Class 12 Lab

Nickolas Beam

In today's class we will work with published RNA-Seq experiment where airway smooth muscle cells (ASMs) were treated with dexamethasone, a synthetic glucocorticoid steroid with anti-inflammatory effects (Himes et al. 2014).

Data iport

We will use good old `read.csv()` to read the two things we need for this analysis: - count data - col data (meta)

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

Q1. How many genes are in this dataset?

```
nrow(counts)
```

```
[1] 38694
```

```
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
```

First we should check the correspondence of the metadata and count data

```
metadata$id
```

```
[1] "SRR1039508" "SRR1039509" "SRR1039512" "SRR1039513" "SRR1039516"
[6] "SRR1039517" "SRR1039520" "SRR1039521"
```

```
colnames(counts)
```

```
[1] "SRR1039508" "SRR1039509" "SRR1039512" "SRR1039513" "SRR1039516"
[6] "SRR1039517" "SRR1039520" "SRR1039521"
```

To check that these are all in the same order we can use `==` test of equality.

```
all(metadata$id == colnames(counts))
```

```
[1] TRUE
```

```
all(c(T,T,T,T,T,T))
```

```
[1] TRUE
```

Analysis via comparisoon of CONTROL vs TREATED

The treated have the dex drug and the control do not. First I need to be able to extract just the "contol" columns in the `counts` data set.

```
control.inds <- metadata$dex == "control"
control <- metadata[control.inds,]
```

Now I can use this to access just the "control" columns in the `counts` data...

```
control.counts <- counts[,control$id]
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
```

Find the mean count value for each transcript/gene by binding the `rowmeans()`.

```
control.mean <- rowMeans(control.counts)
head(control.mean)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
         900.75            0.00          520.50          339.75           97.25
ENSG00000000938
           0.75
```

And noow find a mean value for all the "treated" columns in the same way

```
treated.id <- metadata[metadata$dex == "treated","id"]
treated.mean <- rowMeans(counts[,treated.id])
```

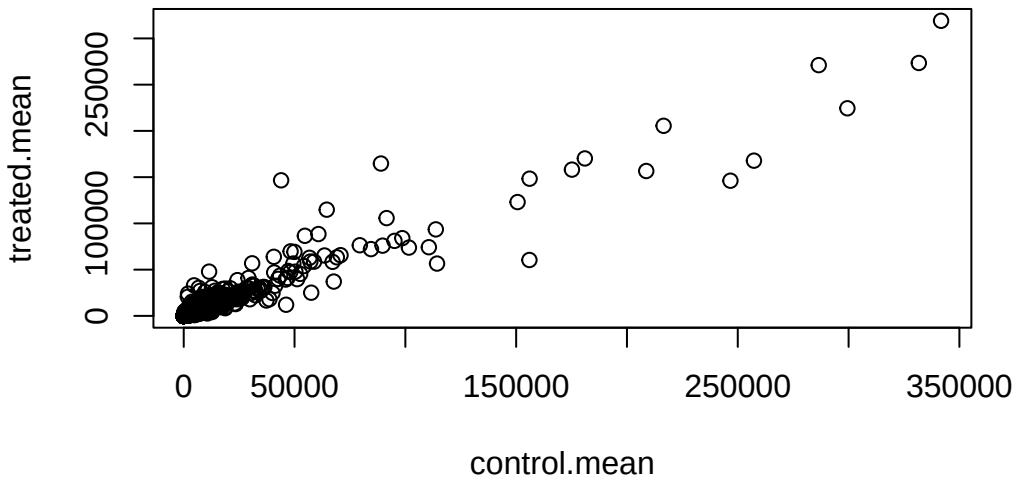
Now I hae `control.mean` and `treated.mean`. Lets put them together for safe keeping and ease of use later.

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

```
                control.mean treated.mean
ENSG00000000003       900.75       658.00
ENSG00000000005         0.00         0.00
ENSG00000000419       520.50       546.00
ENSG00000000457       339.75       316.50
ENSG00000000460        97.25        78.75
ENSG00000000938         0.75         0.00
```

Let's do a quick plot to see how our data looks

```
plot(meancounts)
```

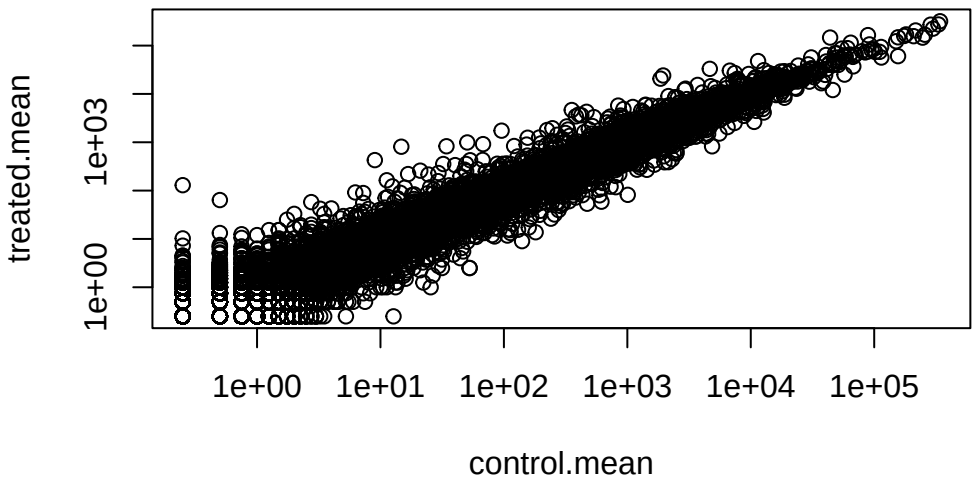


This is very heavily skewed and over a wide range - calling out for a log transform!

```
plot(meancounts, 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
```



We like working with log transformed dat aas it can help make things more straightforward to interpret.

What if we had a doubling

```
log2(40/20)
```

```
[1] 1
```

We like working with log2 fold-change values. Let's calculate them for our data.

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

We want filter out any genes (that is the rows) where we have ZERO count data.

```
to.keep.inds <- rowSums(meancounts[,1:2] == 0) == 0
```

```
mycounts <- meancounts[to.keep.inds,]
nrow(mycounts)
```

```
[1] 21817
```

A common threshold for calling genes as differentially expressed is a log2 fold-change of +2 or -2

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

```
[1] 314
```

What percent is this?

```
round((sum(mycounts$log2fc >= +2) / nrow(mycounts)) * 100,2)
```

```
[1] 1.44
```

and down regulated:

```
round((sum(mycounts$log2fc <= -2)/nrow(mycounts)) *100,2)
```

```
[1] 2.22
```

We need some stats to check if the drug induced difference is significant!

Turn to DESeq2

Let's turn to doiing this the correctt way with the DESeq2 package.

```
library(DESeq2)
```

The main function in the DESeq2 package is called `deseq()`. It wantts oour count data and oour colData (metadata) as input in a specific way.

```
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
```

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

```
estimating size factors

estimating dispersions

gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing
```

```
results(dds)
```

```
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
```

Now what we have got so far is the log2 fold-change and the adj p-value for the significance.

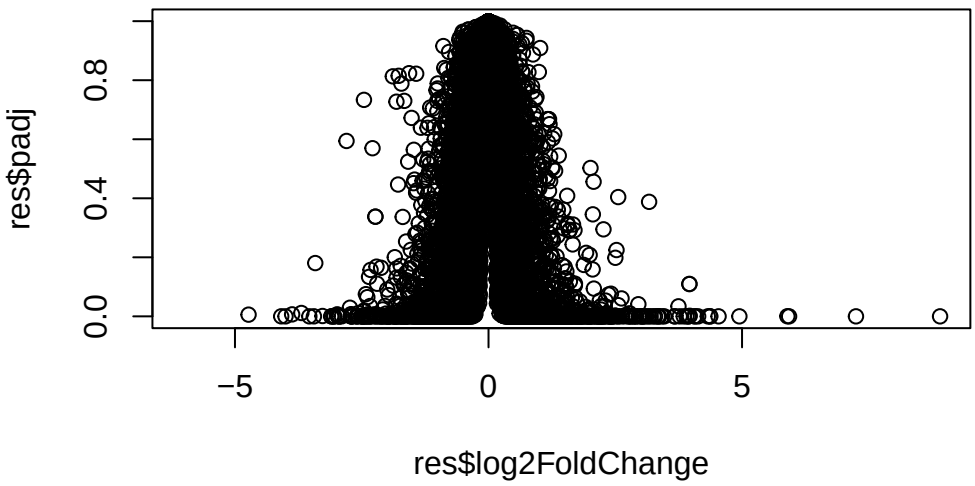
```
res <- results(dds)

head(res)
```

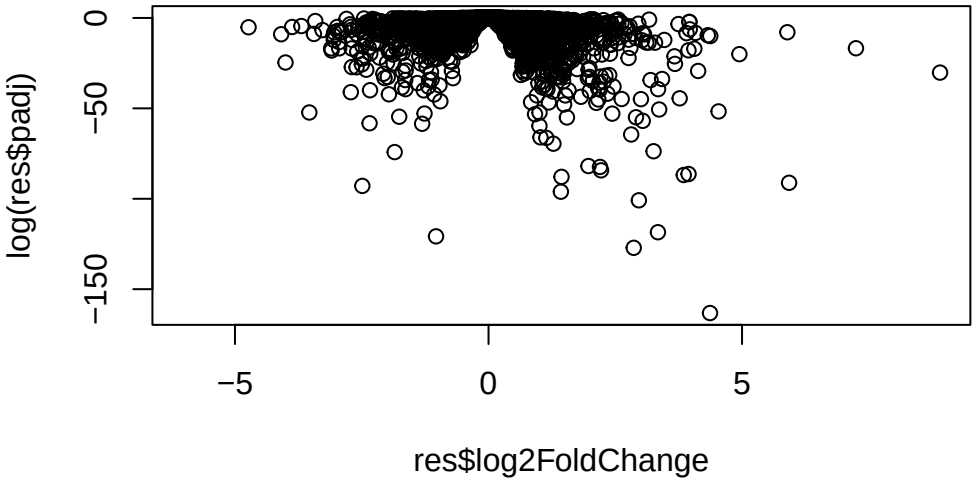
```
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 6 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat    pvalue
                <numeric>      <numeric> <numeric> <numeric> <numeric>
ENSG00000000003 747.194195     -0.3507030  0.168246 -2.084470 0.0371175
ENSG00000000005   0.000000             NA        NA        NA        NA
ENSG00000000419 520.134160      0.2061078  0.101059  2.039475 0.0414026
ENSG00000000457 322.664844      0.0245269  0.145145  0.168982 0.8658106
ENSG00000000460  87.682625     -0.1471420  0.257007 -0.572521 0.5669691
ENSG00000000938   0.319167     -1.7322890  3.493601 -0.495846 0.6200029
                     padj
                <numeric>
ENSG00000000003  0.163035
ENSG00000000005        NA
ENSG00000000419  0.176032
ENSG00000000457  0.961694
ENSG00000000460  0.815849
ENSG00000000938        NA
```

A first plot

```
plot(res$log2FoldChange, res$padj)
```




```
plot(res$log2FoldChange, log(res$padj))
```

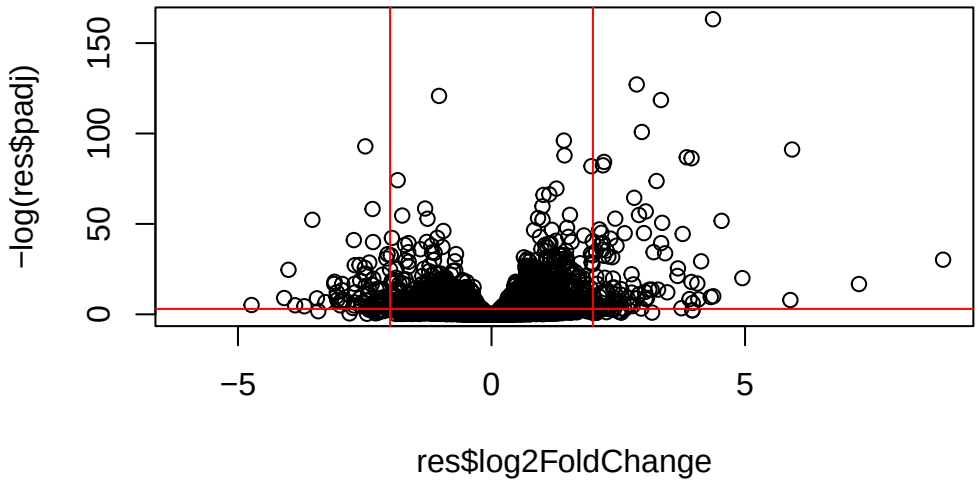


```
log(0.05)
```

```
[1] -2.995732
```

We can flip the y-axis so thte plot does not look upside down.

```
plot(res$log2FoldChange, -log(res$padj))
abline(v=c(-2,+2), col="red")
abline(h=-log(0.05), col="red")
```




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

inds <- (res$padj < 0.01) & (abs(res$log2FoldChange) > 2 )
mycols[ inds ] <- "blue"

# Volcano plot with custom colors
plot( res$log2FoldChange,  -log(res$padj),
 col=mycols, ylab="-Log(P-value)", xlab="Log2(FoldChange)" )
```

```r
# Cut-off lines
abline(v=c(-2,2), col="gray", lty=2)
abline(h=-log(0.1), col="gray", lty=2)
```