Class 13: RNA Seq Intro

Katie Mostoller A17259578

Table of contents

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

We will use the **DESeq2** package for the heavy lifting in a little bit, but first let's read the data and get to know how things work.

Data import

Therea are two datasets that I need for this type of analysis: - **contData**: the transcript abundances (i.e. read counts per gene) - **colData**: metadata about the columns in the countData (i.e. exerimental setup)

```
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
```

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
```

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

```
[1] 4
```

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

```
[1] TRUE
```

Mean counts per condition

Let's find the average gene counts for control and treated conditions (i.e. columns)

- extract all "control" columns/experiments
- find the row average for these columns

```
control.inds <- metadata$dex == "control"
control.counts <- counts[,control.inds]
control.mean <- rowMeans(control.counts)
```

Now do the same for the "treated" clumns to produce `treated.mean`

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

Let's store these mean values all i one data.frame

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

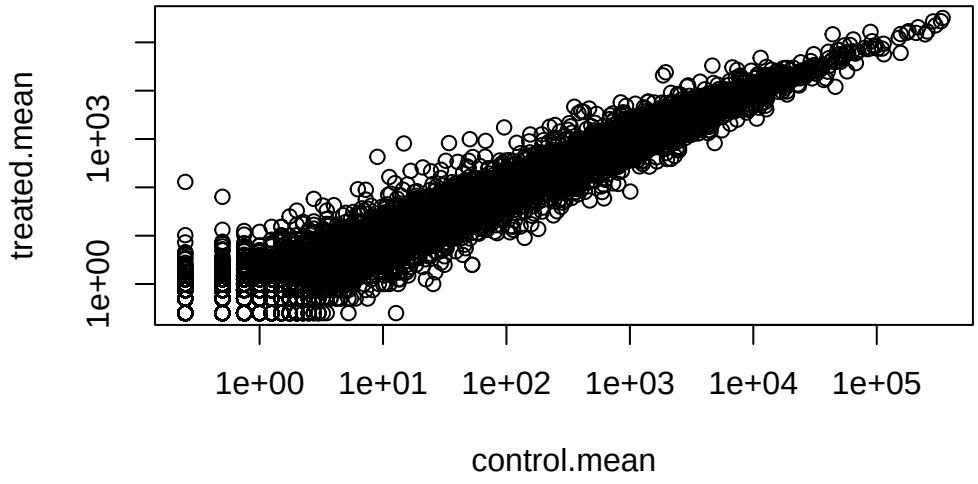
```
                control.mean treated.mean
ENSG00000000003       900.75       658.00
ENSG00000000005         0.00         0.00
ENSG00000000419       520.50       546.00
ENSG00000000457       339.75       316.50
ENSG00000000460        97.25        78.75
ENSG00000000938         0.75         0.00
```

Make a plot of control vs treated

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

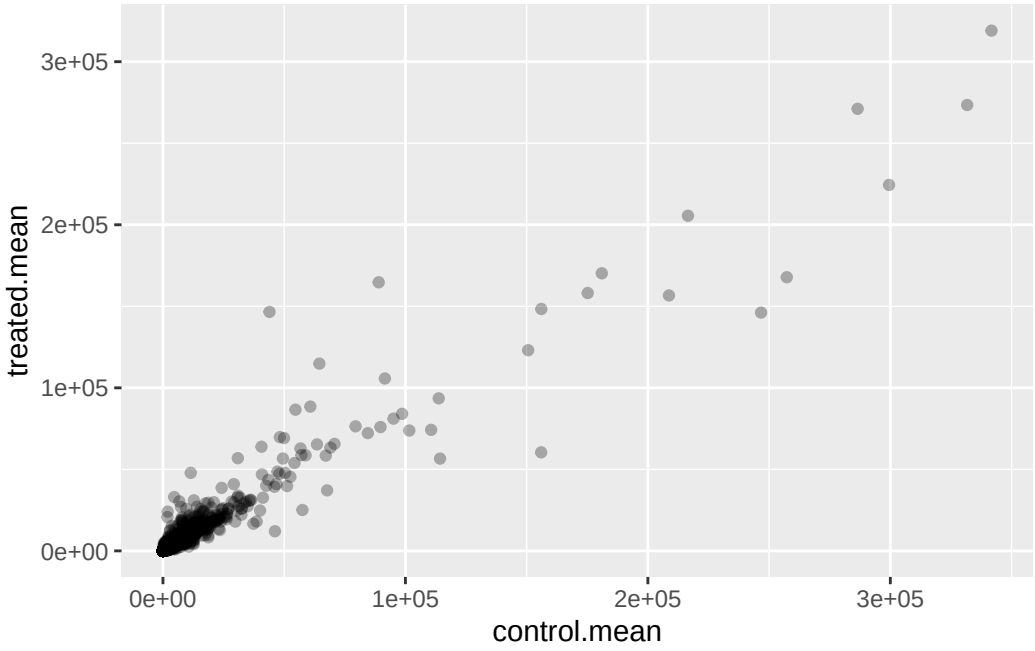
```
Warning in xy.coords(x, y, xlabel, ylabel, log): 15032 x values <= 0 omitted
from logarithmic plot
```

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




```
library(ggplot2)

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



Log fold change

We most often work in log2 units because the interpretation is more straightforward.

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

```
[1] 0
```

When both groups have the same gene expression change, it log2 gives us an answer of 0!

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

```
[1] -1
```

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

```
[1] 1
```

Calculate log2 fold change (`log2fc`) for treated /control

```
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 some weird numbers in the log2fc values like -Inf and NaN because there are zero count genes in our table. We need to filter these out (i.e. remove them) before going any further

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

Q. How many genes do we have left?

```
nrow(mycounts)
```

```
[1] 21817
```

Q. How many genes are upregulated at a log2fc>2?

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

```
[1] 314
```

Q.How many genes are downregulated at a log2fc>2?

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

```
[1] 485
```

Q. Do we trust these results?

We don't know the significance of these results yet

DESeq analysis

To do the analysis properly we can use the bioconductor package **DESeq2**:

```
library(DESeq2)
```

Like most BioConductor packages, DESeq wants it's input 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
```

```
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)
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
```

Save Results

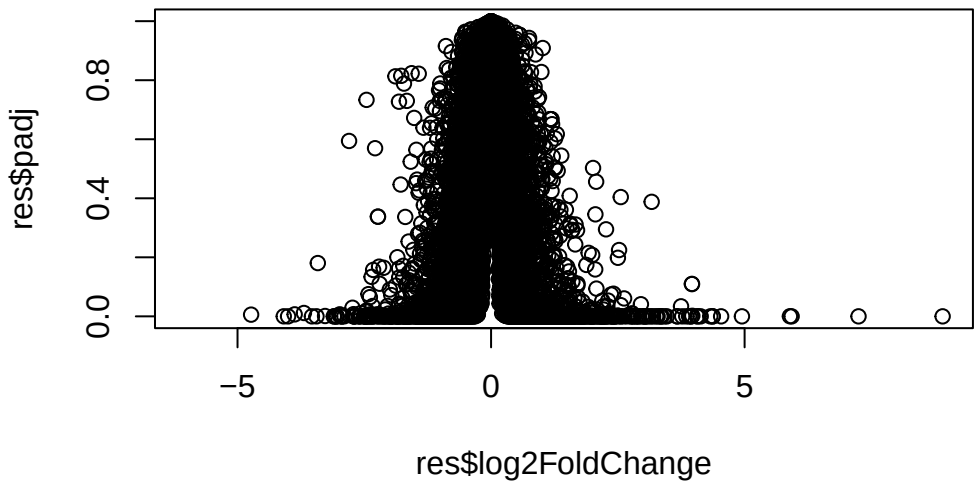
Save our results to CSV file:

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

Volcano Plot

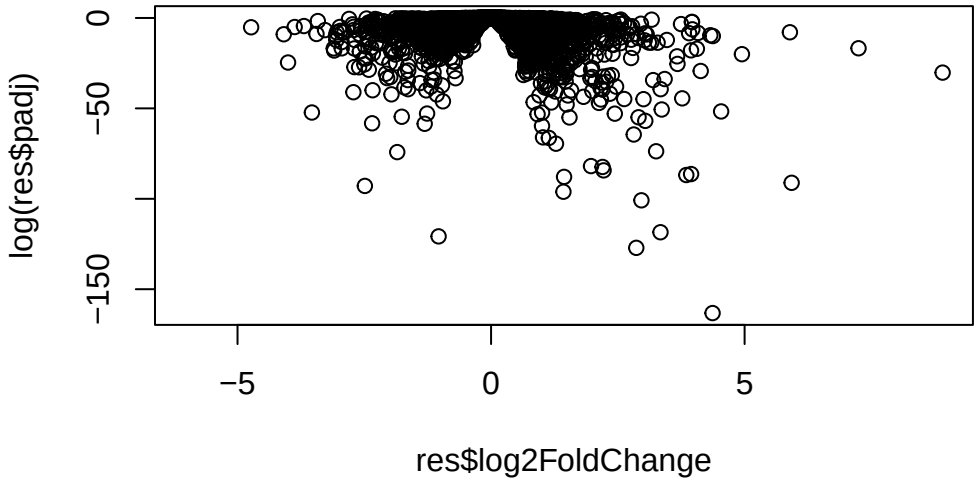
Let's make a common summary plot of our results. Our main results here are the log 2 fold change (log2FoldChange) and correspomding adjusted p value (padj)

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



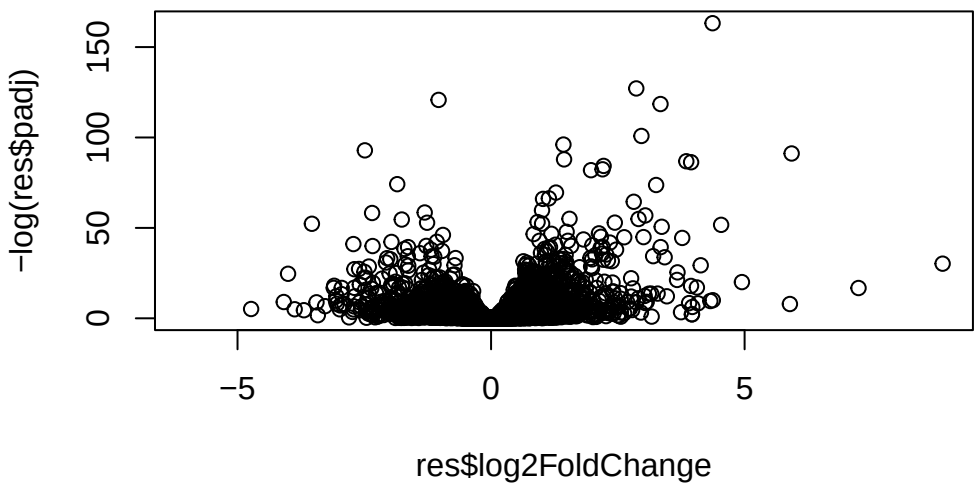
We need to transform the P-value axis so we can see the daat we actually care about (small P-values)

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



To makes folks happy, we need to flip the y-axis so the most significant values are at the top of the graph

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

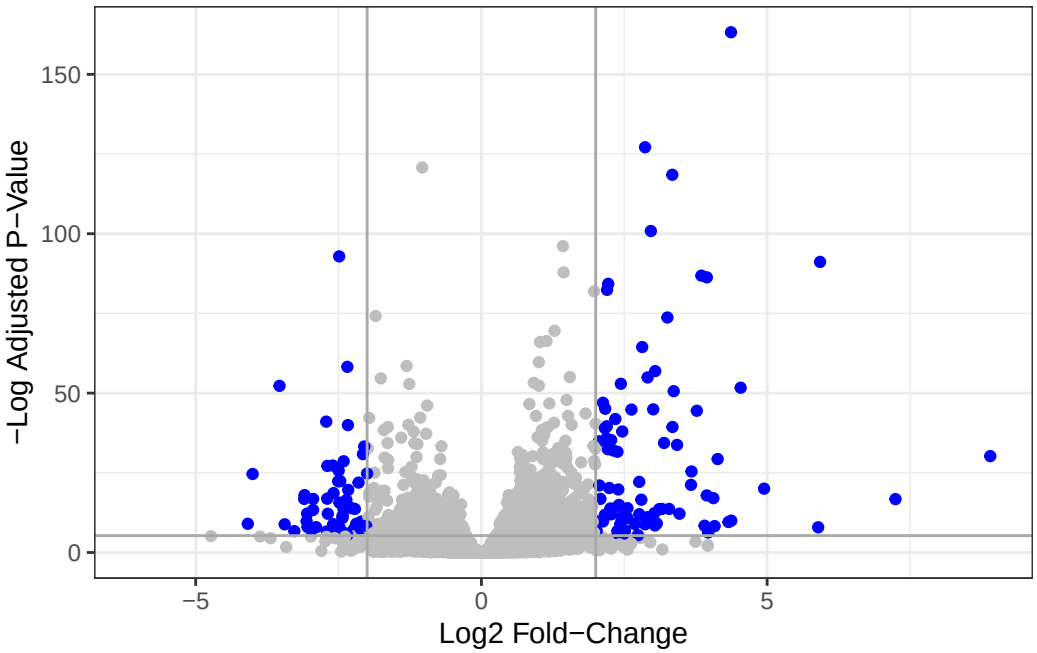


This is our standard volcano plot - let's make this nicer in ggplot. We can use color to highlight the most important subset of transcripts with a log2FC > +2 or < -2 that have a P-value < 0.05. We will need a custom color vector for this `mycols`

```
mycols <- rep("grey", nrow(res))
mycols[res$log2FoldChange >= 2] <- "blue"
mycols[res$log2FoldChange <= -2] <- "blue"
mycols[res$padj > 0.005] <- "grey"
```

```
ggplot(res)+
  aes(log2FoldChange, -log(padj)) +
  geom_point(col = mycols) +
  labs(itle = "Summary Volcano Plot") +
  xlab("Log2 Fold-Change") +
  ylab("-Log Adjusted P-Value") +
    theme_bw() +
geom_vline(xintercept = c(-2,2), color = "darkgray") +
geom_hline(yintercept = -log(0.005), color = "darkgray")
```

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



Adding annotation data

At the minute all we know about the genes in our dataset is their ENSEMBL database id

```
head(row.names(res))
```

```
[1] "ENSG00000000003" "ENSG00000000005" "ENSG00000000419" "ENSG00000000457"
[5] "ENSG00000000460" "ENSG00000000938"
```

We can use a set of BioConductor packages to map these ENSEMBL ids to things like GENE SYMBOL, REFSEQ id, ENTREZ id etc. In oher words what each gene is called is different databses that I might want to use for further analysis

Install these packages with `BioManagerr:: install()` first

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

The different formats that I can convert IDs between include:

```
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"
```

We can use the `mapIds()` function to do this "mapping"/conversion:

```
res$symbol <- mapIds(org.Hs.eg.db,
                     keys=row.names(res), # Our genenames
                     keytype="ENSEMBL", # The format of our genenames
                     column="SYMBOL", # The new format we want to add
                     multiVals="first")
```

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

```
res$genename <- mapIds(org.Hs.eg.db,
                     keys=row.names(res),
                     keytype="ENSEMBL",
                     column="GENENAME", # The new format we want to add
                     multiVals="first")
```

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

```
res$entrez <- mapIds(org.Hs.eg.db,
                     keys=row.names(res),
                     keytype="ENSEMBL",
                     column="ENTREZID", # The new format we want to add
                     multiVals="first")
```

```
'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               genename      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
```

Save results again!

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

Pathway Analysis

Let's use KEGG (kyoto encyclopedia of genes and genomes) to see which pathways my gene sets overlap with - i.e. highlight the biology that may be influenced by the dex drug treatment.

We will use the following packages: BiocManager::install(c("pathview", "gage", "gage-Data"))

```
library(pathview)
```

```
#############################################################################
Pathview is an open source software package distributed under GNU General
Public License version 3 (GPLv3). Details of GPLv3 is available at
http://www.gnu.org/licenses/gpl-3.0.html. Particullary, users are required to
```

```
formally cite the original Pathview paper (not just mention it) in publications
or products. For details, do citation("pathview") within R.

The pathview downloads and uses KEGG data. Non-academic uses may require a KEGG
license agreement (details at http://www.kegg.jp/kegg/legal.html).
##############################################################################
```

```
library(gage)
```

```
library(gageData)

data(kegg.sets.hs)
```

The **gage()** function wants as input a "named vector of importance"

```
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
```

```
keggres <- gage(foldchanges, gsets = kegg.sets.hs)
head(keggres$less)
```

```
                                                      p.geomean stat.mean
hsa05332 Graft-versus-host disease                   0.0004250461 -3.473346
hsa04940 Type I diabetes mellitus                    0.0017820293 -3.002352
hsa05310 Asthma                                      0.0020045888 -3.009050
hsa04672 Intestinal immune network for IgA production 0.0060434515 -2.560547
hsa05330 Allograft rejection                         0.0073678825 -2.501419
hsa04340 Hedgehog signaling pathway                  0.0133239547 -2.248547
                                                             p.val      q.val
hsa05332 Graft-versus-host disease                   0.0004250461 0.09053483
hsa04940 Type I diabetes mellitus                    0.0017820293 0.14232581
hsa05310 Asthma                                      0.0020045888 0.14232581
hsa04672 Intestinal immune network for IgA production 0.0060434515 0.31387180
```

```
hsa05330 Allograft rejection                                0.0073678825 0.31387180
hsa04340 Hedgehog signaling pathway                         0.0133239547 0.47300039
                                                            set.size         exp1
hsa05332 Graft-versus-host disease                                40 0.0004250461
hsa04940 Type I diabetes mellitus                                 42 0.0017820293
hsa05310 Asthma                                                   29 0.0020045888
hsa04672 Intestinal immune network for IgA production             47 0.0060434515
hsa05330 Allograft rejection                                      36 0.0073678825
hsa04340 Hedgehog signaling pathway                               56 0.0133239547
```

We can have a quick look at one of the highlighted pathways e.g. hsa05310

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

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

```
Info: Working in directory /Users/katiemostoller/Desktop/Class 13
```

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