

Lab 13: Transcriptomics RNA Seq

Trevor Hoang (A16371830)

Today we will analyze 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 countData and colData

There are 2 datasets I need to import/read

-`countData` the transcript counts per gene (rows) in the different experiments -`colData` information about the columns (i.e. experiments) in `countData`.

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

We can have a little peak at these with `head()`

```
head(counts)
```

	SRR1039508	SRR1039509	SRR1039512	SRR1039513	SRR1039516
ENSG000000000003	723	486	904	445	1170
ENSG000000000005	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		
ENSG000000000003	1097	806	604		
ENSG000000000005	0	0	0		
ENSG00000000419	781	417	509		
ENSG00000000457	447	330	324		
ENSG00000000460	94	102	74		
ENSG00000000938	0	0	0		

```
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
7 SRR1039520 control    N061011 GSM1275874
8 SRR1039521 treated    N061011 GSM1275875
```

Q1. How many genes are in this dataset?

```
nrow(counts)
```

```
[1] 38694
```

Q2. How many `control` cell lines are in this dataset?

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

```
control treated
        4       4
```

We can find the average (mean) count value per gene for all “control” experiments and compare it to the mean value for “treated”.

- Extract all the “control” columns from the `counts` data
- Find the mean value for each gene in these columns

```
control inds = metadata$dex == "control"
control counts = counts[, control inds]
```

```
dim(control counts)
```

```
[1] 38694      4
```

Now find the row wise mean

```
control.mean = rowSums(control.counts)/ncol(control.counts)
head(control.mean)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
    900.75          0.00      520.50      339.75      97.25
ENSG000000000938
    0.75
```

Q4. Follow the same procedure for the treated samples (i.e. calculate the mean per gene across drug treated samples and assign to a labeled vector called treated.mean)

```
treated inds = metadata$dex == "treated"
treated.counts = counts[, treated.inds]
```

```
dim (treated.counts)
```

```
[1] 38694     4
```

```
treated.mean = rowSums(treated.counts)/ncol(treated.counts)
head(treated.mean)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
    658.00          0.00      546.00      316.50      78.75
ENSG000000000938
    0.00
```

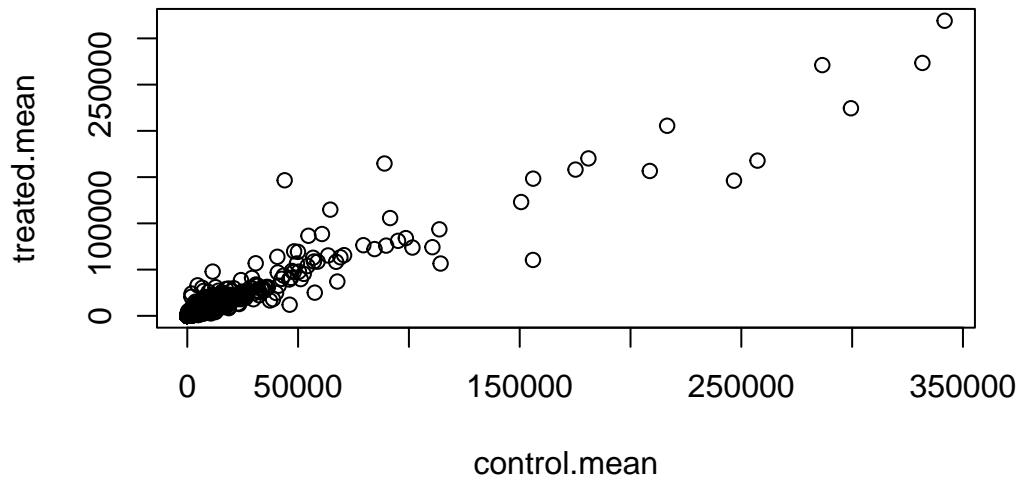
Let's put these two mean values together for easy book keeping

```
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

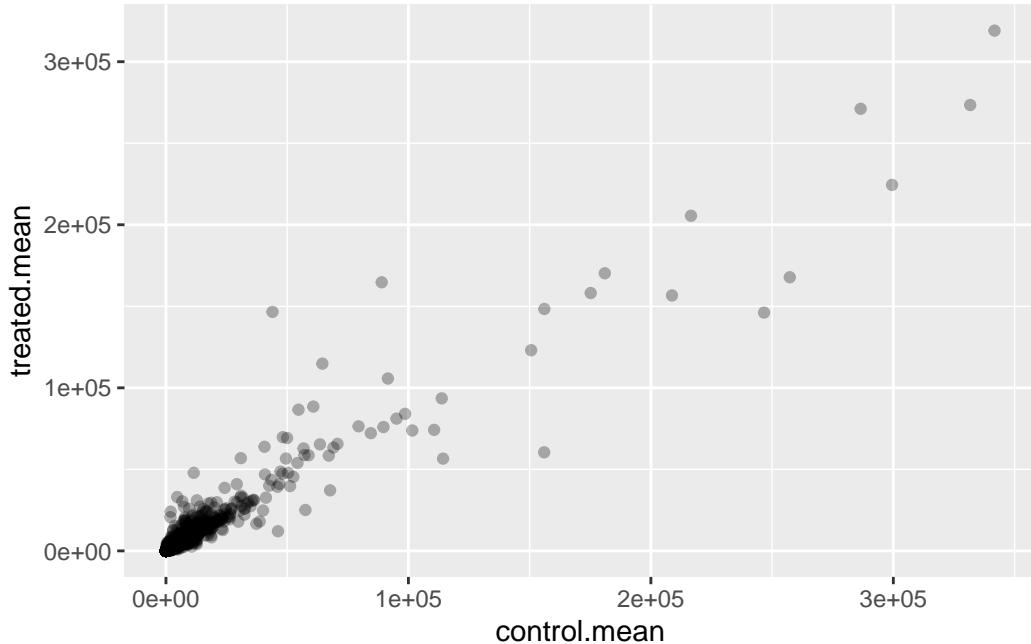
Let's have a look - i.e. plot control.mean vs treated.mean

```
plot(meancounts)
```



```
library(ggplot2)

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

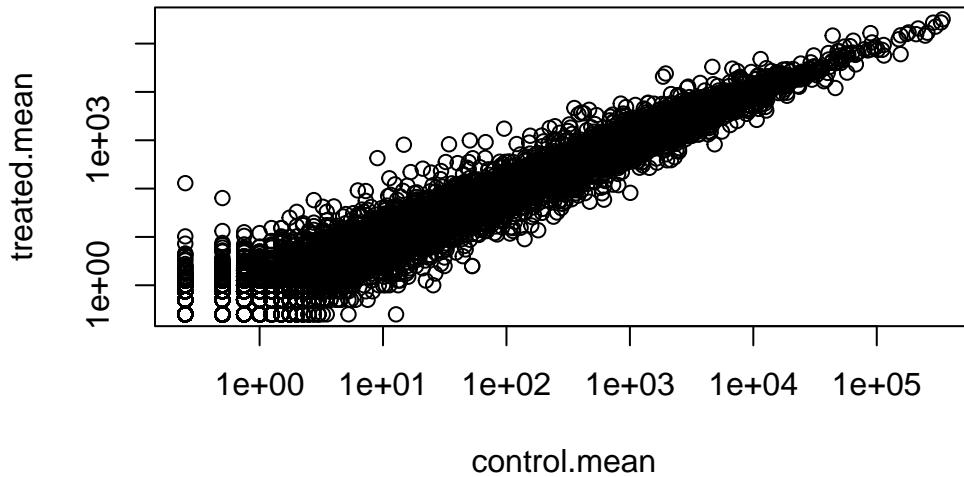


Whenever we see data that is so heavily skewed like this, we often log transform it so we can see what is going on more easily.

```
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 most of ten work in log2 units as this makes the math easier. Let's have a look to see this

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

```
[1] 1
```

We can now add “log2 fold-change” values to our `meancounts` dataset

```
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
ENSG000000000419	520.50	546.00	0.06900279
ENSG000000000457	339.75	316.50	-0.10226805
ENSG000000000460	97.25	78.75	-0.30441833
ENSG000000000938	0.75	0.00	-Inf

We need to filter out zero count genes - i.e. remove the rows (genes) that have a 0 value in either control or treated means.

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

```
[1] 21817
```

How many genes are “up” regulated at the common log2 fold-change threshold of +2.

```
up inds = meancounts$log2fc >= 2  
sum(up inds, na.rm=T)
```

```
[1] 1910
```

How many genes are “down” regulated at the threshold of -2?

```
down inds = meancounts$log2fc <= -2  
sum(down inds, na.rm=T)
```

```
[1] 2330
```

Q10. Do you trust these results? I trust these results because they are within the parameters that I have set.

```
##DESeq2 Analysis
```

To do this the right way we need to consider the significance of the differences not just their magnitude

```
library(DESeq2)
```

To use this package, it wants countData and colData in a specific 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

Extract my results

```
res = results(dds)  
head(res)
```

log2 fold change (MLE): dex treated vs control

Wald test p-value: dex treated vs control

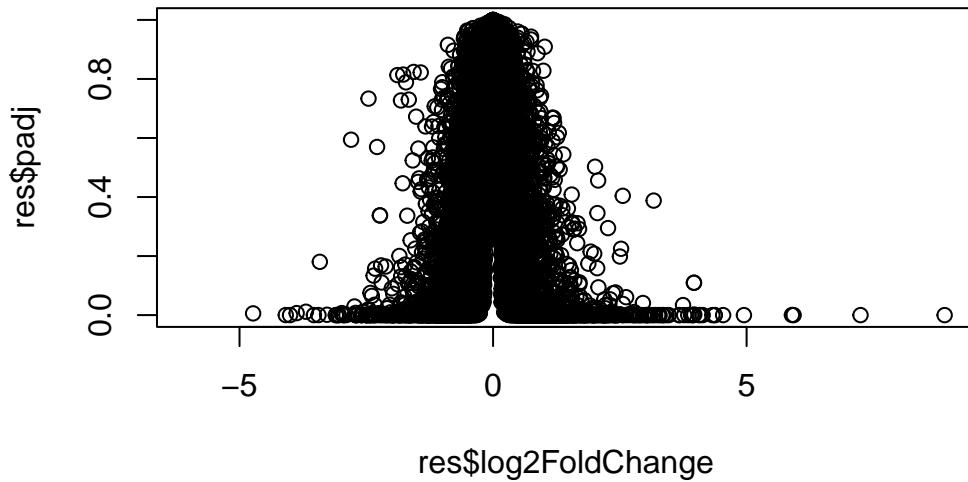
DataFrame with 6 rows and 6 columns

	baseMean	log2FoldChange	lfcSE	stat	pvalue
	<numeric>	<numeric>	<numeric>	<numeric>	<numeric>
ENSG000000000003	747.194195	-0.3507030	0.168246	-2.084470	0.0371175
ENSG000000000005	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>
ENSG000000000003	0.163035
ENSG000000000005	NA
ENSG00000000419	0.176032
ENSG00000000457	0.961694
ENSG00000000460	0.815849
ENSG00000000938	NA

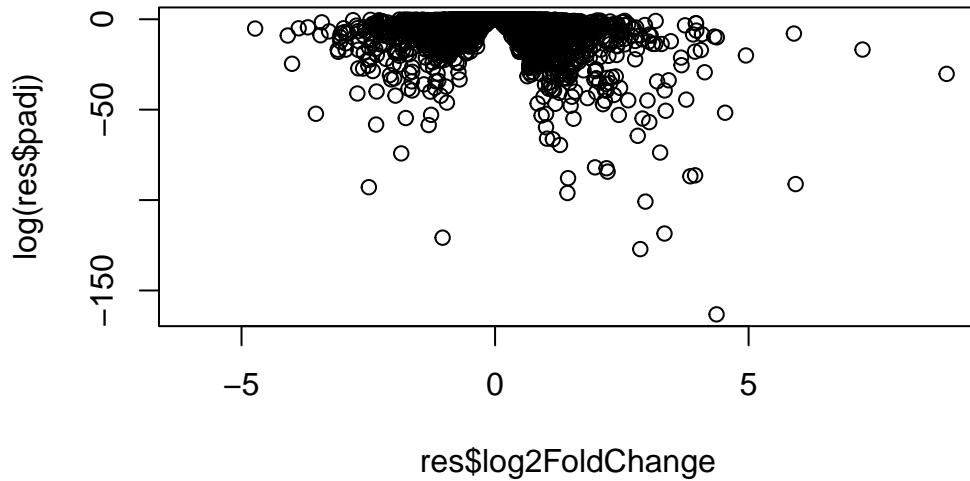
Plot of fold-change vs P-value (adjusted for multiple testing)

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



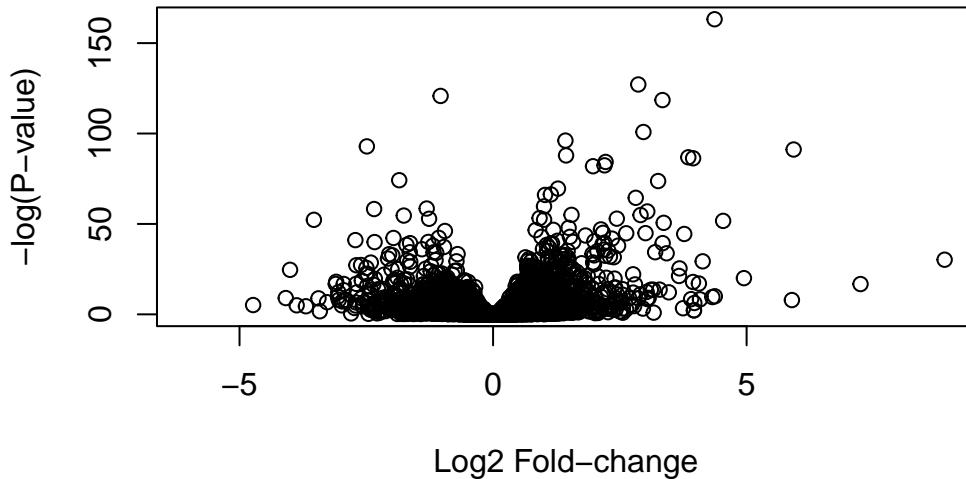
Take the log of the P-value

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



We can just flip the y axis by putting a minus in front of the y axis

```
plot(res$log2FoldChange, -log(res$padj),  
     xlab = "Log2 Fold-change",  
     ylab = "-log(P-value)")
```



Let's save our work to date

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

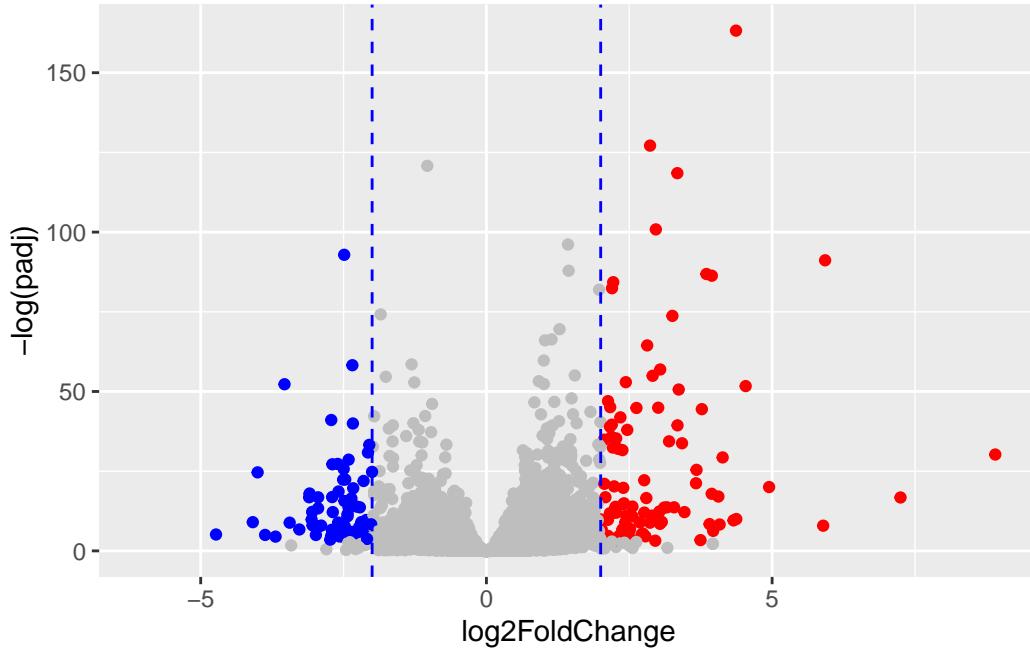
To finish off let's make a nicer volcano plot

- Add the log2 threshold lines of +2/-2
- Add P-value threshold lines at 0.05
- Add color to highlight the subset of genes that meet both of the above thresholds

```
mycols = rep("gray", nrow(res))
mycols[res$log2FoldChange >= 2] <- "red"
mycols[res$log2FoldChange <= -2] <- "blue"
mycols[res$padj > 0.05] = "grey"
```

```
ggplot(res) +
  aes(log2FoldChange, -log(padj)) +
  geom_point(col = mycols) +
  geom_vline(xintercept= c(2,-2), col= "blue", linetype = "dashed")
```

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



Add gene annotation data

Now the question is what are the blue points in the above volcano plots

We will use some BioConductor packages to “map” the ENSEMBLE ids to more useful gene SYMBOL names/ids

We can install these packages with: `BiocManager::install("AnnotationDbi")`

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

What database identifiers can I translate between here:

```
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"
```

We can now use the `mapIDs()` function to translate/map between these different identifier formats.

Let's add SYMBOL, GENENAME and ENTREZID

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

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

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

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

```
res$entrez =mapIds(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>
ENSG000000000003 747.194195 -0.3507030  0.168246 -2.084470 0.0371175
ENSG000000000005  0.000000      NA        NA        NA        NA
ENSG00000000419   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	symbol		genename	entrez
	<numeric>	<character>		<character>	<character>
ENSG000000000003	0.163035	TSPAN6	tetraspanin	6	7105
ENSG000000000005	NA	TNMD	tenomodulin		64102
ENSG000000000419	0.176032	DPM1	dolichyl-phosphate m..		8813
ENSG000000000457	0.961694	SCYL3	SCY1 like pseudokina..		57147
ENSG000000000460	0.815849	FIRRM	FIGNL1 interacting r..		55732
ENSG000000000938	NA	FGR	FGR proto-oncogene, ..		2268

##Pathway analysis Now I know the gene names and their IDs in different databases I want to know what type of biology they are involved in.

This is the job of “pathway analysis” (a.k.a. “gene set enrichment”)

There are tons of different BioConductor packages for pathway analysis here we use just one of them called **gage**, and **pathview**. I will install these packages with `BiocManager::install(c("gage","pathview","gageData"))`

```
library(gage)
```

```
library(gageData)
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>).

```
#####
```

Load up the KEGG genesets

```

data(kegg.sets.hs)

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"

```

We will use these KEGG genesets (aka pathways) and our `res` results to see what overlaps. To do this we will use the `gage()` function.

For input `gage()` wants just a vector of importance - in our case FoldChange values.

```

foldchange = res$log2FoldChange

```

Vectors in R can have “names” that are useful for bookkeeping so we know what a given value corresponds to, e.g.:

```

x = c(10, 100, 20)
names(x) = c("barry", "alex", "chandra")
x

```

```

barry      alex chandra
10        100      20

```

Let’s put names in our `foldchanges` vector - here we will use `res$entrez`

```

names(foldchange) = res$entrez

```

Now we can run “pathway analysis”

```

keggres = gage(foldchange, 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 get a pathway image file with our genesets highlighted via the `pathview()` function.

```

pathview(foldchange, pathway.id = "hsa05310")

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

Info: Working in directory C:/Users/treer/OneDrive/Documents/BIMM 143/Lab 13

Info: Writing image file hsa05310.pathview.png

```

Insert this figure in my report

