Class 13- Transcriptomics and the analysis of RNA-Seq data

Renee Zuhars (PID: A17329856)

Table of contents

Background

Today we will analyze some RNA sequencing data on the effects of a common steroid drug on airway cell lines.

There are two main inputs we need for this analysis:

- `countData`: counts for genes in rows with experiments in the columns
- `colData`: or metadata, tells us about the design of the experiment (i.e. what is in the columns of countData)

Import countData and colData

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

metadata <- read.csv("airway_metadata.csv")
```

Q1. How many genes are in this dataset?

```
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
```

```
nrow(counts)
```

```
[1] 38694
```

There are 38,694 genes in this dataset.

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

option 1:

```
metadata$dex
```

```
[1] "control" "treated" "control" "treated" "control" "treated" "control"
[8] "treated"
```

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

```
control treated
      4       4
```

option 2:

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

```
[1] 4
```

We have 4 control cell lines.

Toy differential gene expression

Let's try finding the average or mean of the "control" and "treated" columns and see if they differ.

- 1. First we need to find all "control" columns
- 2. Extract just the "control" values for each gene
- 3. Calculate the `mean()` for each gene "control" value

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

```
[1] TRUE
```

The $dex column tells me whether we have "control" or "treated"

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

Extract just the "control" values for all genes

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

Calculate the mean value for each gene in these "control" columns

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

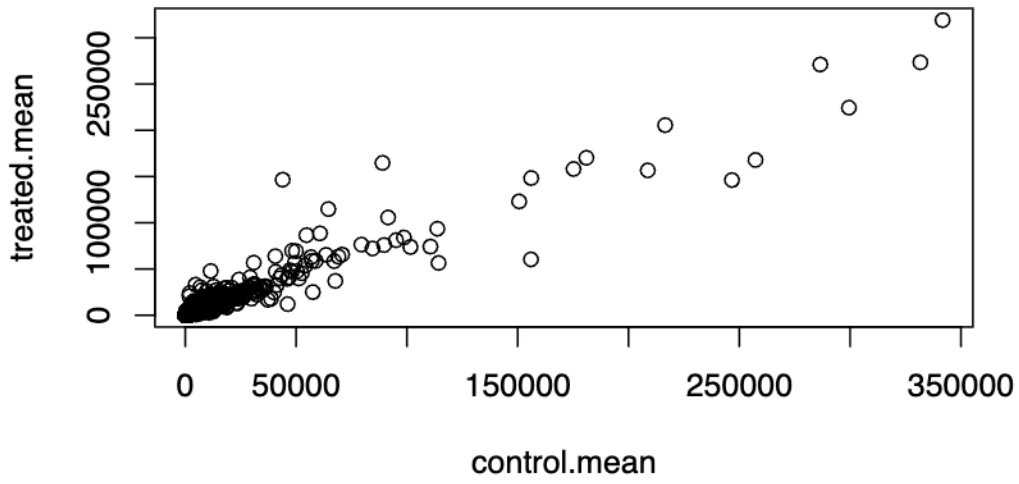
Q3. Do the same for "treated" to get a `treated.mean`

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

treated.counts <- counts[,treated.inds]

treated.mean <- rowMeans(treated.counts)
```

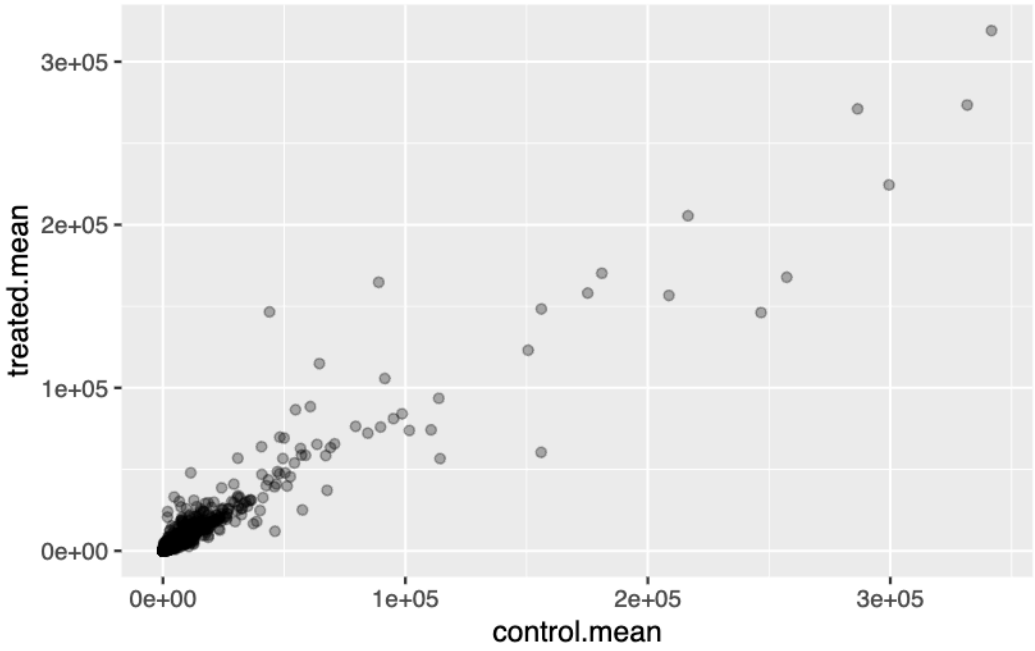
Q4. Make a plot of `control.mean` vs `treated.mean`

Let's store our mean values together in a data.frame for easier book-keeping:

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

plot(meancounts) #plot with base R
```




```
library(ggplot2)

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

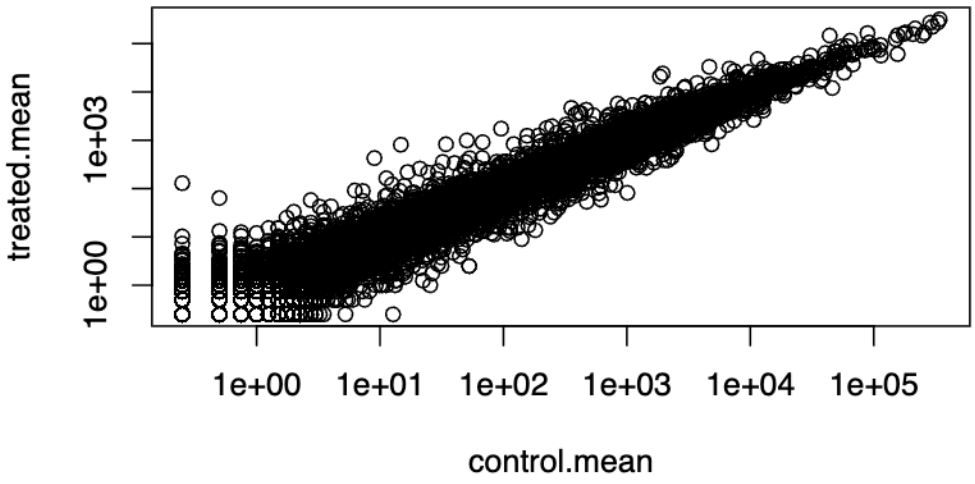


We need to log transform this data as it is so heavily skewed!

```
plot(meancounts, log="xy")
```

```
Warning in xy.coords(x, y, xlabel, ylabel, log): 15032 x values <= 0 omitted
from logarithmic plot
```

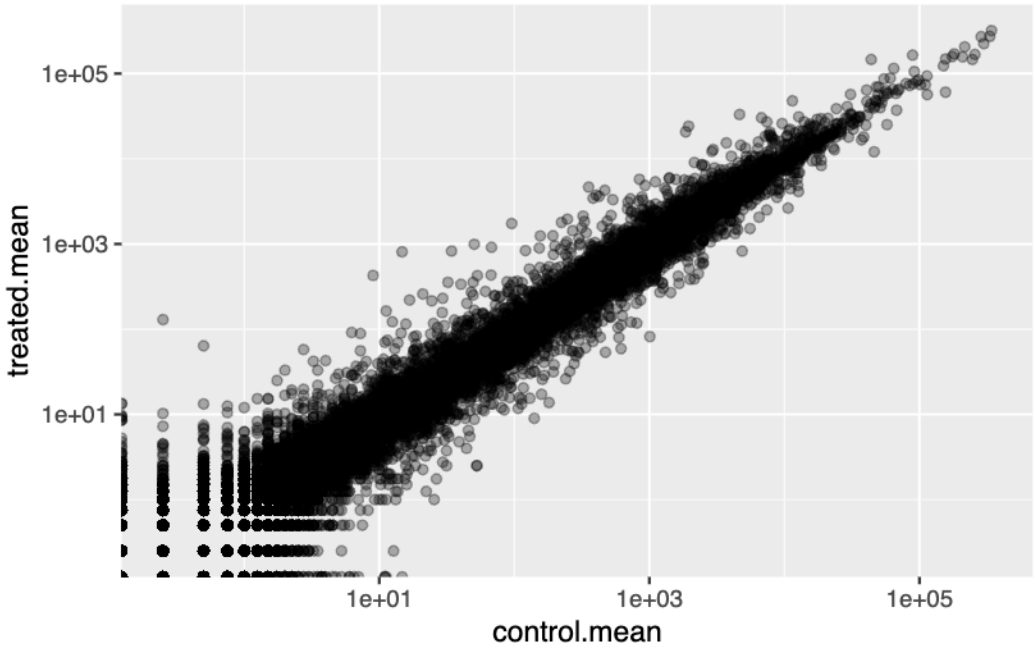
```
Warning in xy.coords(x, y, xlabel, ylabel, log): 15281 y values <= 0 omitted
from logarithmic plot
```




```
ggplot(meancounts) +
  aes(x=control.mean, y=treated.mean) +
  geom_point(alpha=0.3) +
  scale_x_log10() +
  scale_y_log10()
```

```
Warning in scale_x_log10(): log-10 transformation introduced infinite values.
```

```
Warning in scale_y_log10(): log-10 transformation introduced infinite values.
```



A common "rule-of-thumb" is to focus on genes with a log2 "fold-change" of +2 as so called UP REGULATED and -2 as DOWN REGULATED

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

```
[1] 2
```

This means there is 4x more of the gene present.

Let's add a log2 fold-change value to our `meancounts` data.frame

```
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

Q5. Remove any "zero count" genes from our dataset for further analysis.

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

```
[1] 21817
```

```
mycounts <- meancounts[to.keep,]
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
```

Q6. How many genes are "up" regulated at a log2fc threshold of +2?

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

```
[1] 314
```

314 genes out of 21,817 genes are "up" regulated.

Q7. How many genes are "down" regulated at a log2fc threshold of -2?

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

```
[1] 485
```

485 genes out of 21,817 genes are "down" regulated.

*we are missing some statistics- we need a p-value.

DESeq2 analysis

Let's do this properly and consider the stats.

We will use DESeq2 to do this:

```
library(DESeq2)
```

The first function we will use from this package sets up the input in the particular format that DESeq wants:

```
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
```

We can now run our DESeq analysis

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

```
estimating size factors
```

```
estimating dispersions
```

```
gene-wise dispersion estimates
```

```
mean-dispersion relationship
```

```
final dispersion estimates
```

```
fitting model and testing
```

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

Peek at results:

```
head(res)
```

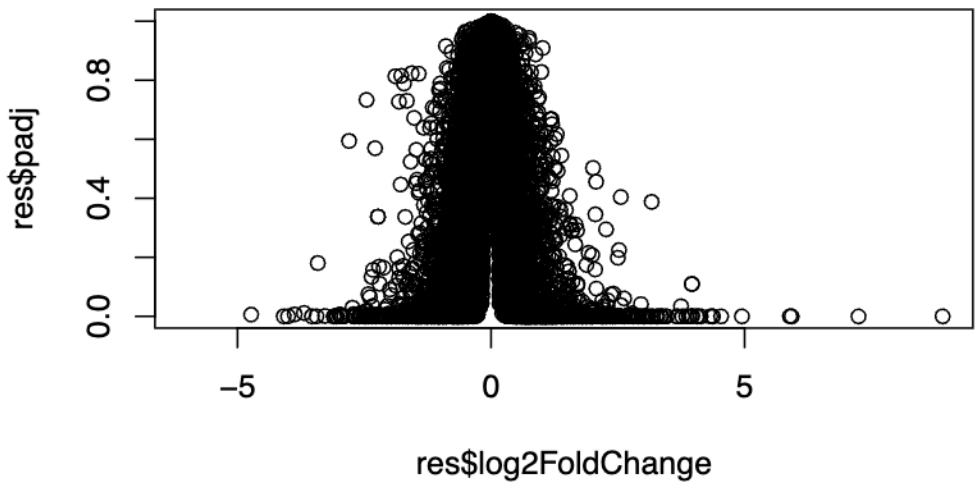
```
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 6 rows and 6 columns
                  baseMean log2FoldChange     lfcSE      stat    pvalue
                 <numeric>      <numeric> <numeric> <numeric> <numeric>
ENSG00000000003 747.194195     -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
```

padj (adjusted p value)- is a multiple testing correction metric

Result figure: Volcano Plot

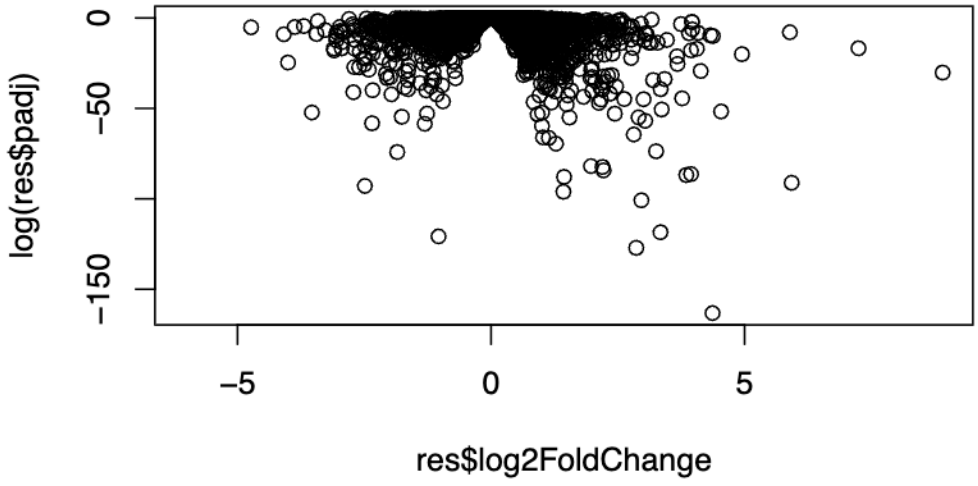
Plot of the Log2FC versus the p-value (the adjusted p value).

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



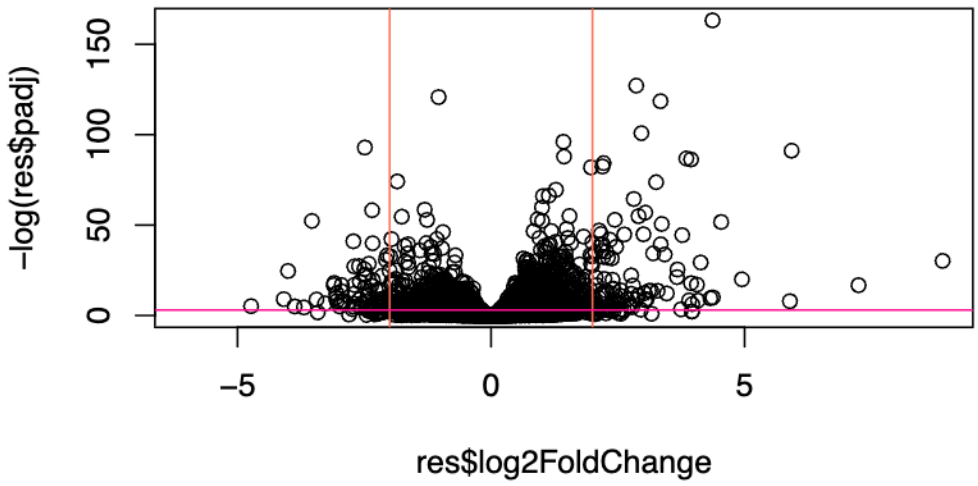
This P-value data is again heavily skewed- so let's log transform it:

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



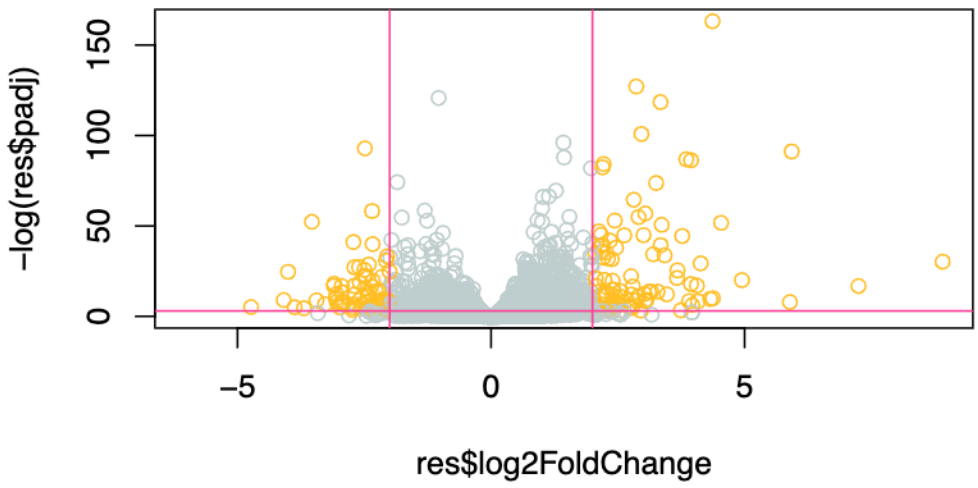
We can flip the y-axis by adding a minus sign. This will make it easier to interpret:

```
plot(res$log2FoldChange, -log(res$padj))
abline(v=-2, col="coral1")
abline(v=+2, col="coral1")
abline(h=-log(0.05), col="deeppink")
```



Let's add some (more) color.

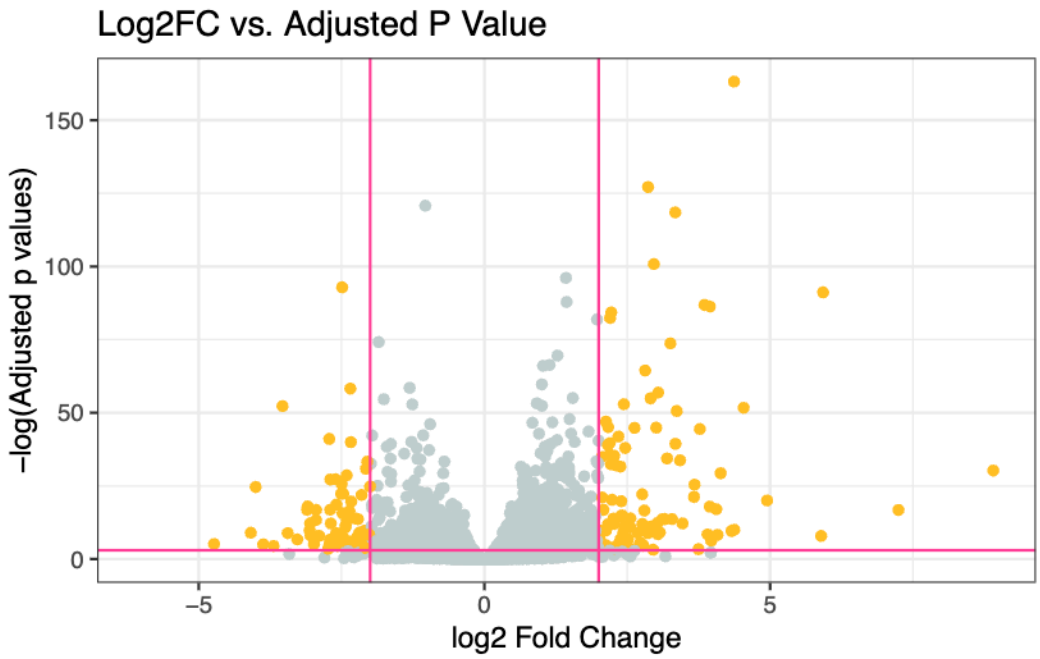
```
mycols <- rep("azure3", nrow(res))
mycols[ res$log2FoldChange <= -2 ] <- "goldenrod1"
mycols[ res$log2FoldChange >= 2 ] <- "goldenrod1"

mycols[ res$padj >= 0.05 ] <-"azure3"

#mycols
plot(res$log2FoldChange, -log(res$padj), col=mycols)
abline(v=-2, col="violetred1")
abline(v=+2, col="violetred1")
abline(h=-log(0.05), col="violetred1")
```



Q8. Make a ggplot volcano plot with colors and lines as annotation along with nice axis labels.

```
ggplot(as.data.frame(res)) +
  aes(res$log2FoldChange, -log(res$padj)) +
  geom_point(col=mycols) +
  geom_vline(xintercept= c(-2, +2), col="violetred1") +
  geom_hline(yintercept = -log(0.05), col="violetred1") +
  labs(x = "log2 Fold Change", y = "-log(Adjusted p values)", title = "Log2FC vs. Adjusted P
  theme_bw()
```

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



Pathway Analysis

We first need to add gene symbols (e.g. HBB, etc) so we know what genes we are dealing with. We need to "translate" between ENSEMBL ids that we

```
library(AnnotationDbi)
library(org.Hs.eg.db)
```

What different database ID types I can translate between:

```
columns(org.Hs.eg.db)
```

```
 [1] "ACCNUM"       "ALIAS"        "ENSEMBL"      "ENSEMBLPROT"  "ENSEMBLTRANS"
 [6] "ENTREZID"     "ENZYME"       "EVIDENCE"     "EVIDENCEALL"  "GENENAME"
[11] "GENETYPE"     "GO"           "GOALL"        "IPI"          "MAP"
[16] "OMIM"         "ONTOLOGY"     "ONTOLOGYALL"  "PATH"         "PFAM"
[21] "PMID"         "PROSITE"      "REFSEQ"       "SYMBOL"       "UCSCKG"
[26] "UNIPROT"
```

Let's "map" between "ENSEMBL" and "SYMBOL" (i.e. gene symbol)

```
res$symbol <- mapIds(x = org.Hs.eg.db,
                     keys = rownames(res),
                     keytype = "ENSEMBL",
                     column = "SYMBOL")
```

```
'select()' returned 1:many mapping between keys and columns
```

```
head(res)
```

```
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 6 rows and 7 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      symbol
                 <numeric> <character>
ENSG00000000003  0.163035      TSPAN6
ENSG00000000005        NA        TNMD
ENSG00000000419  0.176032        DPM1
ENSG00000000457  0.961694       SCYL3
ENSG00000000460  0.815849       FIRRM
ENSG00000000938        NA         FGR
```

Add a few more mappings: Between "ENSEMBL" and "GENENAME" Between "ENSEMBL" and "ENTREZID"

```
res$name <- mapIds(x = org.Hs.eg.db,
                     keys = rownames(res),
                     keytype = "ENSEMBL",
                     column = "GENENAME")
```

```
'select()' returned 1:many mapping between keys and columns
```

```
res$entrez <- mapIds(x = org.Hs.eg.db,
                     keys = rownames(res),
                     keytype = "ENSEMBL",
                     column = "ENTREZID")
```

```
'select()' returned 1:many mapping between keys and columns
```

```
head(res)
```

```
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 6 rows and 9 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      symbol                    name      entrez
                <numeric> <character>             <character> <character>
ENSG00000000003  0.163035      TSPAN6           tetraspanin 6        7105
ENSG00000000005        NA        TNMD             tenomodulin       64102
ENSG00000000419  0.176032        DPM1 dolichyl-phosphate m..        8813
ENSG00000000457  0.961694       SCYL3 SCY1 like pseudokina..       57147
ENSG00000000460  0.815849       FIRRM FIGNL1 interacting r..       55732
ENSG00000000938        NA         FGR FGR proto-oncogene, ..        2268
```

Be sure to save our annotated results to a file.

```
write.csv(res, file = "my_annotated_results_class13.csv")
```

Pathway Analysis

Installed packages (using BiocManager::install(c("pathview", "gage", "gageData")))

```
library(pathview)
library(gage)
library(gageData)
```

Let's peek at the metabolism of caffeine.

```
data(kegg.sets.hs)

# Examine the first 2 pathways in this kegg set for humans
head(kegg.sets.hs, 2)
```

```
$`hsa00232 Caffeine metabolism`
[1] "10"   "1544" "1548" "1549" "1553" "7498" "9"

$`hsa00983 Drug metabolism - other enzymes`
 [1] "10"     "1066"   "10720"  "10941"  "151531" "1548"   "1549"   "1551"
 [9] "1553"   "1576"   "1577"   "1806"   "1807"   "1890"   "221223" "2990"
[17] "3251"   "3614"   "3615"   "3704"   "51733"  "54490"  "54575"  "54576"
[25] "54577"  "54578"  "54579"  "54600"  "54657"  "54658"  "54659"  "54963"
[33] "574537" "64816"  "7083"   "7084"   "7172"   "7363"   "7364"   "7365"
[41] "7366"   "7367"   "7371"   "7372"   "7378"   "7498"   "79799"  "83549"
[49] "8824"   "8833"   "9"      "978"
```

To run pathway analysis, we will use the `gage()` function, and it requires a wee "vector of importance". We will use our LogFC results from our `res` object.

```
foldchanges = res$log2FoldChange
names(foldchanges) = res$entrez
head(foldchanges)
```

```
       7105       64102        8813       57147       55732        2268
-0.35070302          NA  0.20610777  0.02452695 -0.14714205 -1.73228897
```

```
# Get the results
keggres = gage(foldchanges, gsets=kegg.sets.hs)

# What is in the returned `keggres` object:
attributes(keggres)
```

```
$names
[1] "greater" "less"    "stats"
```

```
head(keggres$less, 3) #pathways that have an overlap with my gene
```

```
                                      p.geomean stat.mean        p.val
hsa05332 Graft-versus-host disease 0.0004250461 -3.473346 0.0004250461
hsa04940 Type I diabetes mellitus  0.0017820293 -3.002352 0.0017820293
hsa05310 Asthma                    0.0020045888 -3.009050 0.0020045888
                                        q.val set.size         exp1
hsa05332 Graft-versus-host disease 0.09053483       40 0.0004250461
hsa04940 Type I diabetes mellitus  0.14232581       42 0.0017820293
hsa05310 Asthma                    0.14232581       29 0.0020045888
```

We can pass our foldchanges vector (our results) together with any of these highlighted pathway IDs to see how our genes overlap the pathway.

```
pathview(gene.data=foldchanges, pathway.id="hsa05310")
```

```
'select()' returned 1:1 mapping between keys and columns

Info: Working in directory /Users/reneezuhars/Desktop/BIMM 143/R work/Class 13

Info: Writing image file hsa05310.pathview.png
```

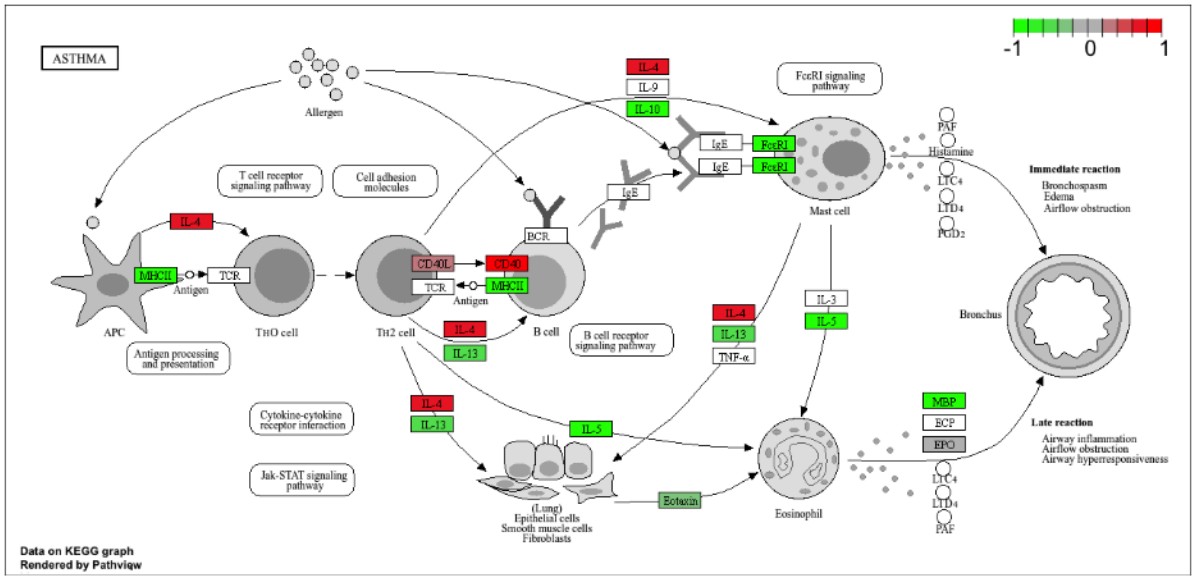


Figure 1: The Asthma pathway overlaps with our differentially expressed genes