# Class 13: Transcriptomics and analysis of RNA-Seq data

Cynthia Perez (A16393492)

The data for today's lab comes from a published RNA-seq experiment where airway smooth muscle cells were treated with dexamethasone, a synthetic glucocorticoid steroid with anti-inflammatory effects.

## BioConductor Setup

Install BiocManager in the R console using `install.packages("BiocManager")`. Then we install DESeq2 package in the R console using `BiocManager::install("DESeq2")`.

## Import countData and colData

We need two things for this analysis: counts and metadata these are called "countData" and "colData" in the DESeq2.

```
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
```

The counts are organized with a gene per row and experiment per column.

Q1. How many genes are in this dataset?

```
nrow(counts)
```

```
[1] 38694
```

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

```
table(metadata$dex)
```

```
control treated
      4       4
```

### Check on match of metadata and coldata

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

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

If you want to know that all elements of a vector are TRUE we can use the `all()` function

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

```
[1] TRUE
```

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

```
[1] FALSE
```

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

```
[1] TRUE
```

## Toy Differential Gene Expression

We will start by comparing the "control" and "treated" columns by using the means for each.

Start by extracting all "control" columns first

```
control.inds <- metadata$dex == "control"
```

```
control.counts <- counts[,control.inds]
```

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

Next we find the mean count value per gene using the `apply()` function.

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

> 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)

Then we mean count value per gene this time for the "treated" column.

```
treated.inds <- metadata$dex == "treated"
treated.counts <- counts[,treated.inds]
treated.mean <- apply(treated.counts, 1, mean)
```

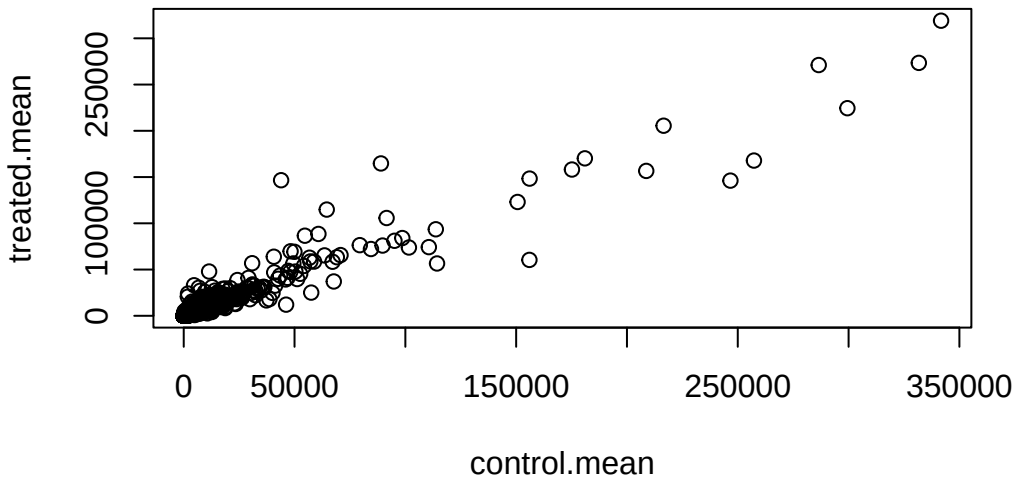
Put these two mean vectors together for ease of book-keeping.

```
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
```

Q5 (a). Create a scatter plot showing the mean of the treated samples against the mean of the control samples.

```
plot(meancounts)
```



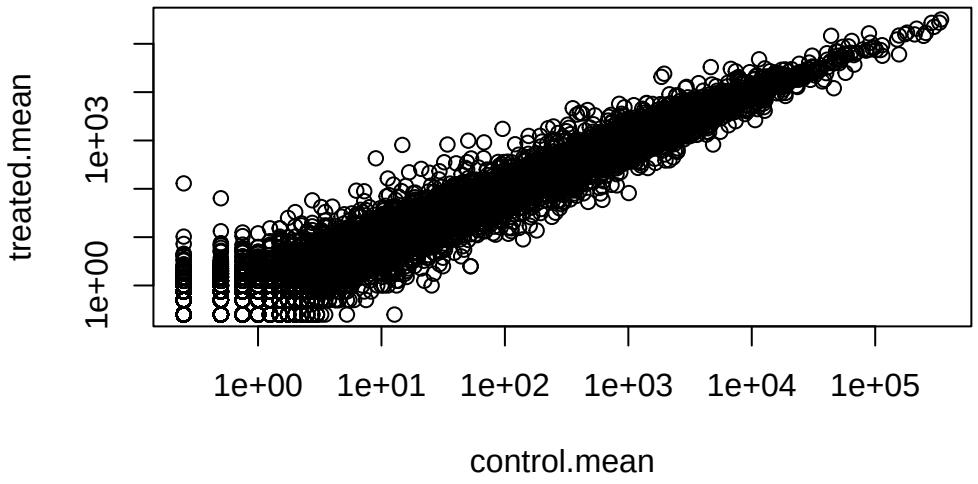

Q6. Try plotting both axes on a log scale

Use log scale to better visualize all the points

```
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
```




Often we use log2 for a better interpretation of our units. Here we calculate the log2 fold-change of treated/control values and add it to our data frame of results. Positive is up-regulated negative is down-regulated. Value of one is doubling, value of two is quadrupled.

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

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
```

There are non-numeric answers in the vector such as NA and -infinity. These are due to zero count genes found in the dataset. We must filter these zero count genes out before we can continue with our analysis.

```
# access first to columns of meancounts
# then ask which are equal to 0
# then sum them up and ask which ones are equal to 0
to.keep.inds <- (rowSums(meancounts[,1:2] == 0) == 0)

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

```
                control.mean treated.mean      log2fc
ENSG00000000003       900.75       658.00 -0.45303916
ENSG00000000419       520.50       546.00  0.06900279
ENSG00000000457       339.75       316.50 -0.10226805
ENSG00000000460        97.25        78.75 -0.30441833
ENSG00000000971      5219.00      6687.50  0.35769358
ENSG00000001036      2327.00      1785.75 -0.38194109
```

Q. how many genes do we have left after taking out zero count genes?

```
nrow(mycounts)
```

```
[1] 21817
```

A common threshold for calling a gene "up" and "down" is a log2 fold change of +2 or -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?

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

```
[1] 314
```

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

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

```
[1] 485
```

## Setting up for DESeq

We are missing the data stats. Need to account for any significant differences.

```
library(DESeq2)
```

To use DESeq we need to get our input data in a very particular format.

```
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 DESeq analysis

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

```
estimating size factors
```

```
estimating dispersions
```

```
gene-wise dispersion estimates
```

```
mean-dispersion relationship
```

```
final dispersion estimates
```

```
fitting model and testing
```

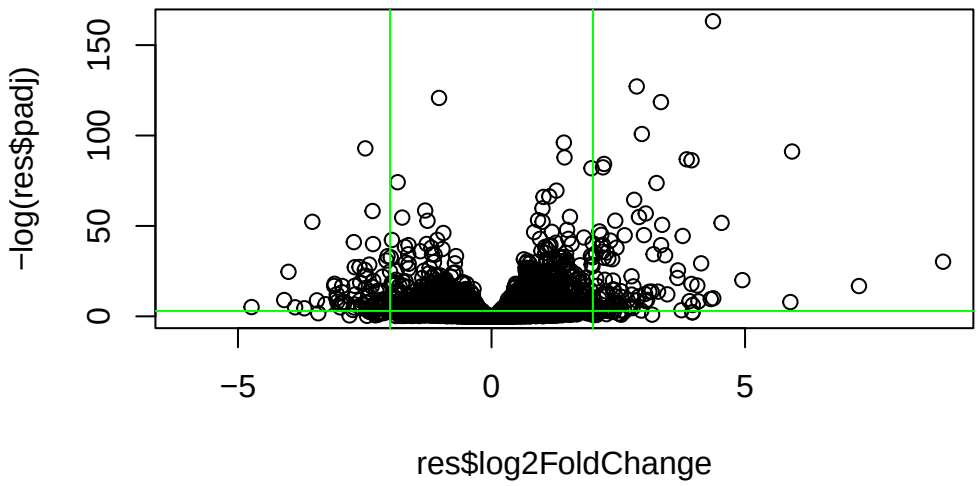
Get the results

```
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
```

We can now make a final figure showing an overview of all the results. Plot **log2 fold change** vs the **adjusted p-value**.

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



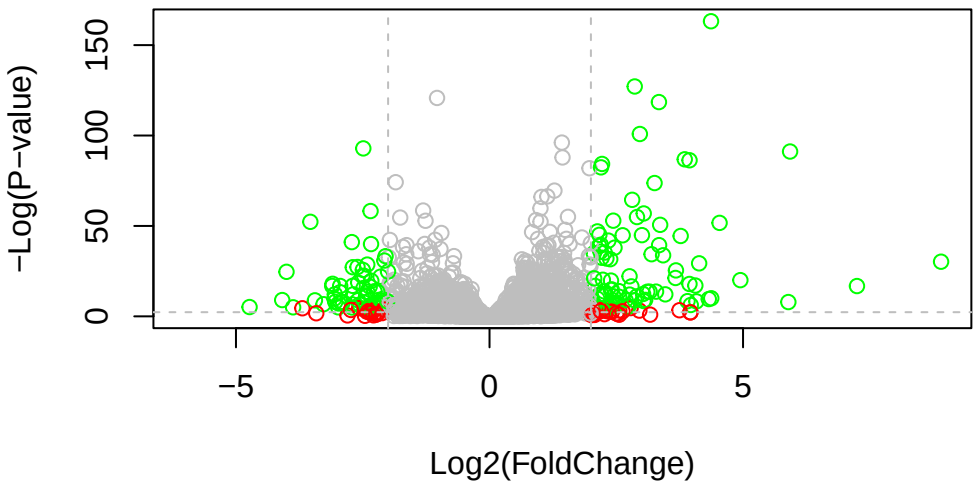

Clean up the plot and make it more legible

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

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

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

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


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