Differential Gene Expression analysis using Deseq2

NGS workshop

Date: 17th May 2023...2nd hour

Mirvat Surakhy

In this script we will be doing DGE analysis using Deseq2 package and view the output using some plots

The codes are from the following tutorials
https://bioconductor.org/packages/devel/bioc/vignettes/EnhancedVolcano/inst/doc/EnhancedVolcano.html
https://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html

https://yulab-smu.github.io/clusterProfiler-book/index.html

Load the required Libraries

```
library(dplyr)
library(ggplot2)
library(DESeq2)
library(RColorBrewer)
library(org.Hs.eg.db)
library(EnhancedVolcano)
```

Read the data files

Read the count data generated in the previous code and load the sample information (metadata). This is extracted from the SRA table when we downloaded the data

```
counts <- read.table(file = "counts/Count_fromtximport_Salmon.txt", header= TRUE, check.names = F)
metadata <- read.csv(file= "/mnt/beegfs/workshop/DGE_results_codes/SampleInfo_ngs.csv", row.names = 1)

#set the levels so the treatment is being compared against the control. The reference is the Control group.

metadata$condition <- factor(metadata$condition,
                             levels = c("Control", "Treatment"))

#check if the sample name is matching the metadata
all(rownames(metadata)%in% colnames(counts))

## [1] TRUE
```

```
## [1] TRUE
all(rownames(metadata)== colnames(counts))

## [1] FALSE

## [1] TRUE
#If the order of rows and columns is  not the same, try do the following
counts<- counts[, row.names(metadata)]
```

Differential Gene Expression

For DGE:

1. Design the matrix, and remove genes that have counts less than 5 reads as this will have an effect on the number of significant results after multiple hypothesis adjustments.
2. Create your DESeq2Dataset object and perform the DGE.
3. The output will be a dds object that contains all the information about your data. The output of the data (the results) shows the raw fold change. To get a better estimate of the log fold change and be more confident about the log fold change, we will run lfcShrink on the dds object. This function will look at the largest fold changes that are not due to low counts and uses these to inform a prior distribution. The large fold changes from genes with lots of statistical information are not shrunk, while the imprecise fold changes are shrunk. This will give you better visualization and ranking of genes.
4. The method that we used for shrinkage estimation is apeglm.

   Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for sequence count data: removing the noise and preserving large differences. Bioinformatics. 10.1093/bioinformatics/bty895

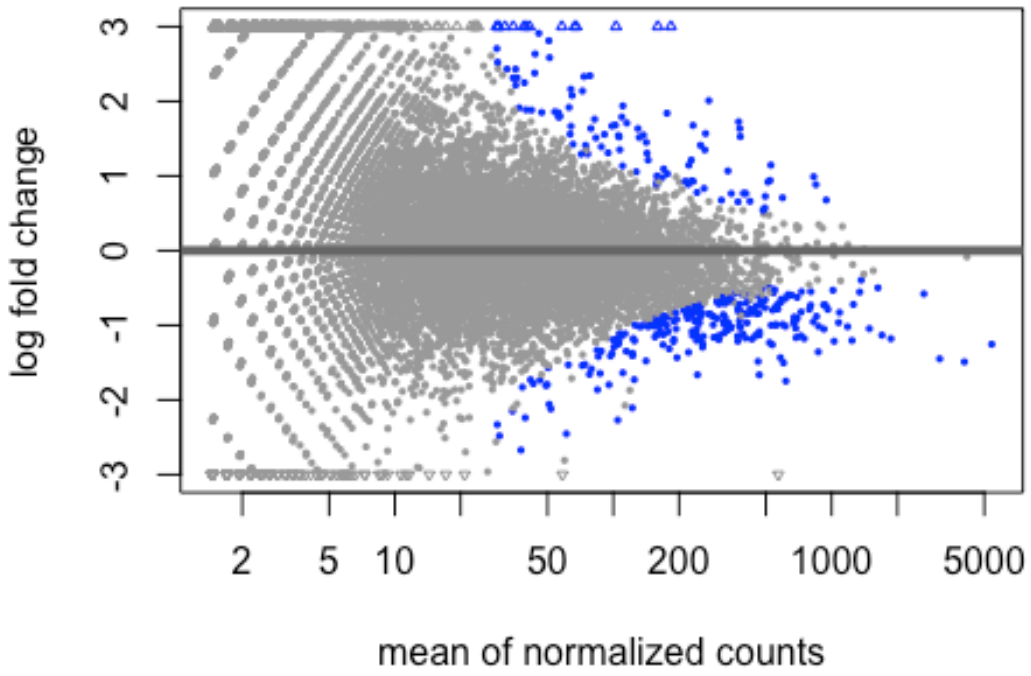
```
######## design the matrix
design <- as.formula(~condition)
model<- model.matrix(design, data= metadata)
keep <- rowSums(counts)>5
countdata<- counts[keep,]
countdata<- as.matrix(countdata)

#### Create DESeq2Dataset object
dds.raw<- DESeqDataSetFromMatrix(countData = countdata,
                                 colData = metadata,
                                design = design)

#Perform the differential gene expression
dds <- DESeq(dds.raw)

## estimating size factors
```

```
## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

### a. Obtain the results from DESeq object
res_05<- results(dds,alpha= 0.05) #alpha indicates the value of padj. By defa
ult the argument alpha is set to 0.1
######
#lists the coefficients and use it with lfcShrink()
resultsNames(dds)

## [1] "Intercept"                      "condition_Treatment_vs_Control"

#[1] "Intercept"                     "condition_Treatment_vs_Control"
#Use the out put in the lfcShrink
resLFC_05 <- lfcShrink(dds, coef="condition_Treatment_vs_Control", type="apeg
lm", res= res_05)

## using 'apeglm' for LFC shrinkage. If used in published research, please ci
te:
##     Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distribut
ions for
##     sequence count data: removing the noise and preserving large differenc
es.
##     Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

Plot and view the Results

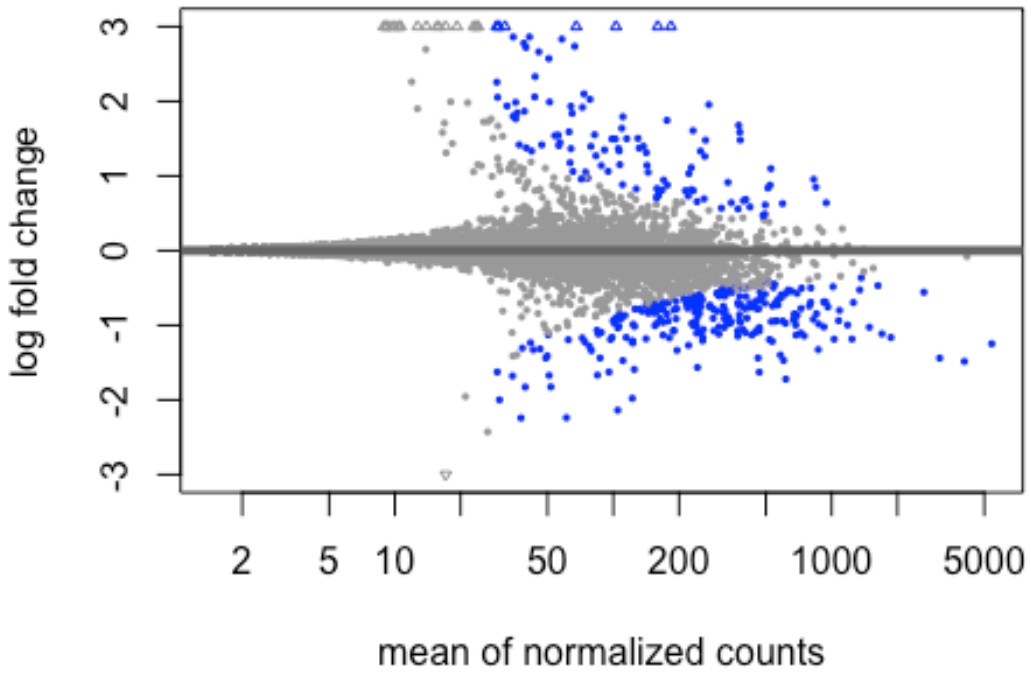
MA plot is a scatter plot showing log2 fold changes (on the y-axis) versus the mean of normalized counts (on the x-axis). Points will be colored blue if the adjusted p-value is less than 0.1. Points which fall out of the window are plotted as open triangles pointing either up or down.

```
plotMA(res_05, ylim=c(-3,3))
```



Plot MA plot after log fold shrinkage. Can you see the effect of lfcShrink

```
plotMA(resLFC_05, ylim=c(-3,3))
```



Volcano Plot

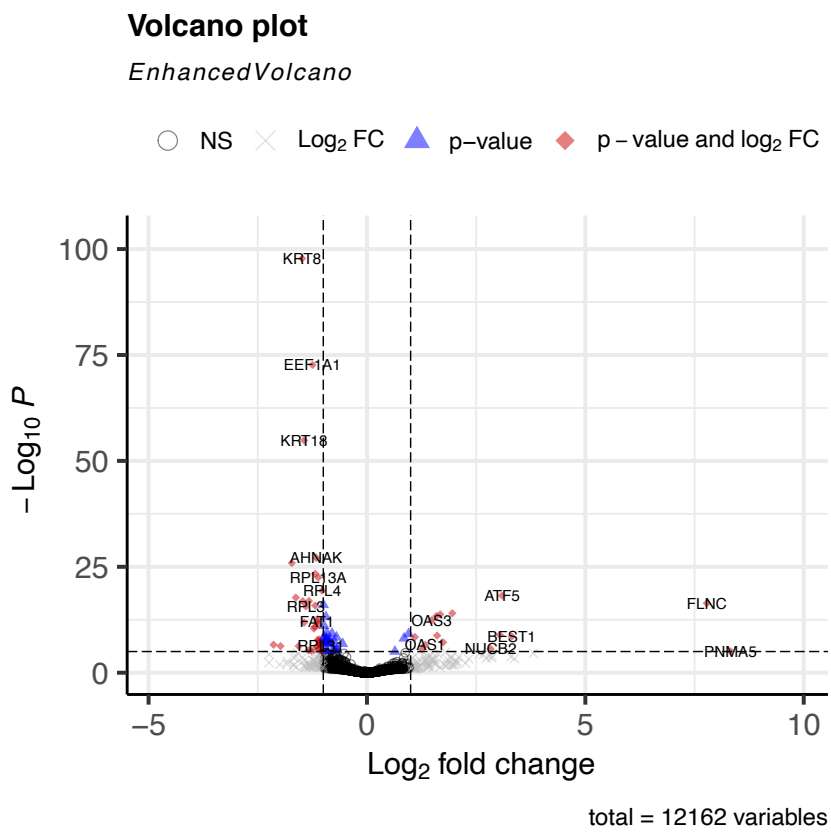
A volcano plot is a type of scatter plot that represents a differential expression of genes. The fold change will be on the x-axis and the p-value on the y-axis. Genes that are to the left of the graph are upregulated in the control group and those at the right of the graph are upregulated in the treatment group.

#Prepare the data

We will be using the library (org.Hs.eg.db) to map gene ids, e.g. symbols and entrez ids.

```
DEG<- as.data.frame(resLFC_05) # assign the results to data frame degenes

# assign symbol(common names) and ENTREZ id according to ENSEMBL id
DEG$symbol<- mapIds(org.Hs.eg.db,
                            keys= rownames(DEG),
                            column = "SYMBOL",
                            keytype = "ENSEMBL",
                            multiVals = "first")

## 'select()' returned 1:many mapping between keys and columns
```

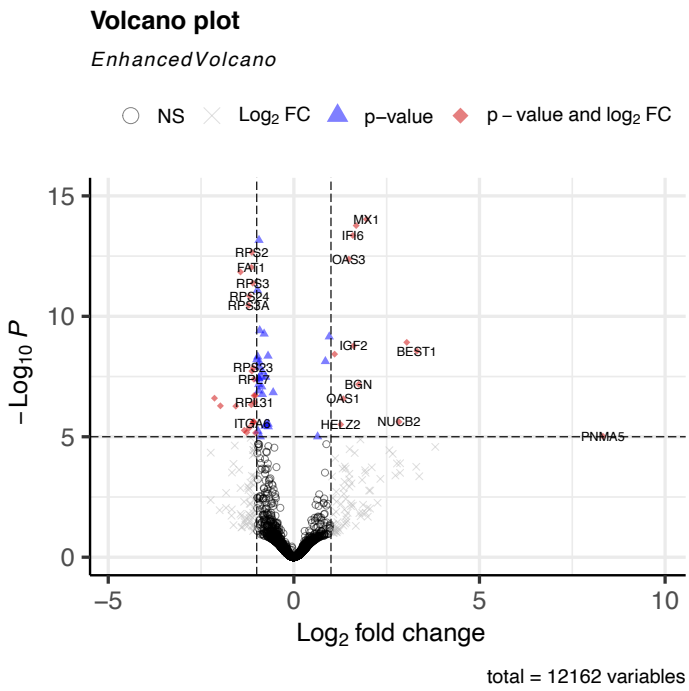
```
DEG$entrez<- mapIds(org.Hs.eg.db,
                          keys= rownames(DEG),
                          column = "ENTREZID",
                          keytype = "ENSEMBL",
                          mutiVals= "first")

## 'select()' returned 1:many mapping between keys and columns

# remove genes that don't have a common name and those with duplicated gene n
ame
DEG_symbol<- DEG[is.na(DEG$symbol)== FALSE,]
dim(DEG_symbol)

## [1] 12557     7

DEG_symbol<- DEG_symbol[!duplicated(DEG_symbol$symbol),]

DEG05_symbol<- subset(DEG_symbol, padj< 0.05  &abs(log2FoldChange)>1)

write.csv(DEG_symbol, "counts/DEGs_5uMaza_treatment_All.csv")

write.csv(DEG05_symbol, "counts/DEGs_5uMaza_treatment_significant.csv")

DEG_symbol$ENSEMBL.ID=row.names(DEG_symbol)
row.names(DEG_symbol) <- DEG_symbol$symbol

EnhancedVolcano(DEG_symbol,
    lab = rownames(DEG_symbol),
    x = 'log2FoldChange',
    y = 'padj',
labSize = 3.0,
  shape = c(1, 4, 17, 18), #if we need to add shape
 col=c('black','gray','blue', 'red3'))
```



Extra exercise:

You can change the limit of the padj value for better visualization. This can be adjusted using `ylim= c(0,15)`

```
EnhancedVolcano(DEG_symbol,
    lab = rownames(DEG_symbol),
    x = 'log2FoldChange',
    y = 'padj',
labSize = 3.0,
  ylim= c(0,15),
  shape = c(1, 4, 17, 18), #if we need to add shape
 col=c('black','gray','blue', 'red3'))
```

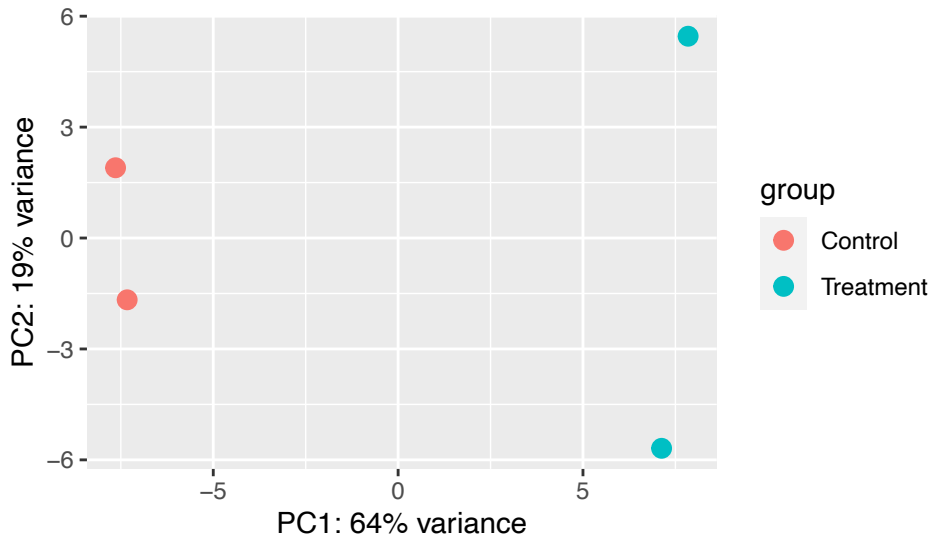


Principle component analysis (PCA)

PCA is a dimensionality reduction method that is used to reduce the dimensionality of large data sets. It transforms a large set of variables into a smaller one that still contains most of the information in the large set.

To do this we need to extract log normalised counts from the dds object. plotPCA is a build in function in the deseq2 package

```
rld <- rlog(dds, blind=TRUE)

plotPCA(rld, intgroup="condition")
```




```
sessionInfo()

R version 4.2.1 (2022-06-23)

Platform: x86_64-pc-linux-gnu (64-bit)

Running under: Ubuntu 20.04.6 LTS

Matrix products: default

BLAS:   /mnt/service/software/packages/r/R-4.2.1/lib/R/lib/libRblas.so

LAPACK: /mnt/service/software/packages/r/R-4.2.1/lib/R/lib/libRlapack.so

locale:

 [1] LC_CTYPE=en_GB.UTF-8       LC_NUMERIC=C               LC_TIME=en_GB.UTF-8

 [4] LC_COLLATE=en_GB.UTF-8     LC_MONETARY=en_GB.UTF-8    LC_MESSAGES=en_GB.UTF-8

 [7] LC_PAPER=en_GB.UTF-8       LC_NAME=C                  LC_ADDRESS=C

[10] LC_TELEPHONE=C             LC_MEASUREMENT=en_GB.UTF-8 LC_IDENTIFICATION=C


attached base packages:

[1] stats4    stats     graphics  grDevices utils     datasets  methods   base

other attached packages:

 [1] apeglm_1.20.0               EnhancedVolcano_1.16.0
```

```
 [3] ggrepel_0.9.3               org.Hs.eg.db_3.16.0
 [5] AnnotationDbi_1.60.2        RColorBrewer_1.1-3
 [7] DESeq2_1.38.0               SummarizedExperiment_1.26.1
 [9] Biobase_2.58.0              MatrixGenerics_1.8.1
[11] matrixStats_0.62.0          GenomicRanges_1.48.0
[13] GenomeInfoDb_1.34.9         IRanges_2.32.0
[15] S4Vectors_0.36.2            BiocGenerics_0.44.0
[17] ggplot2_3.4.2               dplyr_1.1.1
loaded via a namespace (and not attached):
 [1] KEGGREST_1.38.0        tidyselect_1.2.0       xfun_0.39
 [4] colorspace_2.0-3       vctrs_0.5.0            generics_0.1.3
 [7] htmltools_0.5.3        yaml_2.3.6             utf8_1.2.2
[10] blob_1.2.3             rlang_1.0.6            pillar_1.8.1
[13] glue_1.6.2             withr_2.5.0            DBI_1.1.3
[16] bit64_4.0.5            GenomeInfoDbData_1.2.9 lifecycle_1.0.3
[19] zlibbioc_1.44.0        Biostrings_2.66.0      munsell_0.5.0
[22] gtable_0.3.1           evaluate_0.17          memoise_2.0.1
[25] fastmap_1.1.0          GenomeInfoDb_1.34.9    fansi_1.0.3
[28] Rcpp_1.0.9             scales_1.2.1           cachem_1.0.6
[31] XVector_0.38.0         bit_4.0.4              png_0.1-8
[34] digest_0.6.30          grid_4.2.1             cli_3.4.1
[37] tools_4.2.1            bitops_1.0-7           magrittr_2.0.3
[40] RCurl_1.98-1.12        tibble_3.1.8           RSQLite_2.3.1
[43] crayon_1.5.2           pkgconfig_2.0.3        assertthat_0.2.1
[46] httr_1.4.4             rstudioapi_0.14
[49] R6_2.5.1               compiler_4.2.1
```