

Class 12: RNAseq Analysis

Peter Shamasha (A15857589)

Transcriptomics and Analysis of RNA-seq Data

Here we will use the DESeq2 package for RNAseq analysis. The data for todays class comes from a study of airway smooth muscle cells were treated with dexamethasone, a synthetic glucocorticoid steroid with anti-inflammatory effects (Himes et al. 2014).

Import their data

We need two thing sfor this analysis:

- **countData** (counts for every transcript/gene in each experiment)
- **colData** (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")
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(countData)
```

```
[1] 38694
```

There are 38,694 genes in this dataset

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

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

```
control treated
        4         4
```

Another way

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

```
[1] 4
```

- Step 1. Calculate the mean of the control samples (i.e. columns in countData) Calculate the mean of the treated samples
 - (a) we need to find which columns in countData are “control” samples.
 - look in the metadata (a.k.a. metaData), \$dex column

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

- (b) Extract all the control columns from countData and call it control.counts

```
control counts <- countData[, control inds]
```

- (c) Calculate the mean value across 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
ENSG000000000938
         0.75
```

- Step 2. Calculate the mean of the treated samples

```
treated means <- rowMeans( countData[, metadata$dex == "treated"])
head(treated means)
```

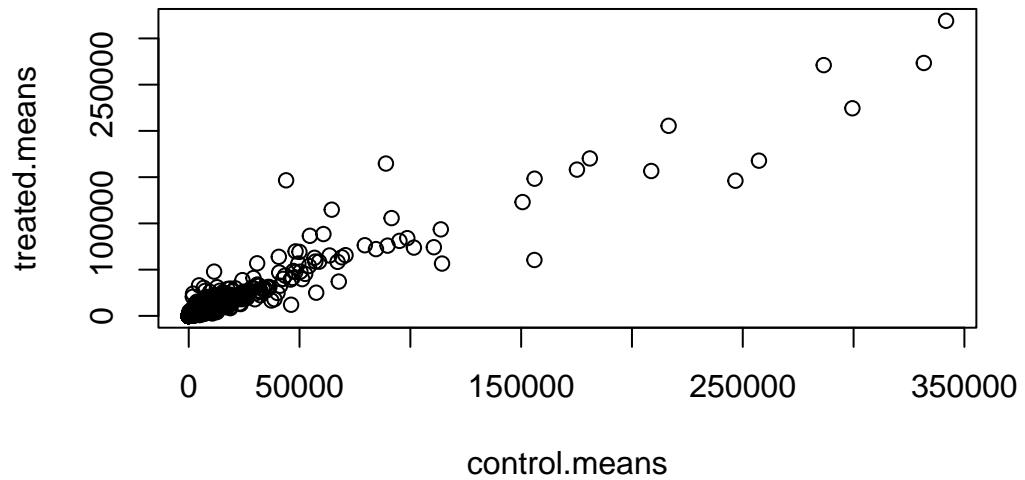
```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
         658.00          0.00        546.00        316.50        78.75
ENSG000000000938
         0.00
```

We now have control and treated mean count values. For ease of book-keeping, I will combine these vectors into a new data.frame called meancounts

```
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
ENSG000000000938	0.75	0.00

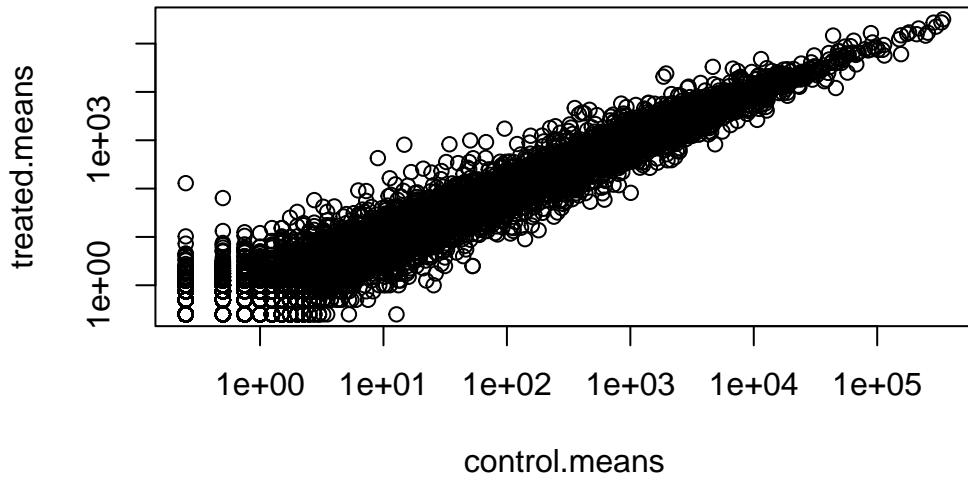
```
plot(meancounts)
```



```
plot.default(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 transforms for skewed data such as this and because we really care most about relative changes in magnitude.

We most often used log2 as our transform as the math is easier to interpret than log10 or others

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

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

```
[1] 0
```

If I have double the amount i.e. 20 compared to 10 for example, I will have a log2 fold-change of 1.

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

```
[1] 1
```

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

```
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
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

Q. How many genes are upregulated at the common threshold of +2 logFC values?

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

```
[1] 1910
```

Hold on what about stats! Yes these are big changes but are these changes significant!

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

DESeq2 Analysis

```
library(DESeq2)
```

To use DESeq2 we need our input countData and colData 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 functions called `DESeq()` with `dds` as input.

```
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 the `dds` object we can use the `results()` function from the package

```
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
```

```

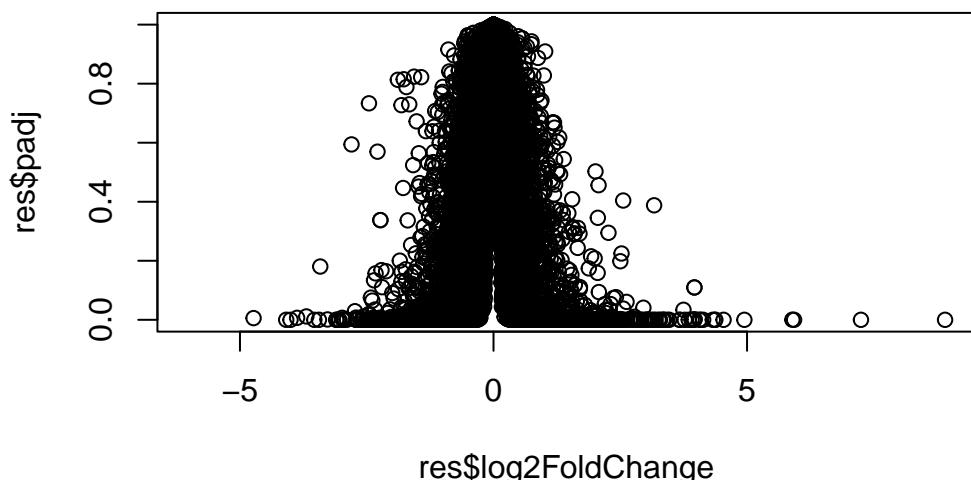
ENSG000000000005  0.000000          NA          NA          NA          NA
ENSG000000000419  520.134160      0.2061078  0.101059  2.039475  0.0414026
ENSG000000000457  322.664844      0.0245269  0.145145  0.168982  0.8658106
ENSG000000000460  87.682625      -0.1471420  0.257007  -0.572521  0.5669691
ENSG000000000938  0.319167      -1.7322890  3.493601  -0.495846  0.6200029
                           padj
                           <numeric>
ENSG000000000003  0.163035
ENSG000000000005  NA
ENSG000000000419  0.176032
ENSG000000000457  0.961694
ENSG000000000460  0.815849
ENSG000000000938  NA

```

Volcano Plot

Let's make a final (for today) plot of log2 fold-change vs the adjusted P-value.

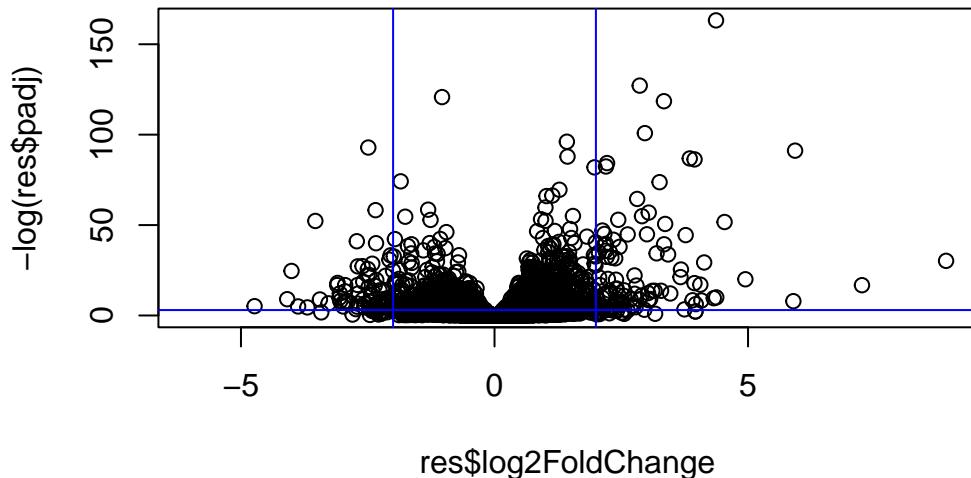
```
plot(res$log2FoldChange, res$padj)
```



It is the low P-values that we care about and these are lost in the skewed plot above. Let's

take the log of \$padj values for the plot.

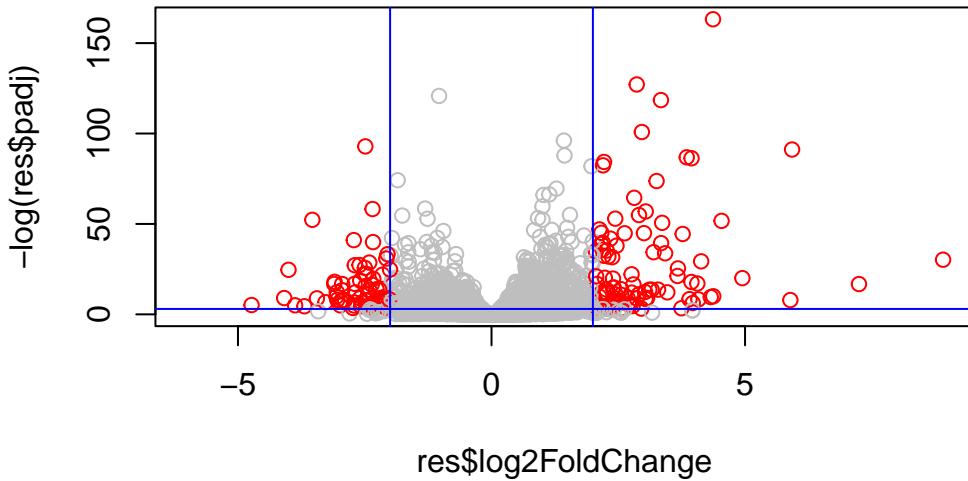
```
plot(res$log2FoldChange, -log(res$padj))
abline(v=c(+2,-2), col="blue")
abline(h=-log(0.05), col="blue")
```



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

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

plot(res$log2FoldChange, -log(res$padj), col=mycols)
abline(v=c(+2,-2), col="blue")
abline(h=-log(0.05), col="blue")
```



Still to do:

- Add annotation (including gene symbols, entrezid and genename)
- save results as 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>
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
ENSG000000000419  0.176032
ENSG000000000457  0.961694
ENSG000000000460  0.815849
ENSG000000000938      NA

```

Adding annotation data

We can use the AnnotationDbi package to add annotation data such as gene identifiers from different sources to our results object.

```

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

```

We can translate (A.K.A. “map”) between all these database id formats:

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

```

[1] "ACCCNUM"        "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"

```

My IDs are stored as the rownames of `res`

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

```

[1] "ENSG000000000003" "ENSG000000000005" "ENSG000000000419" "ENSG000000000457"
[5] "ENSG000000000460" "ENSG000000000938"

```

```

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

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>
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      symbol
  <numeric> <character>
ENSG000000000003 0.163035      TSPAN6
ENSG000000000005  NA          TNMD
ENSG000000000419 0.176032      DPM1
ENSG000000000457 0.961694      SCYL3
ENSG000000000460 0.815849      C1orf112
ENSG000000000938  NA          FGR

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

```

```
    multiVals="first")  
  
'select()' returned 1:many mapping between keys and columns
```

Save our results to date

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

Pathway Analysis

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

```
#1 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>).

```
#####
```

```
library(gage)
```

```
library(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"

head(res$entrez)

ENSG00000000003 ENSG00000000005 ENSG000000000419 ENSG000000000457 ENSG000000000460
"7105"           "64102"          "8813"          "57147"          "55732"
ENSG000000000938
"2268"

Make a new vector of fold-change values that I will use as input for gage this will have the
ENTREZ IDs as names.

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

x <- 1:3
x

[1] 1 2 3

```

```

names(x) <- c("Rob", "Bob", "John")
x

Rob   Bob  John
1     2     3

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)

attributes(keggres)

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

Look at the top 3 "LESS"

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/peter/Desktop/BIMM 143/class13

Info: Writing image file hsa05310.pathview.png

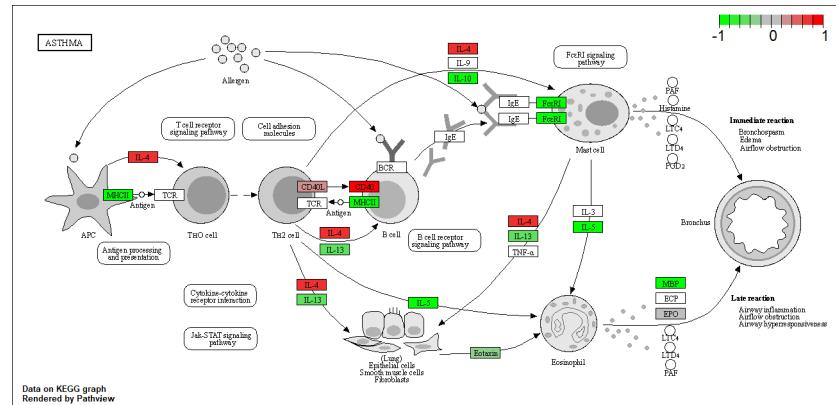


Figure 1: Asthma pathway from KEGG with our genes shown in color