DGE_basic_pipeline

Roger Huerlimann

23/03/2022

```
#=================================================================
### Loading packages ### #=================================================
```

```
sapply(c("DESeq2", "tidyverse"), require, character.only = TRUE)
```

```
## Loading required package: DESeq2

## Loading required package: S4Vectors

## Loading required package: stats4

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

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

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

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:utils':
##
##     findMatches

## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
```

```
## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: GenomicRanges

## Loading required package: GenomeInfoDb

## Warning: package 'GenomeInfoDb' was built under R version 4.3.2

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: Biobase

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

```
##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians

## Loading required package: tidyverse

## Warning: package 'ggplot2' was built under R version 4.3.3

## Warning: package 'tidyr' was built under R version 4.3.2

## Warning: package 'readr' was built under R version 4.3.2

## Warning: package 'dplyr' was built under R version 4.3.3

## Warning: package 'stringr' was built under R version 4.3.2

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::%within%() masks IRanges::%within%()
## x dplyr::collapse()     masks IRanges::collapse()
## x dplyr::combine()      masks Biobase::combine(), BiocGenerics::combine()
## x dplyr::count()        masks matrixStats::count()
## x dplyr::desc()         masks IRanges::desc()
## x tidyr::expand()       masks S4Vectors::expand()
## x dplyr::filter()       masks stats::filter()
## x dplyr::first()        masks S4Vectors::first()
## x dplyr::lag()          masks stats::lag()
## x ggplot2::Position()   masks BiocGenerics::Position(), base::Position()
## x purrr::reduce()       masks GenomicRanges::reduce(), IRanges::reduce()
## x dplyr::rename()       masks S4Vectors::rename()
## x lubridate::second()   masks S4Vectors::second()
## x lubridate::second<-() masks S4Vectors::second<-()
## x dplyr::slice()        masks IRanges::slice()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

##    DESeq2 tidyverse
##      TRUE      TRUE

#=================================================================
### Data preparation ### #=========================================
```

```r
#loading the sample information into R from csv files
sample_table <- read.csv("01_Raw_data/Metadata_all.csv", header=TRUE)

#loading the count data into R from txt files
count_data <- read.delim("01_Raw_data/STAR_gene_counts_all.tsv", header=TRUE)
rownames(count_data) <- count_data[,1]
count_data <- as.matrix(count_data[,-1])


#loading the annotation information into R from the csv file
annotation <- read_tsv("01_Raw_Data/Emal_final_annotation_20211122.txt")
```

```
## Rows: 36407 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (9): SeqName, Description, GO IDs, GO Names, Enzyme Codes, Enzyme Names,...
## dbl (5): Length, #Hits, e-Value, sim mean, #GO
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# removing duplicated gene names and unwanted genes
annotation <- annotation %>%
               filter(!duplicated(annotation$SeqName)) %>%
               filter(grepl("jg", SeqName))
```

#==========

Data processing ### #==========

```r
#Setting up the DESeq model
dds <- DESeqDataSetFromMatrix(countData = count_data,colData=sample_table,design = ~ Stage)
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```r
#Running the actual DESeq fitting process.
dds <- DESeq(dds)
```

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
#Running the reduced model for the LRT analysis
dds_lrt <- DESeq(dds, test="LRT", reduced = ~ 1)
```

```
## 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
```

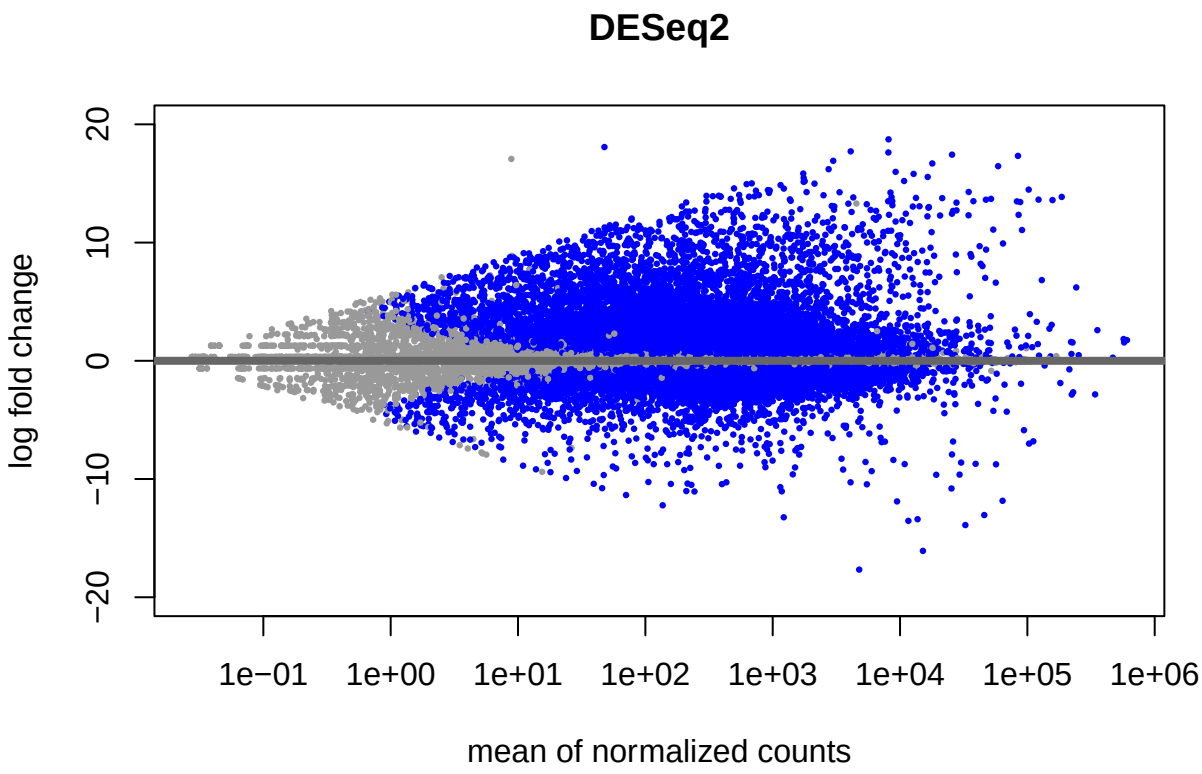
```
res_lrt <- results(dds_lrt)

summary(res_lrt)
```

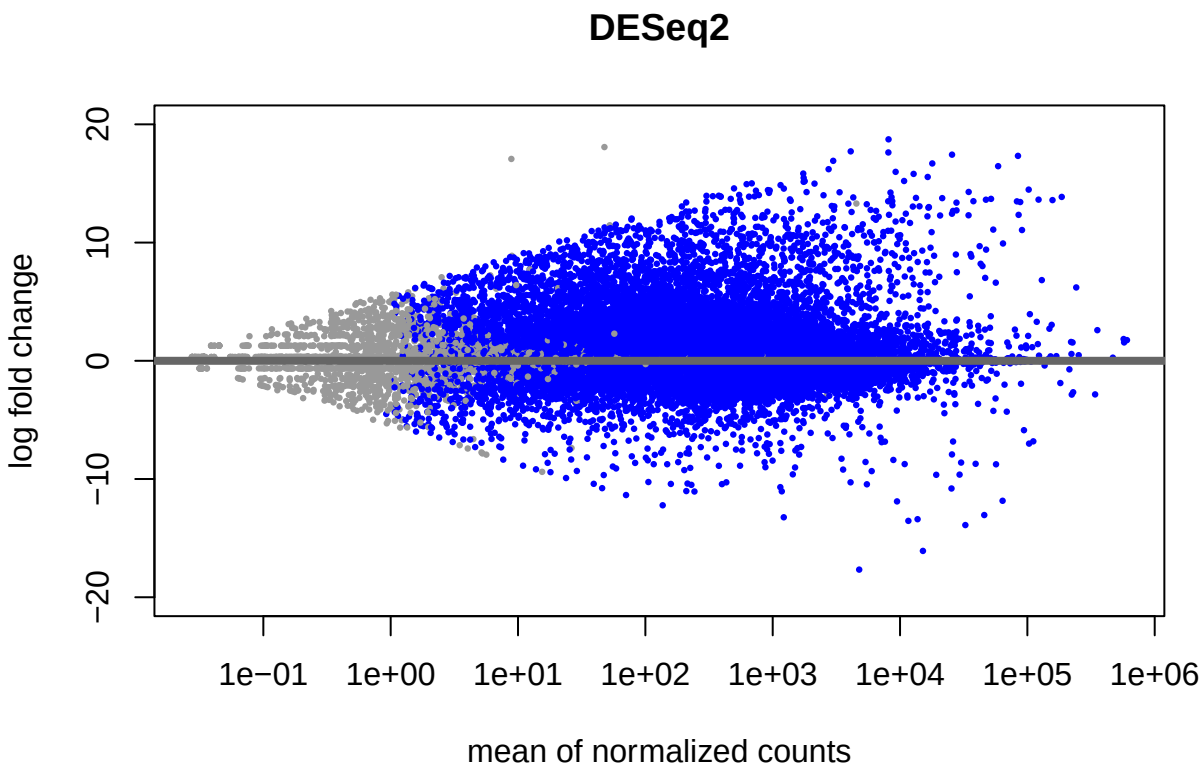
```
##
## out of 25987 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 12890, 50%
## LFC < 0 (down)     : 10866, 42%
## outliers [1]       : 9, 0.035%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

#========== ### Differentially expressed genes ### #==========
#MA plot to plot differentially expressed genes (in blue), at different ranges for log fold change #Since we are looking at completely different tissues, the fold changes are relatively large

```
plotMA(dds,ylim=c(-20,20),main='DESeq2')
```




```
plotMA(dds_lrt,ylim=c(-20,20),main='DESeq2')
```



#Pulling out the different contrasts

```
# setting a slighly more stringent cut-off
padj.cutoff <- 0.01

res_D01_D03 <- results(dds, contrast=c("Stage","D01","D03"))
sig_res_D01_D03 <- res_D01_D03 %>%
  data.frame() %>%
  rownames_to_column(var="Gene") %>%
  as_tibble() %>%
  filter(padj < padj.cutoff) %>%
  filter(between(!log2FoldChange, -.58, .58) )
print("sig_res_D01_D03")
```

```
## [1] "sig_res_D01_D03"
```

```
nrow(sig_res_D01_D03)
```

```
## [1] 14830
```

```
res_D03_D06 <- results(dds, contrast=c("Stage","D03","D06"))
sig_res_D03_D06 <- res_D03_D06 %>%
  data.frame() %>%
  rownames_to_column(var="Gene") %>%
  as_tibble() %>%
```

```
  filter(padj < padj.cutoff) %>%
  filter(between(!log2FoldChange, -.58, .58) )
print("sig_res_D03_D06")
```

```
## [1] "sig_res_D03_D06"
```

```
nrow(sig_res_D03_D06)
```

```
## [1] 8381
```

```
res_D06_D10 <- results(dds, contrast=c("Stage","D06","D10"))
sig_res_D06_D10 <- res_D06_D10 %>%
  data.frame() %>%
  rownames_to_column(var="Gene") %>%
  as_tibble() %>%
  filter(padj < padj.cutoff) %>%
  filter(between(!log2FoldChange, -.58, .58) )
print("sig_res_D06_D10")
```

```
## [1] "sig_res_D06_D10"
```

```
nrow(sig_res_D06_D10)
```

```
## [1] 1844
```

```
res_D10_D13 <- results(dds, contrast=c("Stage","D10","D13"))
sig_res_D10_D13 <- res_D10_D13 %>%
  data.frame() %>%
  rownames_to_column(var="Gene") %>%
  as_tibble() %>%
  filter(padj < padj.cutoff) %>%
  filter(between(!log2FoldChange, -.58, .58) )
print("sig_res_D10_D13")
```

```
## [1] "sig_res_D10_D13"
```

```
nrow(sig_res_D10_D13)
```

```
## [1] 1976
```

```
res_D13_D18 <- results(dds, contrast=c("Stage","D13","D18"))
sig_res_D13_D18 <- res_D13_D18 %>%
  data.frame() %>%
  rownames_to_column(var="Gene") %>%
  as_tibble() %>%
  filter(padj < padj.cutoff) %>%
  filter(between(!log2FoldChange, -.58, .58) )
print("sig_res_D13_D18")
```

```
## [1] "sig_res_D13_D18"
```

```
nrow(sig_res_D13_D18)
```

```
## [1] 191
```

```
res_D18_D32 <- results(dds, contrast=c("Stage","D18","D32"))
sig_res_D18_D32 <- res_D18_D32 %>%
  data.frame() %>%
  rownames_to_column(var="Gene") %>%
  as_tibble() %>%
  filter(padj < padj.cutoff) %>%
  filter(between(!log2FoldChange, -.58, .58) )
print("sig_res_D18_D32")
```

```
## [1] "sig_res_D18_D32"
```

```
nrow(sig_res_D18_D32)
```

```
## [1] 10774
```

```
res_D32_D60 <- results(dds, contrast=c("Stage","D32","D60"))
sig_res_D32_D60 <- res_D32_D60 %>%
  data.frame() %>%
  rownames_to_column(var="Gene") %>%
  as_tibble() %>%
  filter(padj < padj.cutoff) %>%
  filter(between(!log2FoldChange, -.58, .58) )
print("sig_res_D32_D60")
```

```
## [1] "sig_res_D32_D60"
```

```
nrow(sig_res_D32_D60)
```

```
## [1] 2654
```

```
res_D60_D60J <- results(dds, contrast=c("Stage","D60","D60J"))
sig_res_D60_D60J <- res_D60_D60J %>%
  data.frame() %>%
  rownames_to_column(var="Gene") %>%
  as_tibble() %>%
  filter(padj < padj.cutoff) %>%
  filter(between(!log2FoldChange, -.58, .58) )
print("sig_res_D60_D60J")
```

```
## [1] "sig_res_D60_D60J"
```

```
nrow(sig_res_D60_D60J)
```

```
## [1] 6963
```

Create a data frame to store the results in table

```
result_table <- data.frame(
  Contrast = c("sig_res_D01_D03", "sig_res_D03_D06", "sig_res_D06_D10", "sig_res_D10_D13",
               "sig_res_D13_D18", "sig_res_D18_D32", "sig_res_D32_D60", "sig_res_D60_D60J"),
  Total = c(nrow(sig_res_D01_D03), nrow(sig_res_D03_D06), nrow(sig_res_D06_D10), nrow(sig_res_D10_D13),
            nrow(sig_res_D13_D18), nrow(sig_res_D18_D32), nrow(sig_res_D32_D60), nrow(sig_res_D60_D60J)),
  Up = c(nrow(sig_res_D01_D03 %>% filter(log2FoldChange > 0)), nrow(sig_res_D03_D06 %>% filter(log2FoldC
         nrow(sig_res_D06_D10 %>% filter(log2FoldChange > 0)), nrow(sig_res_D10_D13 %>% filter(log2FoldC
         nrow(sig_res_D13_D18 %>% filter(log2FoldChange > 0)), nrow(sig_res_D18_D32 %>% filter(log2FoldC
         nrow(sig_res_D32_D60 %>% filter(log2FoldChange > 0)), nrow(sig_res_D60_D60J %>% filter(log2FolC
  Down = c(nrow(sig_res_D01_D03 %>% filter(log2FoldChange < 0)), nrow(sig_res_D03_D06 %>% filter(log2Fol
           nrow(sig_res_D06_D10 %>% filter(log2FoldChange < 0)), nrow(sig_res_D10_D13 %>% filter(log2Fol
           nrow(sig_res_D13_D18 %>% filter(log2FoldChange < 0)), nrow(sig_res_D18_D32 %>% filter(log2Fol
           nrow(sig_res_D32_D60 %>% filter(log2FoldChange < 0)), nrow(sig_res_D60_D60J %>% filter(log2F
)

# Print the table
print(result_table)
```

```
##           Contrast Total   Up Down
## 1  sig_res_D01_D03 14830 7151 7679
## 2  sig_res_D03_D06  8381 4464 3917
## 3  sig_res_D06_D10  1844  790 1054
## 4  sig_res_D10_D13  1976  671 1305
## 5  sig_res_D13_D18   191   81  110
## 6  sig_res_D18_D32 10774 5320 5454
## 7  sig_res_D32_D60  2654  967 1687
## 8 sig_res_D60_D60J  6963 3259 3704
```

#==========
Plotting PCA ### #==========

Data transformation

```
vst <- varianceStabilizingTransformation(dds, blind=FALSE)
```

PCA of all genes

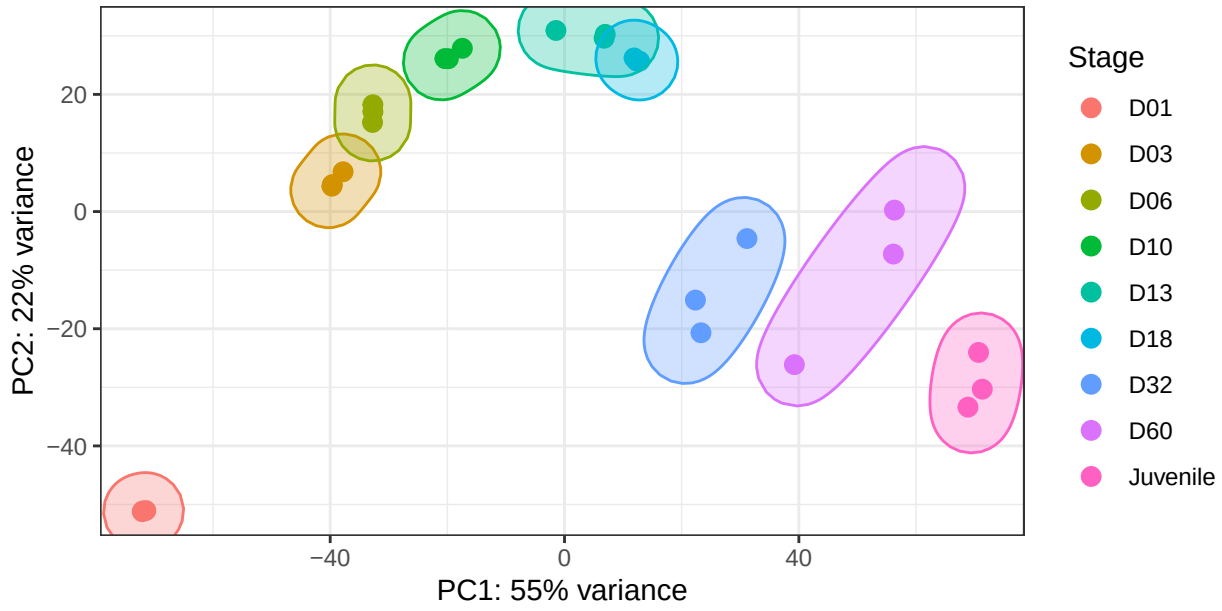
```
pcaData <- plotPCA(vst, intgroup="Stage", returnData=TRUE)
```

```
## using ntop=500 top features by variance
```

```
percentVar <- round(100 * attr(pcaData, "percentVar"))
pcaData$Stage = factor(pcaData$Stage)

library(ggplot2)

PCA <- ggplot(pcaData, aes(PC1, PC2, color=Stage)) +
  geom_point(size=3) +
  xlab(paste0("PC1: ",percentVar[1],"% variance")) +
  ylab(paste0("PC2: ",percentVar[2],"% variance")) +
  coord_equal() +
  scale_color_discrete(labels=c('D01', "D03", "D06", "D10", "D13", "D18", "D32", "D60", "Juvenile")) +
  theme_bw() +
  ggforce::geom_mark_ellipse(aes(fill = Stage, color = Stage), show.legend = FALSE)
PCA
```




```
# saving as jpeg and pdf
jpeg("03_For_publication/PCA_plot.jpg", quality = 100)
PCA
dev.off()
```

```
## pdf
##   2
```

```
pdf("03_For_publication/PCA_plot.pdf")
PCA
dev.off()
```

```
## pdf
##   2
```