class12

Kelsey Fierro

Install DESeq2: install.packages("BiocManager") BiocManager::install("DESeq2")

```r
library(BiocManager)
```

```r
library(DESeq2)
```

```
Loading required package: S4Vectors

Loading required package: stats4

Loading required package: BiocGenerics

Loading required package: generics


Attaching package: 'generics'

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

    as.difftime, as.factor, as.ordered, intersect, is.element, setdiff,
    setequal, union


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, is.unsorted, lapply, Map, mapply, match, mget,
    order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
    rbind, Reduce, rownames, sapply, saveRDS, table, tapply, 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: Seqinfo

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
```

Background

Today we wull analyze some RNAseq data from Himes et al. on the effects of a common steroid (dexmethasone also called "dex") on airway smooth muscle cells (ASMs). For this analysis we need two main inputs: 1. countData: a table of **counts** per gene (in rows) across experiments (in columns) 2. colData: **metadata** about the design of the experiments, the rows here must match the columns in `countData`

Data Import

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

Let's have a wee peek at our `counts` data

```
head(counts)
```

```
                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
```

Add the `metadata`

```
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 experiments (i.e. columns in `counts` or rows in `metadata`) are there?

```
ncol(counts)
```

```
[1] 8
```

> Q3. How many "control" experiments are there in the dataset?

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

```
[1] 4
```

Toy analysis example

1. Extract the "control" columns from `counts`
2. Calculate the mean value for each gene (rows) in these "control" columns 11/9/25, 11:25 PM 10/15/2025 - Posit Cloud https://posit.cloud/spaces/707042/content/11139485 4/7 3-4. Do the same for the "treated" columns
3. Compare these mean values for each gene

Step 1.

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

Step 2.

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

Step 3 and 4.

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

```
                SRR1039509 SRR1039513 SRR1039517 SRR1039521
ENSG00000000003        486        445       1097        604
ENSG00000000005          0          0          0          0
ENSG00000000419        523        371        781        509
ENSG00000000457        258        237        447        324
ENSG00000000460         81         66         94         74
ENSG00000000938          0          0          0          0
```

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

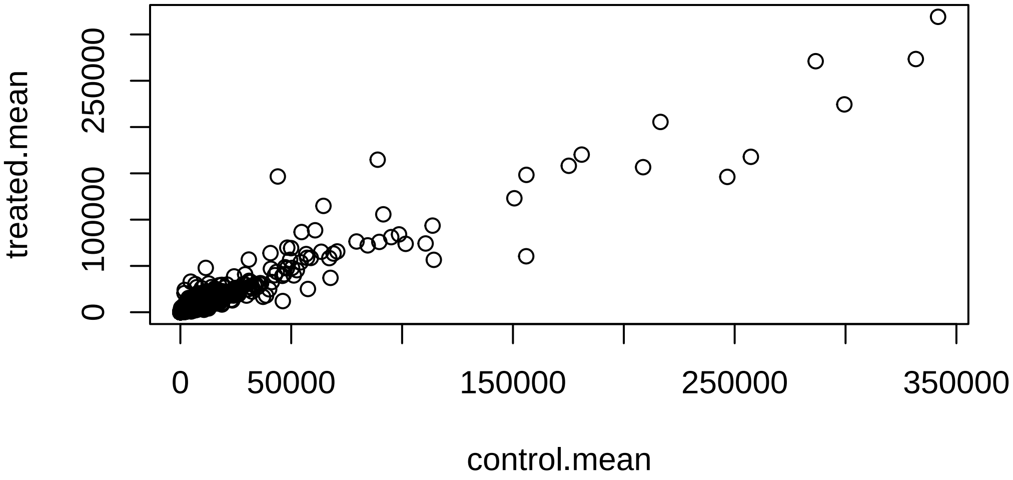
Step 5. For ease of book-keeping we can store these together in one data frame called `meancounts`

```
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
```

Plot these against each other

```
plot(meancounts)
```

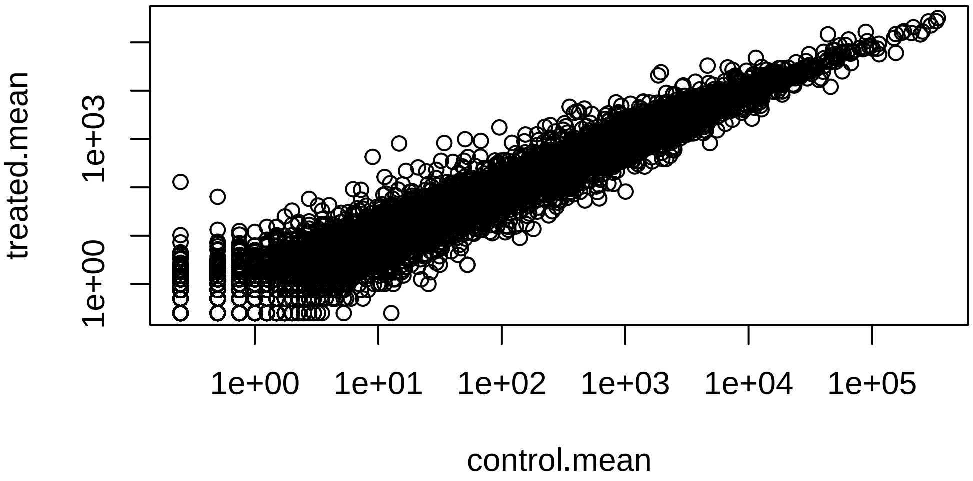


This is screaming at me to log transform!

```
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 log2 "fold-change" as a way to compare.

```
#treated/control
log2(10/10)
```

```
[1] 0
```

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

```
[1] 1
```

```
log2(5/10)
```

```
[1] -1
```

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

```
[1] 2
```

```
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
```

Alternate method find zero values

```
# 11/9/25, 11:25 PM 10/15/2025 - Posit Cloud
# https://posit.cloud/spaces/707042/content/11139485 5/7
x <- c(1,5,0,5)
which(x==0)
```

```
[1] 3
```

```
y <- data.frame(a=c(1,5,0,5), b=c(1,0,5,5))
y
```

```
  a b
1 1 1
2 5 0
3 0 5
4 5 5
```

```
which(y==0, arr.ind = T)
```

```
     row col
[1,]   3   1
[2,]   2   2
```

```r
zero.inds <- which(meancounts[,1:2]==0, arr.ind=T)[,1]
mygenes <- meancounts[-zero.inds,]
```

A common “rule-of-thumb” threshold for calling something “up” regulated is a log2-fold-change of +2 or greater. For down regulated -2 or less.

> Q. How many genes are “up” regulated at the +2 log2FC threshold?

```r
table(meancounts$log2fc>=2)
```

```

FALSE  TRUE 
23348  1910 
```

```r
sum(mygenes$log2fc>=2)
```

```
[1] 314
```

> Q. How many genes are “down” regulated at the -2 log2FC threshold?

```r
table(meancounts$log2fc<=2)
```

```

FALSE  TRUE 
 1846 23412 
```

DESeq analysis

Let’s do this with DESeq2 and put some stats behind these numbers.

```r
library(DESeq2)
```

DESeq wants 3 things for analysis, countData, colData, and design.

```r
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
```

The main function in the DESeq package to run analysis is called `DESeq()`.

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

```
estimating size factors
```

```
estimating dispersions
```

```
gene-wise dispersion estimates
```

```
mean-dispersion relationship
```

```
final dispersion estimates
```

```
fitting model and testing
```

Get the results out of this DESeq object with the function `results()`.

```r
res <- results(dds)
head(res)
```

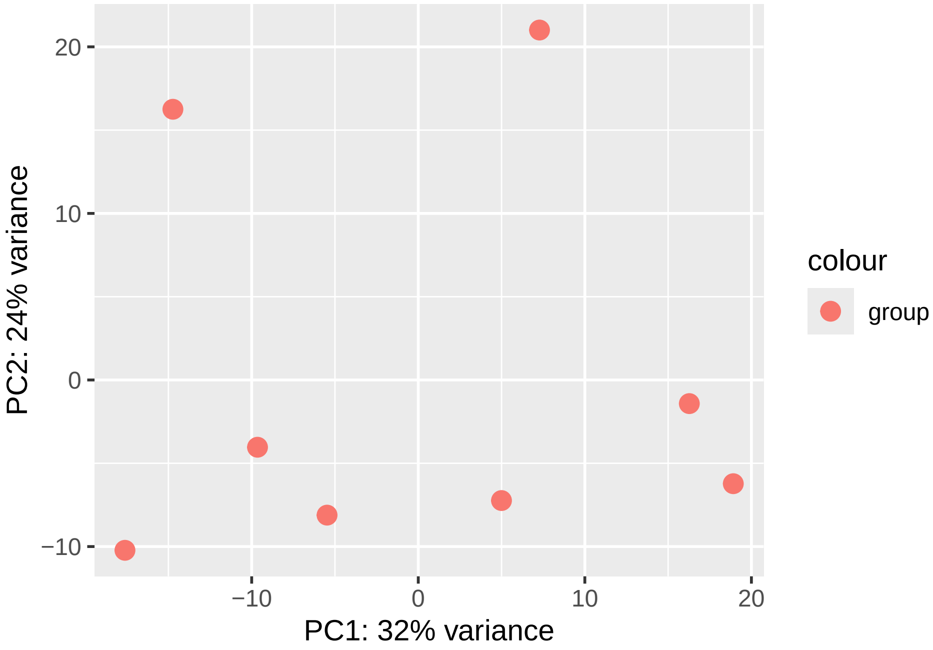
```
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 6 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat    pvalue
                <numeric>      <numeric> <numeric> <numeric> <numeric>
ENSG00000000003 747.194195      -0.350703  0.168242 -2.084514 0.0371134
ENSG00000000005   0.000000             NA        NA        NA        NA
ENSG00000000419 520.134160       0.206107  0.101042  2.039828 0.0413675
ENSG00000000457 322.664844       0.024527  0.145134  0.168996 0.8658000
ENSG00000000460  87.682625      -0.147143  0.256995 -0.572550 0.5669497
ENSG00000000938   0.319167      -1.732289  3.493601 -0.495846 0.6200029
                     padj
                <numeric>
```

```
ENSG00000000003  0.163017
ENSG00000000005        NA
ENSG00000000419  0.175937
ENSG00000000457  0.961682
ENSG00000000460  0.815805
ENSG00000000938        NA
```

PCA

```
vsd <- vst(dds, blind = FALSE)
plotPCA(vsd, intgroup = c("dex"))
```

```
using ntop=500 top features by variance
```



```
pcaData <- plotPCA(vsd, intgroup=c("dex"), returnData=TRUE)
```

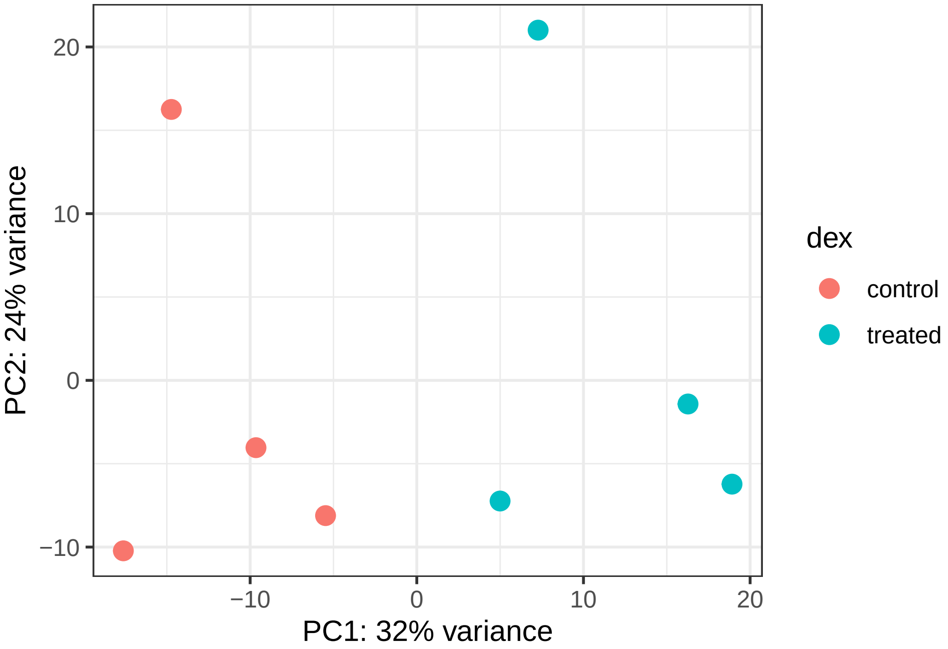
```
using ntop=500 top features by variance
```

```r
head(pcaData)
```

```
                 PC1        PC2   group       name          id     dex celltype
SRR1039508 -17.607922 -10.225252 control SRR1039508 SRR1039508 control   N61311
SRR1039509   4.996738  -7.238117 treated SRR1039509 SRR1039509 treated   N61311
SRR1039512  -5.474456  -8.113993 control SRR1039512 SRR1039512 control  N052611
SRR1039513  18.912974  -6.226041 treated SRR1039513 SRR1039513 treated  N052611
SRR1039516 -14.729173  16.252000 control SRR1039516 SRR1039516 control  N080611
SRR1039517   7.279863  21.008034 treated SRR1039517 SRR1039517 treated  N080611
               geo_id sizeFactor
SRR1039508 GSM1275862  1.0193796
SRR1039509 GSM1275863  0.9005653
SRR1039512 GSM1275866  1.1784239
SRR1039513 GSM1275867  0.6709854
SRR1039516 GSM1275870  1.1731984
SRR1039517 GSM1275871  1.3929361
```

```r
# Calculate percent variance per PC for the plot axis labels
percentVar <- round(100 * attr(pcaData, "percentVar"))
```

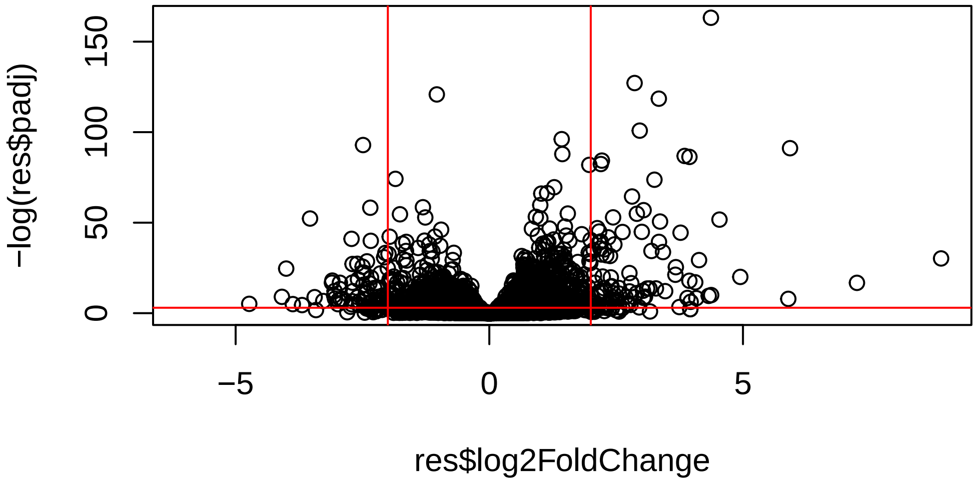
```r
library(ggplot2)
ggplot(pcaData) +
  aes(x = PC1, y = PC2, color = dex) +
  geom_point(size = 3) +
  xlab(paste0("PC1: ", percentVar[1], "% variance")) +
  ylab(paste0("PC2: ", percentVar[2], "% variance")) +
  coord_fixed() +
  theme_bw()
```



Volcano Plot

This is a plot of log2FC vs adjusted p-value

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



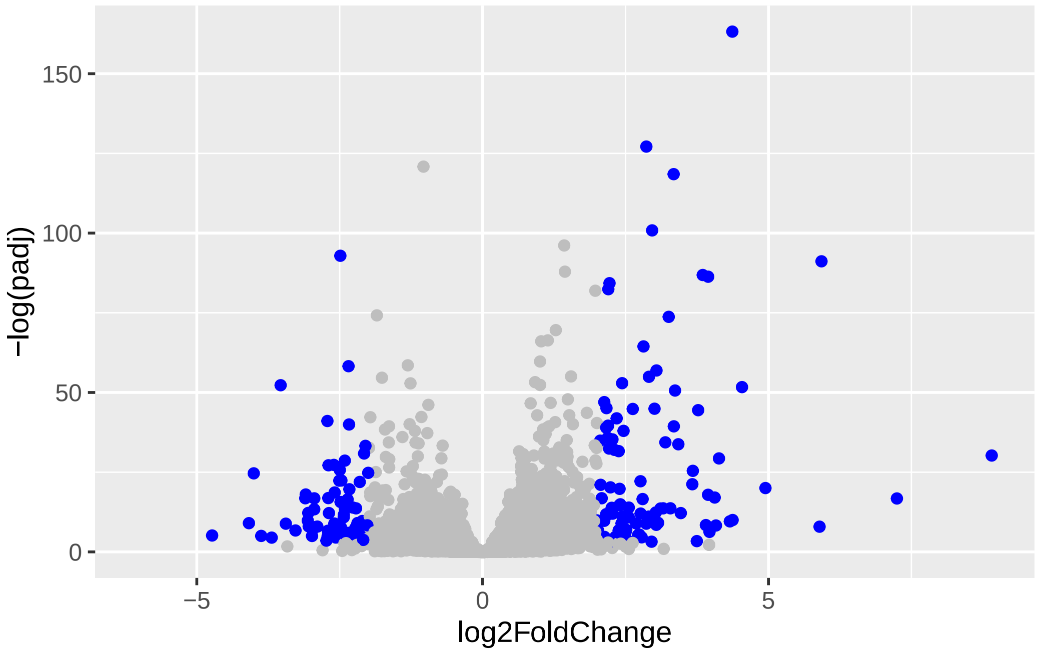
Save our results

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

Make a nicer ggplot with color.

```
library(ggplot2)
mycols <- rep("gray", nrow(res))
mycols[abs(res$log2FoldChange)>2] <- "blue"
mycols[res$padj>=0.05] <- "gray"
ggplot(res)+
  aes(log2FoldChange, -log(padj))+
  geom_point(col=mycols)
```

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



```
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)
```

```r
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"   
```

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

```
[1] -0.35070296          NA  0.20610728  0.02452701 -0.14714263 -1.73228897
```

```r
# Get the results
keggres = gage(foldchanges, gsets=kegg.sets.hs)
```

```r
attributes(keggres)
```

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

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

```
                                         p.geomean stat.mean p.val q.val
hsa00232 Caffeine metabolism                    NA       NaN    NA    NA
hsa00983 Drug metabolism - other enzymes        NA       NaN    NA    NA
```

```
hsa01100 Metabolic pathways                      NA       NaN    NA    NA
                                         set.size exp1
hsa00232 Caffeine metabolism                    0   NA
hsa00983 Drug metabolism - other enzymes        0   NA
hsa01100 Metabolic pathways                     0   NA
```

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

```
Warning: None of the genes or compounds mapped to the pathway!
Argument gene.idtype or cpd.idtype may be wrong.

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

Info: Working in directory /Users/kelseyfierro/Desktop/UCSD/bggn 213 - bioinformatics fall25/

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