Class 13

AUTHOR

Charlie Rezanka (A15837296)

Importing Data

We need tow things for this project:

- Countdata (counts every transcript and gene in the experiment)
- Col data (metadata that describes the experimental setup)

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

```
                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
```

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

```
               dex celltype     geo_id
SRR1039508 control   N61311 GSM1275862
SRR1039509 treated   N61311 GSM1275863
SRR1039512 control  N052611 GSM1275866
SRR1039513 treated  N052611 GSM1275867
SRR1039516 control  N080611 GSM1275870
SRR1039517 treated  N080611 GSM1275871
```

> Q1. How many genes are in this dataset?

```
nrow(countdata)
```

```
[1] 38694
```

> Q2. How many ‘control’ cell lines do we have?

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

```

control treated 
      4       4 
```

4 control cell lines

another way:

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

```
[1] 4
```

- Step 1. Calculate the mean of the control samples (i.e. columns in countdata)

a. We need to find which columns are “control” samples.

- look in the metadata at the $dex column

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

b. Extract all control columns from `countdata` and call it `control.counts`

```
control.counts <- (countdata[,control.inds])
```

c. Calculate the mean value accross the rows of `control.counts` i.e. calculate the mean count values for each gene in the control samples.

```
control.means <- rowMeans(control.counts)
head(control.means)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460 
         900.75            0.00          520.50          339.75           97.25 
ENSG00000000938 
           0.75 
```

- Step 2. Calculate the mean of the treated samples

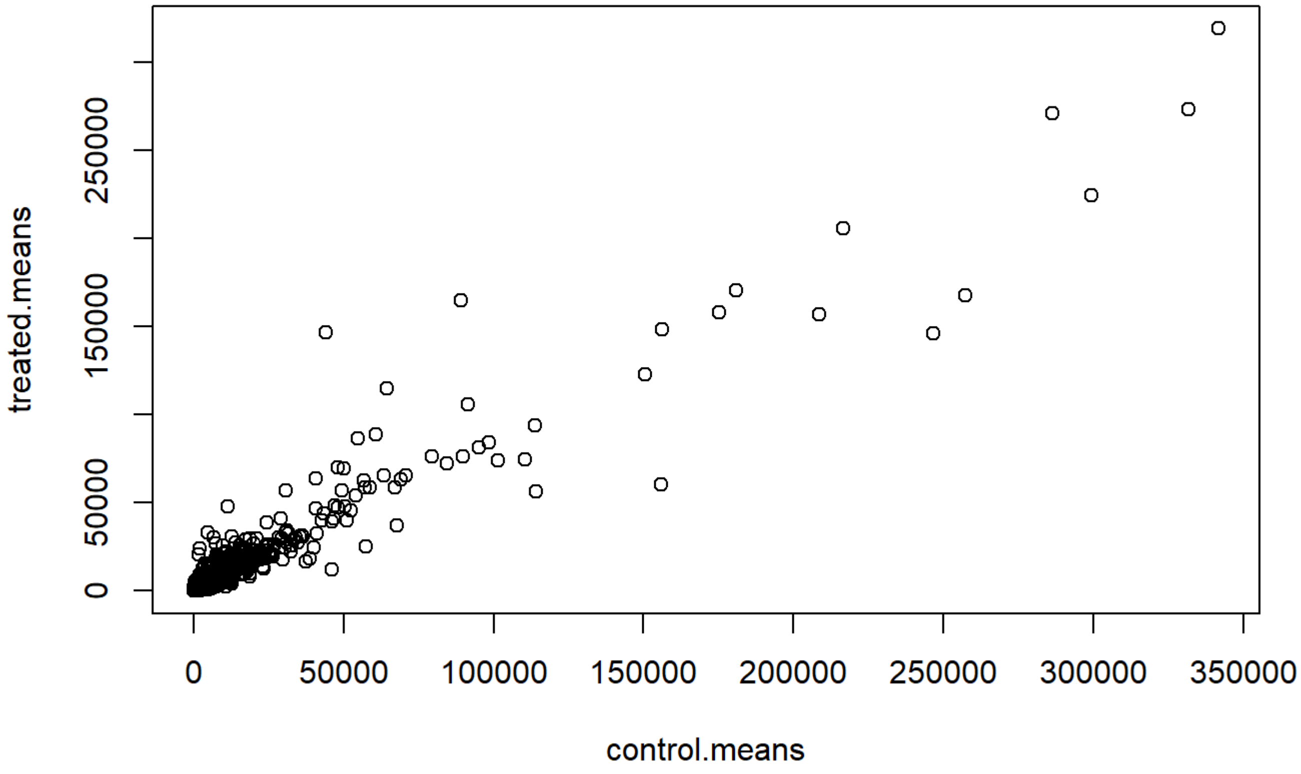
```
treated.inds <- metadata$dex == "treated"
treated.counts <- (countdata[,treated.inds])
treated.means <- rowMeans(treated.counts)
head(treated.means)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
         658.00            0.00          546.00          316.50           78.75
ENSG00000000938
           0.00
```

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

```
                control.means treated.means
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
```

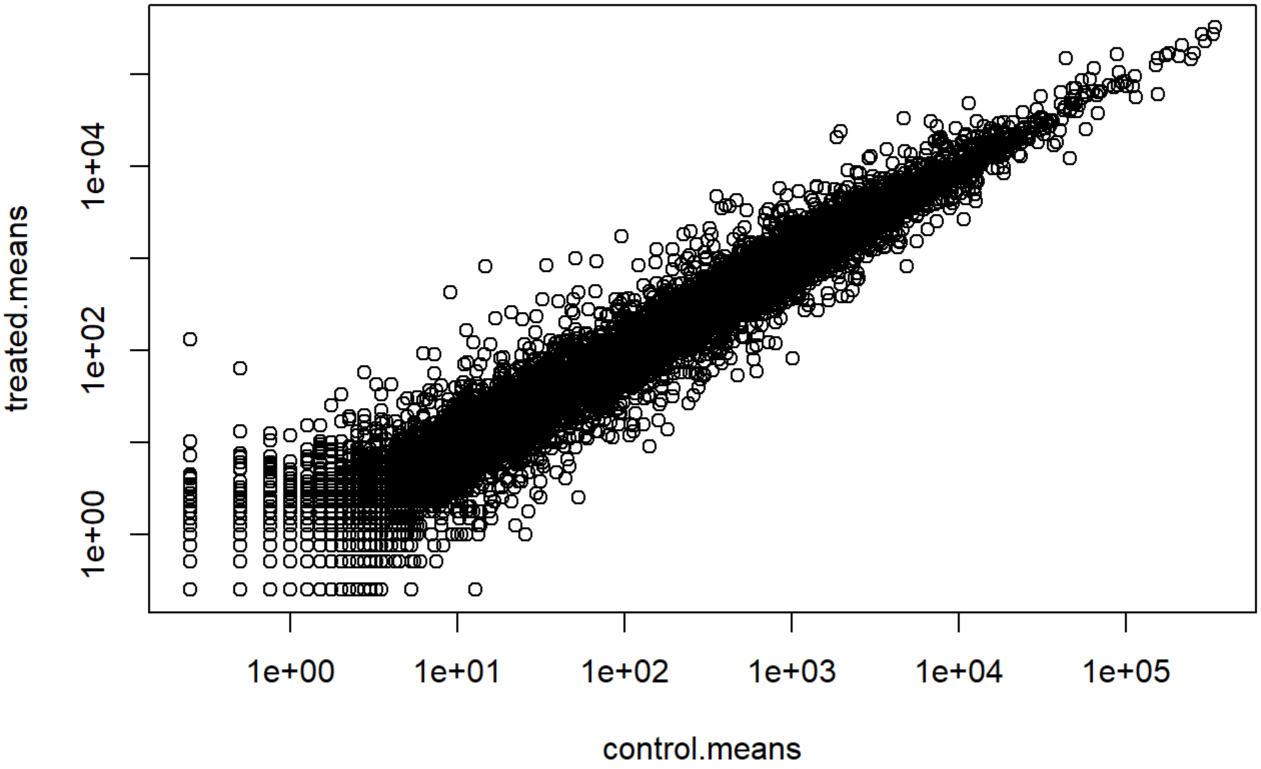
```
plot(meancounts)
```



```
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 use log transformations for skewed data such as this and because we really care most about relative changes in magnitude.

We most often use Log2 to transform as the math is easier to interpret than log10 or ln.

If we have no change - i.e. some values in control and treated we will have a log2 value of 0.

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

```
[1] 0
```

If I have double the amount of (20 compared to 10) I will have a log2 fold-change of +1

```r
log2(20/10)
```

```
[1] 1
```

If I have half the amount I will have a log2 fold-change of -1

```r
log2(10/20)
```

```
[1] -1
```

```
log2(40/10)
```

```
[1] 2
```

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

```
                control.means treated.means      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
```

A common rule of thumb is if the log-fold change is +2 or greater we consider that gene “up-regulated” and a change of at least -2 is considered “down-regulated”.

> Q. How many genes are up-regulated at the common threshold of +2 log2fc values?

```
sum(meancounts$log2fc >= 2 , na.rm = TRUE)
```

```
[1] 1910
```

Wait a damn minute! Yes these are big changes, but are they significant??

To do this properly, we will turn to the DESeq2 package.

DESeq2 Analysis

```
library(DESeq2)
```

To use DESeq, we need our input countdata and metadata in a specific format that DESeq wants:

```
dds <- DESeqDataSetFromMatrix(countData = countdata,
                       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
```

To run the analysis, I can now use the main DESeq2 function called `DESeq()` with `dds` as input

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

```
estimating size factors

estimating dispersions

gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing
```

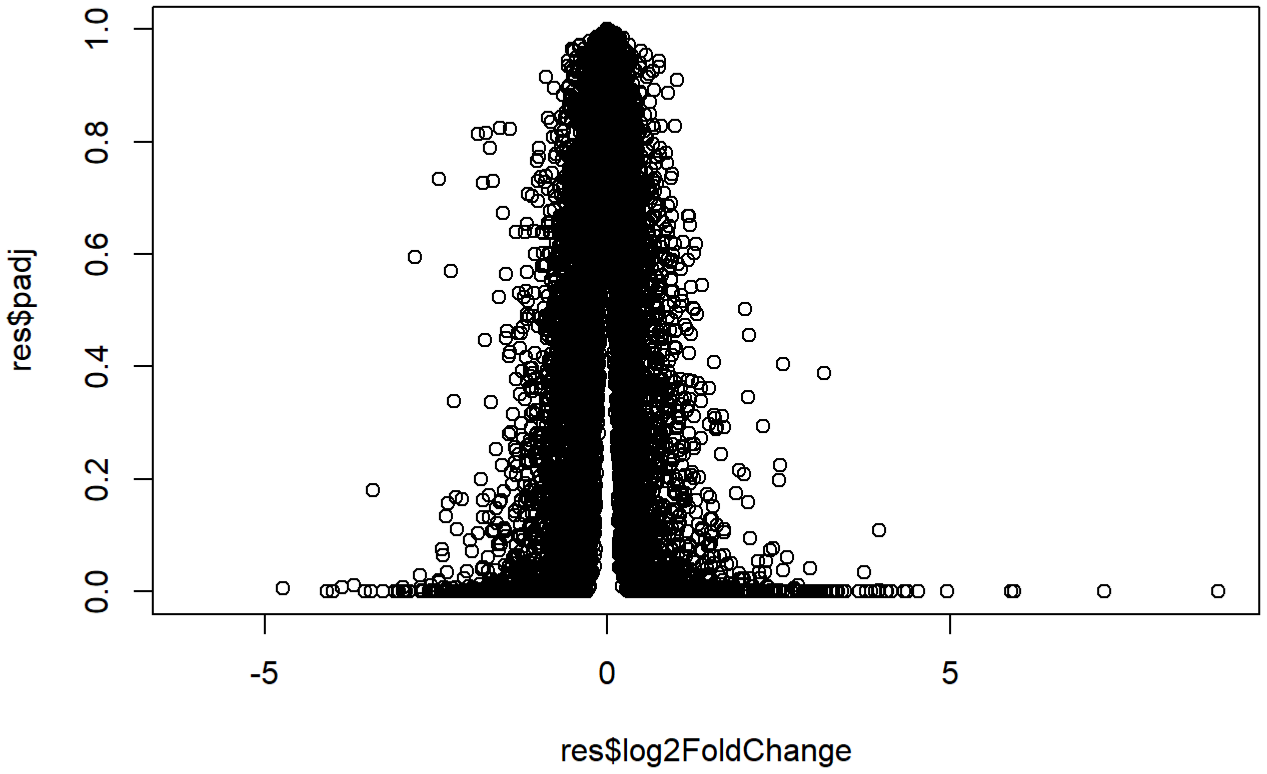
To get the results out of this `dds` object, we can use the `results()` function from the package.

```r
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
```

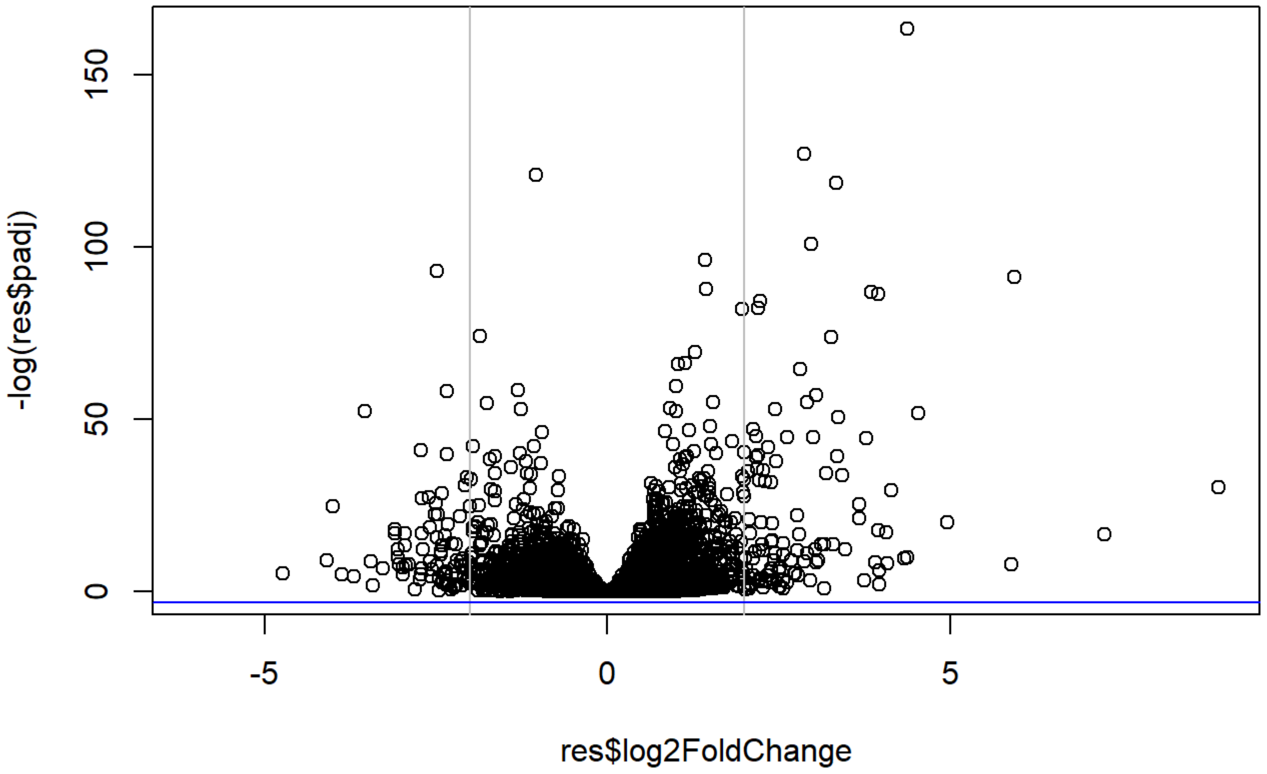
Lets make a final (for today) plot out of the log2fold change vs the adjusted P-value.

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



It is the low P-values that we care about and these are lost in the skered plot above. Let’s take the log of the $padj values for out plot.

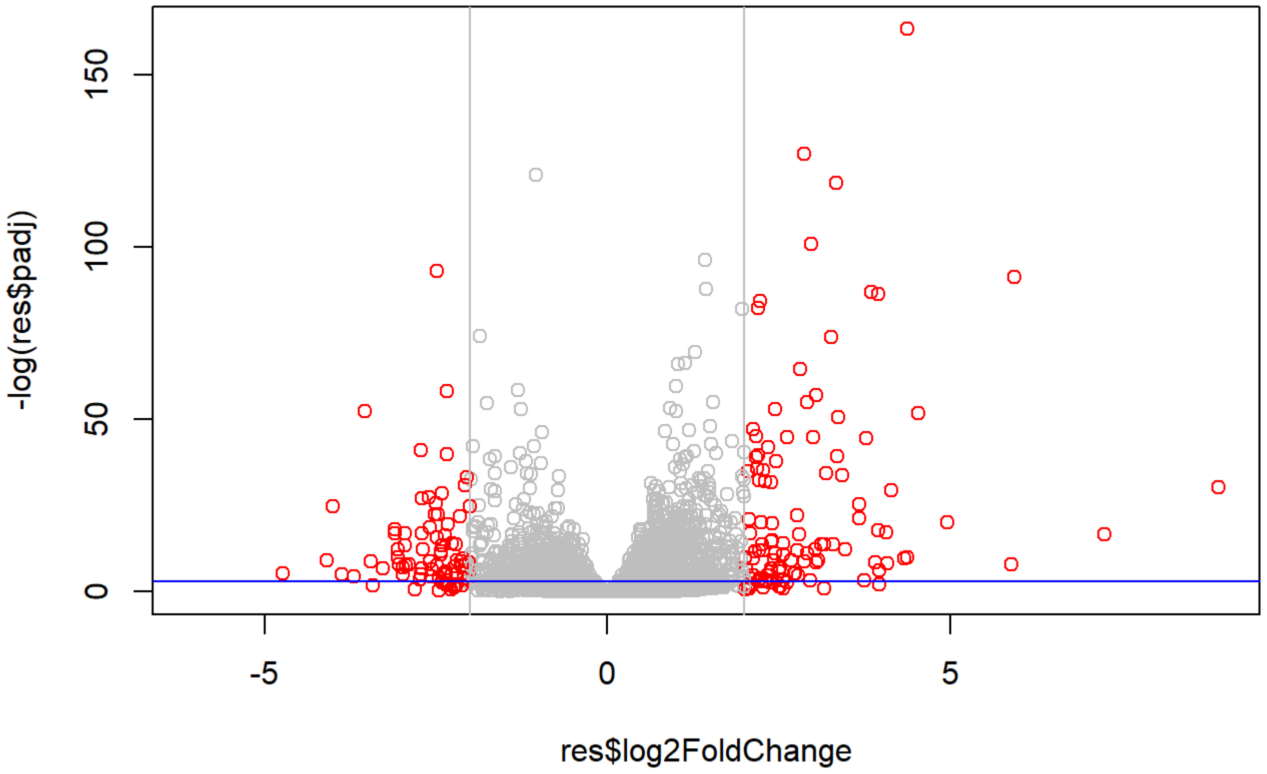
```
plot(res$log2FoldChange, -log(res$padj))
abline(v=c(+2,-2), col="gray")
abline(h=log(.05), col="blue")
```



Finally, we can make a color vector to use i the plot to better highlight the genes we care about.

```
mycols <- rep("gray", nrow(res))
mycols[res$log2FoldChange >= 2] <- "red"
mycols[res$log2FoldChange <= -2] <- "red"
mycols[res$padj > -log(.05)] <- "red"

plot(res$log2FoldChange, -log(res$padj), col=mycols)
abline(v=c(+2,-2), col="gray")
abline(h=-log(.05), col="blue")
```



Still to do: - add annotations (gene name, genome, etc) - save results to a CSV file - do some pathway analysis

```
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
```

Adding Annotation Data

We can use AnnotationDbi to package and add annotation data such as gene identifiers from different sources.

```
BiocManager::install("AnnotationDbi")
```

```
Bioconductor version 3.16 (BiocManager 1.30.20), R 4.2.3 (2023-03-15 ucrt)

Warning: package(s) not installed when version(s) same as or greater than current; use
  `force = TRUE` to re-install: 'AnnotationDbi'

Installation paths not writeable, unable to update packages
  path: C:/Program Files/R/R-4.2.3/library
  packages:
    class, KernSmooth, lattice, MASS, Matrix, nnet, survival

Old packages: 'cachem', 'DelayedArray', 'fs', 'httpuv', 'markdown', 'rlang',
  'sys', 'vctrs', 'xfun'
```

```
BiocManager::install("org.Hs.eg.db")
```

```
Bioconductor version 3.16 (BiocManager 1.30.20), R 4.2.3 (2023-03-15 ucrt)

Warning: package(s) not installed when version(s) same as or greater than current; use
  `force = TRUE` to re-install: 'org.Hs.eg.db'

Installation paths not writeable, unable to update packages
  path: C:/Program Files/R/R-4.2.3/library
  packages:
    class, KernSmooth, lattice, MASS, Matrix, nnet, survival
Old packages: 'cachem', 'DelayedArray', 'fs', 'httpuv', 'markdown', 'rlang',
  'sys', 'vctrs', 'xfun'
```

We can translate (a.k.a “map”) between all these database ID formats:

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

```
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"
```

```
res$symbol <- mapIds(org.Hs.eg.db,
                     keys=row.names(res), #genenames
                     keytype="ENSEMBL",   #current genename format
                     column="SYMBOL",     #new genename format
                     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 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    C1orf112
ENSG00000000938        NA         FGR
```

```
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
```

```
res$genename <- mapIds(org.Hs.eg.db,
                     keys=row.names(res),
                     keytype="ENSEMBL",
                     column="GENENAME",
                     multiVals="first")
```

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

```r
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      entrez               genename
                <numeric> <character> <character>            <character>
ENSG00000000003  0.163035      TSPAN6        7105          tetraspanin 6
ENSG00000000005        NA        TNMD       64102            tenomodulin
ENSG00000000419  0.176032        DPM1        8813 dolichyl-phosphate m..
ENSG00000000457  0.961694       SCYL3       57147 SCY1 like pseudokina..
ENSG00000000460  0.815849    C1orf112       55732 chromosome 1 open re..
ENSG00000000938        NA         FGR        2268 FGR proto-oncogene, ..
```

Save our results as a CSV file

```r
write.csv(res, file="myresultsc12.csv")
```

Pathway Analysis

We can use the KEGG database of biological pathways to get some more insight into our differentially expressed genes and the kinds of biology they are involved in.

```r
#l message: false
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).
##############################################################################
```

```r
library(gage)
```

```r
library(gageData)
```

```r
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"   
```

```r
head(res$entrez)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460 
         "7105"         "64102"          "8813"         "57147"         "55732" 
ENSG00000000938 
         "2268" 
```

Make a new vector of fold-change values that I will use as input ofr `gage()` this will have the ENTREZ IFs as names

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

```r
head(foldchanges)
```

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

```r
keggres = gage(foldchanges, gsets=kegg.sets.hs)
attributes(keggres)
```

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

```
# Look at the first three downregulated (less) pathways
head(keggres$less, 3)
```

```
                                      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
```

Now I can use the **KEGG IDs** of these pathways from gage to view our genes mapped to these pathways.

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

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

```
Info: Working in directory C:/Users/charl/OneDrive/Desktop/BIMM 143/Class 13
```

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

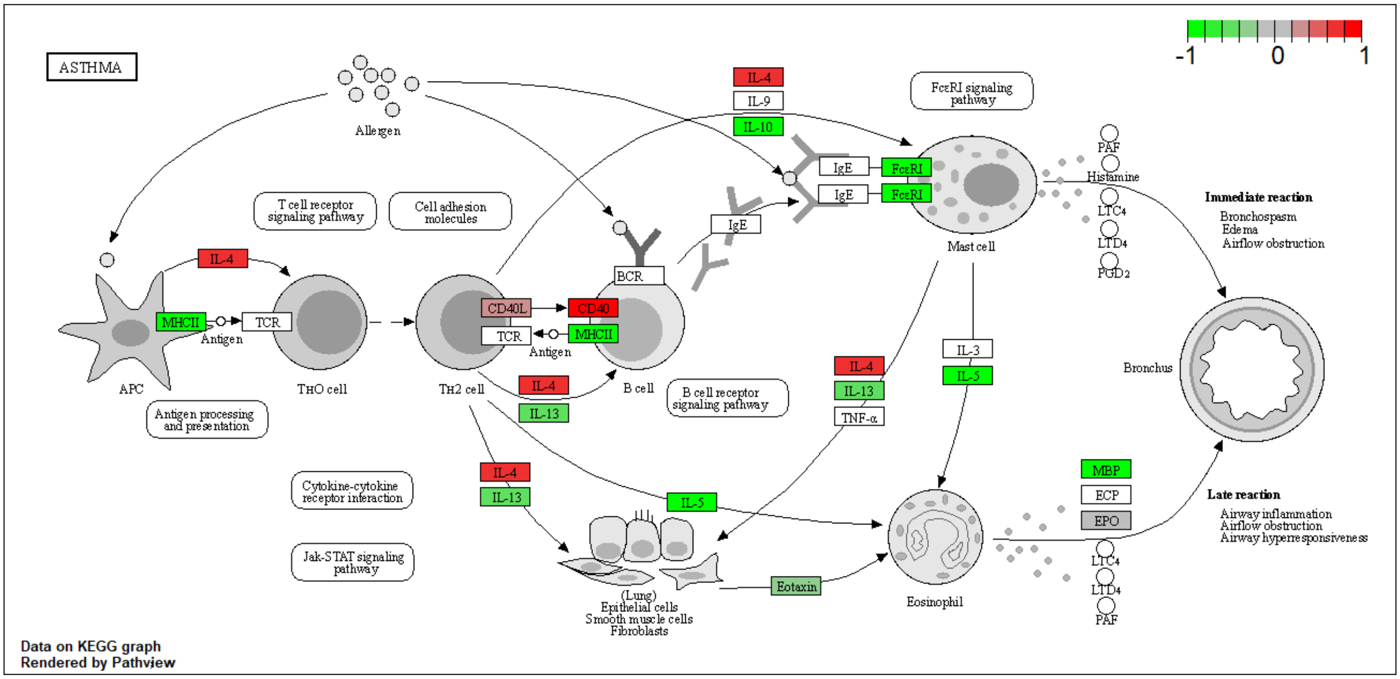


Fig. 1. A schematic overview of the asthma pathway including associated genes and their expression levels. Green coloration dennotes up-regulation during an asthma attack, red dennotes downregulation.