

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

Cienna Santos (PID: A17581026)

The data for this hands-on session comes 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/Read the data from Himes et al.

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

```
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
```

Santify check on correspondence of counts and metadata

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

```
[1] TRUE
```

Q1. How many genes are in this dataset?

There are 38694 genes in this dataset.

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

```
n.control <- sum(metadata$dex == "control")
```

There are 4 control cell lines in this dataset.

### Extract and summarize the control samples

To find out where the control samples are, we need the metadata.

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

ENSG00000000003	ENSG00000000005	ENSG000000000419	ENSG000000000457	ENSG000000000460
900.75	0.00	520.50	339.75	97.25
ENSG00000000938				
0.75				

### Extract and summarize the treated (i.e. drug) samples

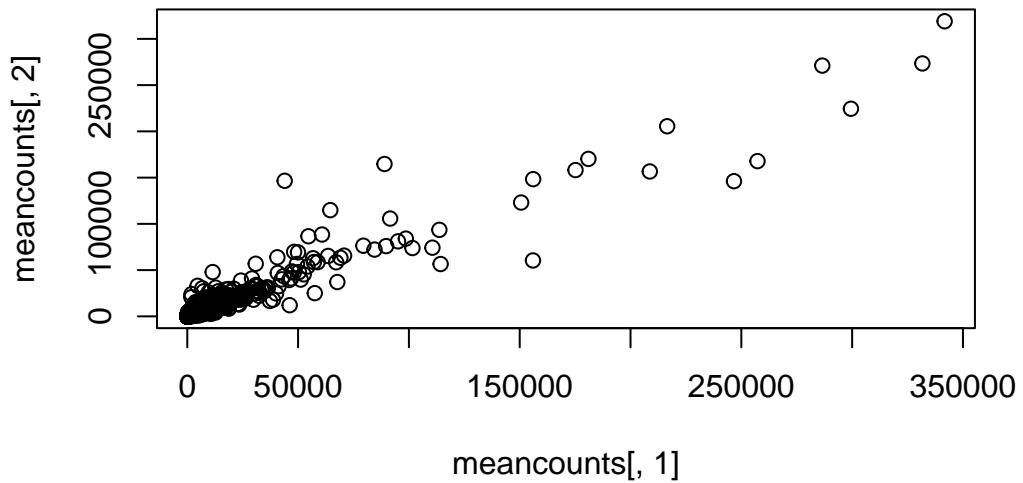
```
treated <- metadata[metadata$dex == "treated", ]
treated.counts <- counts[,treated$id]
treated.mean <- rowMeans(treated.counts)
```

Store these results together in a new data frame called `mean counts`

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

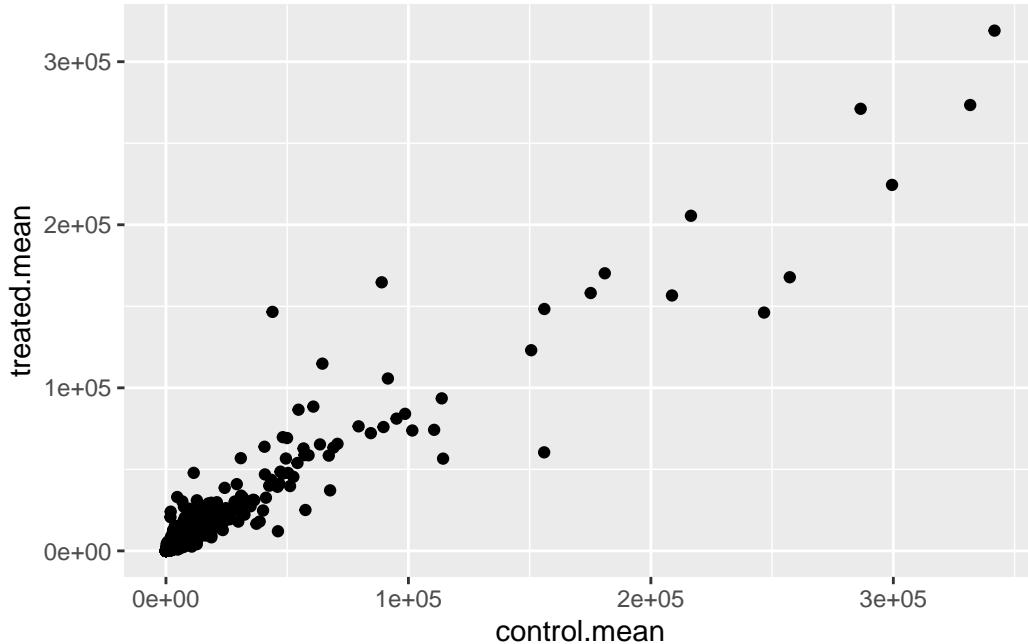
Let’s make a plot to explore the results alittle

```
plot(meancounts[,1], meancounts[,2])
```



```
library(ggplot2)

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

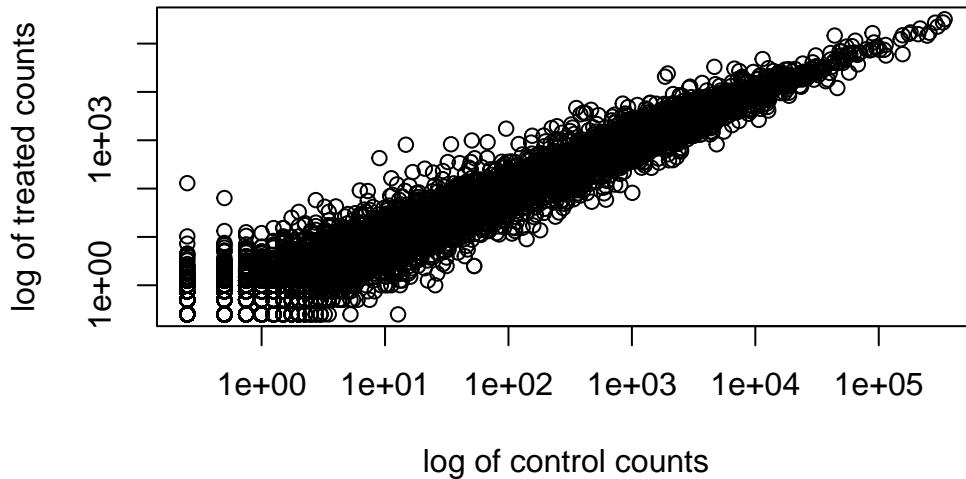


We'll make a log-log plot to draw out this skewed data and see what is going on.

```
plot(meancounts[,1], meancounts[,2], log='xy',
      xlab="log of control counts",
      ylab="log of treated counts")
```

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



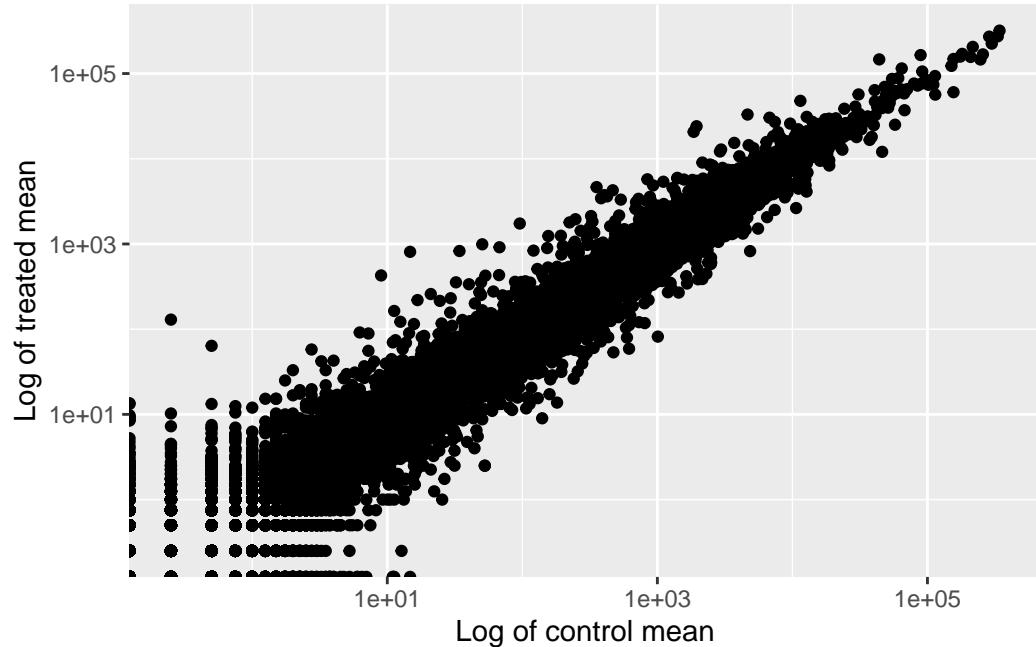
We often log2 transformations when dealing with this sort of data.

```
library(ggplot2)

ggplot(meancounts) +
  aes(control.mean, treated.mean) +
  geom_point() +
  scale_x_log10() +
  scale_y_log10() +
  labs(x = "Log of control mean", y = "Log of treated mean")
```

Warning in scale\_x\_log10(): log-10 transformation introduced infinite values.

Warning in scale\_y\_log10(): log-10 transformation introduced infinite values.



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

```
[1] 0
```

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

```
[1] 1
```

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

```
[1] -1
```

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

```
[1] 2
```

This log2 transformation has this nice property where if there is no change the log2 value will be zero and if it is double the log2 value will be 1 and if halved it will be -1.

So let's add a log2 fold change column to our results so far.

```
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
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

We need to get rid of zero count genes that we can not say anything about.

```
zero.values <- (which(meancounts[,1:2]==0, arr.ind=TRUE))
to.rm <- unique(zero.values[,1])
mycounts <- meancounts[-to.rm,]
```

```
head(mycounts)
```

	control.mean	treated.mean	log2fc
ENSG000000000003	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

How many genes are remaining?

```
nrow(mycounts)
```

```
[1] 21817
```

## Use fold change to see up and down regulated genes.

A common threshold used for calling something differentially expressed is a  $\log_2(\text{FoldChange})$  of greater than 2 or less than -2. Let's filter the dataset both ways to see how many genes are up or down-regulated.

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

```
[1] 250
```

and down regulated

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

```
[1] 367
```

Do we trust these results? Not fully because we don't yet know if these changes are significant...

## DESeq2 analysis

Let's do this the right way. DESeq2 is an R package specifically for analyzing count-based NGS data like RNA-seq.

```
# load up DESeq2
library(DESeq2)
```

```
Loading required package: S4Vectors
```

```
Loading required package: stats4
```

```
Loading required package: BiocGenerics
```

```
Attaching package: 'BiocGenerics'
```

```
The following objects are masked from 'package:stats':
```

```
IQR, mad, sd, var, xtabs
```

The following objects are masked from 'package:base':

```
anyDuplicated, aperm, append, as.data.frame, basename, cbind,
colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
Position, rank, rbind, Reduce, rownames, sapply, saveRDS, setdiff,
table, tapply, union, unique, unsplit, which.max, which.min
```

Attaching package: 'S4Vectors'

The following object is masked from 'package:utils':

```
findMatches
```

The following objects are masked from 'package:base':

```
expand.grid, I, unname
```

Loading required package: IRanges

Loading required package: GenomicRanges

Loading required package: GenomeInfoDb

Loading required package: SummarizedExperiment

Loading required package: MatrixGenerics

Loading required package: matrixStats

Attaching package: 'MatrixGenerics'

```
The following objects are masked from 'package:matrixStats':
```

```
colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
colWeightedMeans, colWeightedMedians, colWeightedSds,
colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
rowWeightedSds, rowWeightedVars
```

```
Loading required package: Biobase
```

```
Welcome to Bioconductor
```

```
Vignettes contain introductory material; view with
'browseVignettes()'. To cite Bioconductor, see
'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
Attaching package: 'Biobase'
```

```
The following object is masked from 'package:MatrixGenerics':
```

```
rowMedians
```

```
The following objects are masked from 'package:matrixStats':
```

```
anyMissing, rowMedians
```

```
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)  
res
```

```
log2 fold change (MLE): dex treated vs control
```

```
Wald test p-value: dex treated vs control
```

```
DataFrame with 38694 rows and 6 columns
```

	baseMean	log2FoldChange	lfcSE	stat	pvalue
	<numeric>	<numeric>	<numeric>	<numeric>	<numeric>
ENSG000000000003	747.1942	-0.3507030	0.168246	-2.084470	0.0371175
ENSG000000000005	0.0000	NA	NA	NA	NA
ENSG000000000419	520.1342	0.2061078	0.101059	2.039475	0.0414026
ENSG000000000457	322.6648	0.0245269	0.145145	0.168982	0.8658106
ENSG000000000460	87.6826	-0.1471420	0.257007	-0.572521	0.5669691
...	...	...	...	...	...
ENSG00000283115	0.000000	NA	NA	NA	NA
ENSG00000283116	0.000000	NA	NA	NA	NA
ENSG00000283119	0.000000	NA	NA	NA	NA
ENSG00000283120	0.974916	-0.668258	1.69456	-0.394354	0.693319
ENSG00000283123	0.000000	NA	NA	NA	NA

```
padj
```

```
<numeric>
ENSG000000000003 0.163035
ENSG000000000005 NA
ENSG000000000419 0.176032
ENSG000000000457 0.961694
ENSG000000000460 0.815849
...
ENSG00000283115 NA
ENSG00000283116 NA
ENSG00000283119 NA
ENSG00000283120 NA
ENSG00000283123 NA
```

We can get some basic summary tallies using the `summary()` function

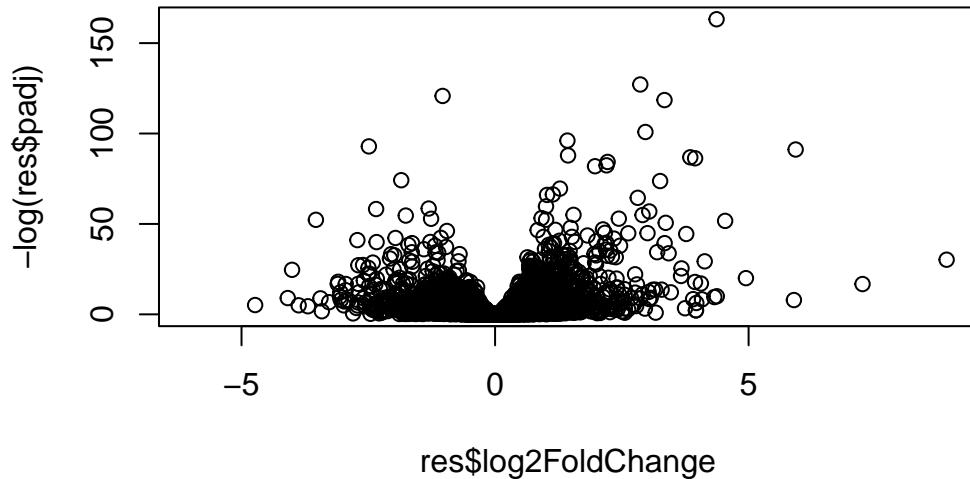
```
summary(res, alpha=0.05)
```

```
out of 25258 with nonzero total read count
adjusted p-value < 0.05
LFC > 0 (up)      : 1242, 4.9%
LFC < 0 (down)     : 939, 3.7%
outliers [1]       : 142, 0.56%
low counts [2]     : 9971, 39%
(mean count < 10)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```

## Volcano plot

Make a summary plot of our results.

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



```
log(0.1)
```

```
[1] -2.302585
```

```
log(0.005)
```

```
[1] -5.298317
```

```
write.csv(res, file="DESeq2.results.csv")
```

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

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

```
[1] "ACNUM"          "ALIAS"           "ENSEMBL"         "ENSEMLPROT"      "ENSEMLTRANS"
[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"
```

Add a few more ID mappings including “GENENAME” 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 8 columns

	baseMean	log2FoldChange	lfcSE	stat	pvalue
	<numeric>	<numeric>	<numeric>	<numeric>	<numeric>
ENSG000000000003	747.194195	-0.3507030	0.168246	-2.084470	0.0371175
ENSG000000000005	0.000000	NA	NA	NA	NA
ENSG000000000419	520.134160	0.2061078	0.101059	2.039475	0.0414026
ENSG000000000457	322.664844	0.0245269	0.145145	0.168982	0.8658106
ENSG000000000460	87.682625	-0.1471420	0.257007	-0.572521	0.5669691
ENSG000000000938	0.319167	-1.7322890	3.493601	-0.495846	0.6200029
	padj		name	entrez	
	<numeric>		<character>	<character>	
ENSG000000000003	0.163035	tetraspanin 6		7105	
ENSG000000000005	NA	tenomodulin		64102	
ENSG000000000419	0.176032	dolichyl-phosphate m..		8813	
ENSG000000000457	0.961694	SCY1 like pseudokina..		57147	
ENSG000000000460	0.815849	FIGNL1 interacting r..		55732	
ENSG000000000938	NA	FGR proto-oncogene, ..		2268	

Be sure to save our annotated results to a file.

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

## Pathway analysis

Install the packages we need for pathway analysis: Run in your R console (i.e. not your Quarto doc!) `BiocManager::install(c("pathview", "gage", "gageData"))`

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

Let's peek at the gageData

```
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 Log2FC 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

```

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

What is in the returned `keggres` object

```
attributes(keggres)
```

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

```
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 <u>Asthma</u>	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 <u>Asthma</u>	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 <u>Asthma</u>	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 pass our foldchanges vector (our results) together with any of these highlighted pathway IDs to see how our genes overlap the pathway.

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

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

Info: Working in directory /Users/ciennasantos/Documents/school/bimm143/class13

Info: Writing image file hsa05310.pathview.png

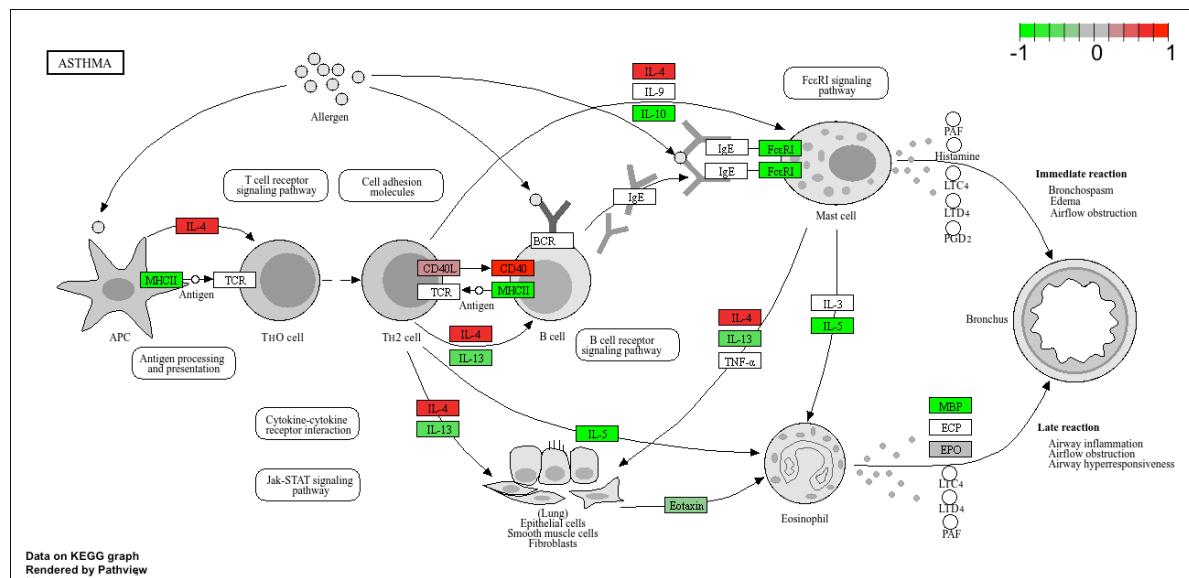


Figure 1: The Asthma pathway overlaps with our differentially expressed genes