_MG1655.sig <- subset(res_Mut_vs_WT_MG1655, padj <= 0.05 & abs(log2FoldChange) >= 1)
sign_genes <- unique(c(rownames(res_WT_vs_Mut_LF82.sig),rownames(res_Mut_vs_WT_MG1655.sig)))
rlog_res.all_strains.names <- rownames(rlog_res.all_strains@assays@data@listData[[1]])
sign_genes <- sign_genes[sign_genes %in% rlog_res.all_strains.names]

p3.heatmap <- pheatmap(assay(rlog_res.all_strains)[sign_genes,],
          cluster_rows=T,
          show_rownames=TRUE,
          cluster_cols=T,
          annotation_col = annot_col,
          annotation_colors = ann_colors, fontsize_row = 8)

print(p3.heatmap)
```



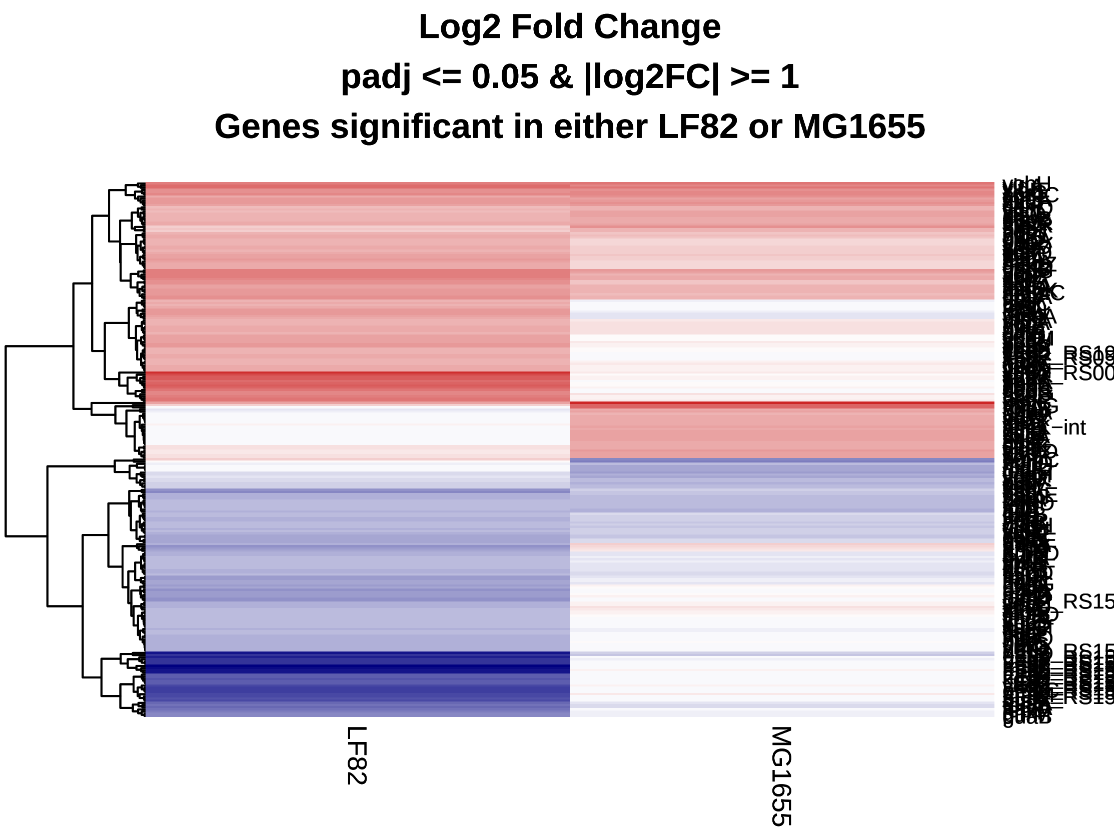
```
ggsave2(filename = "mut_vs_wt_heatmap_sign.pdf", plot = p3.heatmap, path = "./Plots", width = 7, height

sign_genes.both_comp <- c(rownames(res_WT_vs_Mut_LF82.sig),rownames(res_Mut_vs_WT_MG1655.sig))[duplicat

sign_genes.both_comp <- sign_genes.both_comp[sign_genes.both_comp %in% rlog_res.all_strains.names]

rlog_res.all_strains.both <- rlog_res.all_strains[sign_genes.both_comp,]

# gene names
ensembl_to_symbol.both_comp <- subset(ensembl_to_symbol, Ensembl_ID %in% sign_genes.both_comp)
rownames(ensembl_to_symbol.both_comp) <- ensembl_to_symbol.both_comp$Ensembl_ID

ensembl_to_symbol.both_comp <- dplyr::select(ensembl_to_symbol.both_comp, "gene_name")

sort.idx <- match(rownames(ensembl_to_symbol.both_comp),
                        rownames(assay(rlog_res.all_strains.both))
                        )

genes_names <- ensembl_to_symbol.both_comp$gene_name[order(sort.idx)]

p4.heatmap <- pheatmap(assay(rlog_res.all_strains.both),
          cluster_rows=T,
          show_rownames=TRUE,
          cluster_cols=T,
          labels_row = genes_names,
          annotation_col = annot_col,
          annotation_colors = ann_colors,
```

```r
         fontsize_row = 8)

print(p4.heatmap)
```



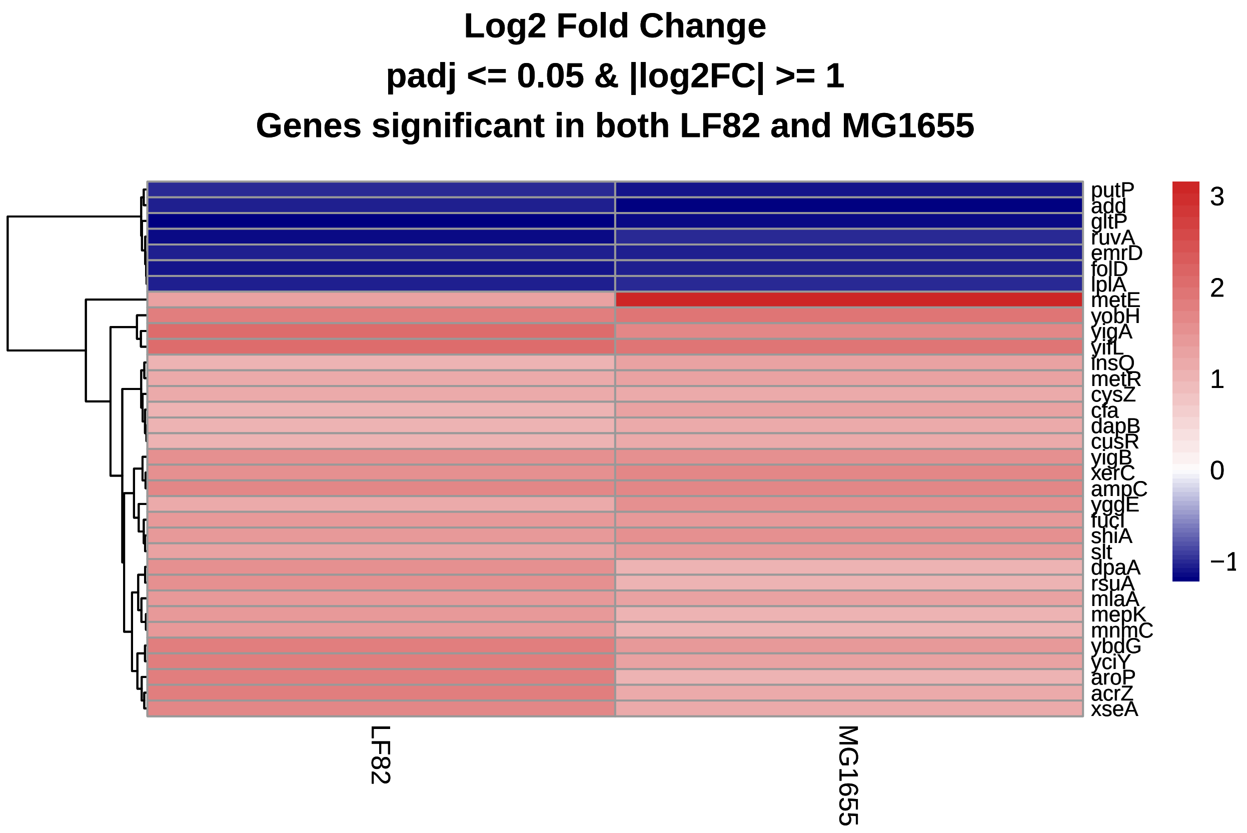
```r
ggsave2(filename = "mut_vs_wt_heatmap_both_sign.pdf", plot = p4.heatmap, path = "./Plots", width = 7, h
```

```r
### Make matrix with Log2FC values res_WT_vs_Mut_LF82_sorted res_Mut_vs_WT_MG1655_sorted
res_WT_vs_Mut_LF82_sorted$row_names <- rownames(res_WT_vs_Mut_LF82_sorted)
res_Mut_vs_WT_MG1655_sorted$row_names <- rownames(res_Mut_vs_WT_MG1655_sorted)

res_LF82_MG1655_merged <- merge(x = res_WT_vs_Mut_LF82_sorted,
                    y = res_Mut_vs_WT_MG1655_sorted,
                    by = "row_names", all = TRUE)

res_LF82_MG1655_merged.sig_any <- subset(res_LF82_MG1655_merged, (padj.x <= 0.05 & abs(log2FoldChange.x)

# Set colors for heatmap
my_matrix <- dplyr::select(res_LF82_MG1655_merged.sig_any, c("log2FoldChange.x","log2FoldChange.y"))

paletteLength <- 50
myBreaks <- c(seq(min(my_matrix), 0, length.out=ceiling(paletteLength/2) + 1),
              seq(max(my_matrix)/paletteLength, max(my_matrix), length.out=floor(paletteLength/2)))
myColor <- colorRampPalette(c("navy", "white", "firebrick3"))(paletteLength)

p5.heatmap <- pheatmap(dplyr::select(res_LF82_MG1655_merged.sig_any, c("log2FoldChange.x","log2FoldChan
          cluster_rows=TRUE,
          show_rownames=TRUE,
          cluster_cols=FALSE,
          labels_row=res_LF82_MG1655_merged.sig_any$Gene_name.y,
          labels_col=c("LF82","MG1655"),
          fontsize_row = 8,
          color = myColor,
          breaks = myBreaks,
          main = "Log2 Fold Change\npadj <= 0.05 & |log2FC| >= 1\nGenes significant in either LF82 or M

print(p5.heatmap)
```

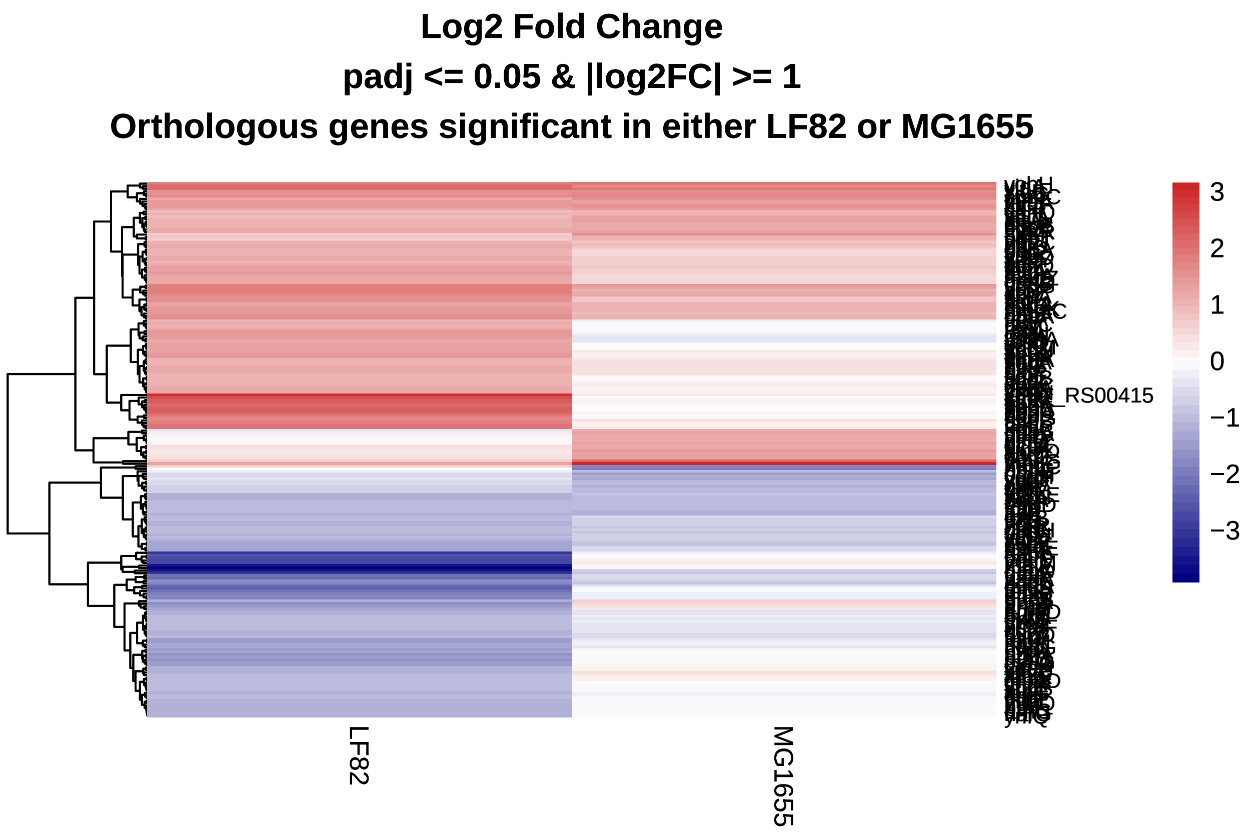


Heatmap using Log2FC

```
ggsave2(filename = "Log2FC_heatmap_sign_either.pdf", plot = p5.heatmap, path = "./Plots", width = 7, he

res_LF82_MG1655_merged.sig_all <- subset(res_LF82_MG1655_merged, (padj.x <= 0.05 & abs(log2FoldChange.x)

# Set colors for heatmap
my_matrix <- dplyr::select(res_LF82_MG1655_merged.sig_all, c("log2FoldChange.x","log2FoldChange.y"))

paletteLength <- 50
myBreaks <- c(seq(min(my_matrix), 0, length.out=ceiling(paletteLength/2) + 1),
              seq(max(my_matrix)/paletteLength, max(my_matrix), length.out=floor(paletteLength/2)))
myColor <- colorRampPalette(c("navy", "white", "firebrick3"))(paletteLength)

p6.heatmap <- pheatmap(my_matrix,
          cluster_rows=TRUE,
          show_rownames=TRUE,
          cluster_cols=FALSE,
          labels_row=res_LF82_MG1655_merged.sig_all$Gene_name.y,
          labels_col=c("LF82","MG1655"),
          fontsize_row = 8,
          color = myColor,
          breaks = myBreaks,
          main = "Log2 Fold Change\npadj <= 0.05 & |log2FC| >= 1\nGenes significant in both LF82 and MG

print(p6.heatmap)
```



```
ggsave2(filename = "Log2FC_heatmap_sign_both.pdf", plot = p6.heatmap, path = "./Plots", width = 4, heig

# Filter out strain-specific genes
keep_genes_in_both <- c(! is.na(res_LF82_MG1655_merged$LF82_IDs) & ! is.na(res_LF82_MG1655_merged$MG165
res_LF82_MG1655_merged_in_both <- res_LF82_MG1655_merged[keep_genes_in_both, ]

res_LF82_MG1655_merged_in_both.sig_any <- subset(res_LF82_MG1655_merged_in_both, (padj.x <= 0.05 & abs(

# Set colors for heatmap
my_matrix <- dplyr::select(res_LF82_MG1655_merged_in_both.sig_any, c("log2FoldChange.x","log2FoldChange

paletteLength <- 50
myBreaks <- c(seq(min(my_matrix), 0, length.out=ceiling(paletteLength/2) + 1),
              seq(max(my_matrix)/paletteLength, max(my_matrix), length.out=floor(paletteLength/2)))
myColor <- colorRampPalette(c("navy", "white", "firebrick3"))(paletteLength)

p7.heatmap <- pheatmap(dplyr::select(res_LF82_MG1655_merged_in_both.sig_any, c("log2FoldChange.x","log2
          cluster_rows=TRUE,
          show_rownames=TRUE,
          cluster_cols=FALSE,
          labels_row=res_LF82_MG1655_merged_in_both.sig_any$Gene_name.y,
          labels_col=c("LF82","MG1655"),
          fontsize_row = 8,
          color = myColor,
          breaks = myBreaks,
          main = "Log2 Fold Change\npadj <= 0.05 & |log2FC| >= 1\nOrthologous genes significant in eith

print(p7.heatmap)
```



```
ggsave2(filename = "Log2FC_heatmap_orthologous_sign_either.pdf", plot = p7.heatmap, path = "./Plots", wi
```

```
# Generate first plot with all genes
my_title <- "Log2(Fold Change) of differential gene expression\nLF82-WT/LF82-Mut and MG1655-Mut/MG1655-

df <- res_LF82_MG1655_merged

df$pvalue.x[is.na(df$pvalue.x)] <- 1
df$pvalue.y[is.na(df$pvalue.y)] <- 1
df$min_pval <- apply(dplyr::select(df,c("padj.x","padj.y")), 1, function(x){ min(x)})
df <- df[order(df$min_pval, decreasing = T),]
df$significance <- as.factor(apply(dplyr::select(df,c("padj.x","padj.y")), 1, function(x){ ifelse( (x[1

myColor <- c("N.S."="gray","Sig both"="orange3","Sig LF82"="green4","Sig MG1655"="yellow3")
```

```r
p3 <- ggplot(df, aes(x=log2FoldChange.x, y=log2FoldChange.y, label = Gene_name.x)) +
  labs(title = my_title) +
  xlab(bquote('LF82 '* ~Log[2]*'(Fold Change)')) +
  ylab(bquote('MG1655 '* ~Log[2]*'(Fold Change)')) +
  geom_point(aes(colour=significance), size = 0.5, alpha = 1 ) +
  ylim(min(df$log2FoldChange.y),max(df$log2FoldChange.y)) +
  xlim(min(df$log2FoldChange.x),max(df$log2FoldChange.x)) +
  geom_abline(slope = 1, intercept = 0, col = "black", size=0.5, linetype="dashed") +
  theme_minimal() + scale_colour_manual(values = myColor) +
  theme(
    legend.direction = "vertical",
    legend.position = c(.95, .10),
    legend.justification = c("right", "top"),
    legend.box.just = "right",
    legend.margin = margin(3, 3, 3, 3),
    legend.text = element_text(size = 8),
    legend.title = element_text(face = "bold" ,size = 8, vjust = 0.9))
```

dotplot using Log2FC

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
p3 <-  p3 + guides(color = guide_legend(override.aes = list(size = 5)))

# Change legend title
p3$labels$colour = paste("Significance" )

# Making genes of interest bigger
#genes_of_interesst <- ifelse(df$Log2_pICWT_pICKO > 4 | df$Log2_pICWT_pICKO < -2,  as.character(df$Gene
my_list <- genes_of_interest.names$gene_name
#genes_of_interest <- ifelse(df$Gene_name.y %in% my_list,  as.character(df$Gene_name.y),NA)


df.subset <- subset(df, Gene_name.y %in% my_list)


p3 <- p3 + geom_point(data = df.subset, aes(x = log2FoldChange.x, y = log2FoldChange.y, fill=significan
ggsave2(filename = "Log2FC_dotplot_no_labels.pdf", plot = p3, path = "./Plots", width = 10, height = 10

genes_of_interest.tmp <- genes_of_interest[genes_of_interest %in% my_list]
# Adding labels to genes of interest
p3 <- p3 + geom_text_repel(data = df.subset, aes(label = genes_of_interest.tmp),
                  colour = "black",
                  label.size = 1,
                  box.padding   = 0.4,
                  point.padding = 0.3,
                  segment.color = 'black',
                  na.rm = TRUE,
                  size = 5,
                  min.segment.length = 0.02,
                  direction = "both",
                  segment.curvature = -0.1,
                  segment.ncp = 3,
                  segment.angle = 20,
                  max.iter = 1e5)
```

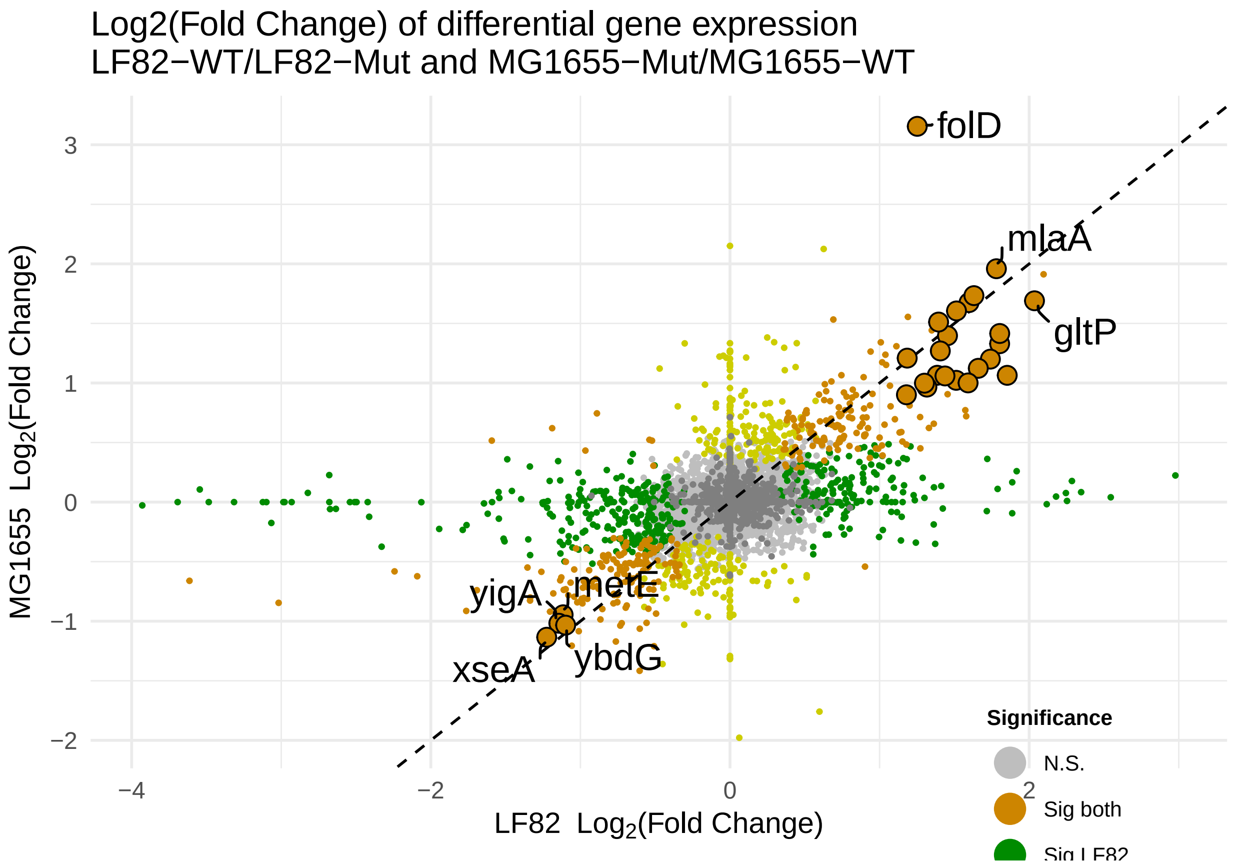
```
## Warning in geom_text_repel(data = df.subset, aes(label =
## genes_of_interest.tmp), : Ignoring unknown parameters: `label.size`
```

```
ggsave2(filename = "Log2FC_dotplot.pdf", plot = p3, path = "./Plots", width = 10, height = 10)
```

```
## Warning: ggrepel: 7 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
print(p3)
```

```
## Warning: ggrepel: 19 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

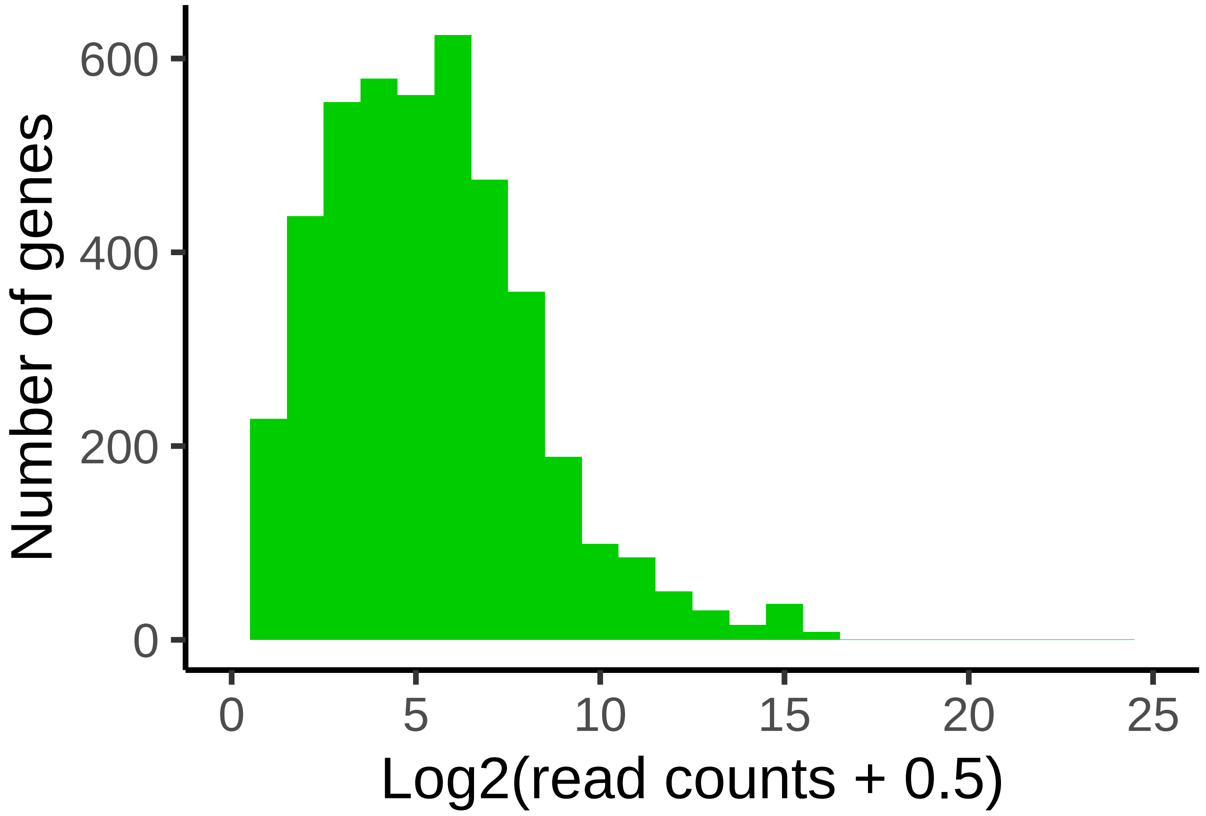


Distributions

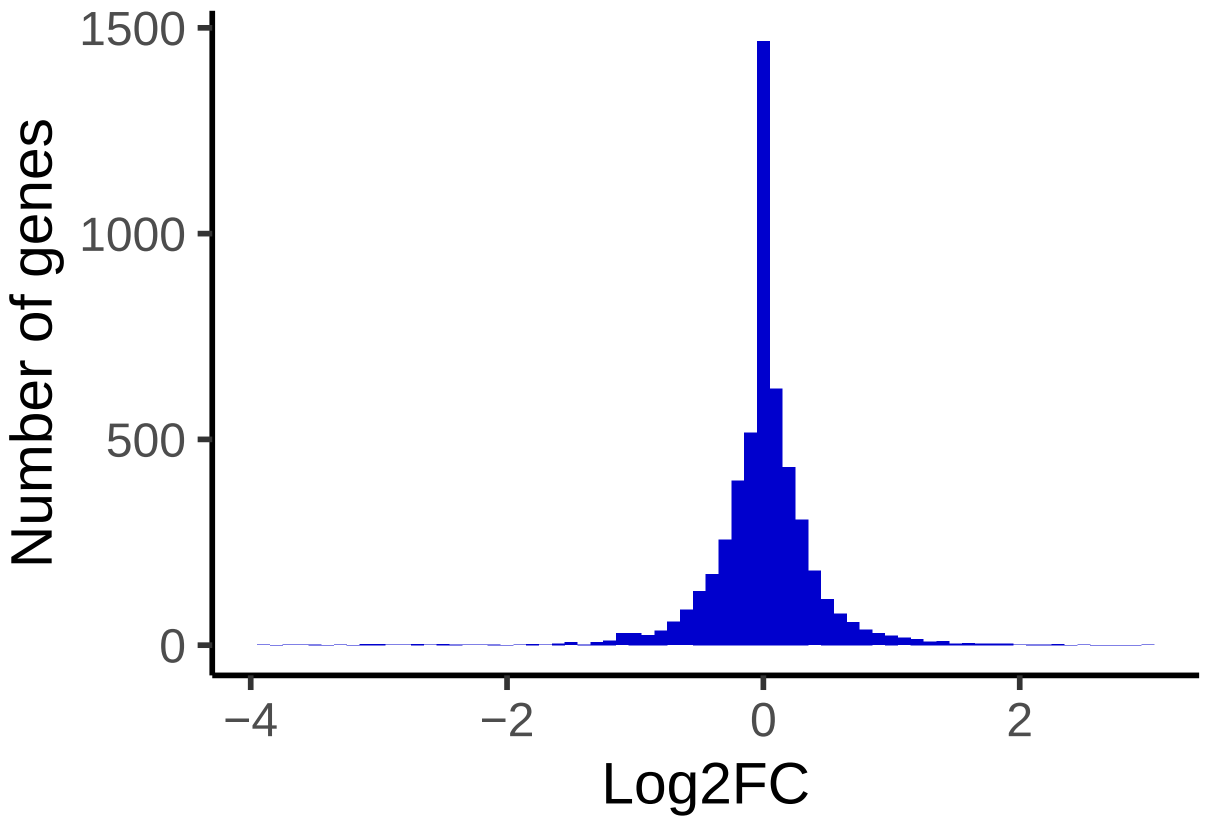
```
p <- ggplot(all, aes(x = log2(all$LF82_WT_1+0.5))) + geom_histogram(binwidth = 1, fill = "green3") + xl
p + theme_classic(base_size = 22)
```

```
## Warning: Removed 904 rows containing non-finite values (`stat_bin()`).
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



```
p1 <- ggplot(as.data.frame(res_WT_vs_Mut_LF82), aes(x = log2FoldChange)) + geom_histogram(binwidth = 0.
p1 + theme_classic(base_size = 22)
```



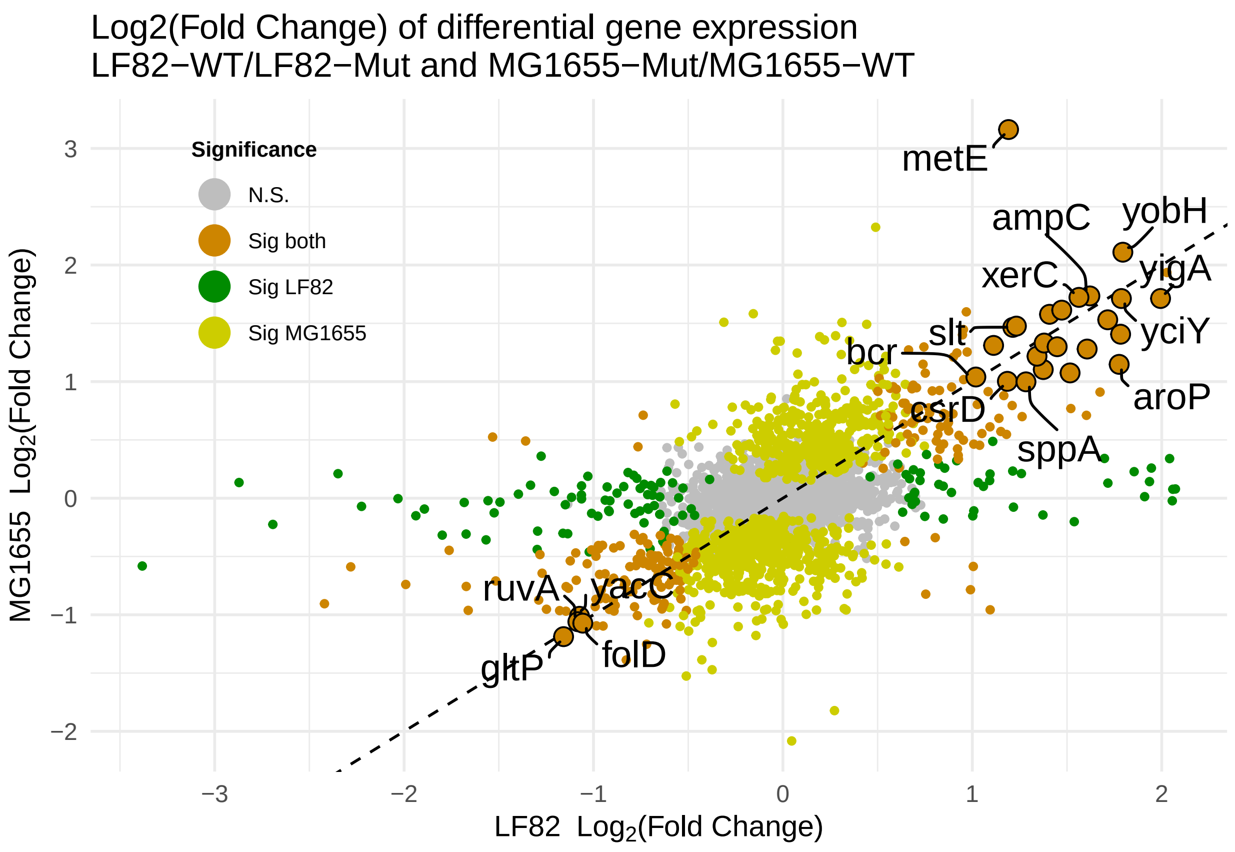
```r
log2fc_lf82_mg <- read.table(file = "./data/Log2FC_LF82_MG.txt", header = T, sep = "\t", row.names = 1)


# Generate first plot with all genes
my_title <- "Log2(Fold Change) of differential gene expression\nLF82-WT/LF82-Mut and MG1655-Mut/MG1655-

df <- log2fc_lf82_mg

# Invert Log2FC for LF82 to reflect WT/Mut rather than the original data Mut/WT
df$Log2FC_LF82 <- df$Log2FC_LF82 * -1

# remove rows where Log2FC LF82/MG == NA
df <- df[!c(is.na(df$Log2FC_MG) | is.na(df$Log2FC_LF82)), ]

df$padj_LF82[is.na(df$padj_LF82)] <- 1
df$padj_MG[is.na(df$padj_MG)] <- 1
df$min_pval <- apply(dplyr::select(df,c("padj_LF82","padj_MG")), 1, function(x){ min(x)})
df <- df[order(df$min_pval, decreasing = T),]
df$significance <- as.factor(apply(dplyr::select(df,c("padj_LF82","padj_MG")), 1, function(x){ ifelse(

myColor <- c("N.S."="gray","Sig both"="orange3","Sig LF82"="green4","Sig MG1655"="yellow3")

p4 <- ggplot(df, aes(x=Log2FC_LF82, y=Log2FC_MG, label = gene_name_MG)) +
  labs(title = my_title) +
  xlab(bquote('LF82 '* ~Log[2]*'(Fold Change)')) +
  ylab(bquote('MG1655 '* ~Log[2]*'(Fold Change)')) +
  geom_point(aes(colour=significance), size = 1, alpha = 1 ) +
  ylim(-3.5,3.5) + #ylim(min(df$Log2FC_MG),max(df$Log2FC_MG)) +
  xlim(-3.5,3.5) + #xlim(min(df$Log2FC_LF82),max(df$Log2FC_LF82)) +
  geom_abline(slope = 1, intercept = 0, col = "black", size=0.5, linetype="dashed") +
  scale_x_continuous(breaks =c(-3,-2,-1,0,1,2,3)) +
  scale_y_continuous(breaks =c(-3,-2,-1,0,1,2,3)) +
  theme_minimal() +
  scale_colour_manual(values = myColor) +
  theme(
    legend.direction = "vertical",
    legend.position = c(.25, .95),
    legend.justification = c("right", "top"),
    legend.box.just = "right",
    legend.margin = margin(3, 3, 3, 3),
    legend.text = element_text(size = 8),
    legend.title = element_text(face = "bold" ,size = 8, vjust = 0.9))
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for y is already present.
## Adding another scale for y, which will replace the existing scale.
```

```r
p4 <-  p4 + guides(color = guide_legend(override.aes = list(size = 5)))

# Change legend title
p4$labels$colour = paste("Significance" )

#genes_of_interest
```

```r
my_list <- genes_of_interest # genes_of_interest.names$gene_name

df.subset <- subset(df, gene_name_LF82 %in% my_list)


p4 <- p4 + geom_point(data = df.subset,
                      aes(x = Log2FC_LF82, y = Log2FC_MG, fill=significance),
                      size = 3,
                      pch=21,
                      colour="black",
                      show.legend = FALSE) +
            scale_fill_manual(values = myColor)

ggsave2(filename = "Log2FC_dotplot_from_orig_DE_no_labels.pdf", plot = p4, path = "./Plots", width = 10

# Adding labels to genes of interest
p4 <- p4 + geom_text_repel(data = df.subset,
                            aes(label = gene_name_MG),
                            colour = "black",
                            box.padding   = 0.4,
                            point.padding = 0.3,
                            segment.color = 'black',
                            na.rm = TRUE,
                            size = 5,
                            min.segment.length = 0.02,
                            direction = "both",
                            segment.curvature = -0.1,
                            segment.ncp = 3,
                            segment.angle = 20,
                            max.iter = 1e6,
                            max.overlaps = 20,
                           max.time = 10)

ggsave2(filename = "Log2FC_dotplot_from_orig_DE.pdf", plot = p4, path = "./Plots", width = 10, height =
p4
```

```
## Warning: ggrepel: 12 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```r
print(sessionInfo())
```

```
## R version 4.3.1 (2023-06-16)
## Platform: x86_64-apple-darwin20 (64-bit)
## Running under: macOS Ventura 13.6.3
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRlapack.dylib;  LAPACK
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] grid      stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] ggraph_2.1.0            broom_1.0.5
##  [3] ggupset_0.3.0           enrichplot_1.22.0
##  [5] cowplot_1.1.1           msigdbr_7.5.1
##  [7] RColorBrewer_1.1-3      viridis_0.6.4
##  [9] viridisLite_0.4.2       ggsci_3.0.0
## [11] GOSemSim_2.28.0         clusterProfiler_4.10.0
## [13] VennDiagram_1.7.3           futile.logger_1.4.3
## [15] pcaExplorer_2.28.0          lubridate_1.9.3
## [17] forcats_1.0.0               dplyr_1.1.4
## [19] purrr_1.0.2                 readr_2.1.4
## [21] tidyr_1.3.0                 tibble_3.2.1
## [23] tidyverse_2.0.0             biomaRt_2.58.0
## [25] stringr_1.5.1               DESeq2_1.42.0
## [27] SummarizedExperiment_1.32.0 MatrixGenerics_1.14.0
## [29] matrixStats_1.1.0           GenomicRanges_1.54.1
## [31] GenomeInfoDb_1.38.1         ggpubr_0.6.0
## [33] EnhancedVolcano_1.20.0      ggrepel_0.9.4
## [35] ggplot2_3.4.4               pheatmap_1.0.12
## [37] AnnotationDbi_1.64.1        IRanges_2.36.0
## [39] S4Vectors_0.40.2            Biobase_2.62.0
## [41] BiocGenerics_0.48.1
##
## loaded via a namespace (and not attached):
##   [1] fs_1.6.3                bitops_1.0-7            HDO.db_0.99.1
##   [4] httr_1.4.7              webshot_0.5.5           doParallel_1.0.17
##   [7] Rgraphviz_2.46.0        tools_4.3.1             backports_1.4.1
##  [10] utf8_1.2.4              R6_2.5.1                DT_0.30
##  [13] lazyeval_0.2.2          withr_2.5.2             prettyunits_1.2.0
##  [16] gridExtra_2.3           textshaping_0.3.7       cli_3.6.1
##  [19] pacman_0.5.1            formatR_1.14            TSP_1.2-4
##  [22] scatterpie_0.2.1        labeling_0.4.3          topGO_2.54.0
##  [25] SQUAREM_2021.1          genefilter_1.84.0       mixsqp_0.3-48
##  [28] systemfonts_1.0.5       yulab.utils_0.1.0       gson_0.1.0
##  [31] DOSE_3.28.1             AnnotationForge_1.44.0  invgamma_1.1
##  [34] limma_3.58.1            rstudioapi_0.15.0       RSQLite_2.3.3
##  [37] gridGraphics_0.5-1      generics_0.1.3          GOstats_2.68.0
##  [40] crosstalk_1.2.1         car_3.1-2               dendextend_1.17.1
##  [43] GO.db_3.18.0            Matrix_1.6-4            fansi_1.0.5
##  [46] abind_1.4-5             lifecycle_1.0.4         yaml_2.3.7
##  [49] carData_3.0-5           qvalue_2.34.0           SparseArray_1.2.2
##  [52] BiocFileCache_2.10.1    blob_1.2.4              promises_1.2.1
##  [55] crayon_1.5.2            shinydashboard_0.7.2    lattice_0.21-8
##  [58] annotate_1.80.0         KEGGREST_1.42.0         pillar_1.9.0
##  [61] knitr_1.45              fgsea_1.28.0            codetools_0.2-19
##  [64] fastmatch_1.1-4         glue_1.6.2              ggfun_0.1.3
##  [67] data.table_1.14.8       treeio_1.27.0.001       vctrs_0.6.5
##  [70] png_0.1-8               gtable_0.3.4            assertthat_0.2.1
##  [73] cachem_1.0.8            xfun_0.41               S4Arrays_1.2.0
##  [76] mime_0.12               tidygraph_1.2.3         survival_3.5-5
##  [79] seriation_1.5.3         iterators_1.0.14        statmod_1.5.0
##  [82] ellipsis_0.3.2          nlme_3.1-162            Category_2.68.0
##  [85] ggtree_3.10.0           bit64_4.0.5             threejs_0.3.3
##  [88] progress_1.2.3          filelock_1.0.2          irlba_2.3.5.1
##  [91] colorspace_2.1-0        DBI_1.1.3               tidyselect_1.2.0
##  [94] bit_4.0.5               compiler_4.3.1          curl_5.1.0
##  [97] graph_1.80.0            SparseM_1.81            xml2_1.3.6
## [100] DelayedArray_0.28.0     plotly_4.10.3           shadowtext_0.1.2
## [103] scales_1.3.0            RBGL_1.78.0             NMF_0.26
## [106] rappdirs_0.3.3          digest_0.6.33           shinyBS_0.61.1
## [109] rmarkdown_2.25          ca_0.71.1               XVector_0.42.0
## [112] htmltools_0.5.7         pkgconfig_2.0.3         base64enc_0.1-3
## [115] highr_0.10              dbplyr_2.4.0            fastmap_1.1.1
## [118] rlang_1.1.2             htmlwidgets_1.6.4       shiny_1.8.0
## [121] farver_2.1.1            jsonlite_1.8.8          BiocParallel_1.36.0
## [124] RCurl_1.98-1.13         magrittr_2.0.3          ggplotify_0.1.2
## [127] GenomeInfoDbData_1.2.11 patchwork_1.1.3         munsell_0.5.0
## [130] Rcpp_1.0.11             babelgene_22.9          ape_5.7-1
## [133] stringi_1.8.2           zlibbioc_1.48.0         MASS_7.3-60
## [136] plyr_1.8.9              parallel_4.3.1          Biostrings_2.70.1
## [139] graphlayouts_1.0.2      splines_4.3.1           hms_1.1.3
## [142] locfit_1.5-9.8          igraph_1.5.1            ggsignif_0.6.4
## [145] rngtools_1.5.2          reshape2_1.4.4          futile.options_1.0.1
## [148] XML_3.99-0.16           evaluate_0.23           lambda.r_1.2.4
## [151] BiocManager_1.30.22     tzdb_0.4.0              foreach_1.5.2
## [154] tweenr_2.0.2            httpuv_1.6.13           polyclip_1.10-6
## [157] heatmaply_1.5.0         ashr_2.2-63             gridBase_0.4-7
## [160] ggforce_0.4.1           xtable_1.8-4            tidytree_0.4.5
## [163] rstatix_0.7.2           later_1.3.2             ragg_1.2.6
## [166] truncnorm_1.0-9         aplot_0.2.2             memoise_2.0.1
## [169] registry_0.5-1          cluster_2.1.4           timechange_0.2.0
## [172] shinyAce_0.4.2          GSEABase_1.64.0
```