Class 13: RNA seq data

Bryn Baxter (PID A69038039)

Table of contents

In today's class we will analyze some 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).

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

Data import

There are two datasets that I need for this type of analysis:

-countData: the transcript abundances (counts per gene) -colData: metadata about the columns in countData (i.e experiment setup)

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

Q1. How many genes/transcripts are in this dataset?

```
dim(counts)
```

```
[1] 38694     8
```

There are 38,694 gene transcripts in this dataset.

Q2. How many "control" experiments are there in the dataset?

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

```
control treated
      4       4
```

4.

Mean counts per condition

We want to average all our control groups and treatment groups and compare them.

-extract all "control" columns/experiment -then find the row wise average for these columns

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

```
[1] 38694     4
```

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

```
                SRR1039508 SRR1039512 SRR1039516 SRR1039520
ENSG00000000003        723        904       1170        806
ENSG00000000005          0          0          0          0
ENSG00000000419        467        616        582        417
ENSG00000000457        347        364        318        330
ENSG00000000460         96         73        118        102
ENSG00000000938          0          1          2          0
```

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

Now we need to do that same for the treated group to produce `treated.mean`

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

Lets store these mean values on one dataframe.

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

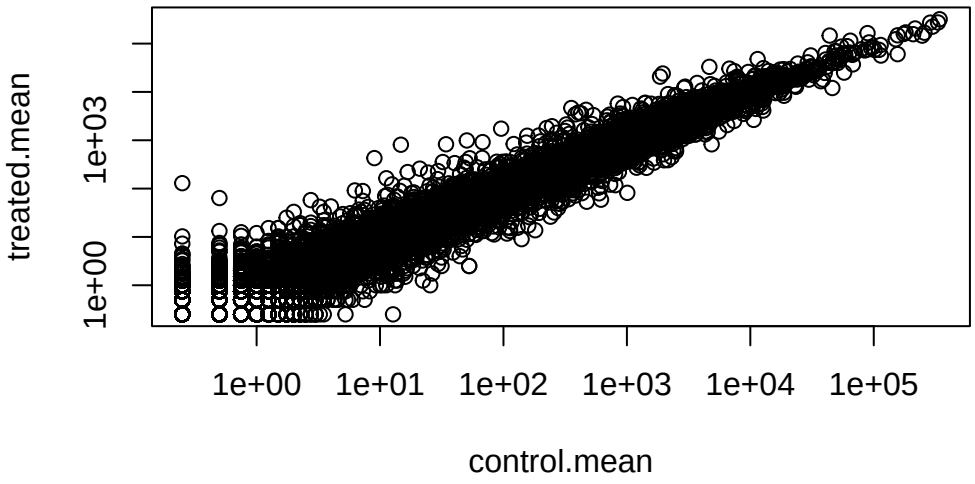
```
                control.mean treated.mean
ENSG00000000003       900.75       658.00
ENSG00000000005         0.00         0.00
ENSG00000000419       520.50       546.00
ENSG00000000457       339.75       316.50
ENSG00000000460        97.25        78.75
ENSG00000000938         0.75         0.00
```

Make a plot of control vs. treated

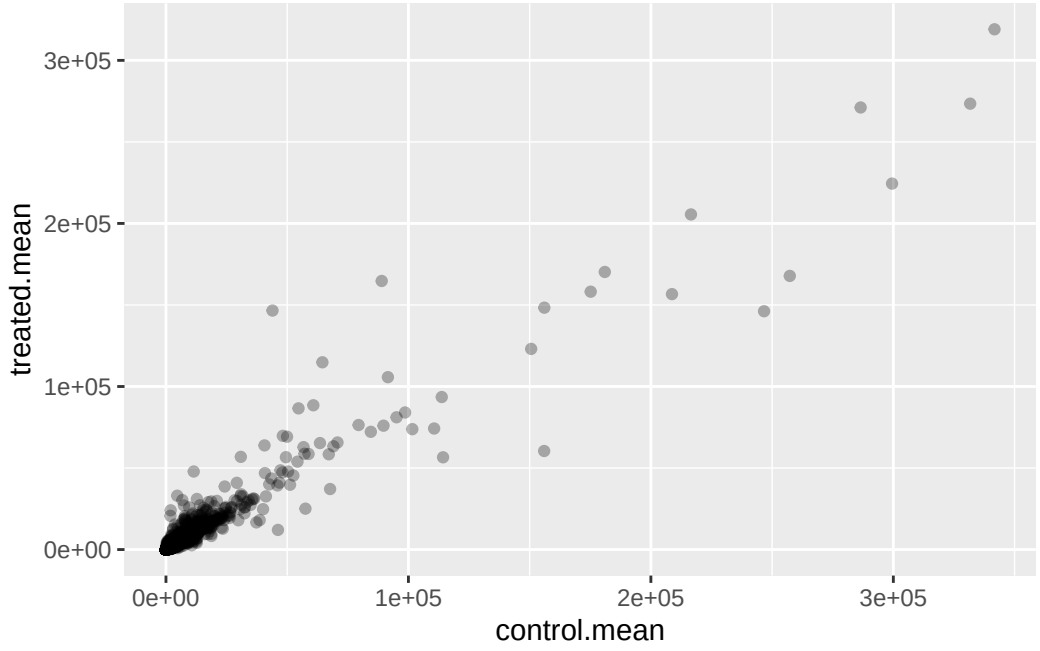
```
plot(meancounts, log="xy")
```

```
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 often work in log2 units- why? Because the interpertation is much more straightforward.

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

```
[1] 0
```

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

```
[1] -1
```

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

```
[1] 1
```

Calculate log2 fold change (`log2fc`) of trated/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 NaN and -Inf all because I have zero count genes. I 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, ]
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
```

Q3. How many non-zero count genes do we have left?

```
nrow(mycounts)
```

```
[1] 21817
```

Q4. How many genes are upregulated at a log2fc > 2?

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

```
[1] 314
```

Q5. How many genes are downregulated at a log2fc < 2

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

```
[1] 21503
```

Q5. Do we trust these results?

No, we have completed neglected the differences in the means and we do not know if these are significant. We are missing the statistics.

DESeq analysis

To do this 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
```

```
class: DESeqDataSet
dim: 38694 8
metadata(1): version
assays(1): counts
rownames(38694): ENSG00000000003 ENSG00000000005 ... ENSG00000283120
  ENSG00000283123
rowData names(0):
colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
colData names(4): id dex celltype geo_id
```

```
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 out results to CSV file:

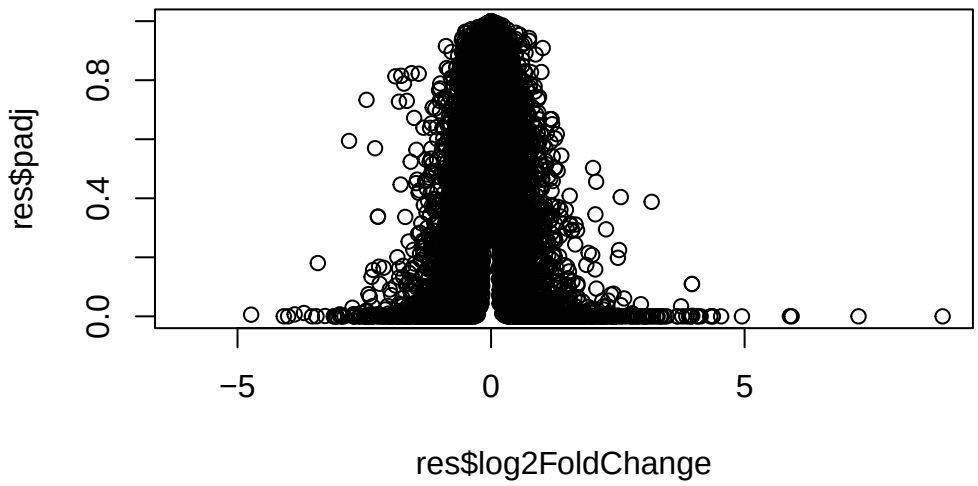
```
write.csv(res, file="myresults.csv")
```

Volcano plots

Lets make a common summary plot of our results.

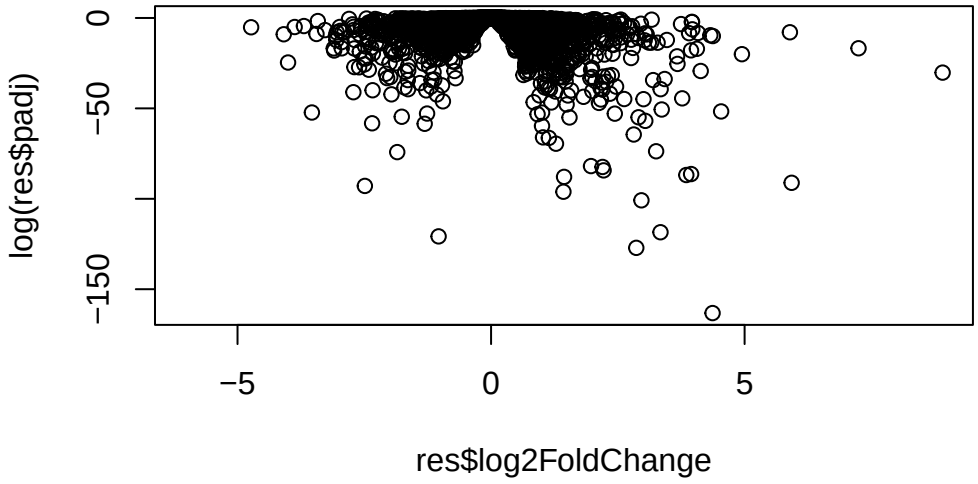
Our main results here are the log2 fold change and the adjusted p-value..

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



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

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



```
log(0.005)
```

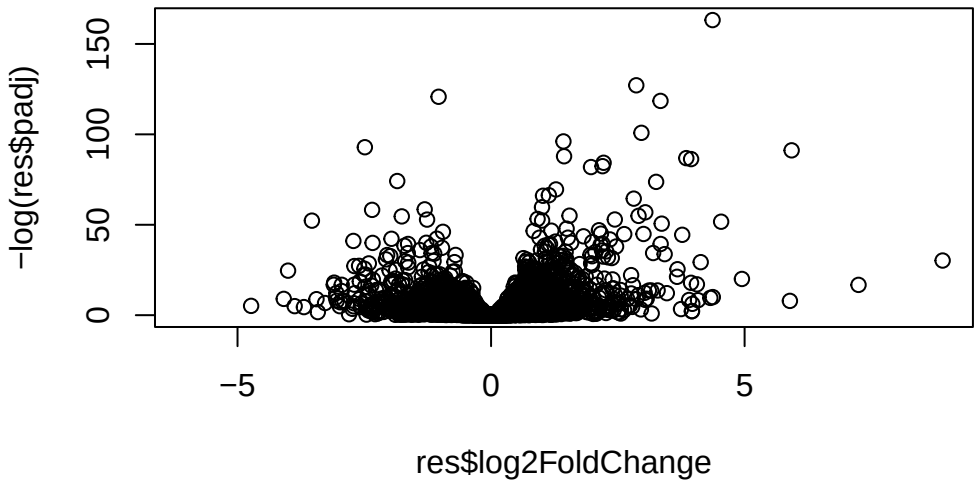
```
[1] -5.298317
```

```
log(0.00000005)
```

```
[1] -16.81124
```

To make folks happy we need to flip our y axis so the most important/significant data points are on the top

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



This is our "standard" volcano plot.

We can use color to highlight the most important subset of transcripts with a log2FC >+2 and <-2 that have a P-value <0.05.

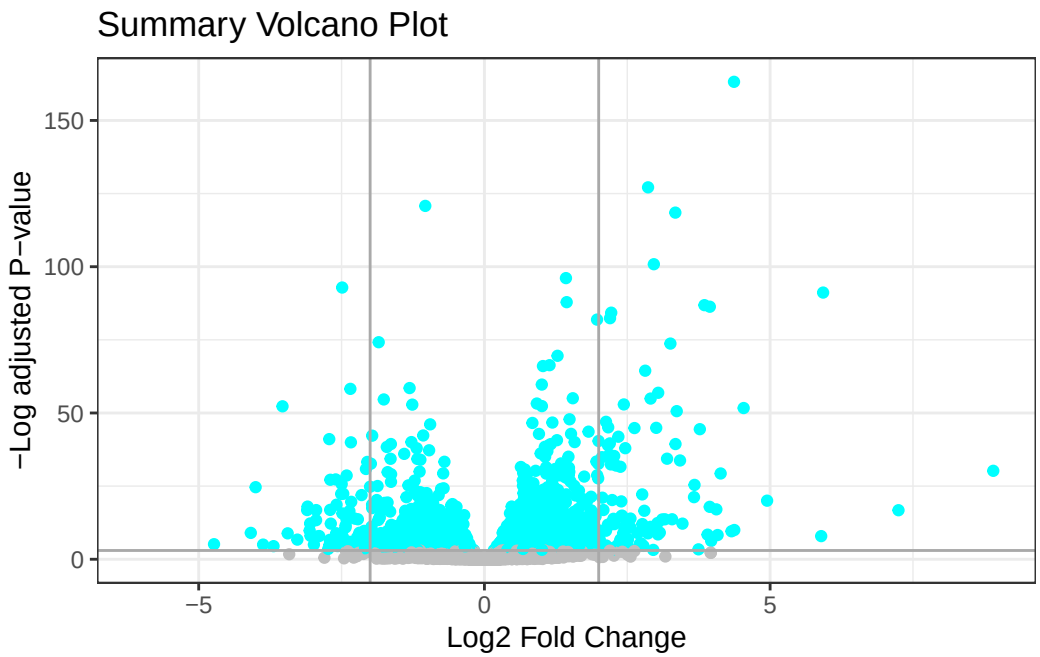
Setup our custom color vector

```
mycols <- rep("gray", nrow(res))
mycols[abs(res$log2FoldChange) >= 2] <- "cyan"
mycols[abs(res$log2FoldChange) < 2] <- "cyan"

mycols[res$padj> 0.05]<- "gray"
```

```
ggplot(res)+
  aes(log2FoldChange, -log(padj))+
  geom_point(col=mycols)+
  labs(title="Summary Volcano Plot")+
  xlab("Log2 Fold Change")+
  ylab("-Log adjusted P-value")+
  geom_vline(xintercept = c(-2,2), col="darkgray")+
  geom_hline(yintercept = -log(0.05), col="darkgray")+
  theme_bw()
```

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



Setup our custom color vector

```
mycols <- rep("gray", nrow(res))
```

```
head( rownames(res))
```

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

We can use a set of BioConductor packages to map these ESEMBLE ids to things like GENE SYMBOL,RESEQ id, ENTREZ id, etc. In other words what each gene is called in different databases that I might want to use in further analysis.

I install these packages with `BioManager::install()`

```
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",
                    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
```

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

Pathway Analysis

Lets use KEGG to see which pathway my gene set overlap with-i.e. highlight the biology that may be influences by the dex drug treatment.

```
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
```

```
#Get the results
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
```

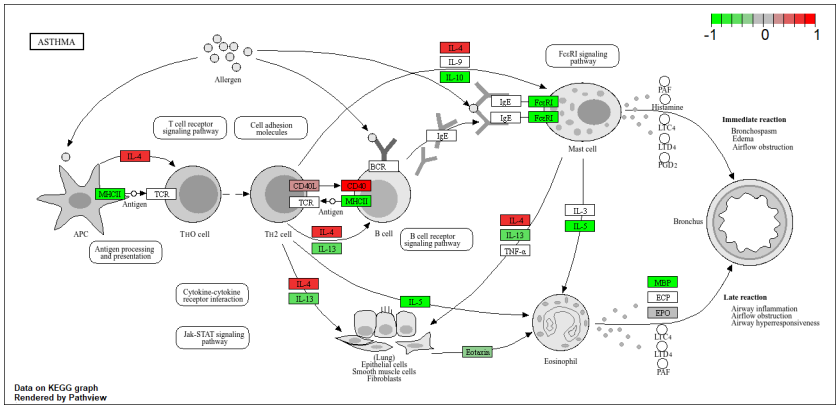
We can have a quick look at one of the highlighted pathways e.g.hsa05310

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

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

Info: Working in directory C:/Users/Bryn Baxter/Documents/BIO213/class 13

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


ASTHMA
-1
0
1
Allergen
IL-4
IL-9
IL-10
FcεRI signaling pathway
IgE
FcεRI
Mast cell
PAF
Histamine
LTC4
LTD4
PGD2
Immediate reaction
Bronchospasm
Edema
Airflow obstruction
T cell receptor signaling pathway
Cell adhesion molecules
IL-4
MHC II
TCR
Antigen
APC
Th0 cell
Th2 cell
BCR
CD40L
CD40
TCR
MHC II
Antigen
B cell
IL-4
IL-13
B cell receptor signaling pathway
Antigen processing and presentation
IL-4
IL-13
IL-5
IL-4
IL-13
TNF-α
IL-3
IL-5
Bronchus
Cytokine-cytokine receptor interaction
Jak-STAT signaling pathway
Eotaxin
(Lung)
Epithelial cells
Smooth muscle cells
Fibroblasts
Eosinophil
MBP
ECP
EPO
LTC4
LTD4
PAF
Late reaction
Airway inflammation
Airflow obstruction
Airway hyperresponsiveness
Data on KEGG graph
Rendered by Pathview