# lab 13: Transcriptomics and the analysis of RNA-Seq data

Elsa Chen (A16632961)

Analyzing a published RNA-seq experiment where airway smooth muscle cells were treated with dexamethasone, a synthetic glucocortiroid steroid with anti-inflammatory effects.

## Import data

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

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.3     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.0
v ggplot2   3.4.3     v tibble    3.2.1
v lubridate 1.9.2     v tidyr     1.3.0
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

## Examine data

How many genes are in this dataset?

```
nrow(counts)
```

```
[1] 38694
```

How many control cell lines do we have?

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

```
control treated 
      4       4 
```

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

```
[1] TRUE
```

## Analysis

Start by comparing control and treated columns. I will find the average for each gene in all control columns and the average for all treated columns

```
# finding control averages
control.ind <- metadata$dex == "control"
control.cts <- counts[,control.ind]
control.avg <- apply(control.cts, 1, mean)
```

```
# finding treated averages
treated.avg <- apply(counts[,metadata$dex == "treated"], 1, mean)
```

```
# combine the two
meancounts <- data.frame(control.avg, treated.avg)
```

plot the means against each other to see if drug has an effect on the samples

```
ggplot(meancounts, aes(control.avg, treated.avg)) +
  geom_point(alpha = 0.2) +
  geom_smooth(method = lm) +
  scale_x_continuous(trans = "log2") +
  scale_y_continuous(trans = "log2")
```

```
Warning: Transformation introduced infinite values in continuous x-axis
```

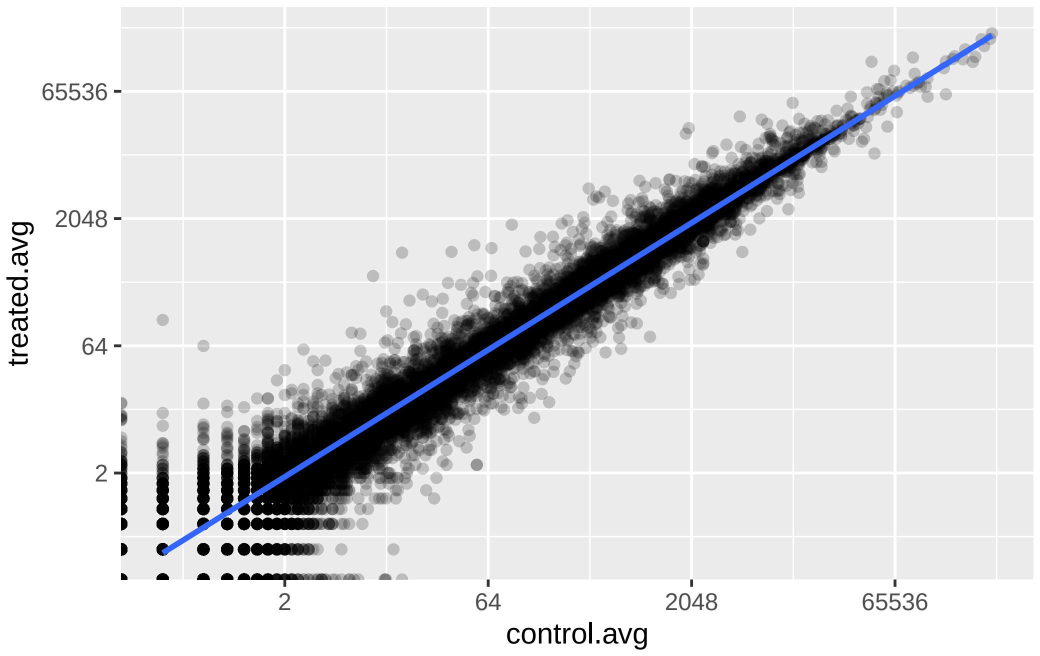
```
Warning: Transformation introduced infinite values in continuous y-axis

Warning: Transformation introduced infinite values in continuous x-axis

Warning: Transformation introduced infinite values in continuous y-axis

`geom_smooth()` using formula = 'y ~ x'

Warning: Removed 16877 rows containing non-finite values (`stat_smooth()`).
```



log2 units are the most common because they have simple interpretations.

we will calculate the LFC of treated/control values and add it to our df

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

```
                control.avg treated.avg      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 some -Inf and NaN because of the 0 reads in the dataset. we can filter the data to exclude these

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

How many genes do we have left after filtering?

```
nrow(mycounts)
```

```
[1] 21817
```

a common threshold for up or down is $|\text{LFC}| > 2$

How many up regulated genes?

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

```
[1] 314
```

How many down regulated?

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

```
[1] 485
```

## DESeq analysis

We are missing the statistics, we can get that properly with DESeq

```
library(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
```

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

```
estimating size factors
```

```
estimating dispersions
```

```
gene-wise dispersion estimates
```

```
mean-dispersion relationship
```

```
final dispersion estimates
```

```
fitting model and testing
```

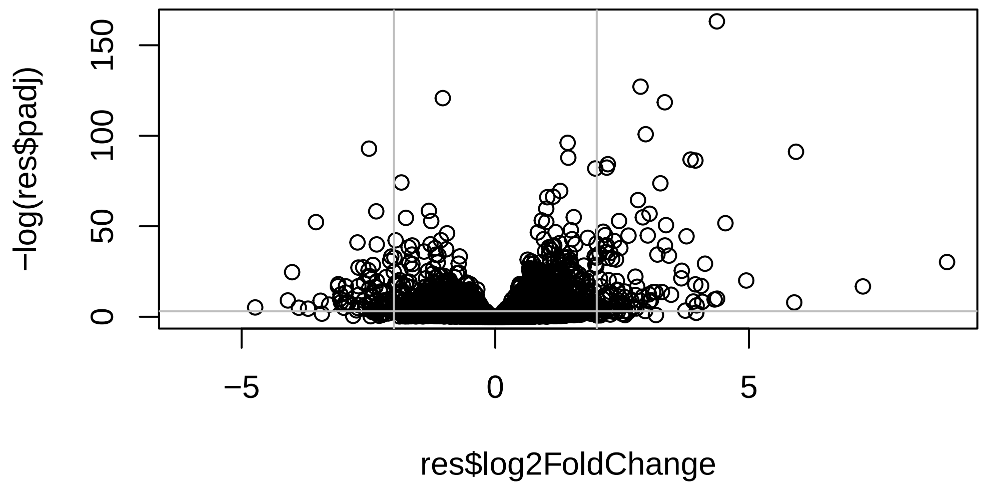
```r
# get results
res <- results(dds)
head(res)
```

```
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
```

make a figure showing LFC vs padj

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

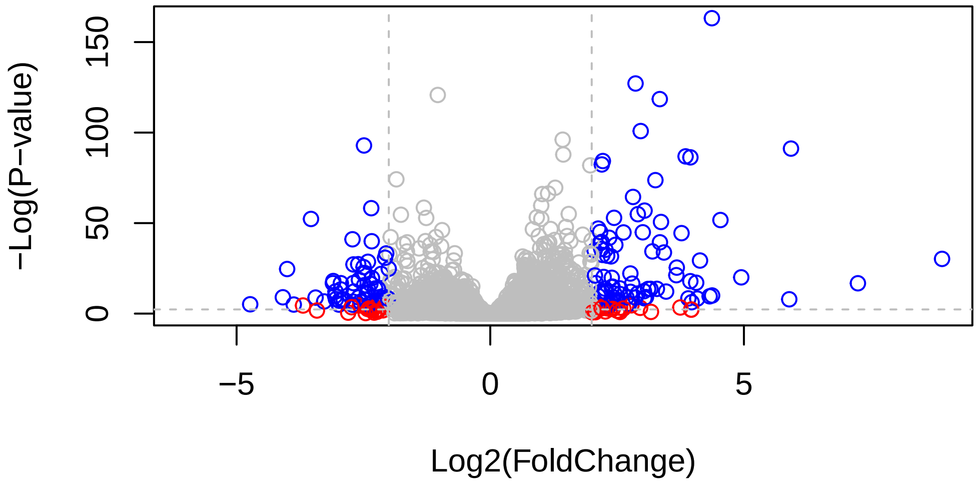


```
# add some colors
mycols <- rep("gray", nrow(res))
mycols[ abs(res$log2FoldChange) > 2 ]  <- "red"
```

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

```r
# 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)
Log2(FoldChange)