2019_PlacentalBiologyCourse_DESeq2.Rmd

R.Hamilton, M.Prater, X.Zhao

17/06/2019

Authors : Dr Russell S. Hamilton, Dr Malwina Prater, Dr Xiaohui Zhao

Emails : rsh46@cam.ac.uk, mn367@cam.ac.uk, xz289@cam.ac.uk

Twitter : @drrshamilton

Web : http://www.trophoblast.cam.ac.uk/directory/Russell-Hamilton

Placental Biology Course 2019 (Centre for Trophoblast Research, University of Cambridge)

R-Script to perform differential transcript analysis of Placenta vs Yolk WT samples

Data derived from: `10.1242/dev.130369` Stumpo DJ et al (2016) Deficiency of the placenta- and yolk sac-specific tristetraprolin family member ZFP36L3 identifies likely mRNA targets and an unexpected link to placental iron metabolism. *Development*, **143(8)**:1424-33

Install the required external libraries / packages if needed:

Load in the required external libraries / packages:

```
library(dplyr)
library(DESeq2)
library(ggplot2)
library(ggrepel)
library(cowplot)
library(ggplot2)
library(ggalt)
library(ggrepel)
library(matrixStats)
library(ggdendro)
library(pheatmap)
library(clusterProfiler)
library(org.Mm.eg.db)
library(Rgraphviz)
```

```
library(tximport)
library(readr)
```

Set up the working directories, they should point to the location of the data:

```
base_dir <- "/home/ctr-teaching-test"
setwd(base_dir)
list.files(base_dir)
```

```
##  [1] "2018_PlacentalBiologyCourse_Presentation.pptx"
##  [2] "2019_PlacentalBiologyCourse_Commands_Rversion.Rmd"
##  [3] "2019_PlacentalBiologyCourse_DESeq2.Rmd"
##  [4] "2019_PlacentalBiologyCourse_Practical_1.pdf"
##  [5] "2019_PlacentalBiologyCourse_Practical_1.pptx"
##  [6] "DESeq2_kallisto_results_table.csv"
##  [7] "ensEMBL2id.csv"
##  [8] "ENST_ENSG_GeneName.GRCm38.kallisto.table"
##  [9] "multiqc.html"
## [10] "Mus_musculus.GRCm38.cdna.all.idx"
## [11] "SRR1811706_ES610_WT_Yolk_Sac"
## [12] "SRR1811707_ES611_WT_Yolk_Sac"
## [13] "SRR1811708_ES612_WT_Yolk_Sac"
## [14] "SRR1811709_ES613_WT_Yolk_Sac"
## [15] "SRR1823638_ES51_WT_Placenta"
## [16] "SRR1823638_sub_1.fastq.gz"
## [17] "SRR1823638_sub_2.fastq.gz"
## [18] "SRR1823639_ES51_WT_Placenta"
## [19] "SRR1823640_ES52_WT_Placenta"
## [20] "SRR1823641_ES52_WT_Placenta"
## [21] "SRR1823642_ES53_WT_Placenta"
## [22] "SRR1823643_ES54_WT_Placenta"
## [23] "SRR1823644_ES55_WT_Placenta"
## [24] "stumpo_2016_Development.pdf"
## [25] "TMP"
```

```
l2fc <- 2
significance <- 0.05
```

Read in the locations of the kallisto_output directories. Print out to screen the directories, you should see a list of 11 directories.

```
dirs <- grep("SRR.*/SRR.*_kallisto_output",list.dirs(base_dir,recursive=TRUE),value=TRUE)
print(dirs)
```

```
##  [1] "/home/ctr-teaching-test/SRR1811706_ES610_WT_Yolk_Sac/SRR1811706_1_val_1.fq.gz_kallisto_output"
##  [2] "/home/ctr-teaching-test/SRR1811707_ES611_WT_Yolk_Sac/SRR1811707_1_val_1.fq.gz_kallisto_output"
##  [3] "/home/ctr-teaching-test/SRR1811708_ES612_WT_Yolk_Sac/SRR1811708_1_val_1.fq.gz_kallisto_output"
##  [4] "/home/ctr-teaching-test/SRR1811709_ES613_WT_Yolk_Sac/SRR1811709_1_val_1.fq.gz_kallisto_output"
##  [5] "/home/ctr-teaching-test/SRR1823638_ES51_WT_Placenta/SRR1823638_1_val_1.fq.gz_kallisto_output"
##  [6] "/home/ctr-teaching-test/SRR1823639_ES51_WT_Placenta/SRR1823639_1_val_1.fq.gz_kallisto_output"
##  [7] "/home/ctr-teaching-test/SRR1823640_ES52_WT_Placenta/SRR1823640_1_val_1.fq.gz_kallisto_output"
```

```
##  [8] "/home/ctr-teaching-test/SRR1823641_ES52_WT_Placenta/SRR1823641_1_val_1.fq.gz_kallisto_output"
##  [9] "/home/ctr-teaching-test/SRR1823642_ES53_WT_Placenta/SRR1823642_1_val_1.fq.gz_kallisto_output"
## [10] "/home/ctr-teaching-test/SRR1823643_ES54_WT_Placenta/SRR1823643_1_val_1.fq.gz_kallisto_output"
## [11] "/home/ctr-teaching-test/SRR1823644_ES55_WT_Placenta/SRR1823644_1_val_1.fq.gz_kallisto_output"
```

Parse out the short sample names for nicer displays in plots later on in the analysis:

```
sample_id <- gsub("_1_val_1.fq.gz_kallisto_output", "", dirs)
sample_id <- gsub(".*/", "", sample_id)
# print to screen the new short names, they should look like "SRR1811706" ...
print(sample_id)
```

```
##  [1] "SRR1811706" "SRR1811707" "SRR1811708" "SRR1811709" "SRR1823638"
##  [6] "SRR1823639" "SRR1823640" "SRR1823641" "SRR1823642" "SRR1823643"
## [11] "SRR1823644"
```

Make sample table:

```
sample <-    c("SRR1811706", "SRR1811707", "SRR1811708", "SRR1811709", "SRR1823638",
               "SRR1823639", "SRR1823640", "SRR1823641", "SRR1823642", "SRR1823643", "SRR1823644")
condition <- c("YolkSac",    "YolkSac",    "YolkSac",    "YolkSac",    "Placenta",
               "Placenta",   "Placenta",   "Placenta",   "Placenta",   "Placenta",   "Placenta")
sample_table <- data.frame(sample, condition)
sample_table <- dplyr::select(sample_table, sample = sample, condition = condition)
sample_table <- dplyr::mutate(sample_table, path = dirs)

# Lets have a look at the sample table linkes the sample, condition directories / filenames
print(sample_table)
```

```
##        sample condition
## 1  SRR1811706   YolkSac
## 2  SRR1811707   YolkSac
## 3  SRR1811708   YolkSac
## 4  SRR1811709   YolkSac
## 5  SRR1823638  Placenta
## 6  SRR1823639  Placenta
## 7  SRR1823640  Placenta
## 8  SRR1823641  Placenta
## 9  SRR1823642  Placenta
## 10 SRR1823643  Placenta
## 11 SRR1823644  Placenta
##                                                                                                  path
## 1  /home/ctr-teaching-test/SRR1811706_ES610_WT_Yolk_Sac/SRR1811706_1_val_1.fq.gz_kallisto_output
## 2  /home/ctr-teaching-test/SRR1811707_ES611_WT_Yolk_Sac/SRR1811707_1_val_1.fq.gz_kallisto_output
## 3  /home/ctr-teaching-test/SRR1811708_ES612_WT_Yolk_Sac/SRR1811708_1_val_1.fq.gz_kallisto_output
## 4  /home/ctr-teaching-test/SRR1811709_ES613_WT_Yolk_Sac/SRR1811709_1_val_1.fq.gz_kallisto_output
## 5   /home/ctr-teaching-test/SRR1823638_ES51_WT_Placenta/SRR1823638_1_val_1.fq.gz_kallisto_output
## 6   /home/ctr-teaching-test/SRR1823639_ES51_WT_Placenta/SRR1823639_1_val_1.fq.gz_kallisto_output
## 7   /home/ctr-teaching-test/SRR1823640_ES52_WT_Placenta/SRR1823640_1_val_1.fq.gz_kallisto_output
## 8   /home/ctr-teaching-test/SRR1823641_ES52_WT_Placenta/SRR1823641_1_val_1.fq.gz_kallisto_output
## 9   /home/ctr-teaching-test/SRR1823642_ES53_WT_Placenta/SRR1823642_1_val_1.fq.gz_kallisto_output
## 10  /home/ctr-teaching-test/SRR1823643_ES54_WT_Placenta/SRR1823643_1_val_1.fq.gz_kallisto_output
## 11  /home/ctr-teaching-test/SRR1823644_ES55_WT_Placenta/SRR1823644_1_val_1.fq.gz_kallisto_output
```

Now you need to read in the annotations for transcripts. Usually it is best to pull the data directly from the ensEMBL website using biomart. However, for this practical we have premade the annotation file "ENST_ENSG_GeneName.GRCm38.kallisto.table". This can still take a little while to load in...

```
t2g       <- read.table(file.path(base_dir, "ENST_ENSG_GeneName.GRCm38.kallisto.table"),
                        header = TRUE, stringsAsFactors=FALSE)
t2g_deseq2 <- dplyr::rename(t2g, target_id = ensembl_transcript_id, ens_gene = ensembl_gene_id,
                            ext_gene = external_gene_name)
t2g_deseq2$ens_gene  <- gsub("\\.[0-9]*", "", t2g_deseq2$ens_gene)

head(t2g_deseq2)
```

```
##              target_id           ens_gene external_gene_shortname
## 1 ENSMUST00000178537.1 ENSMUSG00000095668                    Trbd1
## 2 ENSMUST00000178862.1 ENSMUSG00000094569                    Trbd2
## 3 ENSMUST00000177564.1 ENSMUSG00000096176                    Trdd2
## 4 ENSMUST00000196221.1 ENSMUSG00000096749                    Trdd1
## 5 ENSMUST00000179664.1 ENSMUSG00000096749                    Trdd1
## 6 ENSMUST00000179520.1 ENSMUSG00000094028                  Ighd4-1
##               external_gene_fullname
## 1              T_cell_receptor_beta
## 2              T_cell_receptor_beta
## 3  T_cell_receptor_delta_diversity_2
## 4  T_cell_receptor_delta_diversity_1
## 5  T_cell_receptor_delta_diversity_1
## 6 immunoglobulin_heavy_diversity_4-1
##                                    ext_gene
## 1                Trbd1:T_cell_receptor_beta
## 2                Trbd2:T_cell_receptor_beta
## 3    Trdd2:T_cell_receptor_delta_diversity_2
## 4    Trdd1:T_cell_receptor_delta_diversity_1
## 5    Trdd1:T_cell_receptor_delta_diversity_1
## 6 Ighd4-1:immunoglobulin_heavy_diversity_4-1
```

Differential gene analysis using DESeq2

- The package DESeq2 provides methods to test for differential expression by use of negative binomial generalized linear models; the estimates of dispersion and logarithmic fold changes incorporate data-driven prior distributions.
- method uses shrinkage estimation for dispersions and fold changes to improve stability and interpretability of estimates

See publication: Love, M.I., Huber, W., Anders, S. (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biology, 15:550. 10.1186/s13059-014-0550-8.

DESeq2 accepts inputs: an unnormalised count matrix (called here txi) and a table of sample information (here called sample_table).

Create a list of count files:

```
files <- file.path(dirs, "abundance.tsv")
names(files) <- sample_id
all(file.exists(files))
```

```
## [1] TRUE
```

Import the files and examine raw counts.

```
txi <- tximport(files, type = "kallisto", tx2gene = t2g_deseq2[,c(1,2)])
names(txi)
```

```
## [1] "abundance"           "counts"              "infReps"
## [4] "length"              "countsFromAbundance"
```

```
head(txi$counts)
```

```
##                     SRR1811706 SRR1811707 SRR1811708 SRR1811709  SRR1823638
## ENSMUSG00000000001 5341.00000  6693.0000  7405.0000 7096.00000 7864.000000
## ENSMUSG00000000003    0.00000     0.0000     0.0000    0.00000    0.000000
## ENSMUSG00000000028  135.00007   123.0004   161.9997  100.00000   89.000020
## ENSMUSG00000000037   54.99994    17.0000    59.0000   49.99998    4.000009
## ENSMUSG00000000049 1372.00000  1847.0000  3567.0000 1083.00000   16.000000
## ENSMUSG00000000056  718.00000  1028.5903  1059.0000 1280.00058  465.000070
##                     SRR1823639  SRR1823640  SRR1823641  SRR1823642
## ENSMUSG00000000001 4983.00000 8210.000000 4807.000000 5771.000000
## ENSMUSG00000000003    0.00000    0.000000    0.000000    0.000000
## ENSMUSG00000000028   82.00004  108.000000   88.000020  137.000280
## ENSMUSG00000000037    2.00000    4.999994    4.999996    6.999995
## ENSMUSG00000000049    7.00000   15.000000   13.000000   51.000000
## ENSMUSG00000000056  458.00000  772.000320  528.000380  631.999700
##                     SRR1823643  SRR1823644
## ENSMUSG00000000001 7391.000000 8631.000000
## ENSMUSG00000000003    0.000000    0.000000
## ENSMUSG00000000028  104.000030  162.474710
## ENSMUSG00000000037    5.000002    7.999999
## ENSMUSG00000000049   12.000000   23.999951
## ENSMUSG00000000056  742.931510  721.000310
```

Differential expression analysis steps are wrapped into a single function, `DESeq()`.

```
dds <- DESeqDataSetFromTximport(txi, sample_table, ~condition)
dds <- DESeq(dds)
dds
```

```
## class: DESeqDataSet
## dim: 32360 11
## metadata(1): version
## assays(8): counts avgTxLength ... replaceCounts replaceCooks
## rownames(32360): ENSMUSG00000000001 ENSMUSG00000000003 ...
##   ENSMUSG00000109577 ENSMUSG00000109578
## rowData names(23): baseMean baseVar ... maxCooks replace
## colnames(11): SRR1811706 SRR1811707 ... SRR1823643 SRR1823644
## colData names(4): sample condition path replaceable
```

```
resultsNames(dds)
```

```
## [1] "Intercept"                      "condition_YolkSac_vs_Placenta"
```

Results tables are generated using the function `results()`.

```
res <- lfcShrink(dds, coef="condition_YolkSac_vs_Placenta", type="normal")
```

```
## using 'normal' for LFC shrinkage, the Normal prior from Love et al (2014).
##
## Note that type='apeglm' and type='ashr' have shown to have less bias than type='normal'.
## See ?lfcShrink for more details on shrinkage type, and the DESeq2 vignette.
## Reference: https://doi.org/10.1093/bioinformatics/bty895
```

```
res <-res[order(res$padj),]
head(res)
```

```
## log2 fold change (MAP): condition YolkSac vs Placenta
## Wald test p-value: condition YolkSac vs Placenta
## DataFrame with 6 rows and 6 columns
##                            baseMean    log2FoldChange             lfcSE
##                           <numeric>         <numeric>         <numeric>
## ENSMUSG00000000440 2219.12285828675 -7.10958330505989 0.186344754569847
## ENSMUSG00000009281 8876.48378414179 -6.00270342284512 0.133075034853347
## ENSMUSG00000020689 4641.61735631465 -5.62632456821533 0.122261700633162
## ENSMUSG00000022464 18491.6039475728 -4.43855884206903 0.107783180583526
## ENSMUSG00000029648 36196.1632492297 -5.74430814123885 0.115512871553728
## ENSMUSG00000032666 4897.95564153874 -4.79076111606392 0.120304830960984
##                                stat    pvalue      padj
##                           <numeric> <numeric> <numeric>
## ENSMUSG00000000440 -37.8923173072965         0         0
## ENSMUSG00000009281 -45.0881431296822         0         0
## ENSMUSG00000020689 -45.9794060680792         0         0
## ENSMUSG00000022464 -41.1784694271336         0         0
## ENSMUSG00000029648 -49.7237759244842         0         0
## ENSMUSG00000032666 -39.8127151067746         0         0
```

```
summary(res)
```

```
##
## out of 25509 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 6397, 25%
## LFC < 0 (down)     : 7233, 28%
## outliers [1]       : 20, 0.078%
## low counts [2]     : 4878, 19%
## (mean count < 1)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
# we can save the results table:
#write.csv(res, file=paste("DESeq2", "_kallisto_results_table.csv", sep =""))
```

How many adjusted p-values were less than 0.05?

```
sum(res$padj < 0.05, na.rm=TRUE)
```

```
## [1] 12535
```

Now annotate DESeq2 results table `res` and lets call it `results_deseq2`.

```
#library(biomaRt)
#ensembl    =  useEnsembl(biomart="ensembl", dataset="mmusculus_gene_ensembl")
#ensEMBL2id <- getBM(attributes=c('ensembl_gene_id', 'external_gene_name', 'entrezgene',
#               'description'), mart = ensembl)
ensEMBL2id <- read.csv("ensEMBL2id.csv")
ensEMBL2id <- ensEMBL2id[,-1]
head(ensEMBL2id)
```

```
##      ensembl_gene_id external_gene_name entrezgene
## 1 ENSMUSG00000064336              mt-Tf         NA
## 2 ENSMUSG00000064337            mt-Rnr1         NA
## 3 ENSMUSG00000064338              mt-Tv         NA
## 4 ENSMUSG00000064339            mt-Rnr2         NA
## 5 ENSMUSG00000064340             mt-Tl1         NA
## 6 ENSMUSG00000064341             mt-Nd1      17716
##                                                                      description
## 1   mitochondrially encoded tRNA phenylalanine [Source:MGI Symbol;Acc:MGI:102487]
## 2             mitochondrially encoded 12S rRNA [Source:MGI Symbol;Acc:MGI:102493]
## 3          mitochondrially encoded tRNA valine [Source:MGI Symbol;Acc:MGI:102472]
## 4             mitochondrially encoded 16S rRNA [Source:MGI Symbol;Acc:MGI:102492]
## 5       mitochondrially encoded tRNA leucine 1 [Source:MGI Symbol;Acc:MGI:102482]
## 6 mitochondrially encoded NADH dehydrogenase 1 [Source:MGI Symbol;Acc:MGI:101787]
```

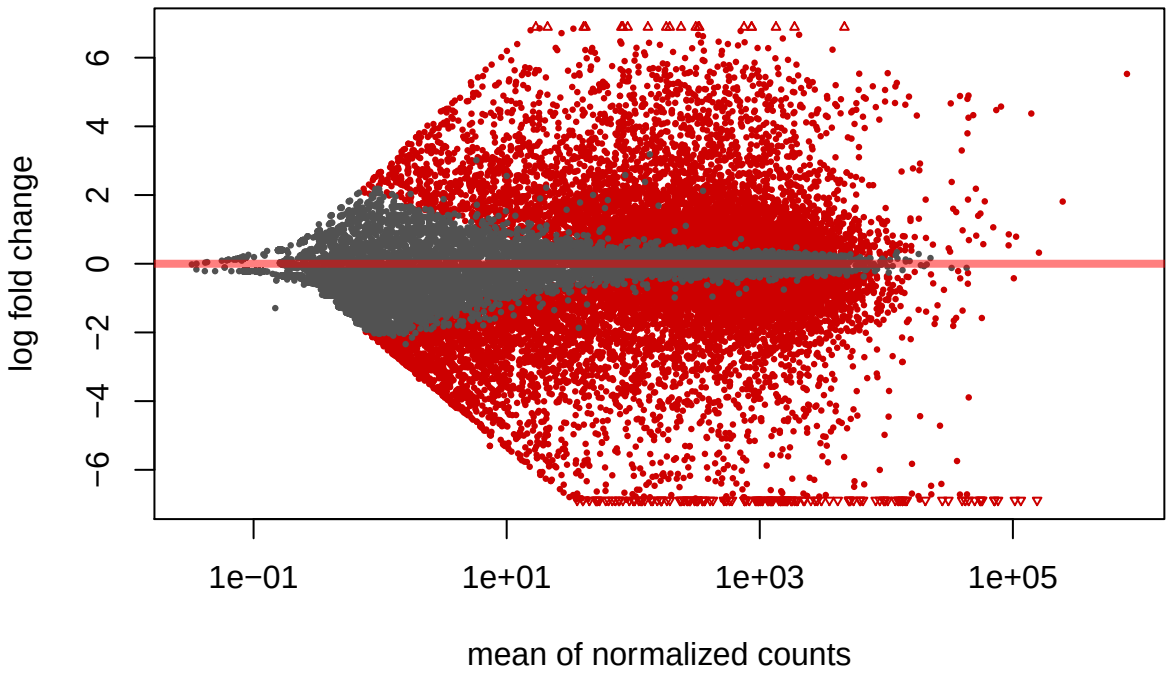
```
results_deseq2          <- as.data.frame(res)
results_deseq2$ens_gene <- rownames(results_deseq2)
results_deseq2          <- unique(merge(results_deseq2, ensEMBL2id, by.x = "ens_gene",
                                        by.y = "ensembl_gene_id", all.x= TRUE))
head(results_deseq2)
```

```
##             ens_gene   baseMean log2FoldChange     lfcSE      stat
## 1 ENSMUSG00000000001 6721.31215     -0.3137032 0.1455953 -2.154624
## 2 ENSMUSG00000000003    0.00000             NA        NA        NA
## 3 ENSMUSG00000000028  116.92043     -0.2211728 0.2140354 -1.033333
## 4 ENSMUSG00000000037   15.64743      1.7600994 0.6032649  2.923309
## 5 ENSMUSG00000000049  572.59420      5.7812896 0.5007215 11.510782
## 6 ENSMUSG00000000056  742.43279      0.3178982 0.1816325  1.750234
##         pvalue         padj external_gene_name entrezgene
## 1 3.119125e-02 5.116140e-02              Gnai3      14679
## 2           NA           NA               Pbsn      54192
## 3 3.014479e-01 3.720138e-01              Cdc45      12544
```

```
## 4 3.463324e-03 6.897535e-03              Scml2     107815
## 5 1.164182e-30 1.455263e-29               Apoh      11818
## 6 8.007801e-02 1.184449e-01               Narf      67608
##                                                                                                 descriptio
## 1 guanine nucleotide binding protein (G protein), alpha inhibiting 3 [Source:MGI Symbol;Acc:MGI:95773
## 2                                                                 probasin [Source:MGI Symbol;Acc:MGI:186048
## 3                                                  cell division cycle 45 [Source:MGI Symbol;Acc:MGI:133807
## 4                                  Scm polycomb group protein like 2 [Source:MGI Symbol;Acc:MGI:134004
## 5                                                  apolipoprotein H [Source:MGI Symbol;Acc:MGI:88058
## 6                               nuclear prelamin A recognition factor [Source:MGI Symbol;Acc:MGI:191485
```

Explore data using default DeSeq2 Functions:

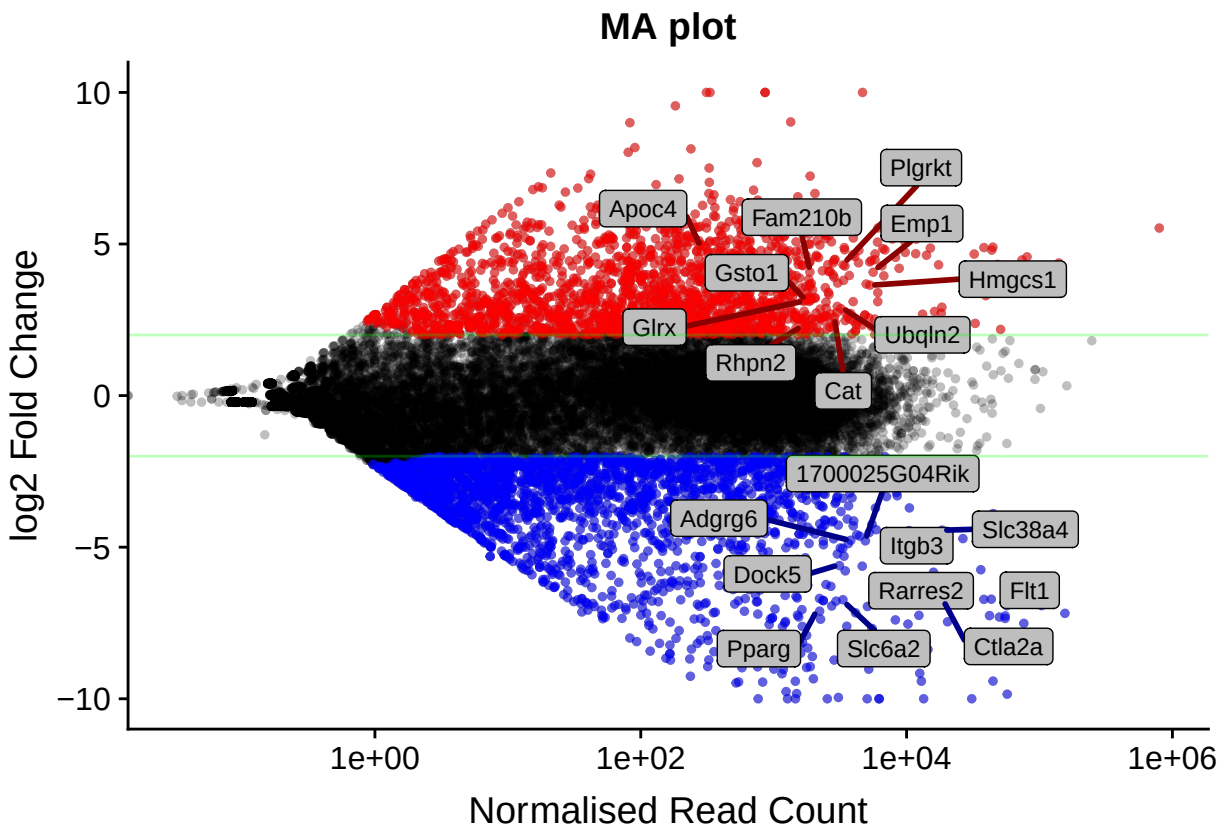
```
DESeq2::plotMA(res)
```



Explore data using custom functions:

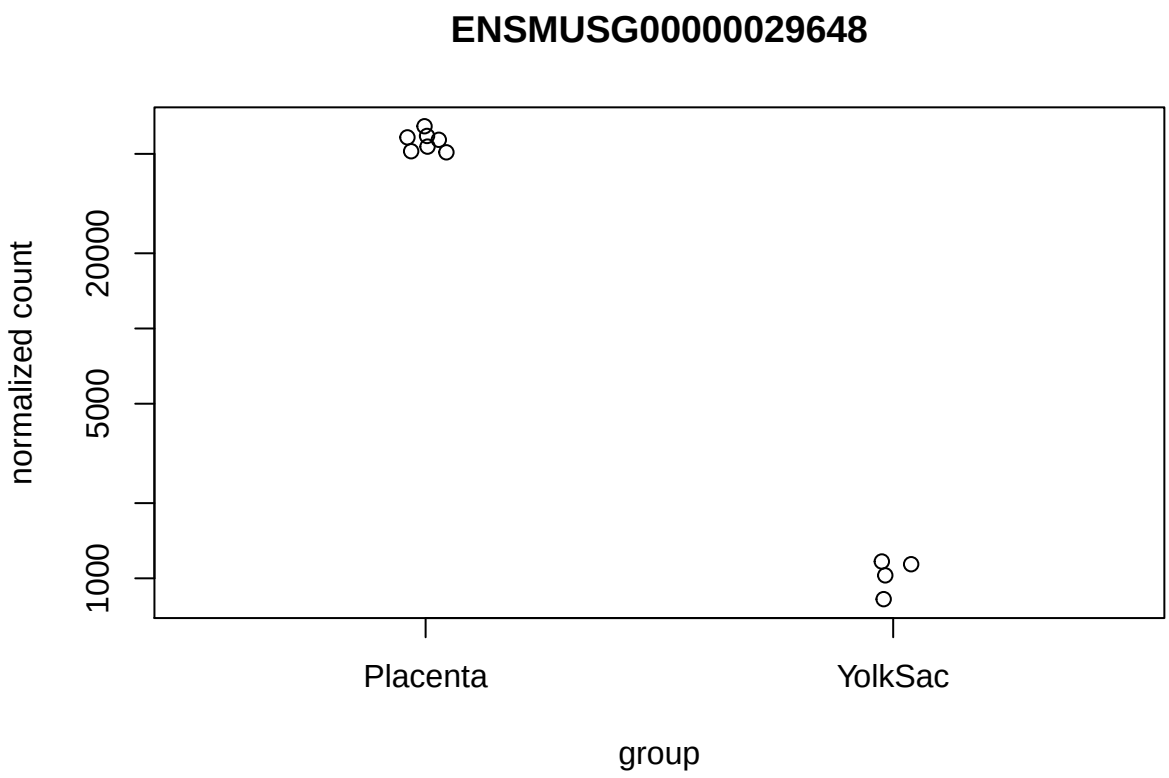
```
Title <- "MA plot"

results_deseq2.ma <- results_deseq2
results_deseq2.ma$log2FoldChange[results_deseq2.ma$log2FoldChange > 10]  <- 10
results_deseq2.ma$log2FoldChange[results_deseq2.ma$log2FoldChange < -10] <- -10

# select genes for labeling
results_deseq2.label      <- subset(results_deseq2.ma, abs(log2FoldChange) >= l2fc &
                                    (padj < significance | padj == 0))
```

```
results_deseq2.label.up   <- subset(results_deseq2.label, log2FoldChange > 0 )
results_deseq2.label.up   <- results_deseq2.label.up[order(results_deseq2.label.up$padj, decreasing=FALS
results_deseq2.label.down <- subset(results_deseq2.label, log2FoldChange < 0 )
results_deseq2.label.down <- results_deseq2.label.down[order(results_deseq2.label.down$padj, decreasing=

# plot using ggplot2 package

ggplot(data = results_deseq2.ma, aes(x=baseMean, y=log2FoldChange)) +
  geom_point(size=1, alpha=0.25, col="black") +
  geom_point(data=subset(results_deseq2.ma, (padj <= significance & log2FoldChange >=  l2fc)),
             size=1, alpha=0.5,  col="red") +
  geom_point(data=subset(results_deseq2.ma, (padj <= significance & log2FoldChange <= -l2fc)),
             size=1, alpha=0.5,  col="blue") +
  geom_label_repel(data=results_deseq2.label.up[1:10,],
                   aes( x=baseMean, y=log2FoldChange, label=external_gene_name),
                   fill='gray', colour='black', point.padding = unit(0.25, "lines"),
                   size=3, segment.size = 1, segment.color = 'darkred',  nudge_x = 0, nudge_y=0) +
  geom_label_repel(data=results_deseq2.label.down[1:10,],
                   aes( x=baseMean, y=log2FoldChange, label=external_gene_name),
                   fill='gray', colour='black', point.padding = unit(0.25, "lines"),
                   size=3, segment.size = 1, segment.color = 'darkblue',  nudge_x = 0, nudge_y=0) +
  scale_x_log10() +
  xlab("Normalised Read Count") + ylab("log2 Fold Change") + ggtitle(paste("MA plot")) +
  geom_abline(intercept = l2fc, slope = 0, colour='green', alpha=0.25) +
  geom_abline(intercept = -l2fc, slope = 0, colour='green', alpha=0.25)
```



Lets pick a gene to examine individually - Flt1 (ENSMUSG00000029648).

```
plotCounts(dds, gene="ENSMUSG00000029648", intgroup="condition")
```



rlog transformation

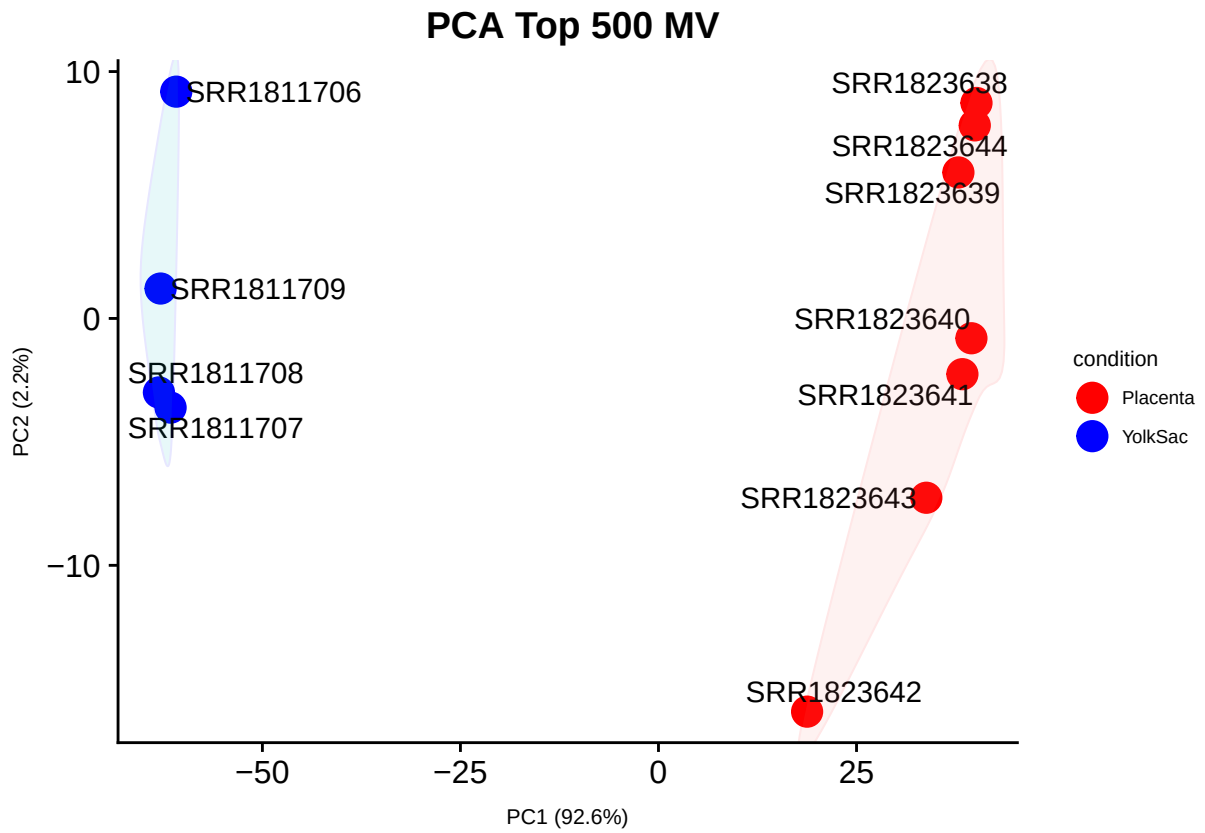
This function transforms the count data to the log2 scale in a way which minimizes differences between samples for rows with small counts, and which normalizes with respect to library size. Note: This an take upto a minute to run!!!

```
rld <- rlogTransformation(dds)
```

Plot PCA:

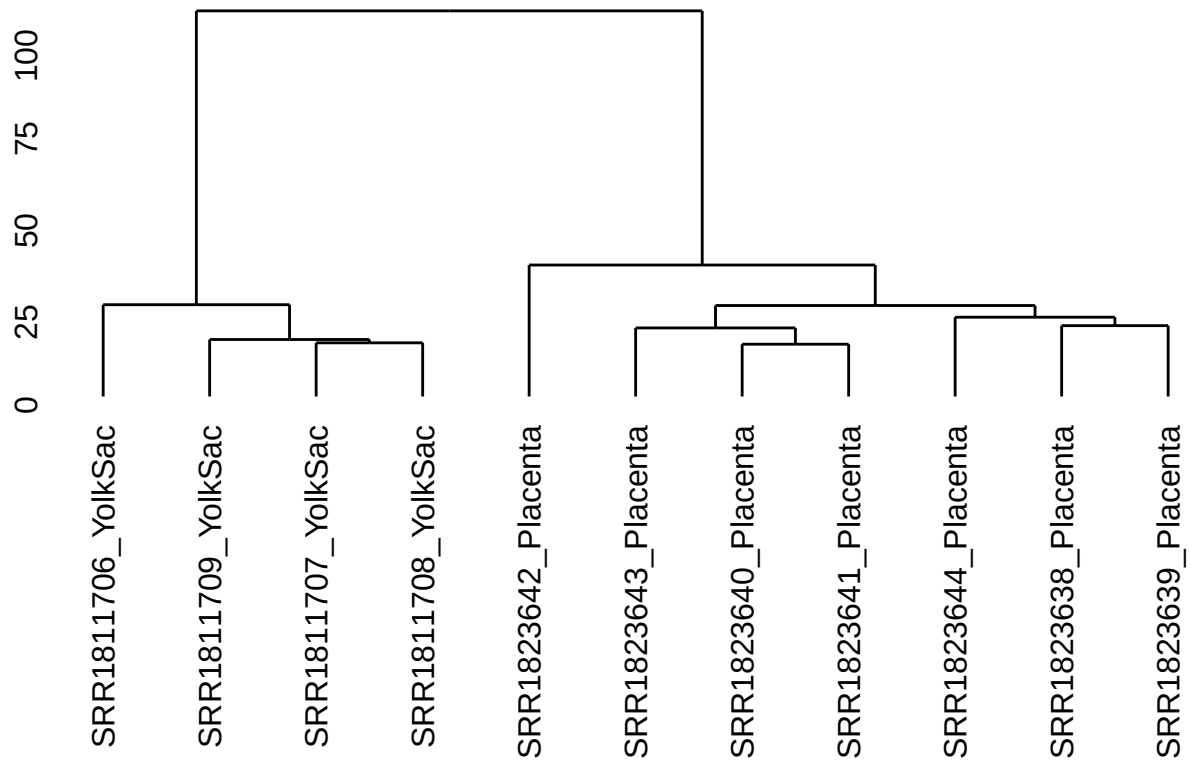
```
elementTextSize <- 8
topNum = 500

pca = prcomp(t(assay(rld)))
rv = rowVars(assay(rld))
select = order(rv, decreasing = TRUE)[seq_len(min(topNum, length(rv)))]
pca = prcomp(t(assay(rld)[select, ]))

pc1var <- round(summary(pca)$importance[2,1]*100, digits=1)
pc2var <- round(summary(pca)$importance[2,2]*100, digits=1)
pc1lab <- paste0("PC1 (",as.character(pc1var),"%)")
pc2lab <- paste0("PC2 (",as.character(pc2var),"%)")
```

```
scores <- data.frame(sample_id, pca$x, sample_table)

ggplot(scores, aes(x = PC1, y = PC2, col = condition )) +
  geom_point(size = 5 ) +
  geom_text_repel(aes(label=sample_id), col = "black") +
  scale_colour_manual(name="condition", values = c(YolkSac = "blue", Placenta= "red")) +
  geom_encircle(alpha = 0.1, show.legend = FALSE, aes(fill=condition)) +
  xlab(pc1lab) + ylab(pc2lab) +
  ggtitle(paste(" PCA Top ", topNum, " MV", sep="")) +
  theme(text = element_text(size=elementTextSize))
```



Example of hierarchical clustering:

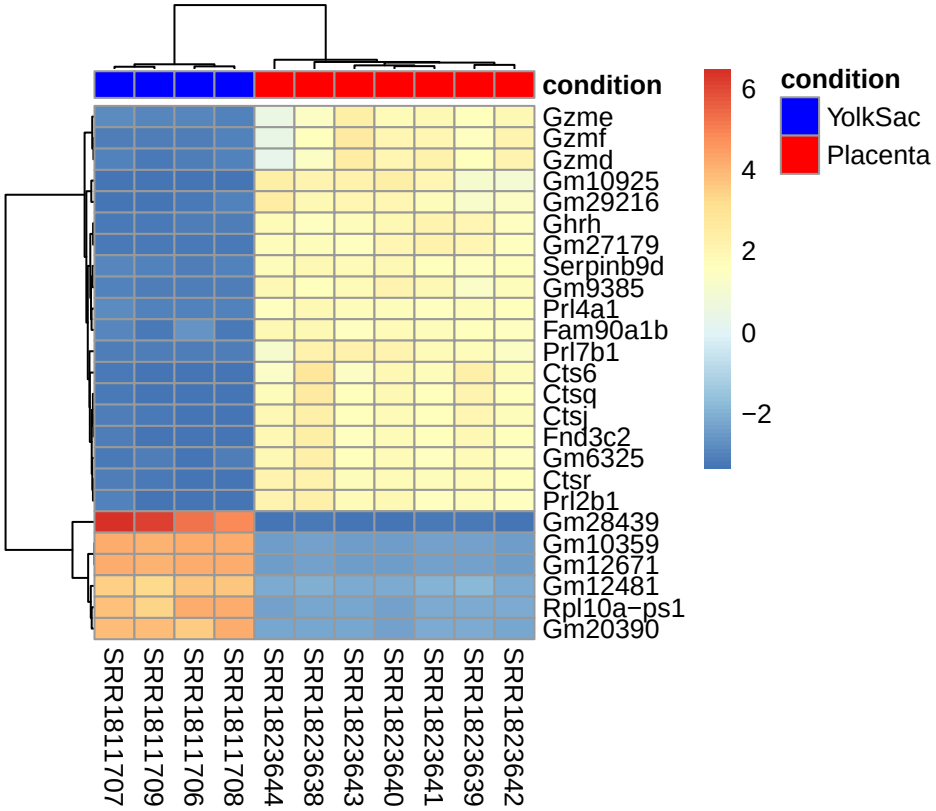
```
rld_name <- rld
colnames(rld_name) <- paste(colnames(rld_name),  sample_table$condition, sep = "_" )

sample_distances <- dist(t(assay(rld_name)[select, ]))

ggdendrogram(hclust(sample_distances), rotate = FALSE, segments = TRUE)
```



Heatmap of top DEGs:

```
#### plot top 25  genes:
selected_genes    <- subset(results_deseq2, results_deseq2$padj < 0.00000001)
selected_genes    <- head(selected_genes[order(abs(selected_genes$log2FoldChange),
                                               decreasing = TRUE),],  25)
selected_genes_id <- selected_genes$ens_gene
genes2plot        <- unique( selected_genes_id )

#### alternatively you can plot selected genes of interest e.g.:
#selected_genes    <- c("Itgb3", "Lepr", "Synb", "Sct", "Ghrh", "Psg16")
#selected_genes_id <- results_deseq2.ma[results_deseq2.ma$external_gene_shortname
#                     %in% selected_genes,]
#genes2plot <- unique(selected_genes_id$ens_gene)

rows         <- match(genes2plot, row.names(assay(rld)))
mat          <- assay(rld)[rows,]
mat          <- as.data.frame(mat)
#mat$YolkSac  <- rowMeans(mat[,c(1:4)])
#mat$Placenta <- rowMeans(mat[,c(5:11)])
#mat          <- mat[,c(15,16)]
mat          <- mat - rowMeans(mat)    #    MeanCentred
mat.df       <- data.frame(ens_gene=rownames(mat),mat)
mat.ann      <- unique(merge(mat.df, t2g_deseq2[,c(2,3)], by="ens_gene"))
mat.ann      <- mat.ann[!duplicated(mat.ann$external_gene_shortname),]
```

```
rownames(mat.ann) <- mat.ann$external_gene_shortname
mat.new <- mat.ann[ ,-c(1, ncol(mat.ann))]


annotation_col = data.frame(condition = sample_table[ , 2])
rownames(annotation_col) <- colnames(mat.new)
ann_colors = list(condition = c(YolkSac = "blue", Placenta= "red"))


pheatmap(mat.new, fontsize=10, fontsize_row=10, show_rownames=TRUE, cluster_cols = TRUE,
         cluster_rows = TRUE, cellwidth = 15, cellheight = 8, treeheight_row = 30,
         treeheight_col = 20,  annotation_col = annotation_col, annotation_colors = ann_colors)
```



GO analysis using `clusterProfiler`. Entrez gene id usually needed for GO analysis.

```
# create results table with significant genes that have absolute log2FC > 2
RESULTS_l2fc2 <- subset(results_deseq2, results_deseq2$padj < 0.05 & abs(results_deseq2$log2FoldChange)
RESULTS_l2fc2 <- RESULTS_l2fc2[order(RESULTS_l2fc2$log2FoldChange, decreasing = TRUE),]


# for GO analysis, create a vector of all genes and DEGs with their log2FC's:
geneList <- results_deseq2$log2FoldChange
names(geneList) <- results_deseq2$entrezgene
geneList <- geneList[!is.na(names(geneList))]

SigGeneList <- RESULTS_l2fc2$log2FoldChange
```

```
names(SigGeneList) <- RESULTS_l2fc2$entrezgene
SigGeneList <- SigGeneList[!is.na(names(SigGeneList))]
SigGeneList <- SigGeneList[!is.na(SigGeneList)]

# sort the vector
SigGeneList <- sort(SigGeneList, decreasing = T )
gene <- names(SigGeneList)

head(SigGeneList)
```

```
## 100303744    394434 100042514     20210     22239     12824
##  9.025674  8.181758  8.137693  7.685343  7.505321  7.239479
```

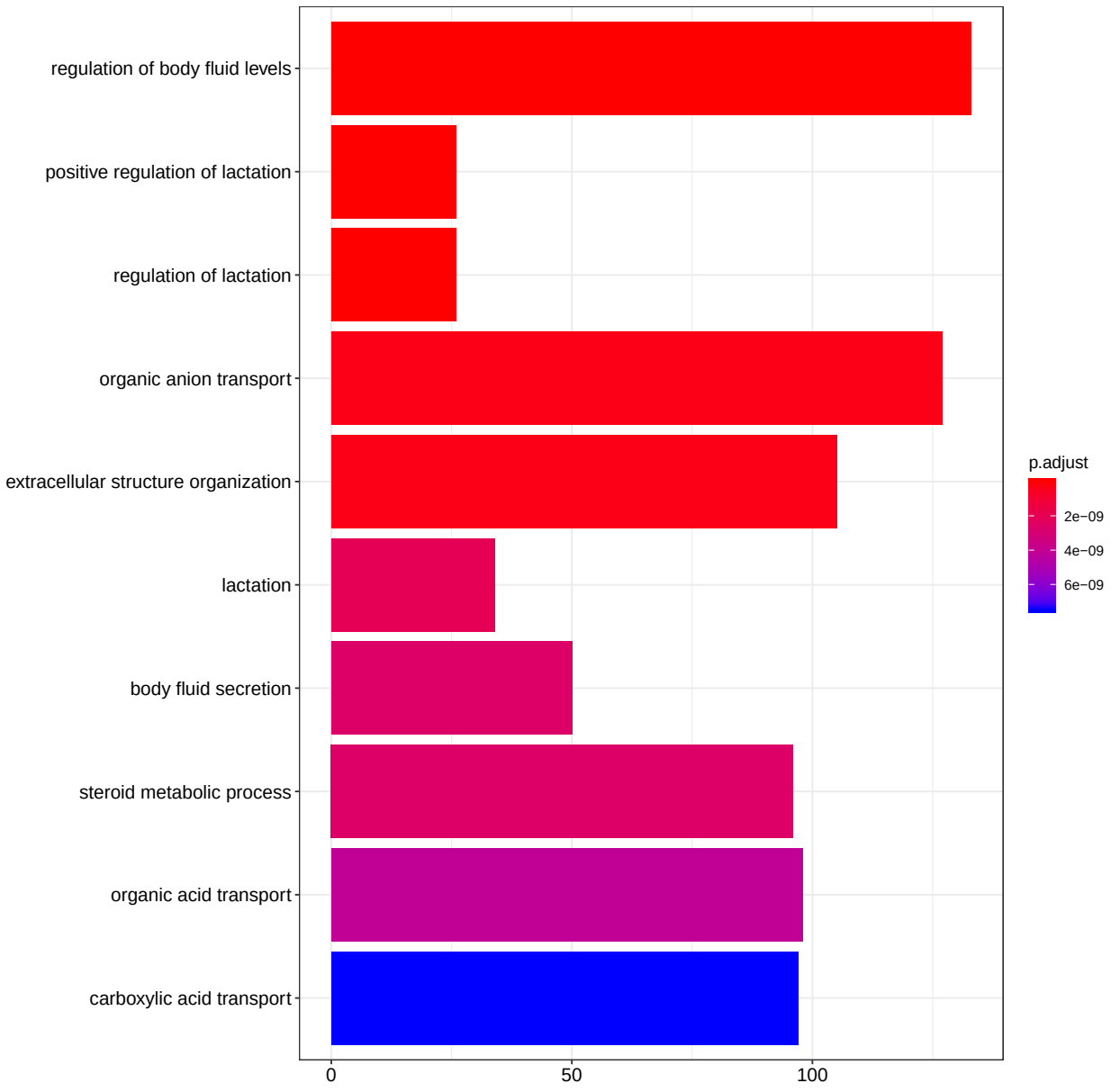
```
head(gene)
```

```
## [1] "100303744" "394434"    "100042514" "20210"     "22239"     "12824"
```

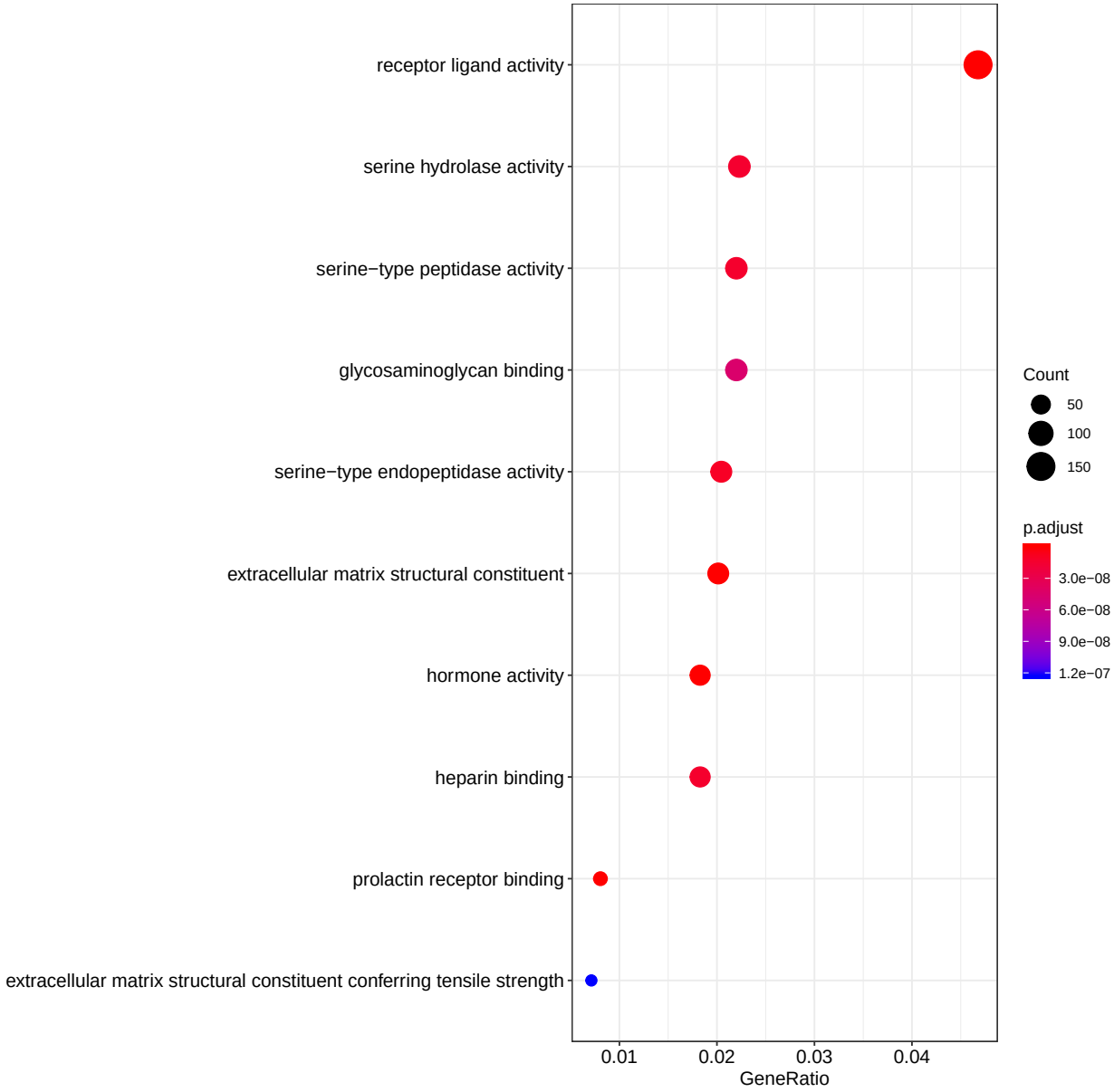
```
# GO over-representation test
ego_bp <- enrichGO(gene = gene, universe = names(geneList), OrgDb = org.Mm.eg.db, ont = "BP",
                   pAdjustMethod = "BH", pvalueCutoff  = 0.05, qvalueCutoff  = 0.05, readable      = TRU
ego_mf <- enrichGO(gene = gene, universe = names(geneList), OrgDb = org.Mm.eg.db, ont = "MF",
                   pAdjustMethod = "BH", pvalueCutoff  = 0.05, qvalueCutoff  = 0.05, readable      = TRU
ego_cc <- enrichGO(gene = gene, universe = names(geneList), OrgDb = org.Mm.eg.db, ont = "CC",
                   pAdjustMethod = "BH", pvalueCutoff  = 0.05, qvalueCutoff  = 0.05, readable      = TRU

# view GO results in a table
ego_bp_df <- as.data.frame(ego_bp)
ego_cc_df <- as.data.frame(ego_cc)
ego_mf_df <- as.data.frame(ego_mf)
```

Some ways to visualize GO results:

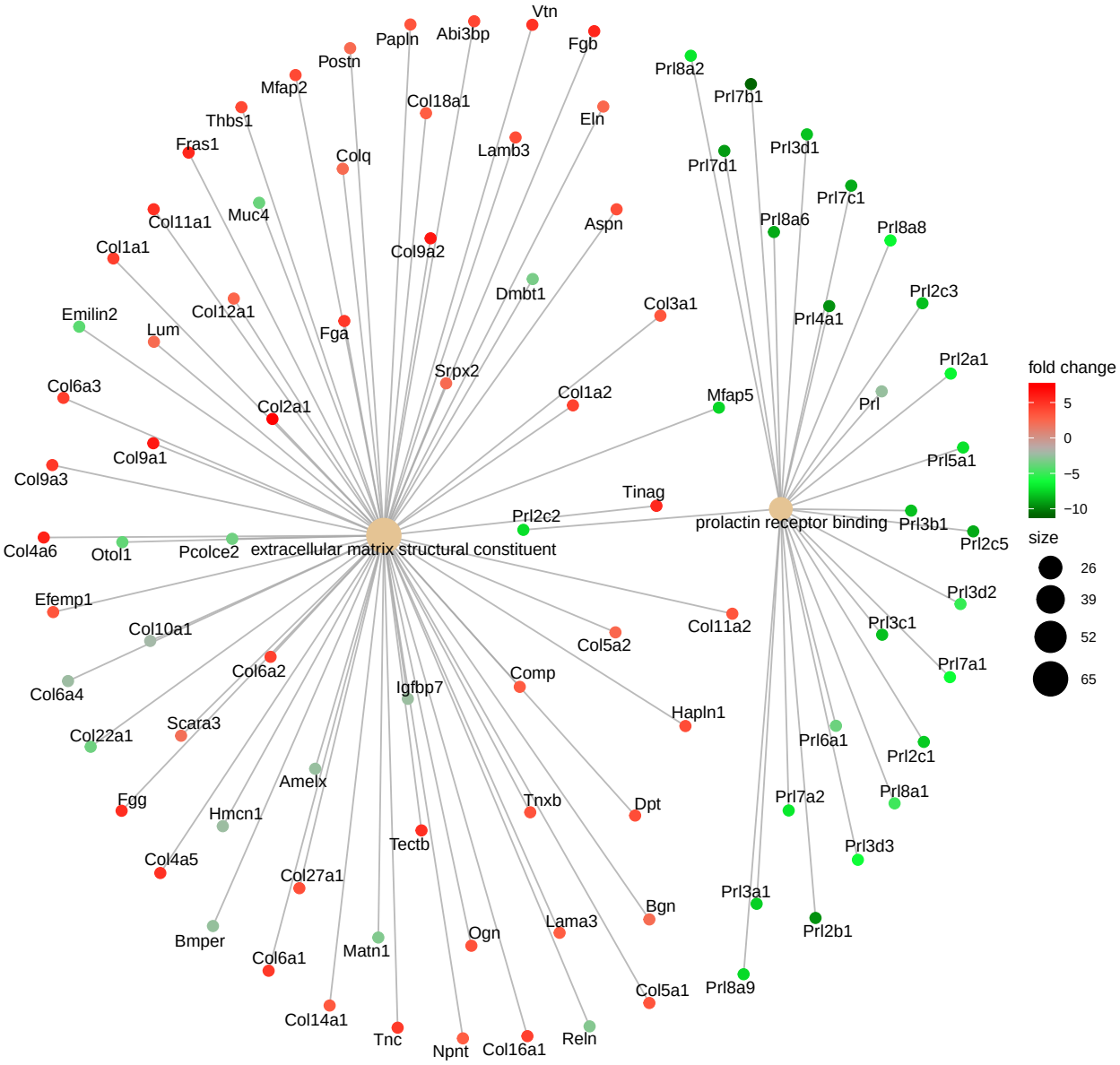
```
# Bar plot is the most widely used method to visualize enriched terms. It depicts the enrichment scores
# and gene count or ratio as bar height and color.
barplot(ego_bp, showCategory = 10)
```



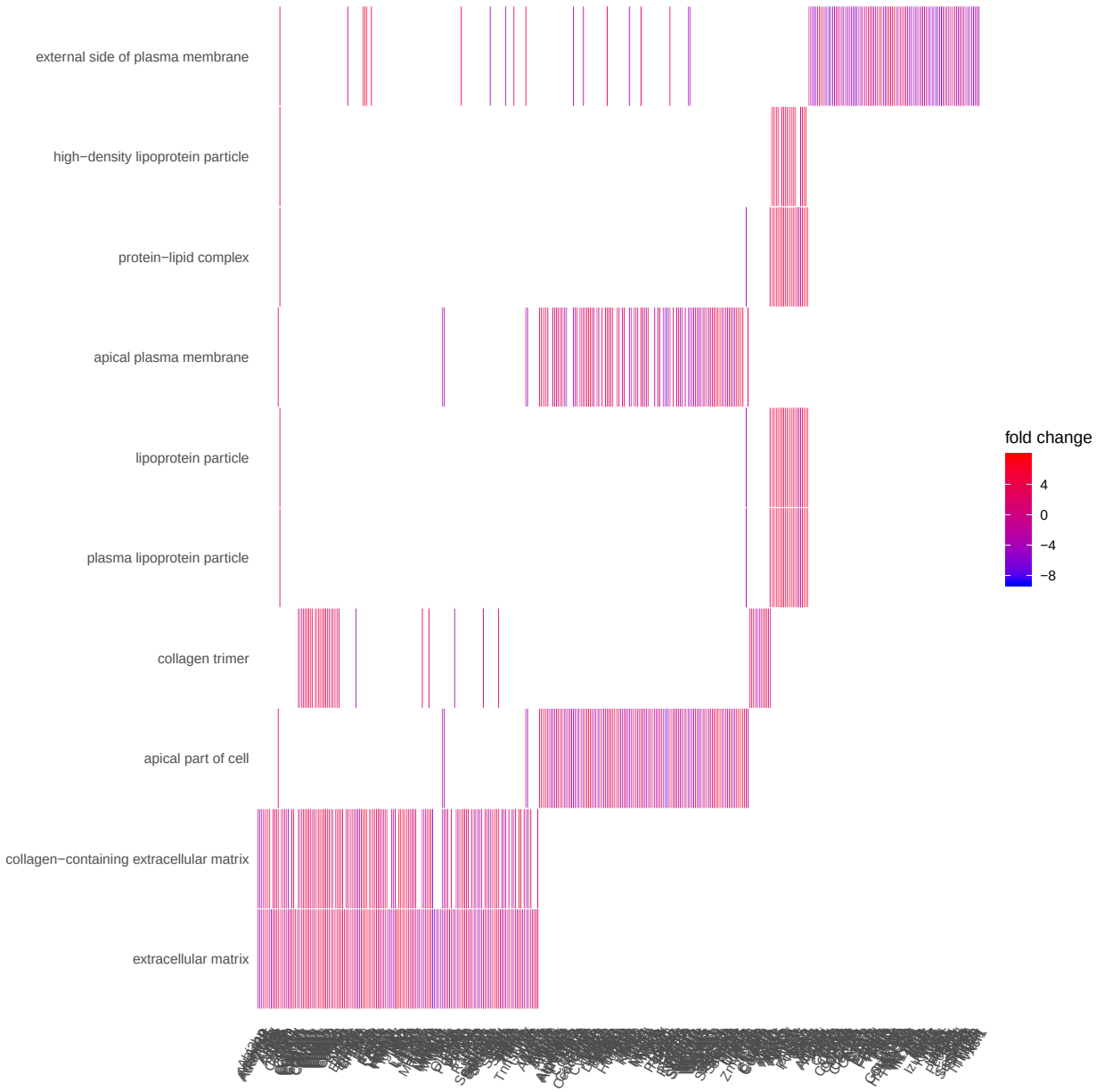

```
# Dot plot is similar to bar plot with the capability to encode another score as dot size
enrichplot::dotplot(ego_mf, showCategory=10, orderBy = "GeneRatio")
```



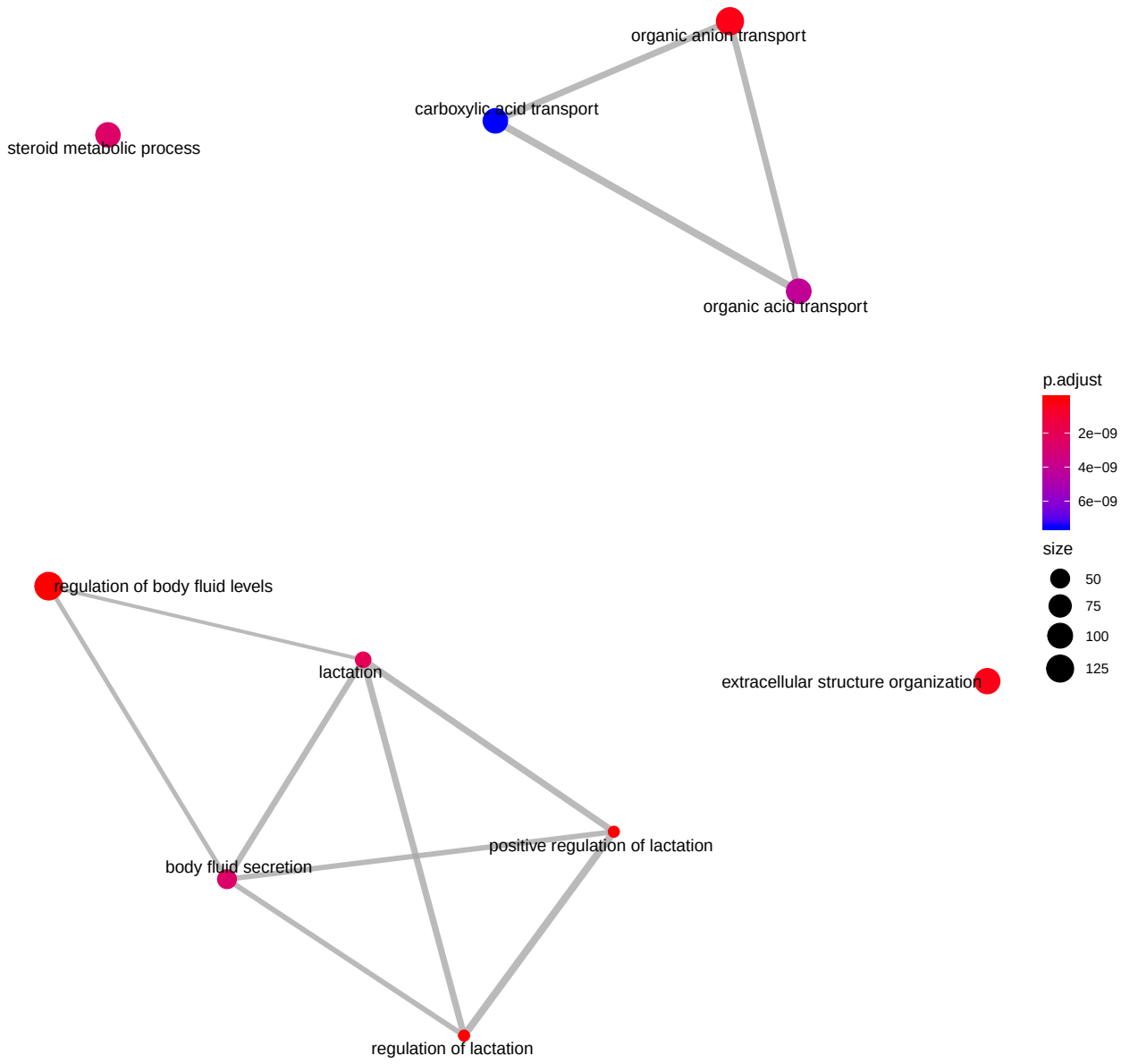

```
# category-gene-net plot
cnetplot(ego_mf, foldChange=SigGeneList, circular = FALSE, showCategory = 2)
```



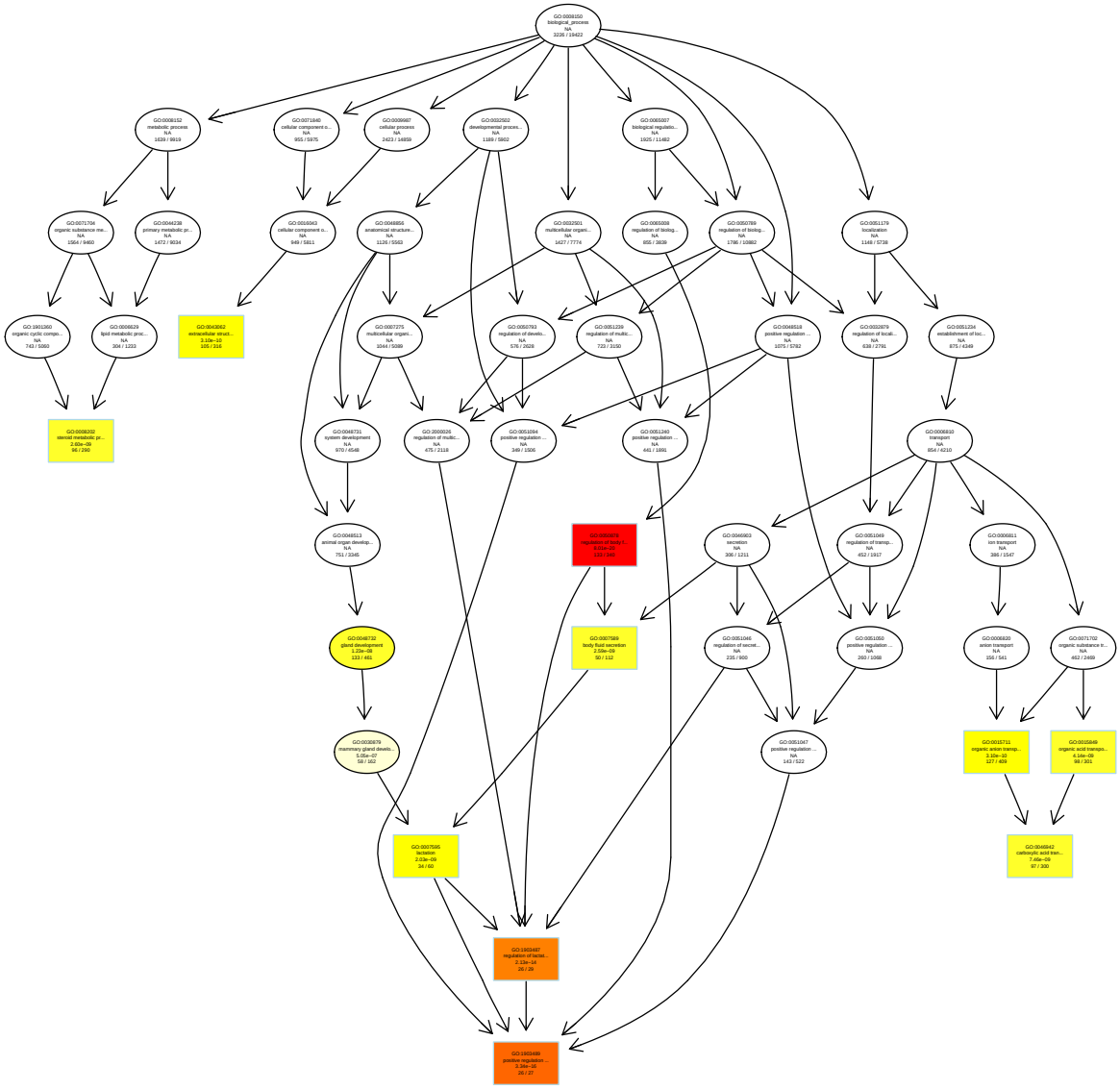

```
# The heatplot is similar to cnetplot, while displaying the relationships as a heatmap.
heatplot(ego_cc, foldChange=SigGeneList, showCategory = 10)
```




```
# Enrichment map organizes enriched terms into a network with edges connecting overlapping gene sets.
# In this way, mutually overlapping gene sets are tend to cluster together, making it easy to identify
emapplot(ego_bp, showCategory=10, color = "p.adjust")
```




```
# Show significant GO nodes
plotGOgraph(ego_bp, useFullNames = TRUE )
```



```
## $dag
## A graphNEL graph with directed edges
## Number of Nodes = 48
## Number of Edges = 81
##
## $complete.dag
## [1] "A graph with 48 nodes."
```