RNA Seq-Assignment

Viz

3/16/2019

Contents

Introduction

This is a report on RNA-seq analysis of the data Spry1,2,4 fl/fl Unsync (VEC_U) and Spry1,2,4 -/- Unsync (CRE_U), using Salmon/tximport/DESeq2 pipeline. We look for differences in expression between the Spry wild-type and knockout in the unsynchronised state.

Indexing

The genome is indexed using salmon index

```
salmon index -t Mus_musculus.GRCm38.cdna.all.fa.gz -i index_file
```

where, -t is target transcriptome -i is index file

Salmon Quantification

Salmon quant is run using a bash script.

```
#!/bin/bash
for i in *_1.fastq;
do
        prefix=$(basename $i _1.fastq)
        /data4/nextgen2018/data/rna_seq/assign/salmon quant -i index -l A -r ${prefix}_1.fastq -o quant/
done
```

where, -i is index file created in previous step -l A tells salmon it should automatically determine the library type of the sequencing reads. -r tells salmon the reads -o specifies directory where salmons quantification results go

The resulting quant/ folder is copied to local machine.

Analysis in R

The libraries are imported for processing the quantified samples

```
library(tximport)
library(GenomicFeatures)
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

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

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

```
## Loading required package: S4Vectors
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'S4Vectors'
```

```
## The following object is masked from 'package:base':
##
##     expand.grid
```

```
## Loading required package: IRanges
```

```
## Loading required package: GenomeInfoDb
```

```
## Loading required package: GenomicRanges
```

```
## Loading required package: AnnotationDbi
```

```
## 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")'.
```

```
library(readr)
```

tximport: Can import salmon's transcript-level quantifications and aggregate them to the gene level for gene-level differential expression analysis. GenomicsFeatures: Tools for making and manipulating transcript centric annotations. readr: Provides a fast and friendly way to read rectangular data.

```
txdb <- makeTxDbFromGFF("/home/rstudio/Data/Mus_musculus.GRCm38.95.chr.gtf.gz")
```

```
## Import genomic features from the file as a GRanges object ... OK
## Prepare the 'metadata' data frame ... OK
## Make the TxDb object ...
```

```
## Warning in .get_cds_IDX(type, phase): The "phase" metadata column contains non-NA values for features
##   type stop_codon. This information was ignored.
```

```
## OK
```

```
#makeTxdb makes Txdb object from annotation at GTF file
k <- keys(txdb, keytype = "TXNAME")
tx2gene <- select(txdb, k, "GENEID", "TXNAME") #selects GENEID and TXNAME to be displayed
```

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

```
head(tx2gene)
```

```
##               TXNAME             GENEID
## 1 ENSMUST00000193812 ENSMUSG00000102693
## 2 ENSMUST00000082908 ENSMUSG00000064842
## 3 ENSMUST00000192857 ENSMUSG00000102851
## 4 ENSMUST00000161581 ENSMUSG00000089699
## 5 ENSMUST00000192183 ENSMUSG00000103147
## 6 ENSMUST00000193244 ENSMUSG00000102348
```

TxDb class is a container for storing transcript annotations. It maps untranslated regions, protein coding sequences and exons from mRNA transcripts to their associated genome

```
samples <- read.table("/home/rstudio/Data/quant/samples.txt", header = TRUE)
samples
```

```
##       sample condition
## 1 SRR1658038       VEC
## 2 SRR1658041       CRE
## 3 SRR1658044       VEC
## 4 SRR1658047       CRE
## 5 SRR1658050       VEC
## 6 SRR1658053       CRE
```

```
files <- file.path("/home/rstudio/Data/quant", samples$sample, "quant.sf")
names(files) <- paste0(samples$sample) #pastes the two together, paste0 is just paste with sep ""
txi.salmon <- tximport(files, type = "salmon", tx2gene = tx2gene,ignoreTxVersion = TRUE) #Import and su
```

```
## reading in files with read_tsv
```

```
## 1 2 3 4 5 6
## transcripts missing from tx2gene: 1680
## summarizing abundance
## summarizing counts
## summarizing length
```

```
head(txi.salmon$counts)
```

```
##                     SRR1658038 SRR1658041 SRR1658044 SRR1658047 SRR1658050
## ENSMUSG00000000001 5620.00000  7785.0000 5240.00000  8216.0000 5516.00000
## ENSMUSG00000000003    0.00000     0.0000    0.00000     0.0000    0.00000
## ENSMUSG00000000028 1063.41075  1442.2765 1010.86283  1503.8199 1159.73250
## ENSMUSG00000000037   22.99996   171.0000   27.99997   181.0000   23.00003
## ENSMUSG00000000049    6.00000     2.0000    4.00000     0.0000    5.00000
## ENSMUSG00000000056  816.99991   583.9999  736.00024   583.9999  812.00009
##                     SRR1658053
## ENSMUSG00000000001  8832.0000
## ENSMUSG00000000003     0.0000
## ENSMUSG00000000028  1585.2850
## ENSMUSG00000000037   215.9996
## ENSMUSG00000000049     6.0000
## ENSMUSG00000000056   614.0004
```

Differential Gene Expression

DESeq2: This package estimates variance-mean dependence in count data and tests for differential expression based on a model using the negative binomial distribution.

```
library(DESeq2)
```

```
## Loading required package: SummarizedExperiment
## Loading required package: DelayedArray
## Loading required package: matrixStats
##
## Attaching package: 'matrixStats'
## The following objects are masked from 'package:Biobase':
##
##     anyMissing, rowMedians
## Loading required package: BiocParallel
##
## Attaching package: 'DelayedArray'
## The following objects are masked from 'package:matrixStats':
##
##     colMaxs, colMins, colRanges, rowMaxs, rowMins, rowRanges
## The following objects are masked from 'package:base':
##
##     aperm, apply
```

The negative binomial distribution is a probability distribution that is used with discrete random variables. This type of distribution concerns the number of trials that must occur in order to have a predetermined number of successes

DESeq2 reports the number of sequence fragments that have been assigned to each gene in each sample. The inference in DESeq2 relies on an estimation of the typical relationship between the data's variance and their mean,or, equivalently, between the data's dispersion and their mean.

```
dds <- DESeqDataSetFromTximport(txi.salmon, samples, ~condition) # enforces non-negative integer values
```

```
## using counts and average transcript lengths from tximport
```

```
dds <- DESeq(dds) #calculates differential expression
```

```
## estimating size factors
## using 'avgTxLength' from assays(dds), correcting for library size
## estimating dispersions
## gene-wise dispersion estimates
## mean-dispersion relationship
## final dispersion estimates
## fitting model and testing
```

```
res <- results(dds)
summary(res)
```

```
##
## out of 25324 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 4005, 16%
## LFC < 0 (down)     : 4183, 17%
## outliers [1]       : 1, 0.0039%
## low counts [2]     : 6246, 25%
## (mean count < 2)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

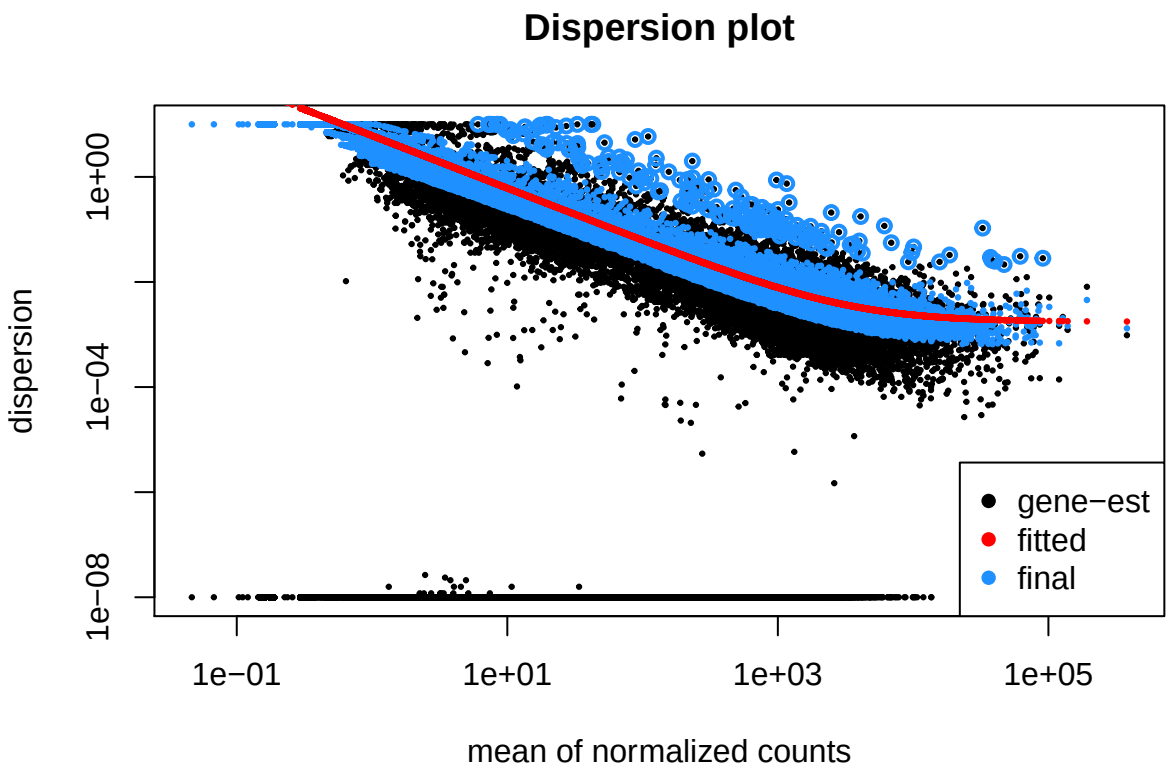
```
sig <- subset(res)
nrow(subset(sig))
```

```
## [1] 35642
```

Summary of results performs independent filtering on mean normalized counts for every gene. Out of 25324 with non-zero read count it uses an adjusted p-value <.1 and finds genes with log fold criteria less than or greater than 0, outliers, low counts and mean count. "sig" is added to find that there are 35642 significant genes.

Dispersion Plot

```
plotDispEsts(dds, main="Dispersion plot")
```



Dispersion Plot provides the per-gene dispersion estimates together with the fitted mean-dispersion relationship. Gives three Gene-ests (black dots which are gene wide dispersion estimates), final (blue dots which are the final estimates used for testing), fit which is red and are the fitted estimates. Genes in the model have high variability and that is the reason for the higher dispersion.

Rlog

For clustering and heatmaps, the data must be transformed.

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

```
##                    SRR1658038 SRR1658041 SRR1658044 SRR1658047 SRR1658050
## ENSMUSG00000000001  12.601234  12.830851  12.595433  12.829756  12.573120
## ENSMUSG00000000003   0.000000   0.000000   0.000000   0.000000   0.000000
## ENSMUSG00000000028  10.217222  10.368063  10.255740  10.367786  10.234886
## ENSMUSG00000000037   6.024650   6.682608   6.089365   6.680471   5.963573
## ENSMUSG00000000049   2.282904   2.188270   2.270730   2.180838   2.205468
## ENSMUSG00000000056   9.587200   9.371626   9.514725   9.216443   9.595873
##                    SRR1658053
## ENSMUSG00000000001  12.859020
## ENSMUSG00000000003   0.000000
## ENSMUSG00000000028  10.457074
## ENSMUSG00000000037   6.613845
## ENSMUSG00000000049   2.236178
## ENSMUSG00000000056   9.245698
```

We find that the highest differentially expressed gene is ENSMUSG00000000001, while the lowest is ENSMUSG00000000003

```
library(RColorBrewer) #Creates nice looking colour palettes
library(gplots) #Plots given data
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:IRanges':
##
##     space
```

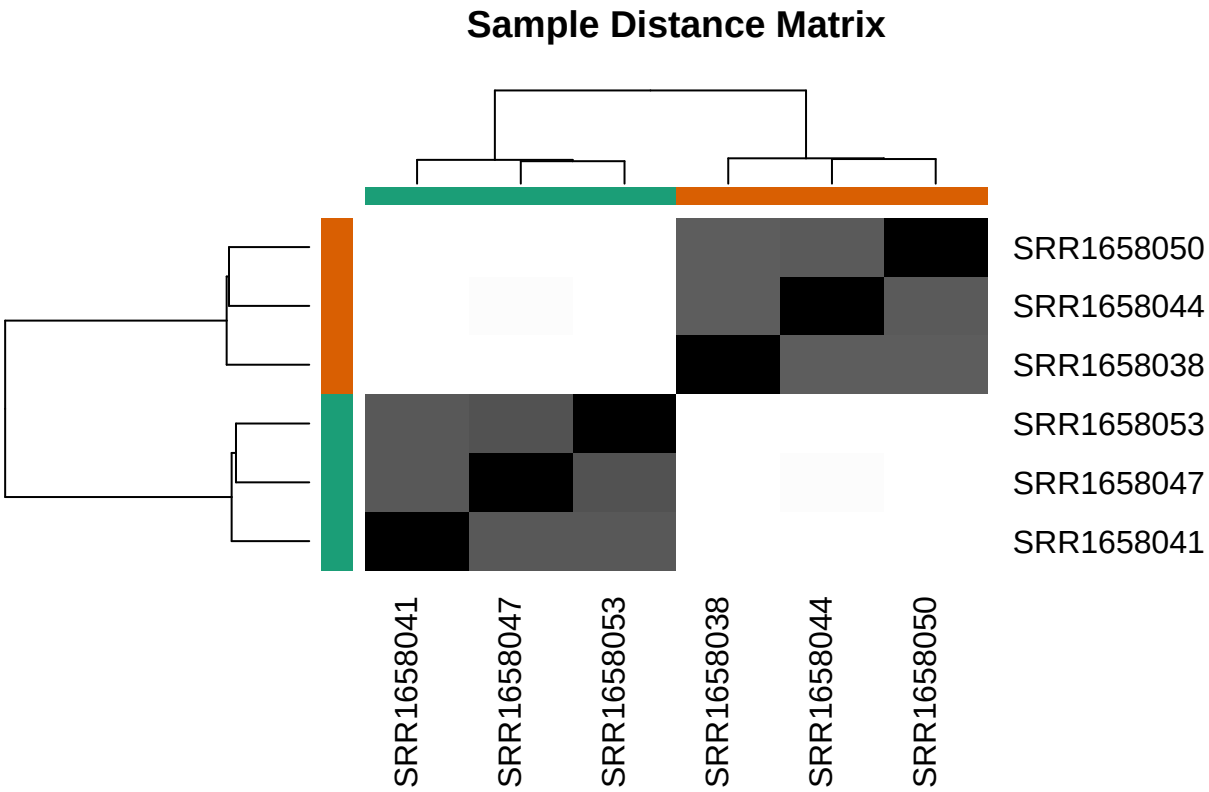
```
## The following object is masked from 'package:S4Vectors':
##
##     space
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(SummarizedExperiment)
```

Heatmaps

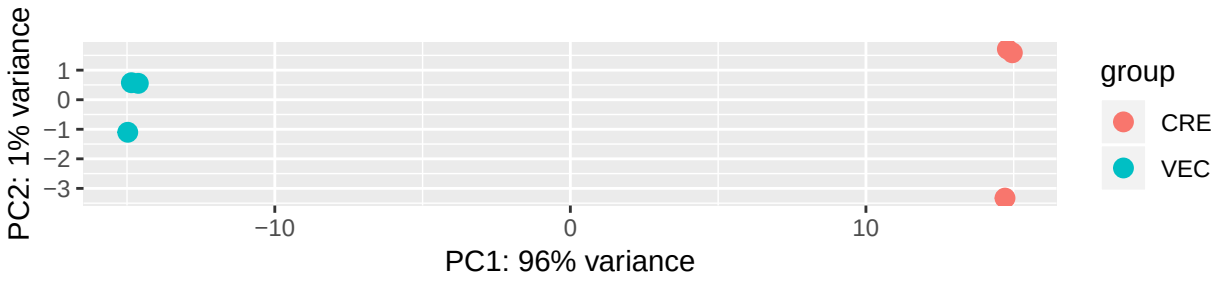
```
mycols <- brewer.pal(8, "Dark2")[1:length(unique(samples$condition))] #used for creating different colo

sampleDists <- as.matrix(dist(t(assay(rld)))) #calculates distance matrix using specified distance meas

heatmap.2(as.matrix(sampleDists), key=F, trace="none",          col=colorpanel(100, "black", "white"),
```



Heatmaps are used to illustrate our data graphically. By creating a matrix of our sampleDists, the heatmap represents all the individual values as colours. In our case, we are using the for cluster analysis of genes. Notable gene clusters are SRR1658053, SRR1658041, and SRR1658047 and also SRR1658038, SRR1658050, and SRR1658044.

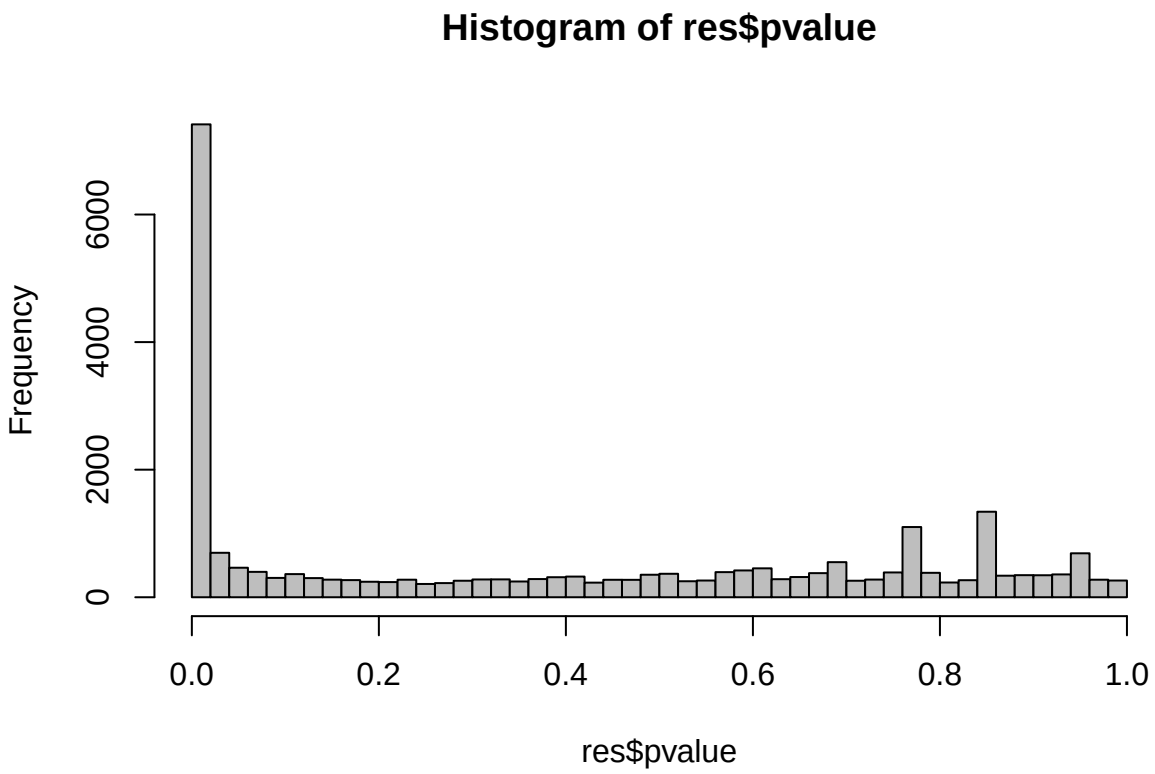
PCA

```
DESeq2::plotPCA(rld, intgroup="condition")
```



We can see that both the VEC and CRE cluster together.

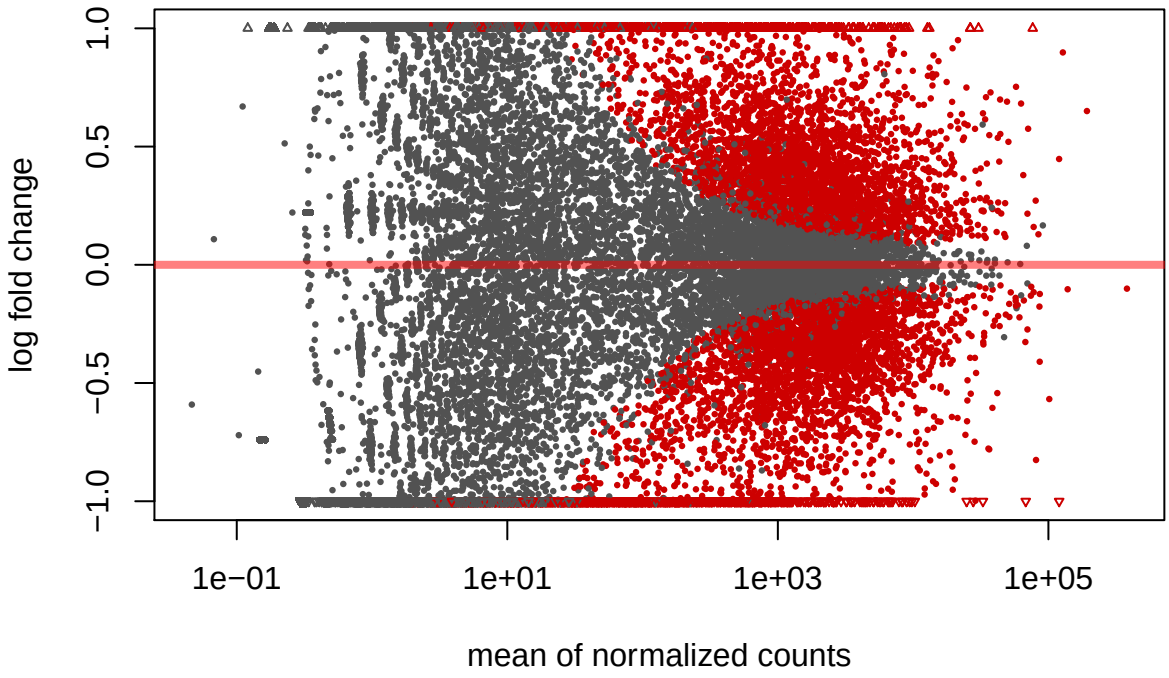
```
hist(res$pvalue, breaks=50, col="grey")
```



In the histogram, lower p-values have higher differential expression (>6000) while higher p-values have lower differential expression (<2000)

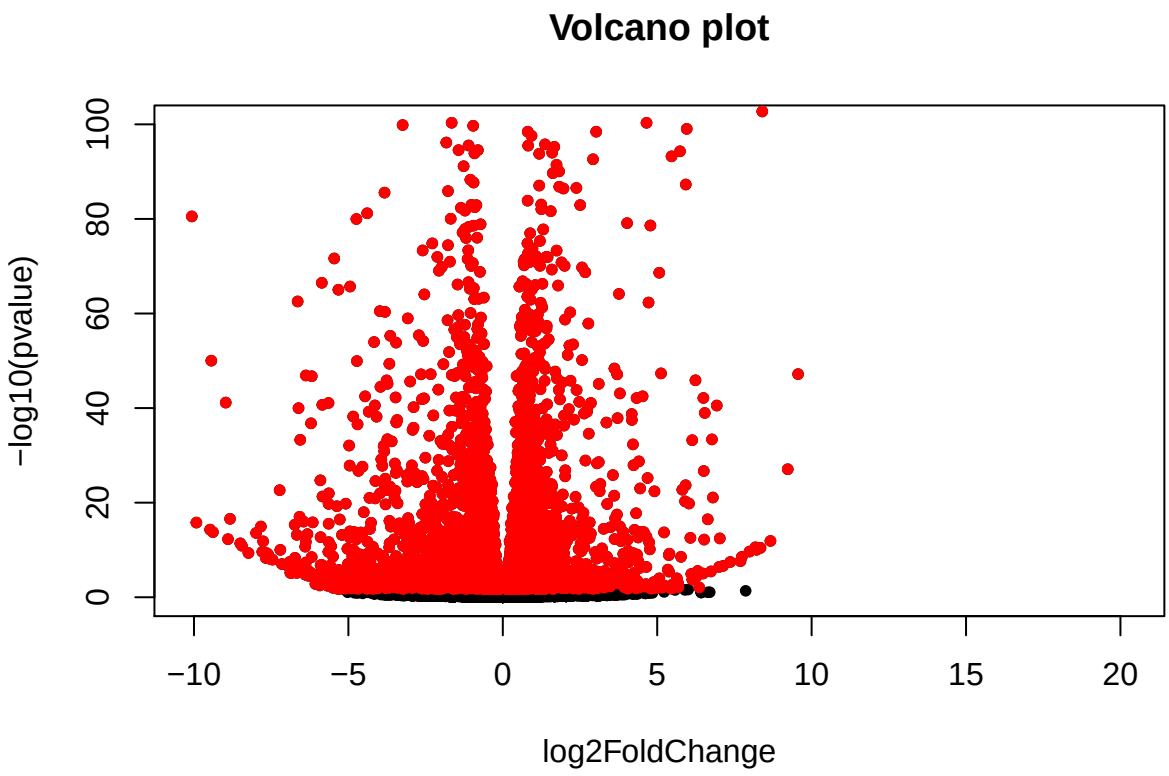
Maximum-posterior Estimate

```
DESeq2::plotMA(dds, ylim=c(-1,1))
```



This is a maximum posterior estimate plot. Shrinkage depends on log fold change, with the log fold increasing with distance from 0

```
with(res, plot(log2FoldChange, -log10(pvalue), pch=20, main="Volcano plot", ylim=c(0,100)))

with(subset(res, padj<.05 ), points(log2FoldChange, -log10(pvalue), pch=20, col="red"))
```



These are the commands for creating volcano plots, with statistical significance (p value) on the y-axis and log2foldChange on x-axis. The resulting volcano plot shows that as the log fold change increases or decreases from 0, the p-values tend to become more significant

The following command creates a table of differentially expressed genes:

```
#Table of significant results
table(res$padj<0.05) #p -values <.05
```

```
##
## FALSE  TRUE
## 11693  7384
```

```
res <- res[order(res$padj), ]
resdata <- merge(as.data.frame(res), as.data.frame(counts(dds, normalized=TRUE)), by="row.names", sort=
#merging results
names(resdata)[1] <- "Gene"
head(resdata)
```

```
##                 Gene  baseMean log2FoldChange      lfcSE      stat pvalue
## 1 ENSMUSG00000003746  8908.252       2.521490 0.05053313  49.89775      0
## 2 ENSMUSG00000019772 30432.133       2.834424 0.05615107  50.47855      0
## 3 ENSMUSG00000019970 13176.651       1.493402 0.03948411  37.82286      0
## 4 ENSMUSG00000024066  8123.058       3.118949 0.06190813  50.38027      0
## 5 ENSMUSG00000024087  6051.400      -4.013698 0.05989581 -67.01133      0
## 6 ENSMUSG00000024512  1415.641       5.533572 0.12544818  44.11043      0
##   padj SRR1658038  SRR1658041 SRR1658044  SRR1658047 SRR1658050
## 1    0 14889.0546  2774.23431 15169.5163  2521.12495 15461.2652
```

```
## 2    0 53132.3432  6924.58817 54333.4825  7562.60753 52677.4269
## 3    0 19752.9525  6926.42507 18849.9389  6909.89249 19735.8952
## 4    0 14918.8042  1591.00879 14529.1806  1824.36378 14257.8685
## 5    0   713.2213 11141.44830   740.0855 11338.77735   664.8198
## 6    0  2848.5042    56.51697  2708.2738    61.79139  2757.7442
##    SRR1658053
## 1  2634.31911
## 2  7962.35182
## 3  6884.80000
## 4  1617.12318
## 5 11710.04882
## 6    61.01295
```

Pathway Analysis

```
library("AnnotationDbi") #Provides user interface + data connection
library("org.Mm.eg.db") #Genome wide annotation for mouse
```

```
##
```

```
library(pathview) #Used for data integration and visualization
```

```
## Loading required package: org.Hs.eg.db
```

```
##
```

```
## ############################################################################
## Pathview is an open source software package distributed under GNU General
## Public License version 3 (GPLv3). Details of GPLv3 is available at
## http://www.gnu.org/licenses/gpl-3.0.html. Particullary, users are required to
## formally cite the original Pathview paper (not just mention it) in publications
## or products. For details, do citation("pathview") within R.
##
## The pathview downloads and uses KEGG data. Non-academic uses may require a KEGG
## license agreement (details at http://www.kegg.jp/kegg/legal.html).
## ############################################################################
```

```
library(gage) #Used for pathway analysis
library(gageData) #Used for visualizing gage
```

```
columns(org.Mm.eg.db)
```

```
##  [1] "ACCNUM"       "ALIAS"        "ENSEMBL"      "ENSEMBLPROT"
##  [5] "ENSEMBLTRANS" "ENTREZID"     "ENZYME"       "EVIDENCE"
##  [9] "EVIDENCEALL"  "GENENAME"     "GO"           "GOALL"
## [13] "IPI"          "MGI"          "ONTOLOGY"     "ONTOLOGYALL"
## [17] "PATH"         "PFAM"         "PMID"         "PROSITE"
## [21] "REFSEQ"       "SYMBOL"       "UNIGENE"      "UNIPROT"
```

This gives all the available column in the genome wide mouse annotation

```
head(res)
```

```
## log2 fold change (MLE): condition VEC vs CRE
## Wald test p-value: condition VEC vs CRE
## DataFrame with 6 rows and 6 columns
##                            baseMean    log2FoldChange             lfcSE
```

```
##                          <numeric>         <numeric>          <numeric>
## ENSMUSG00000003746 8908.25242397344  2.52148963740489  0.050533130628039
## ENSMUSG00000019772 30432.1333510323  2.83442424242077 0.0561510674774333
## ENSMUSG00000019970  13176.650697085  1.49340224150888 0.0394841137478273
## ENSMUSG00000024066  8123.0581766285  3.11894855725278 0.0619081321134631
## ENSMUSG00000024087 6051.40017214858 -4.01369807744582 0.0598958101389703
## ENSMUSG00000024512  1415.6405886606  5.53357232820359  0.125448175174428
##                               stat    pvalue      padj
##                          <numeric> <numeric> <numeric>
## ENSMUSG00000003746  49.8977523471663         0         0
## ENSMUSG00000019772  50.4785459966528         0         0
## ENSMUSG00000019970  37.8228634191151         0         0
## ENSMUSG00000024066  50.3802723612542         0         0
## ENSMUSG00000024087 -67.0113329819437         0         0
## ENSMUSG00000024512  44.1104250461155         0         0
```

The top 6 genes are the most significantly differentially expressed.

We now map the geneIDs to their symbol, entrez ID and gene names:

```
res$symbol <- mapIds(org.Mm.eg.db,keys=row.names(res),column="SYMBOL",keytype="ENSEMBL", multiVals="firs
```

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

```
res$entrez <- mapIds(org.Mm.eg.db,keys=row.names(res),column="ENTREZID",keytype="ENSEMBL", multiVals="fi
```

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

```
res$name <- mapIds(org.Mm.eg.db,keys = row.names(res),column = "GENENAME",keytype = "ENSEMBL",multiVals
```

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

```
head(res)
```

```
## log2 fold change (MLE): condition VEC vs CRE
## Wald test p-value: condition VEC vs CRE
## DataFrame with 6 rows and 9 columns
##                          baseMean    log2FoldChange             lfcSE
##                         <numeric>         <numeric>         <numeric>
## ENSMUSG00000003746 8908.25242397344  2.52148963740489  0.050533130628039
## ENSMUSG00000019772 30432.1333510323  2.83442424242077 0.0561510674774333
## ENSMUSG00000019970  13176.650697085  1.49340224150888 0.0394841137478273
## ENSMUSG00000024066  8123.0581766285  3.11894855725278 0.0619081321134631
## ENSMUSG00000024087 6051.40017214858 -4.01369807744582 0.0598958101389703
## ENSMUSG00000024512  1415.6405886606  5.53357232820359  0.125448175174428
##                               stat    pvalue      padj      symbol
##                          <numeric> <numeric> <numeric> <character>
## ENSMUSG00000003746  49.8977523471663         0         0       Man1a
## ENSMUSG00000019772  50.4785459966528         0         0         Vip
## ENSMUSG00000019970  37.8228634191151         0         0        Sgk1
## ENSMUSG00000024066  50.3802723612542         0         0         Xdh
## ENSMUSG00000024087 -67.0113329819437         0         0      Cyp1b1
## ENSMUSG00000024512  44.1104250461155         0         0       Dynap
##                         entrez
##                    <character>
## ENSMUSG00000003746       17155
## ENSMUSG00000019772       22353
## ENSMUSG00000019970       20393
```

```
## ENSMUSG00000024066       22436
## ENSMUSG00000024087       13078
## ENSMUSG00000024512       75577
##                                                                    name
##                                                             <character>
## ENSMUSG00000003746                                  mannosidase 1, alpha
## ENSMUSG00000019772                    vasoactive intestinal polypeptide
## ENSMUSG00000019970                serum/glucocorticoid regulated kinase 1
## ENSMUSG00000024066                                 xanthine dehydrogenase
## ENSMUSG00000024087 cytochrome P450, family 1, subfamily b, polypeptide 1
## ENSMUSG00000024512                           dynactin associated protein
```

Differentially expressed are mannosidase 1, alpha vasoactive intestinal polypeptide, serum/glucocorticoid regulated kinase 1, xanthine dehydrogenase cytochrome P450, family 1, subfamily b, polypeptide 1, dynactin associated protein.

KEGG pathways

The gageData package has pre-compiled databases mapping genes to KEGG pathways and GO terms for comprehensively studied genomes, in our case mouse.

```
data(kegg.sets.mm)
data(sigmet.idx.mm)
kegg.sets.mm <- kegg.sets.mm[sigmet.idx.mm]
head(kegg.sets.mm, 3)
```

```
## $`mmu04144 Endocytosis`
##   [1] "100017"    "100039024" "100044874" "100045864" "103967"
##   [6] "106572"    "106952"    "107305"    "107568"    "109689"
##  [11] "110355"    "110557"    "110696"    "11554"     "11555"
##  [16] "11556"     "116733"    "11771"     "11772"     "11773"
##  [21] "11845"     "11848"     "12389"     "12390"     "12391"
##  [26] "12402"     "12540"     "12757"     "12765"     "12767"
##  [31] "12774"     "12978"     "13132"     "13196"     "13429"
##  [36] "13430"     "13645"     "13649"     "13660"     "13854"
##  [41] "13855"     "13858"     "13867"     "13869"     "140500"
##  [46] "14062"     "14183"     "14184"     "14186"     "14254"
##  [51] "14275"     "14276"     "14772"     "14773"     "14963"
##  [56] "14964"     "14972"     "14985"     "14990"     "14991"
##  [61] "14997"     "15006"     "15007"     "15013"     "15018"
##  [66] "15019"     "15024"     "15039"     "15040"     "15042"
##  [71] "15043"     "15051"     "15239"     "15461"     "15481"
##  [76] "15482"     "15511"     "15512"     "16001"     "16184"
##  [81] "16185"     "16186"     "16396"     "16542"     "16590"
##  [86] "16835"     "17126"     "17127"     "17130"     "17131"
##  [91] "17246"     "17295"     "17999"     "18211"     "18571"
##  [96] "18595"     "18717"     "18719"     "18720"     "18759"
## [101] "18762"     "18805"     "18806"     "18854"     "19326"
## [106] "19334"     "19341"     "19344"     "19345"     "19349"
## [111] "193740"    "194309"    "19713"     "20404"     "20405"
## [116] "20408"     "20479"     "20779"     "208092"    "20844"
## [121] "208650"    "211914"    "212285"    "213990"    "215445"
## [126] "215632"    "216238"    "216439"    "216724"    "216859"
## [131] "216869"    "216963"    "21803"     "21808"     "21809"
```

```
## [136] "21812"     "21813"     "218441"    "22034"     "22042"
## [141] "22088"     "22365"     "224753"    "224754"    "224756"
## [146] "224761"    "227288"    "227700"    "227733"    "228998"
## [151] "230597"    "230837"    "232227"    "232910"    "234353"
## [156] "234852"    "24013"     "243621"    "245666"    "259300"
## [161] "26385"     "26431"     "268451"    "271457"    "27681"
## [166] "28084"     "330192"    "333715"    "347722"    "399549"
## [171] "433749"    "52055"     "52348"     "53869"     "54189"
## [176] "54673"     "547349"    "56324"     "56513"     "57440"
## [181] "58194"     "58220"     "64931"     "66201"     "66251"
## [186] "66313"     "66371"     "66700"     "66914"     "67028"
## [191] "67064"     "67300"     "67588"     "68942"     "68953"
## [196] "69710"     "69780"     "70160"     "70527"     "71770"
## [201] "71889"     "72543"     "72685"     "73711"     "73728"
## [206] "74002"     "74006"     "74325"     "74998"     "75608"
## [211] "75767"     "75788"     "76959"     "77038"     "78287"
## [216] "78618"     "80794"     "83814"     "84092"     "93737"
## [221] "93742"     "98366"     "98878"
##
## $`mmu03008 Ribosome biogenesis in eukaryotes`
##  [1] "100019"    "100044829" "100045148" "100045968" "100045999"
##  [6] "100047957" "100503708" "100505330" "101592"    "102462"
## [11] "102614"    "103573"    "104444"    "105372"    "108143"
## [16] "110816"    "117109"    "12995"     "13000"     "13001"
## [21] "14000"     "14113"     "14791"     "16418"     "170722"
## [26] "17724"     "17725"     "19384"     "19428"     "195434"
## [31] "20826"     "208366"    "213773"    "213895"    "21453"
## [36] "216987"    "217109"    "21771"     "217995"    "224092"
## [41] "225348"    "227522"    "230082"    "230737"    "237082"
## [46] "237107"    "24127"     "24128"     "245474"    "245610"
## [51] "269470"    "27993"     "30877"     "434401"    "52530"
## [56] "53319"     "54364"     "55989"     "56488"     "57815"
## [61] "59028"     "633406"    "633966"    "66161"     "66181"
## [66] "66711"     "66932"     "67045"     "67134"     "67459"
## [71] "67619"     "67724"     "67973"     "68147"     "68272"
## [76] "69237"     "69961"     "71340"     "72515"     "72554"
## [81] "73674"     "73736"     "74097"     "74778"     "75471"
## [86] "83454"     "97112"     "98956"
##
## $`mmu04141 Protein processing in endoplasmic reticulum`
##   [1] "100037258" "100041121" "100042561" "100043027" "100043039"
##   [6] "100046078" "100046302" "100504754" "103963"    "105245"
##  [11] "107513"    "108687"    "109815"    "110379"    "110616"
##  [16] "116891"    "11911"     "12017"     "12018"     "12028"
##  [21] "12043"     "12282"     "12304"     "12317"     "12330"
##  [26] "12333"     "12334"     "12364"     "12915"     "12954"
##  [31] "12955"     "13002"     "13135"     "13198"     "13200"
##  [36] "13418"     "13665"     "13666"     "140499"    "140740"
##  [41] "14376"     "14827"     "14828"     "15467"     "15481"
##  [46] "15482"     "15502"     "15505"     "15511"     "15512"
##  [51] "15516"     "15519"     "16430"     "17155"     "17156"
##  [56] "17872"     "18024"     "18415"     "18453"     "18786"
##  [61] "19089"     "19106"     "192193"    "19358"     "19359"
##  [66] "193740"    "20014"     "20224"     "20334"     "20335"
```

```
##  [71] "20338"     "20832"     "213539"    "21402"     "216080"
##  [76] "216197"    "216440"    "217365"    "218793"    "218811"
##  [81] "22027"     "22030"     "22193"     "22194"     "22213"
##  [86] "22230"     "223455"    "22393"     "22433"     "226418"
##  [91] "226641"    "227619"    "230815"    "230904"    "231098"
##  [96] "235416"    "23802"     "240667"    "244178"    "244179"
## [101] "26400"     "26408"     "26414"     "26419"     "26420"
## [106] "269523"    "26965"     "27054"     "27061"     "270669"
## [111] "27103"     "320011"    "386649"    "50527"     "50762"
## [116] "50873"     "50907"     "53421"     "54197"     "54609"
## [121] "56085"     "56228"     "56424"     "56438"     "56445"
## [126] "56453"     "56550"     "56709"     "56812"     "57377"
## [131] "57743"     "59007"     "632883"    "63958"     "64209"
## [136] "66105"     "66212"     "66245"     "66256"     "66326"
## [141] "66397"     "66435"     "66530"     "66753"     "66861"
## [146] "66890"     "66967"     "67128"     "67397"     "67437"
## [151] "67475"     "67819"     "67838"     "68292"     "69162"
## [156] "69276"     "69608"     "70361"     "70377"     "71853"
## [161] "72265"     "74126"     "75744"     "77371"     "78943"
## [166] "80286"     "81489"     "81500"     "81910"     "94232"
## [171] "99683"
```

```
foldchanges <- res$log2FoldChange #We want to find fold change
names(foldchanges) <- res$entrez #replaces entrezID wth gene names
keggres <- gage(foldchanges, gsets=kegg.sets.mm, same.dir=TRUE)
lapply(keggres, head)
```

```
## $greater
##                                                      p.geomean stat.mean
## mmu00860 Porphyrin and chlorophyll metabolism        0.03124850  1.892618
## mmu04370 VEGF signaling pathway                      0.05260698  1.630050
## mmu03010 Ribosome                                    0.08297274  1.395881
## mmu04141 Protein processing in endoplasmic reticulum 0.09594830  1.308458
## mmu00591 Linoleic acid metabolism                    0.09812319  1.302207
## mmu04620 Toll-like receptor signaling pathway        0.10336279  1.267021
##                                                           p.val     q.val
## mmu00860 Porphyrin and chlorophyll metabolism        0.03124850 0.9959051
## mmu04370 VEGF signaling pathway                      0.05260698 0.9959051
## mmu03010 Ribosome                                    0.08297274 0.9959051
## mmu04141 Protein processing in endoplasmic reticulum 0.09594830 0.9959051
## mmu00591 Linoleic acid metabolism                    0.09812319 0.9959051
## mmu04620 Toll-like receptor signaling pathway        0.10336279 0.9959051
##                                                      set.size       exp1
## mmu00860 Porphyrin and chlorophyll metabolism              39 0.03124850
## mmu04370 VEGF signaling pathway                            76 0.05260698
## mmu03010 Ribosome                                          88 0.08297274
## mmu04141 Protein processing in endoplasmic reticulum      164 0.09594830
## mmu00591 Linoleic acid metabolism                          45 0.09812319
## mmu04620 Toll-like receptor signaling pathway             101 0.10336279
##
## $less
##                                    p.geomean stat.mean       p.val
## mmu04976 Bile secretion          0.004094927 -2.685337 0.004094927
## mmu04360 Axon guidance           0.009797120 -2.351241 0.009797120
## mmu04510 Focal adhesion          0.013859326 -2.209523 0.013859326
```

```
## mmu04512 ECM-receptor interaction 0.022293849 -2.024417 0.022293849
## mmu04972 Pancreatic secretion     0.022952506 -2.010372 0.022952506
## mmu04540 Gap junction             0.026804675 -1.945026 0.026804675
##                                       q.val set.size        exp1
## mmu04976 Bile secretion           0.4108435       71 0.004094927
## mmu04360 Axon guidance            0.4108435      130 0.009797120
## mmu04510 Focal adhesion           0.4108435      199 0.013859326
## mmu04512 ECM-receptor interaction 0.4108435       86 0.022293849
## mmu04972 Pancreatic secretion     0.4108435      103 0.022952506
## mmu04540 Gap junction             0.4108435       87 0.026804675
##
## $stats
##                                                    stat.mean     exp1
## mmu00860 Porphyrin and chlorophyll metabolism       1.892618 1.892618
## mmu04370 VEGF signaling pathway                     1.630050 1.630050
## mmu03010 Ribosome                                   1.395881 1.395881
## mmu04141 Protein processing in endoplasmic reticulum 1.308458 1.308458
## mmu00591 Linoleic acid metabolism                   1.302207 1.302207
## mmu04620 Toll-like receptor signaling pathway       1.267021 1.267021
```

Here we use gage for our pathway analysis. Pull out the top 5 upregulated pathways, then further process that just to get the IDs. We'll use these KEGG pathway IDs downstream for plotting. The dplyr package is required to use the pipe (%>%) construct.

```
library(dplyr) #Used for manipulating data using flexible grammar
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:matrixStats':
##
##     count

## The following object is masked from 'package:AnnotationDbi':
##
##     select

## The following object is masked from 'package:Biobase':
##
##     combine

## The following objects are masked from 'package:GenomicRanges':
##
##     intersect, setdiff, union

## The following object is masked from 'package:GenomeInfoDb':
##
##     intersect

## The following objects are masked from 'package:IRanges':
##
##     collapse, desc, intersect, setdiff, slice, union

## The following objects are masked from 'package:S4Vectors':
##
##     first, intersect, rename, setdiff, setequal, union

## The following objects are masked from 'package:BiocGenerics':
##
```

```
##     combine, intersect, setdiff, union
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Get the pathways
keggrespathways <- data.frame(id=rownames(keggres$greater), keggres$greater) %>%
  tbl_df() %>%
  filter(row_number()<=5) %>%
  .$id %>%
  as.character()
keggrespathways
```

```
## [1] "mmu00860 Porphyrin and chlorophyll metabolism"
## [2] "mmu04370 VEGF signaling pathway"
## [3] "mmu03010 Ribosome"
## [4] "mmu04141 Protein processing in endoplasmic reticulum"
## [5] "mmu00591 Linoleic acid metabolism"
```

```
# Get the IDs.
keggresids <- substr(keggrespathways, start=1, stop=8)
keggresids
```

```
## [1] "mmu00860" "mmu04370" "mmu03010" "mmu04141" "mmu00591"
```

The KEGG pathway is a collection of manually drawn pathway maps that represents our knowledge on the molecular reaction, interaction, and relation networks for metabolism, genetic and environmental information processing cellular processes, human diseases and drug developemnt. Here we compare the 5 pathways with this collection and found that they were diffeentially regulated. Second part isolates pathway IDs

```
# Define plotting function for applying later
plot_pathway <- function(pid) pathview(gene.data=foldchanges, pathway.id=pid, species="mouse", new.signa

# Unload dplyr since it conflicts with the next line
detach("package:dplyr", unload=T)

# plot multiple pathways (plots saved to disk and returns a throwaway list object)
tmp <- sapply(keggresids, function(pid) pathview(gene.data=foldchanges, pathway.id=pid, species="mouse"
```

```
## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.
```

```
## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.
```

```
## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.
```

```
## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.
## 'select()' returned 1:1 mapping between keys and columns
## Info: Working in directory /home/rstudio/Data
## Info: Writing image file mmu00860.pathview.png
## 'select()' returned 1:1 mapping between keys and columns
## Info: Working in directory /home/rstudio/Data
## Info: Writing image file mmu04370.pathview.png
## 'select()' returned 1:1 mapping between keys and columns
## Info: Working in directory /home/rstudio/Data
## Info: Writing image file mmu03010.pathview.png
## 'select()' returned 1:1 mapping between keys and columns
## Info: Working in directory /home/rstudio/Data
## Info: Writing image file mmu04141.pathview.png
## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.
```

```
## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.
```

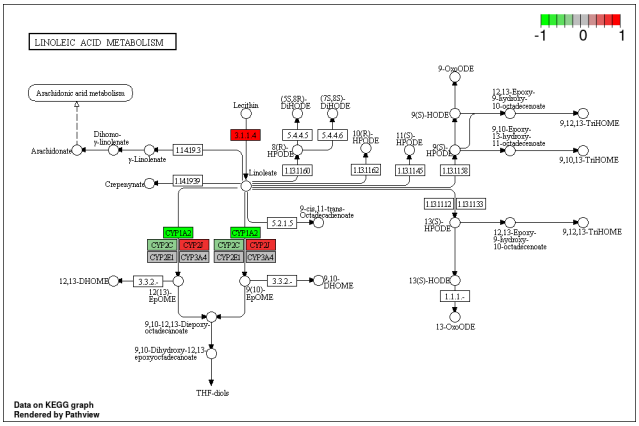


Figure 1: Dysregulated pathways

```
## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

## Warning in structure(x$children, class = "XMLNodeList"): Calling 'structure(NULL, *)' is deprecated,
##   Consider 'structure(list(), *)' instead.

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

## Info: Working in directory /home/rstudio/Data

## Info: Writing image file mmu00591.pathview.png
```

Here we obtain png images of all dysregulated pathways:

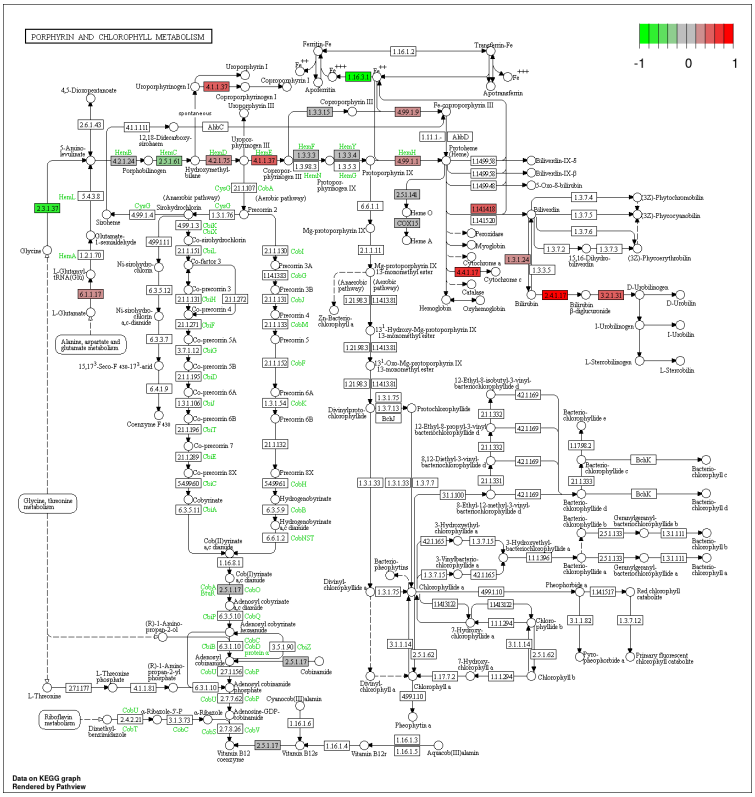


Figure 2: Dysregulated pathways

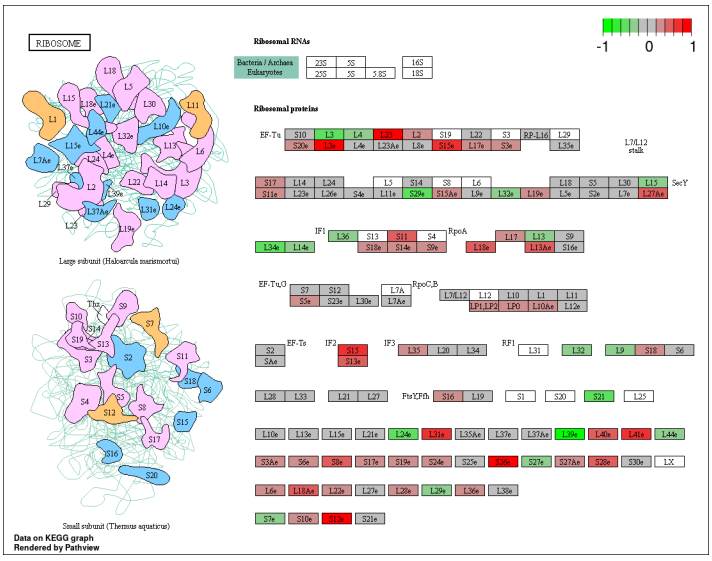


Figure 3: Dysregulated pathways

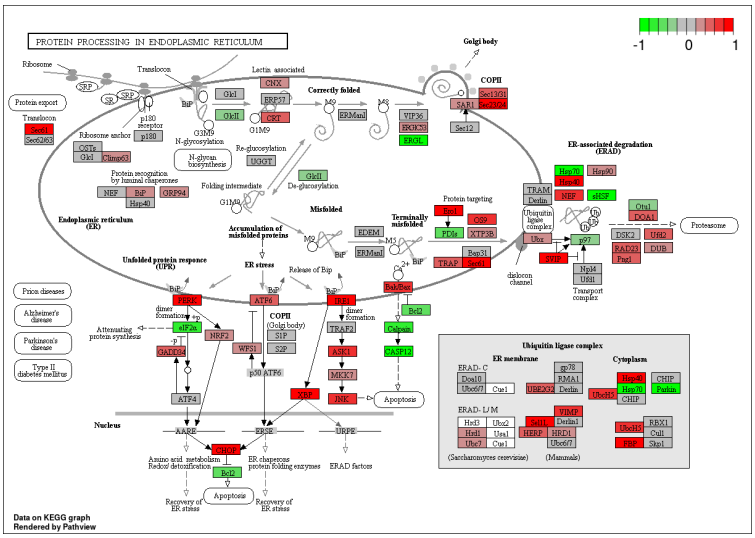


Figure 4: Dysregulated pathways

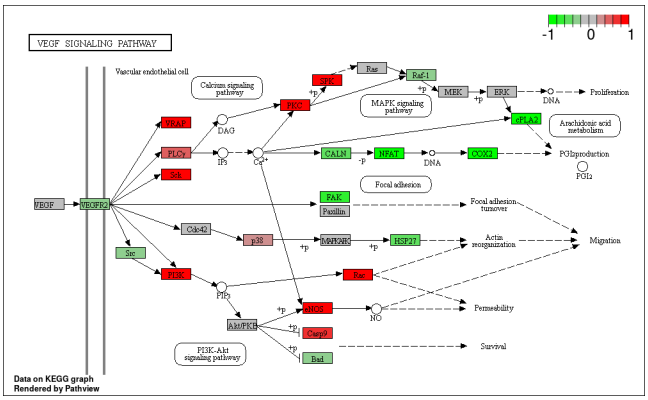


Figure 5: Dysregulated pathways