

Class13: RNA Seq (pt.1)

Sion Kang (PID: A17052234)

2025-02-18

Today we will analyze data from 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).

Import countData and colData

There are two datasets I need to import/read

- `countData` the transcript counts per gene (rows) in the different experiments
- `colData` information (a.k.a. metadata) about the columns (i.e. experiments) in `countData`.

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

We can have a peak at these with `head()`

```
head(counts)
```

	SRR1039508	SRR1039509	SRR1039512	SRR1039513	SRR1039516
ENSG000000000003	723	486	904	445	1170
ENSG000000000005	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		
ENSG000000000003	1097	806	604		
ENSG000000000005	0	0	0		

```

ENSG00000000419      781      417      509
ENSG00000000457      447      330      324
ENSG00000000460      94       102      74
ENSG00000000938      0        0        0

```

```
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
7 SRR1039520 control   N061011 GSM1275874
8 SRR1039521 treated   N061011 GSM1275875

```

Q1. How many genes are in this dataset?

```
nrow(counts)
```

```
[1] 38694
```

There are 38694 genes in this dataset. > Q2. How many ‘control’ cell lines do we have?

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

```

control treated
4         4

```

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

```
[1] 4
```

We can find the average (mean) count values per gene for all “control” experiments and compare it to the mean values for “treated”.

- Extract all “control” columns from the `counts` data
- Find the mean value for each gene in these columns

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

```
head(control.counts)
```

	SRR1039508	SRR1039512	SRR1039516	SRR1039520
ENSG000000000003	723	904	1170	806
ENSG000000000005	0	0	0	0
ENSG000000000419	467	616	582	417
ENSG000000000457	347	364	318	330
ENSG000000000460	96	73	118	102
ENSG000000000938	0	1	2	0

Now find the row wise mean

```
control.mean <- rowSums(control.counts)/ncol(control.counts)  
head(control.mean)
```

ENSG000000000003	ENSG000000000005	ENSG000000000419	ENSG000000000457	ENSG000000000460
900.75	0.00	520.50	339.75	97.25
ENSG000000000938				
	0.75			

- Extracting all the “treated” columns and finding the mean value for each gene in these columns

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

ENSG000000000003	ENSG000000000005	ENSG000000000419	ENSG000000000457	ENSG000000000460
658.00	0.00	546.00	316.50	78.75
ENSG000000000938				
	0.00			

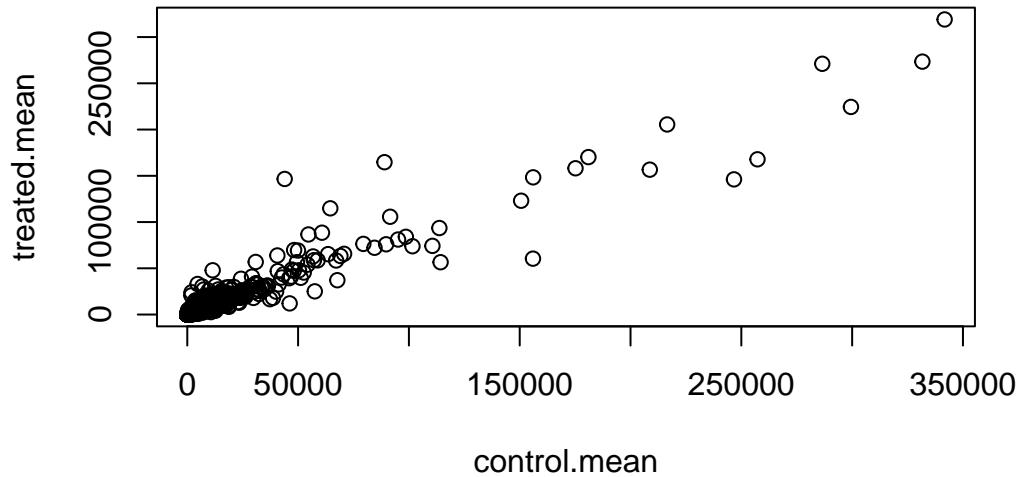
Let's put these two mean values together for easy book-keeping

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

	control.mean	treated.mean
ENSG000000000003	900.75	658.00
ENSG000000000005	0.00	0.00
ENSG00000000419	520.50	546.00
ENSG00000000457	339.75	316.50
ENSG00000000460	97.25	78.75
ENSG00000000938	0.75	0.00

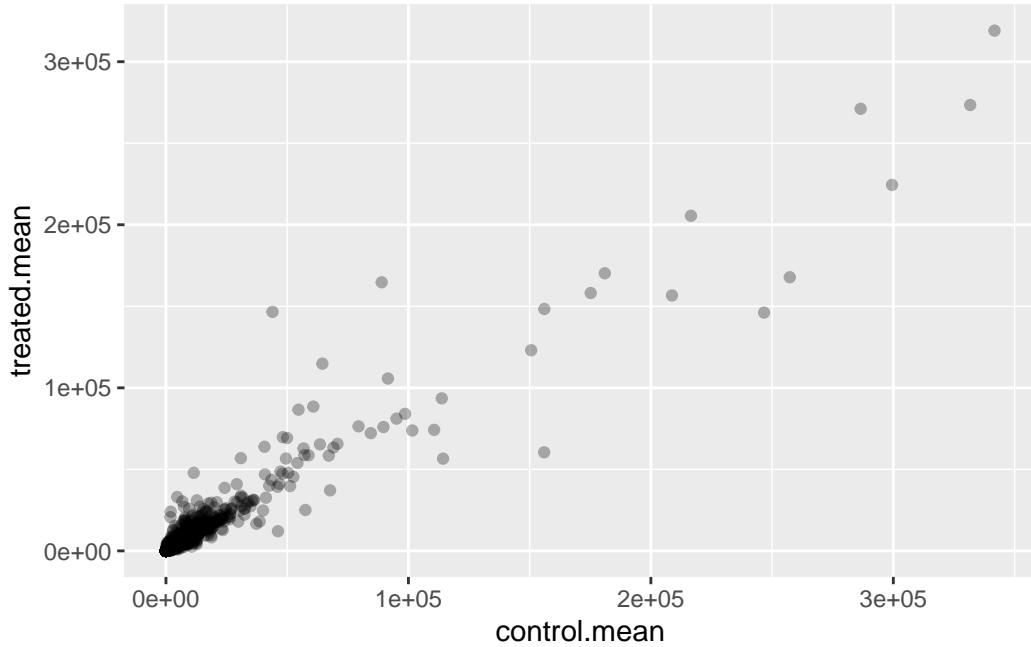
Let's have a look - i.e. plot control.mean vs. treated.mean

```
plot(meancounts)
```



```
library(ggplot2)

ggplot(meancounts) +
  aes(control.mean, treated.mean) +
  geom_point(alpha=0.3)
```

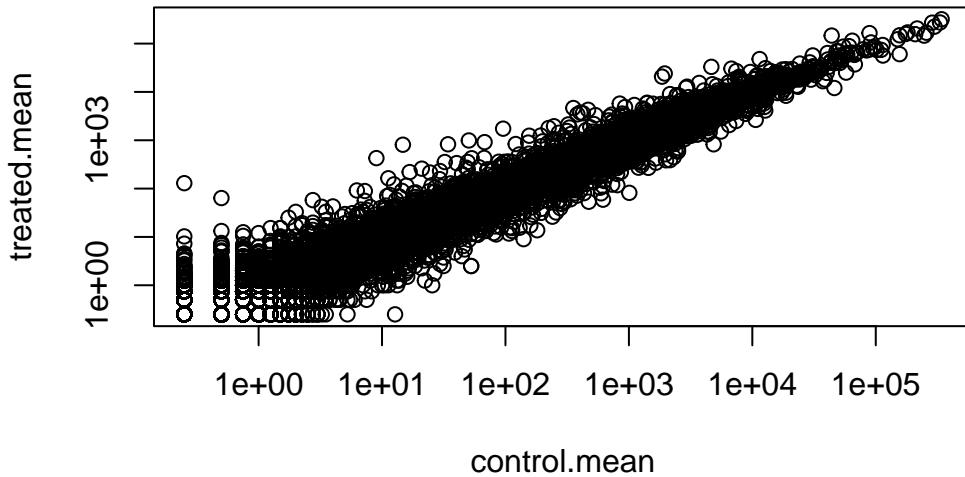


Whenever we see data that is so heavily skewed like this we often log transform it so we can see what is going on more easily.

```
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 most often work in log₂ units as this makes the math easier. Let's have a play to see this.

```
# treated / control
log2(40/20)
```

```
[1] 1
```

```
log2(80/20)
```

```
[1] 2
```

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

```
[1] -1
```

We can now add “log₂ fold-change” values to our `meancounts` dataset.

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

	control.mean	treated.mean	log2fc
ENSG000000000003	900.75	658.00	-0.45303916
ENSG000000000005	0.00	0.00	NaN
ENSG000000000419	520.50	546.00	0.06900279
ENSG000000000457	339.75	316.50	-0.10226805
ENSG000000000460	97.25	78.75	-0.30441833
ENSG000000000938	0.75	0.00	-Inf

We need to filter out zero count genes - i.e. remove the rows (genes) that have a 0 value in either control or treated means.

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

	control.mean	treated.mean	log2fc
ENSG000000000003	900.75	658.00	-0.45303916
ENSG000000000419	520.50	546.00	0.06900279
ENSG000000000457	339.75	316.50	-0.10226805
ENSG000000000460	97.25	78.75	-0.30441833
ENSG000000000971	5219.00	6687.50	0.35769358
ENSG000000001036	2327.00	1785.75	-0.38194109

How many genes are “up” regulated at the common log2 fold-change threshold of +2

```
up.ind <- mycounts$log2fc >= 2
sum(up.ind, na.rm=T)
```

[1] 314

How many genes are “down” regulated at the threshold of -2?

```
down.ind <- mycounts$log2fc <= (-2)
sum(down.ind, na.rm=T)
```

[1] 485

DESeq2 analysis

To do this the right way we need to consider the significance of the differences not just their magnitude.

```
library(DESeq2)
```

To use this package it wants countData and colData in a specific 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

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

estimating size factors

estimating dispersions

gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing

Extract my results

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

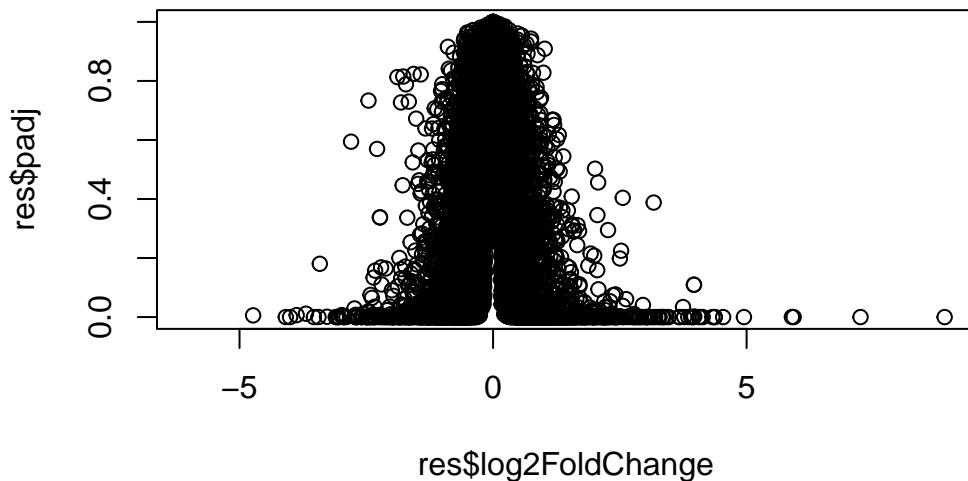
```

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>
ENSG000000000003 747.194195 -0.3507030 0.168246 -2.084470 0.0371175
ENSG000000000005 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>
ENSG000000000003 0.163035
ENSG000000000005  NA
ENSG00000000419 0.176032
ENSG00000000457 0.961694
ENSG00000000460 0.815849
ENSG00000000938  NA

```

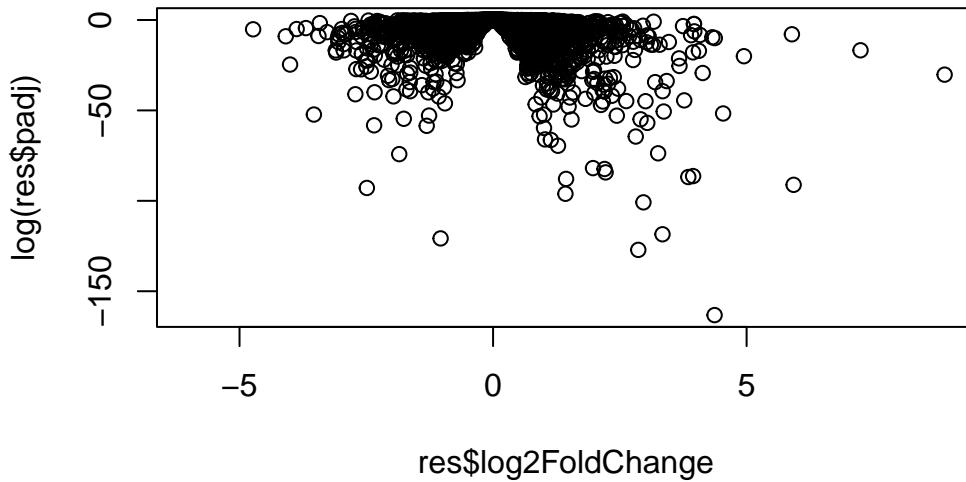
Plot of fold-change vs P-value (adjusted for multiple testing)

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



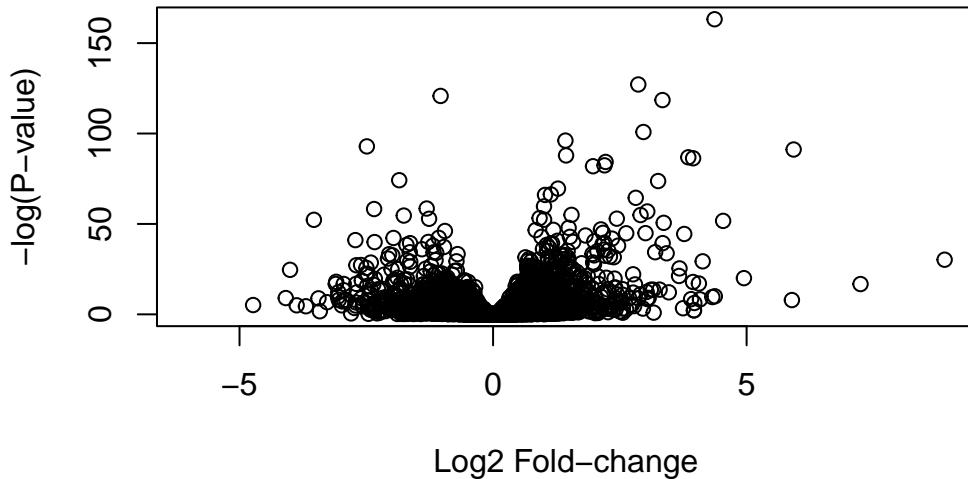
Take the log of the P-value

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



We can just flip that y-axis by putting a minus sign on it

```
plot(res$log2FoldChange, -log(res$padj),
     xlab= "Log2 Fold-change",
     ylab= "-log(P-value)")
```



Let's save our work to date

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

To finish off let's make a nicer volcano plot

Add the log2 threshold lines at $+2/-2$ Add P-value threshold line at 0.05 Add color to highlight the subset of genes that meet both of the above thresholds

Make it with ggplot

```
mycols <- rep("gray", nrow(res))
mycols[res$log2FoldChange >= 2] <- "red"
mycols[res$log2FoldChange <= -2] <- "blue"
mycols[res$padj > 0.05] <- "gray"

ggplot(res) +
  aes(log2FoldChange, -log(padj)) +
  geom_point(col=mycols) +
  geom_vline(xintercept = c(-2,2), linetype="dashed", col="red")+
  geom_hline(yintercept= -log(0.05), linetype="dashed", color = "red")
```

Warning: Removed 23549 rows containing missing values or values outside the scale range
(`geom_point()`).

