Class 13: RNASez analysis with DESeq2

Richard Gao (PID: A16490010)

In today's class we will explore and analyze data frmo a published RNA-seq experiment where airway smooth muscle cells were treated with dexamethasone, a synthetic glucocorticoid steroid with anti-inflammatory effects (Himes et al. 2014).

Data Import

We have two input files, so-called "count data" and "col data".

```r
# Complete the missing code
counts <- read.csv("airway_scaledcounts.csv", row.names=1)
metadata <-  read.csv("airway_metadata.csv")
```

Data Explore

```r
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
```

```r
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?

```r
nrow(counts)
```

```
[1] 38694
```

> 2. How many 'control' cell lines do we have?

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

```
[1] 4
```

Toy differential gene expression

Time to do some analysis.

We have 4 control and 4 treated samples/experiments/columns.

Make sure the metadata id column matches the columns in our countdata.

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

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

To check that all elements of a vector are TRUE we can use the `all()` function.

```r
all(c(T, T, T, F))
```

```
[1] FALSE
```

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

```
[1] TRUE
```

To start I will calculate the `control.mean` and `treated.mean` values and compare them.

- Identify and extract the `control` only columns
- Determine the mean value for each gene (i.e. row)
- Do the same for `treated`.

> Q3. How would you make the above code in either approach more robust? Is there a function that could help here?

apply

> Q4. Follow the same procedure for the treated samples (i.e. calculate the mean per gene across drug treated samples and assign to a labeled vector called treated.mean)

```r
# Where does it tell me which columns are control?
control.inds <- metadata$dex == "control"
control.counts <- counts[ , control.inds]
control.mean <- apply(control.counts, 1, mean)

# Where does it tell me which columns are treated?
treated.inds <- metadata$dex == "treated"
treated.counts <- counts[ , treated.inds]
treated.mean <- apply(treated.counts, 1, mean)
```

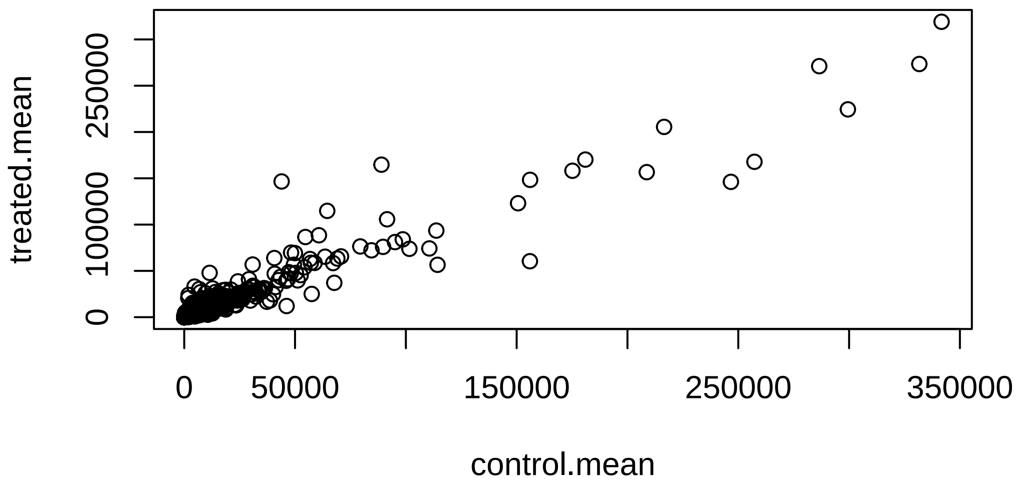
Let's store these together for ease of book-keeping

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

> Q5 (a). Create a scatter plot showing the mean of the treated samples against the mean of the control samples. Your plot should look something like the following. Q5 (b).You could also use the ggplot2 package to make this figure producing the plot below. What geom_?() function would you use for this plot?

Have a quick view of this data:

```r
plot(meancounts)
```



Would use geom_point().

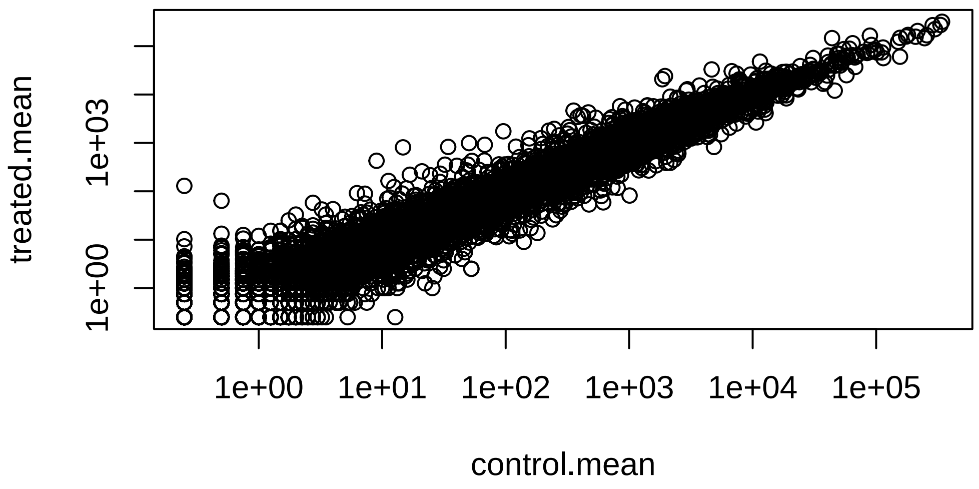
> Q6. Try plotting both axes on a log scale. What is the argument to plot() that allows you to do this?

The data is screaming at us to log transform as it is so heavily skewed and over such a wide range.

```
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
```



I want to compare the treated and the control values here and we will use fold change in log2 units to do this. log2(treated/control)

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

Why log2?

```r
# 0 indicates no change
log2(20/20)
```

```
[1] 0
```

```r
# 1 indicates a doubling in the treated
log2(20/10)
```

```
[1] 1
```

```r
# -1 indicates a halving in the treated
log2(5/10)
```

```
[1] -1
```

A common rule of thumb cut-off for calling a gene "differentially expressed" is a log2 fold-change value of either > +2 or < -2 for "up regulated" and "down regulated" respectively.

```r
head(meancounts)
```

```
                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
```

We first need to remove zero count genes - as we can't say anything about these genes anyway and their division of log values are messing things up (divide by zero) or the -infinity log problem.

```r
to.rm.ind <- rowSums(meancounts[,1:2]==0) > 0
mycounts <- meancounts[!to.rm.ind, ]
```

> Q. How many genes do we have left that we can say something about (i.e. they don't have zero counts)?

```r
nrow(mycounts)
```

```
[1] 21817
```

A common threshold used for calling something differentially expressed is a log2(FoldChange) of greater than 2 or less than -2. Let's filter the dataset both ways to see how many genes are up or down-regulated.

```r
up.ind <- mycounts$log2fc > 2
down.ind <- mycounts$log2fc < (-2)
```

> Q8. Using the up.ind vector above can you determine how many up regulated genes we have at the greater than 2 fc level?

```r
sum(up.ind)
```

```
[1] 250
```

> Q9. Using the down.ind vector above can you determine how many down regulated genes we have at the greater than 2 fc level?

```r
sum(down.ind)
```

```
[1] 367
```

> Q10. Do you trust these results? Why or why not?

No since we are comparing the fold change of means, which can be large without being statistically significant.

DESez analysis

Let's do this properly with the help of the DESeq2 package

```r
library(DESeq2)
```

We have to use a specific data object for working with DESeq2

```r
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
```

Run our main analysis with the `DESeq()` function

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

```
estimating size factors
estimating dispersions

gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing
```

To get the results out of our `dds` object we can use the DESeq function called `results()`:

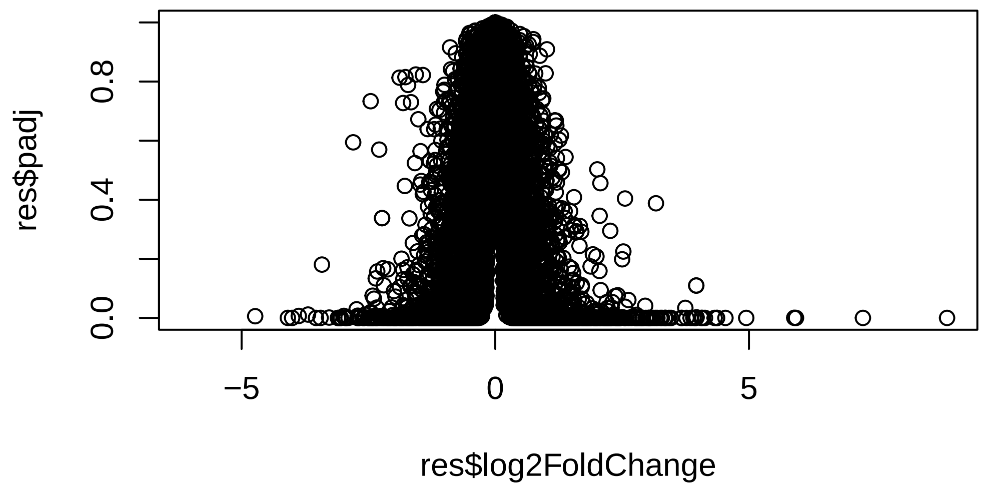
```r
res <- results(dds)
head(res)
```

```
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
```

Volcano Plot

A very common and useful summary results figure from this type of analysis is called a volcano plot - a plot of log2FC vs P-value. We use the `padj` adjusted P-value for multiple testing.

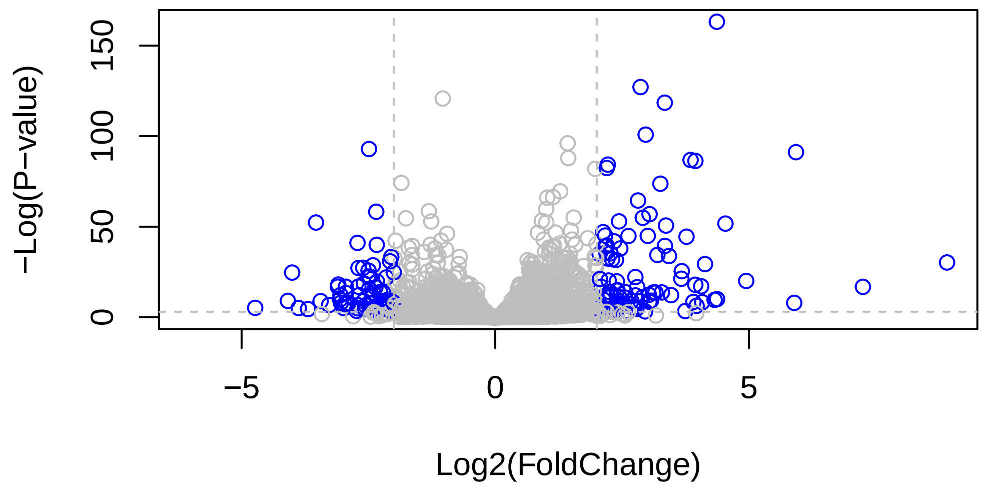
```r
plot(res$log2FoldChange, res$padj)
```



```r
# Setup our custom point color vector
mycols <- rep("gray", nrow(res))
mycols[ res$log2FoldChange > 2 ]  <- "blue"
mycols[ res$log2FoldChange < -2 ]  <- "blue"
mycols[ res$padj > 0.05] <- "gray"

plot(res$log2FoldChange, -log(res$padj),
     xlab = "Log2(FoldChange)", ylab = "-Log(P-value)",
     col = mycols)

# Cut-off lines
abline(v=c(-2,2), col="gray", lty=2)
abline(h=-log(0.05), col="gray", lty=2)
```


−Log(P−value)
0
50
100
150
−5
0
5
Log2(FoldChange)