Lab13

Gretel Warmuth

Analyzing RNA sequence data from Himes et al. and the effects of desamethasone (dex) a synthetic glucocortocoid

```
counts <- read.csv("airway_scaledcounts.csv", row.names=1)
metadata <-  read.csv("airway_metadata.csv")
```

```
head(counts)
```

```
                SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
ENSG00000000003        723        486        904        445       1170
ENSG00000000005          0          0          0          0          0
ENSG00000000419        467        523        616        371        582
ENSG00000000457        347        258        364        237        318
ENSG00000000460         96         81         73         66        118
ENSG00000000938          0          0          1          0          2
                SRR1039517 SRR1039520 SRR1039521
ENSG00000000003       1097        806        604
ENSG00000000005          0          0          0
ENSG00000000419        781        417        509
ENSG00000000457        447        330        324
ENSG00000000460         94        102         74
ENSG00000000938          0          0          0
```

```
head(metadata)
```

```
          id     dex celltype     geo_id
1 SRR1039508 control   N61311 GSM1275862
2 SRR1039509 treated   N61311 GSM1275863
3 SRR1039512 control  N052611 GSM1275866
4 SRR1039513 treated  N052611 GSM1275867
5 SRR1039516 control  N080611 GSM1275870
6 SRR1039517 treated  N080611 GSM1275871
```

Q1. How many genes are in this dataset?

```
nrow(counts)
```

```
[1] 38694
```

Q2. How many 'control' cell lines do we have?

```
sum(metadata$dex == "control")
```

```
[1] 4
```

Toy Differential Expression Analysis

Calculating the mean per gene count values for all "control" samples (i.e. columns in `counts`) and do the same for "treated" and then compare:

1. Find all "control" values/columns in `counts`

```
control.inds <- metadata$dex == "control"
control.counts <- counts[, control.inds]
head(control.counts)
```

```
                SRR1039508 SRR1039512 SRR1039516 SRR1039520
ENSG00000000003        723        904       1170        806
ENSG00000000005          0          0          0          0
ENSG00000000419        467        616        582        417
ENSG00000000457        347        364        318        330
ENSG00000000460         96         73        118        102
ENSG00000000938          0          1          2          0
```

2. Calculating the mean of each gene across all control columns

```
control.mean <- apply(control.counts, 1, mean)
```

3. Do the same to find the mean for the treated columns

```
control.treat <- metadata$dex == "treated"
control.treatcount <- counts[, control.treat]
head(control.treatcount)
```

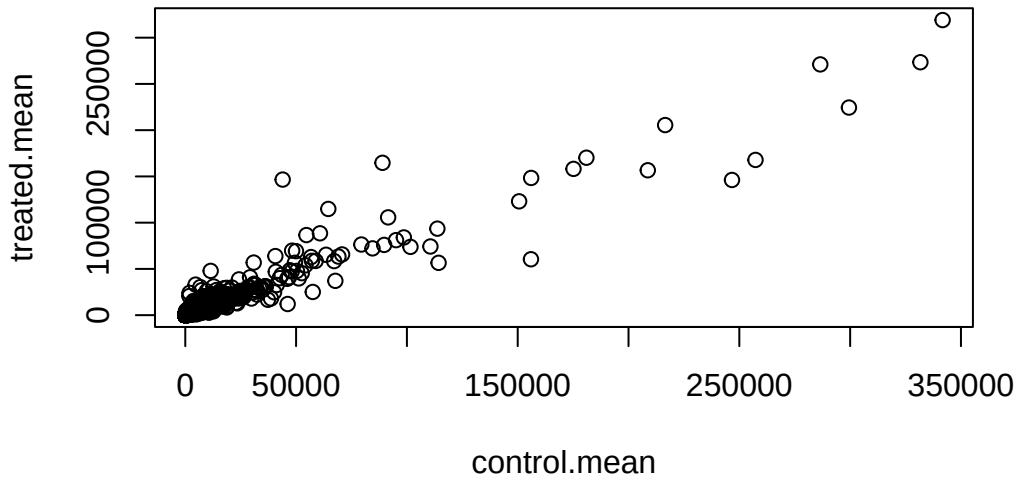
	SRR1039509	SRR1039513	SRR1039517	SRR1039521
ENSG00000000003	486	445	1097	604
ENSG00000000005	0	0	0	0
ENSG00000000419	523	371	781	509
ENSG00000000457	258	237	447	324
ENSG00000000460	81	66	94	74
ENSG00000000938	0	0	0	0

```
treated.mean <- apply(control.treatcount, 1, mean)
```

4. Plot of the means

```
mean.counts <- data.frame(control.mean, treated.mean)
```

```
plot(mean.counts)
```

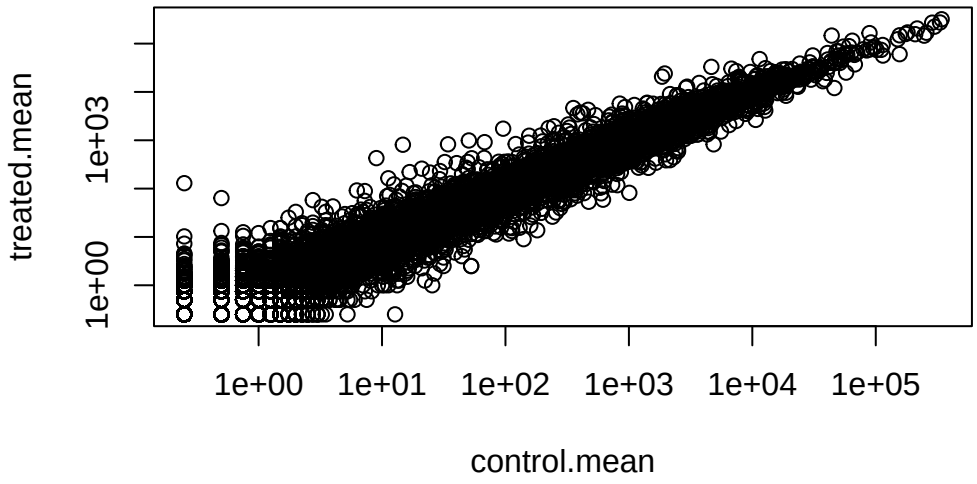


5. Plotting the log of the means

```
plot(mean.counts, log = "xy")
```

```
Warning in xy.coords(x, y, xlabel, ylabel, log): 15032 x values <= 0 omitted
from logarithmic plot
```

```
Warning in xy.coords(x, y, xlabel, ylabel, log): 15281 y values <= 0 omitted
from logarithmic plot
```



Mostly, log2 is used for this type of data:

```
log2(10/10)
```

```
[1] 0
```

```
log2(20/10)
```

```
[1] 1
```

```
log2(10/20)
```

```
[1] -1
```

These log2 values make the interpretation of a "fold-change" a little easier and a rule-of-thumb in the file is a log2 fold-change of +2 or -2 where we start to pay attention.

```
log2(40/10)
```

```
[1] 2
```

Finding the log2(fold-change) and adding it to our `mean.counts`

```
mean.counts$log2fc <- log2(mean.counts$treated.mean/mean.counts$control.mean)
head(mean.counts)
```

```
                control.mean treated.mean      log2fc
ENSG00000000003       900.75       658.00 -0.45303916
ENSG00000000005         0.00         0.00         NaN
ENSG00000000419       520.50       546.00  0.06900279
ENSG00000000457       339.75       316.50 -0.10226805
ENSG00000000460        97.25        78.75 -0.30441833
ENSG00000000938         0.75         0.00        -Inf
```

```
to.rm <- (mean.counts[,1:2] == 0) > 0
mycounts <- mean.counts[!to.rm,]
```

Q. How many genes are left after the zero count filtering?

```
nrow(mycounts)
```

```
[1] 47075
```

Q. How many genes are "up" regulated upon drug treatment (a threshold of +2 log2 fodld-change)?

1. Extract the log2fc values
2. Find those that are above +2
3. Sum them up

```
sum(mycounts$log2fc > 2)
```

```
[1] NA
```

Q. How many genes are "down" regulated upon drug treatment (a threshold of -2 log2 fodld-change)?

```
sum(mycounts$log2fc < -2)
```

```
[1] NA
```

The stats are missing. Finding a difference in the mean counts significance using the **DESeq2** package

DESeq Analysis

```
library(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, saveRDS, setdiff,
    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

Loading required package: GenomicRanges

Loading required package: GenomeInfoDb

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
```

The first dunction that will be used will set up the data in the way DESeq wants to:

```
dds <- DESeqDataSetFromMatrix(countData = counts,
                              colData = metadata,
                              design = ~dex)
```

```
converting counts to integer mode

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

The function in the package is called `DESeq()` and `dds` can be run on it

```
dds <-DESeq(dds)
```

```
estimating size factors

estimating dispersions
```

```
gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing
```

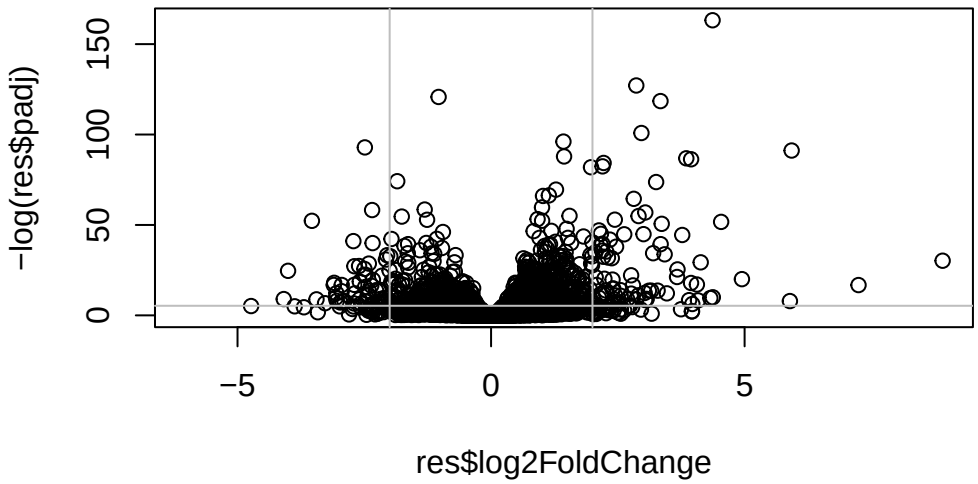
Using `results()`for dds:

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

```
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 38694 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat    pvalue
                <numeric>      <numeric> <numeric> <numeric> <numeric>
ENSG00000000003  747.1942     -0.3507030  0.168246 -2.084470 0.0371175
ENSG00000000005    0.0000             NA        NA        NA        NA
ENSG00000000419  520.1342      0.2061078  0.101059  2.039475 0.0414026
ENSG00000000457  322.6648      0.0245269  0.145145  0.168982 0.8658106
ENSG00000000460   87.6826     -0.1471420  0.257007 -0.572521 0.5669691
...                    ...            ...       ...       ...       ...
ENSG00000283115  0.000000             NA        NA        NA        NA
ENSG00000283116  0.000000             NA        NA        NA        NA
ENSG00000283119  0.000000             NA        NA        NA        NA
ENSG00000283120  0.974916      -0.668258   1.69456 -0.394354  0.693319
ENSG00000283123  0.000000             NA        NA        NA        NA
                     padj
                <numeric>
ENSG00000000003  0.163035
ENSG00000000005        NA
ENSG00000000419  0.176032
ENSG00000000457  0.961694
ENSG00000000460  0.815849
...                   ...
ENSG00000283115        NA
ENSG00000283116        NA
ENSG00000283119        NA
ENSG00000283120        NA
ENSG00000283123        NA
```

Making a common overall results figure from the analysis:

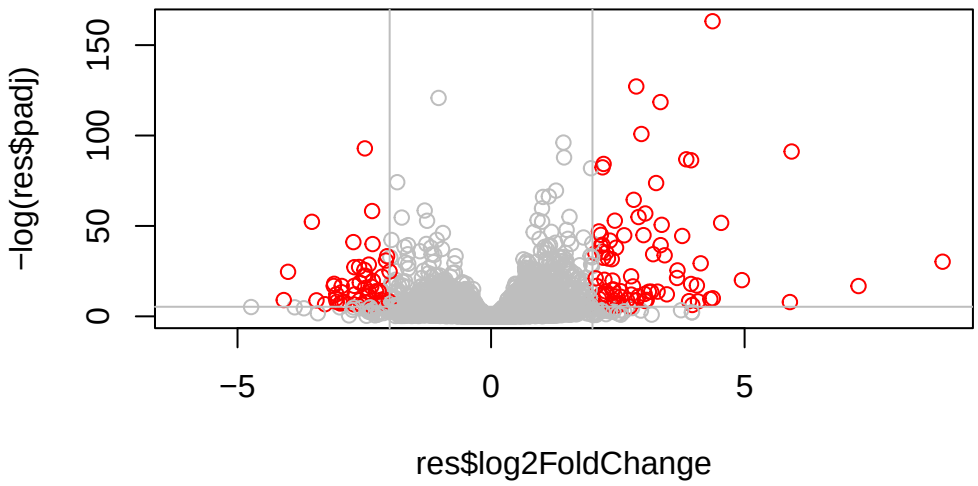
```
plot(res$log2FoldChange, -log(res$padj) )
abline(v = c(-2, 2), col = "gray")
abline(h = -log(0.005), col ="gray")
```



Adding color:

```
mycols <- rep("gray", nrow(res))
mycols[res$log2FoldChange > 2] <- "red"
mycols[res$log2FoldChange < -2] <- "red"
mycols[res$padj > 0.005] <- "gray"

plot(res$log2FoldChange, -log(res$padj), col = mycols )
abline(v = c(-2, 2), col = "gray")
abline(h = -log(0.005), col ="gray")
```



Saving results to a disc:

```
write.csv(res, file= "myresults.csv")
```

Translating gene identifiers "ENSG0000..." into gene names that are more understandable.

To do this annotation, AnnotationDBI will be used:

```
library(BiocManager)
library(stats4)
library(BiocGenerics)
library(AnnotationDbi)
library(org.Hs.eg.db)
```

Using `mapIds()`

```
head(mapIds(org.Hs.eg.db,
       keys = rownames(res),
       keytype = "ENSEMBL",
       column = "SYMBOL"))
```

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

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
       "TSPAN6"          "TNMD"          "DPM1"         "SCYL3"         "FIRRM"
ENSG00000000938
          "FGR"
```

```
write.csv(res, file = "results_annotated.csv")
```

Pathway Analysis

Pathway mapping can now be done with added annotations.

Using the **gage** package to look for KEGG pathways in the results (genes of interest). The package **pathview** will be installed to draw pathway figures.

```
library(pathview)
```

```
################################################################################
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)
```

```
library(gageData)

kegg <- data(kegg.sets.hs)
```

Need a "vector of importance" as the input for **gage**. This will be the log2FC:

```
foldchanges <- res$log2FoldChange
names(foldchanges) = res$entrez
head(foldchanges)
```

```
[1] -0.35070302          NA  0.20610777  0.02452695 -0.14714205 -1.73228897
```

Running the gage pathway analysis:

```
# Get the results
keggres = gage(foldchanges, gsets=kegg.sets.hs)
```

Attributes of `keggres`:

```
attributes(keggres)
```

```
$names
[1] "greater" "less"    "stats"
```

```
head(keggres$less, 3)
```

```
                                      p.geomean stat.mean p.val q.val
hsa00232 Caffeine metabolism                 NA       NaN    NA    NA
hsa00983 Drug metabolism - other enzymes     NA       NaN    NA    NA
hsa01100 Metabolic pathways                  NA       NaN    NA    NA
                                      set.size exp1
hsa00232 Caffeine metabolism                 0   NA
hsa00983 Drug metabolism - other enzymes     0   NA
hsa01100 Metabolic pathways                  0   NA
```

Using the pathview function to look at one of the highlighted KEGG pathways with the genes highlighted:

```
pathview(gene.data = foldchanges, pathway.id = "hsa-5310")
```

```
Info: Downloading xml files for hsa-5310, 1/1 pathways..
```

```
Warning in download.file(xml.url, xml.target, quiet = T): cannot open URL
'https://rest.kegg.jp/get/hsa-5310/kgml': HTTP status was '400 Bad Request'
```

```
Warning: Download of hsa-5310 xml file failed!
This pathway may not exist!

Info: Downloading png files for hsa-5310, 1/1 pathways..

Warning: Download of hsa-5310 png file failed!
This pathway may not exist!

Warning: Failed to download KEGG xml/png files, hsa-5310 skipped!
```