

# Class12

Laura Biggs

2022-11-03

```
#Import countData and colData
```

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

```
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
```

```
dim(counts)
```

```
## [1] 38694     8
```

```
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
```

#Q1. How many genes are in this dataset? 38694 genes

#Q2. How many ‘control’ cell lines do we have? 4 control cell lines

#Mean counts per gene; method 1

```

control <- metadata[metadata[, "dex"]=="control",]
control.counts <- counts[,control$id]
control.mean <- rowSums( control.counts )/4
head(control.mean)

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

```

#Method 2

```

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.1.3

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

control <- metadata %>% filter(dex=="control")
control.counts <- counts %>% select(control$id)
control.mean <- rowSums(control.counts)/4
head(control.mean)

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

```

#Q3. How would you make the above code in either approach more robust? Create a function

#Q4. Follow the same procedure for the treated samples?

```

treated <- metadata %>% filter(dex=="treated")
treated.counts <- counts %>% select(treated$id)
treated.mean <- rowSums(treated.counts)/4
head(treated.mean)

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

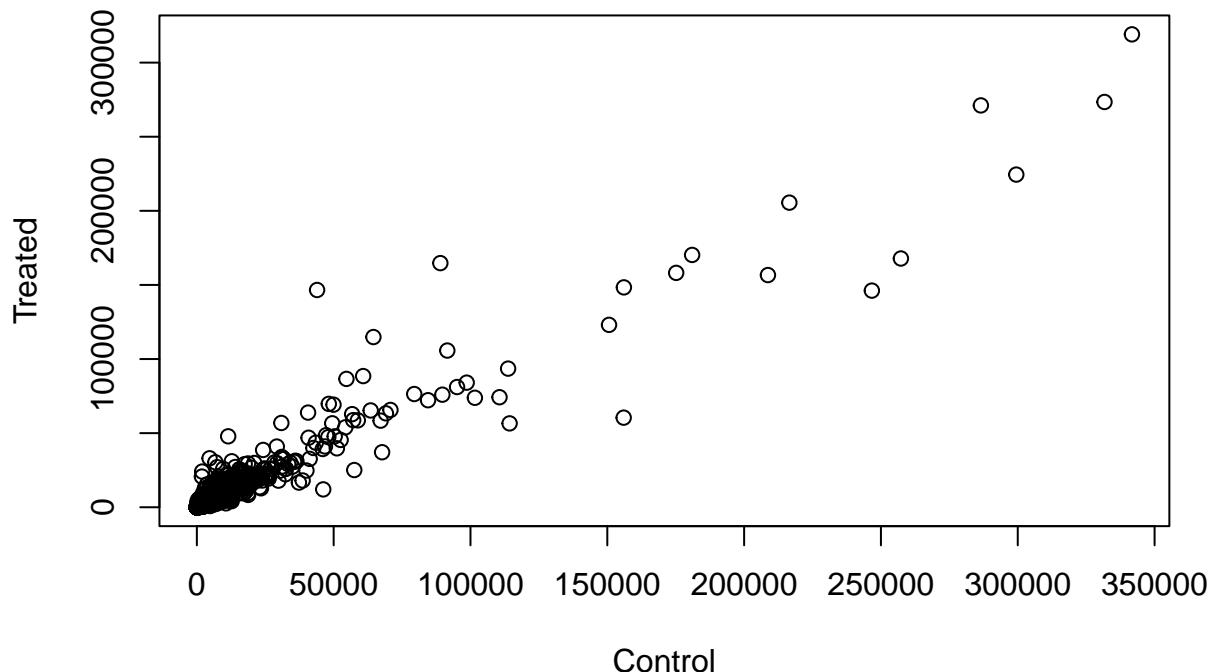
```

#Combined meancount data

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

#Q5a. Create a scatter plot showing the mean of the treated samples against the mean of the control samples.

```
plot(meancounts, xlab = "Control", ylab = "Treated")
```



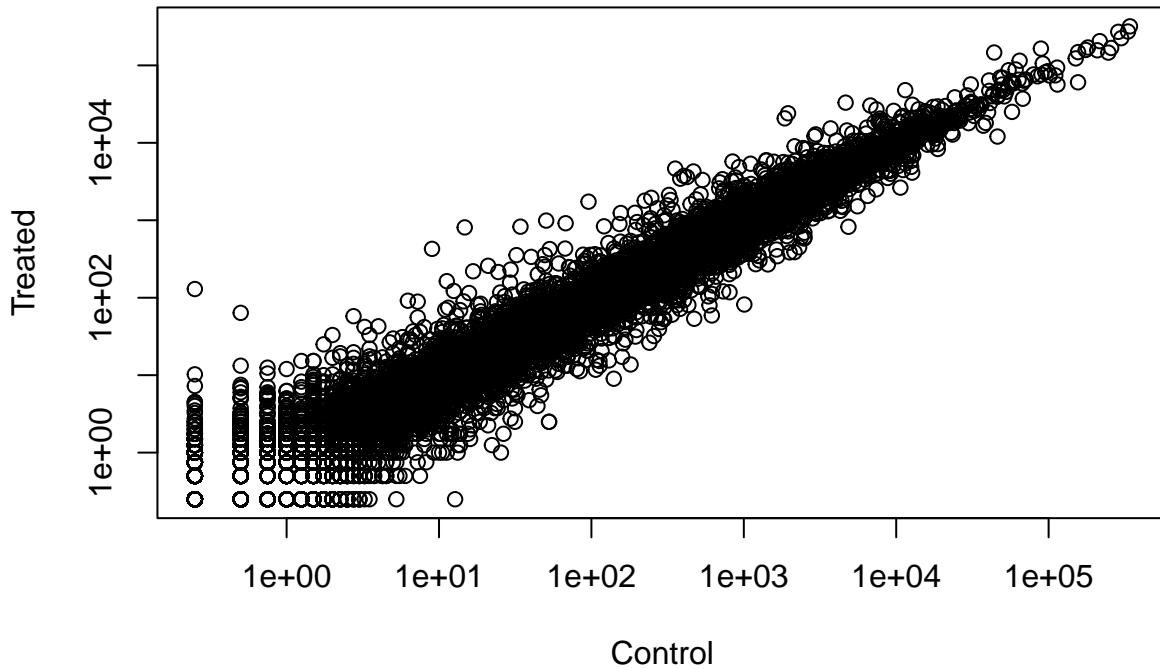
#Q5b. You could also use the ggplot2 package to make this figure producing the plot below. What geom function would you use? #geom\_point()

#Q6. Try plotting both axes on a log scale. What is the argument to plot() that allows you to do this?

```
plot(meancounts, xlab = "Control", ylab = "Treated", 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
```



```
#Calculate the log2 fold change
```

```
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

```
#Filter out zero values
```

```
zero.vals <- which(meancounts[, 1:2] == 0, arr.ind=TRUE)

to.rm <- unique(zero.vals[, 1])
mycounts <- meancounts[-to.rm,]
head(mycounts)
```

	control.mean	treated.mean	log2fc
## ENSG00000000003	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
```

#Q7. What is the purpose of the arr.ind argument in the which() function call above? Why would we then take the first column of the output and need to call the unique() function? #?which() #?unique()  
arr.ind returns row and column indices that are zero values. As zero values are in columns, we want to purposely exclude these values with the unique function.

#Filter by log2 values

```
up.ind <- mycounts$log2fc > 2
down.ind <- mycounts$log2fc < (-2)
```

#Q8. Using the up.ind vector can you determine how many upregulated genes that are greater than 2 fc?  
There are 250 genes that are upregulated.

```
sum(up.ind, na.rm = TRUE)
```

```
## [1] 250
```

```
table(up.ind) ['TRUE']
```

```
## TRUE
## 250
```

#Q9. Using the down.ind vector can you determine how many downregulated genes that are greater than 2 fc?  
There are 367 genes that are downregulated.

```
sum(down.ind, na.rm = TRUE)
```

```
## [1] 367
```

#Q10. Do you trust these results? Why or why not? These results don't tell us whether the fold changes are statistically significant, so further analysis is required.

#DESeq2 normalization

```
library(DESeq2)
```

```
## Loading required package: S4Vectors
## Loading required package: stats4
## Loading required package: BiocGenerics
## Loading required package: parallel
##
## Attaching package: 'BiocGenerics'
```

```

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, 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, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min

##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:dplyr':
##
##     first, rename

## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: GenomicRanges

## Loading required package: GenomeInfoDb

```

```

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

##
## Attaching package: 'matrixStats'

## The following object is masked from 'package:dplyr':
## 
##     count

##
## 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

```

```

citation("DESeq2")

##
##   Love, M.I., Huber, W., Anders, S. Moderated estimation of fold change
##   and dispersion for RNA-seq data with DESeq2 Genome Biology 15(12):550
##   (2014)
##
## A BibTeX entry for LaTeX users is
##
## @Article{,
##   title = {Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2},
##   author = {Michael I. Love and Wolfgang Huber and Simon Anders},
##   year = {2014},
##   journal = {Genome Biology},
##   doi = {10.1186/s13059-014-0550-8},
##   volume = {15},
##   issue = {12},
##   pages = {550},
## }
}

#DESeqDataSet

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

## class: DESeqDataSet
## dim: 38694 8
## metadata(1): version
## assays(1): counts
## rownames(38694): ENSG00000000003 ENSG00000000005 ... ENSG00000283120
##   ENSG00000283123
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(4): id dex celltype geo_id

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

res <- as.data.frame(res)
View(res)
summary(res)

##           baseMean      log2FoldChange      lfcSE      stat
##          Min. : 0.0  Min. :-6.030  Min. : 0.057  Min. :-15.894
##  1st Qu.: 0.0  1st Qu.:-0.425  1st Qu.: 0.174  1st Qu.: -0.643
##  Median : 1.1  Median :-0.009  Median : 0.445  Median : -0.027
##  Mean   : 570.2  Mean  :-0.011  Mean   : 1.136  Mean   :  0.045
##  3rd Qu.: 201.8  3rd Qu.: 0.306  3rd Qu.: 1.848  3rd Qu.:  0.593
##  Max.   :329280.4  Max.  : 8.906  Max.  : 3.534  Max.  : 18.422
##           pvalue      padj
##          NA's       :13436  NA's       :13436  NA's       :13436
```

```

##   Min.    :0.000   Min.    :0.000
## 1st Qu.:0.168   1st Qu.:0.203
## Median :0.533   Median :0.606
## Mean   :0.495   Mean   :0.539
## 3rd Qu.:0.800   3rd Qu.:0.866
## Max.    :1.000   Max.    :1.000
## NA's    :13578   NA's    :23549

res05 <- results(dds, alpha=0.05)
summary(res05)

##
## out of 25258 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)      : 1236, 4.9%
## LFC < 0 (down)    : 933, 3.7%
## outliers [1]       : 142, 0.56%
## low counts [2]     : 9033, 36%
## (mean count < 6)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results

#Add annotation data

library("AnnotationDbi")

##
## Attaching package: 'AnnotationDbi'

## The following object is masked from 'package:dplyr':
## 
##     select

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

##
##help("REFSEQ")

#Add individual columns to results

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

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

```

```

head(res)

##           baseMean log2FoldChange      lfcSE      stat     pvalue
## ENSG000000000003 747.1941954    -0.35070302 0.1682457 -2.0844697 0.03711747
## ENSG000000000005 0.0000000          NA         NA        NA       NA
## ENSG000000000419 520.1341601     0.20610777 0.1010592  2.0394752 0.04140263
## ENSG000000000457 322.6648439     0.02452695 0.1451451  0.1689823 0.86581056
## ENSG000000000460 87.6826252     -0.14714205 0.2570073 -0.5725210 0.56696907
## ENSG000000000938 0.3191666     -1.73228897 3.4936010 -0.4958463 0.62000288
##            padj   symbol
## ENSG000000000003 0.1630348 TSPAN6
## ENSG000000000005  NA     TNMD
## ENSG000000000419 0.1760317 DPM1
## ENSG000000000457 0.9616942 SCYL3
## ENSG000000000460 0.8158486 C1orf112
## ENSG000000000938  NA     FGR

```

#Q11. Run the mapIds() function two more times to add the Entrez ID and UniProt accession and GENE-NAME as new columns called res\$entrez, res\$uniprot, res\$genename.

```

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$uniprot <- mapIds(org.Hs.eg.db,
                      keys=row.names(res),
                      keytype = "ENSEMBL",
                      column="UNIPROT",
                      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

head(res)

```

```

##           baseMean log2FoldChange      lfcSE      stat     pvalue
## ENSG000000000003 747.1941954    -0.35070302 0.1682457 -2.0844697 0.03711747
## ENSG000000000005 0.0000000          NA         NA        NA       NA
## ENSG000000000419 520.1341601     0.20610777 0.1010592  2.0394752 0.04140263

```

```

## ENSG00000000457 322.6648439      0.02452695 0.1451451 0.1689823 0.86581056
## ENSG00000000460 87.6826252      -0.14714205 0.2570073 -0.5725210 0.56696907
## ENSG00000000938 0.3191666      -1.73228897 3.4936010 -0.4958463 0.62000288
##           padj symbol entrez uniprot
## ENSG00000000003 0.1630348     TSPAN6    7105 AOA024RCI0
## ENSG00000000005      NA     TNMD   64102     Q9H2S6
## ENSG00000000419 0.1760317     DPM1    8813     060762
## ENSG00000000457 0.9616942     SCYL3   57147     Q8IZE3
## ENSG00000000460 0.8158486 C1orf112 55732 AOA024R922
## ENSG00000000938      NA     FGR    2268     P09769
##                                     genename
## ENSG00000000003          tetraspanin 6
## ENSG00000000005          tenomodulin
## ENSG00000000419 dolichyl-phosphate mannosyltransferase subunit 1, catalytic
## ENSG00000000457                      SCY1 like pseudokinase 3
## ENSG00000000460          chromosome 1 open reading frame 112
## ENSG00000000938          FGR proto-oncogene, Src family tyrosine kinase

```

#Arrange the list by p adj

```

ord <- order( res$padj )
View(res[ord,])
head(res[ord,])

```

```

##           baseMean log2FoldChange      lfcSE      stat      pvalue
## ENSG0000152583 954.7709      4.368359 0.23712679 18.42204 8.744898e-76
## ENSG0000179094 743.2527      2.863889 0.17556931 16.31201 8.107836e-60
## ENSG0000116584 2277.9135     -1.034701 0.06509844 -15.89440 6.928546e-57
## ENSG0000189221 2383.7537      3.341544 0.21240579 15.73189 9.144326e-56
## ENSG0000120129 3440.7038      2.965211 0.20369513 14.55710 5.264243e-48
## ENSG0000148175 13493.9204     1.427168 0.10038904 14.21638 7.251278e-46
##           padj symbol entrez uniprot
## ENSG0000152583 1.324415e-71 SPARCL1 8404 AOA024RDE1
## ENSG0000179094 6.139658e-56 PER1    5187     015534
## ENSG0000116584 3.497761e-53 ARHGEF2 9181     Q92974
## ENSG0000189221 3.462270e-52 MAOA    4128     P21397
## ENSG0000120129 1.594539e-44 DUSP1    1843     B4DU40
## ENSG0000148175 1.830344e-42 STOM    2040     F8VSL7
##                                     genename
## ENSG0000152583          SPARC like 1
## ENSG0000179094          period circadian regulator 1
## ENSG0000116584 Rho/Rac guanine nucleotide exchange factor 2
## ENSG0000189221          monoamine oxidase A
## ENSG0000120129          dual specificity phosphatase 1
## ENSG0000148175          stomatin

```

#Create csv file with ordered annotations

```

write.csv(res[ord,], "deseq_results.csv")

```

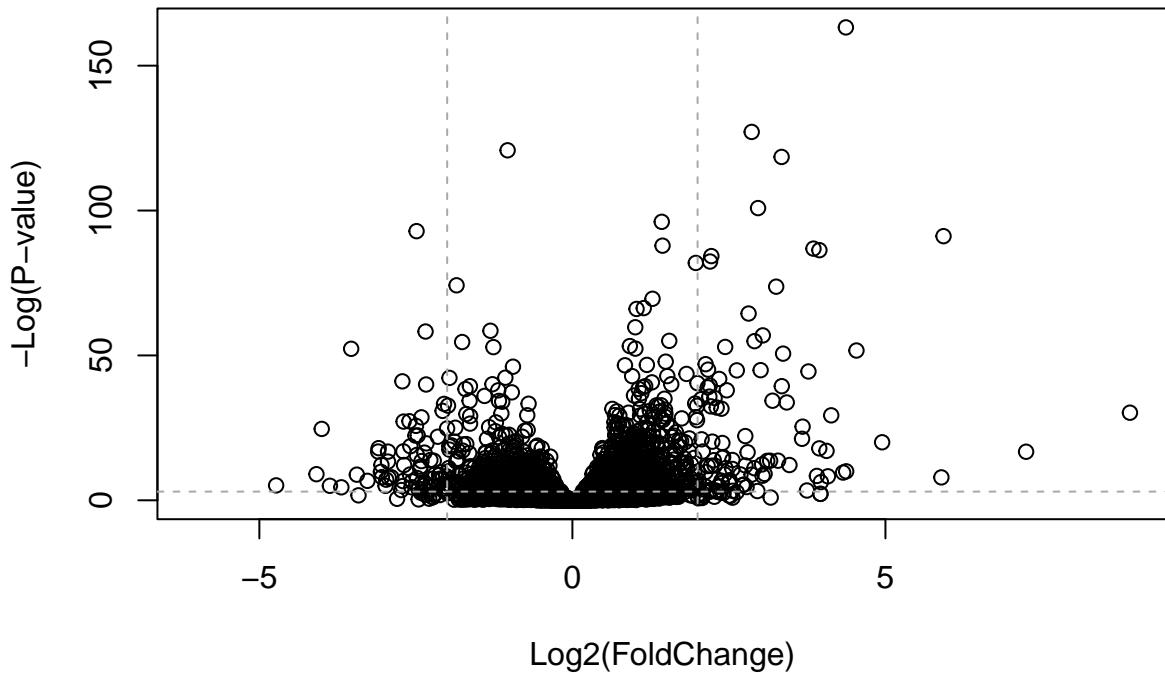
#Volcano plots

```

plot( res$log2FoldChange, -log(res$padj),
      xlab="Log2(FoldChange)",
      ylab="-Log(P-value)")

#Add lines
abline(v=c(-2,2), col="darkgray", lty=2)
abline(h=-log(0.05), col="darkgray", lty=2)

```



```

#Custom point color vector
mycols <- rep("gray", nrow(res))
mycols[ abs(res$log2FoldChange) > 2 ] <- "red"

inds <- (res$padj < 0.01) & (abs(res$log2FoldChange) > 2 )
mycols[ inds ] <- "blue"

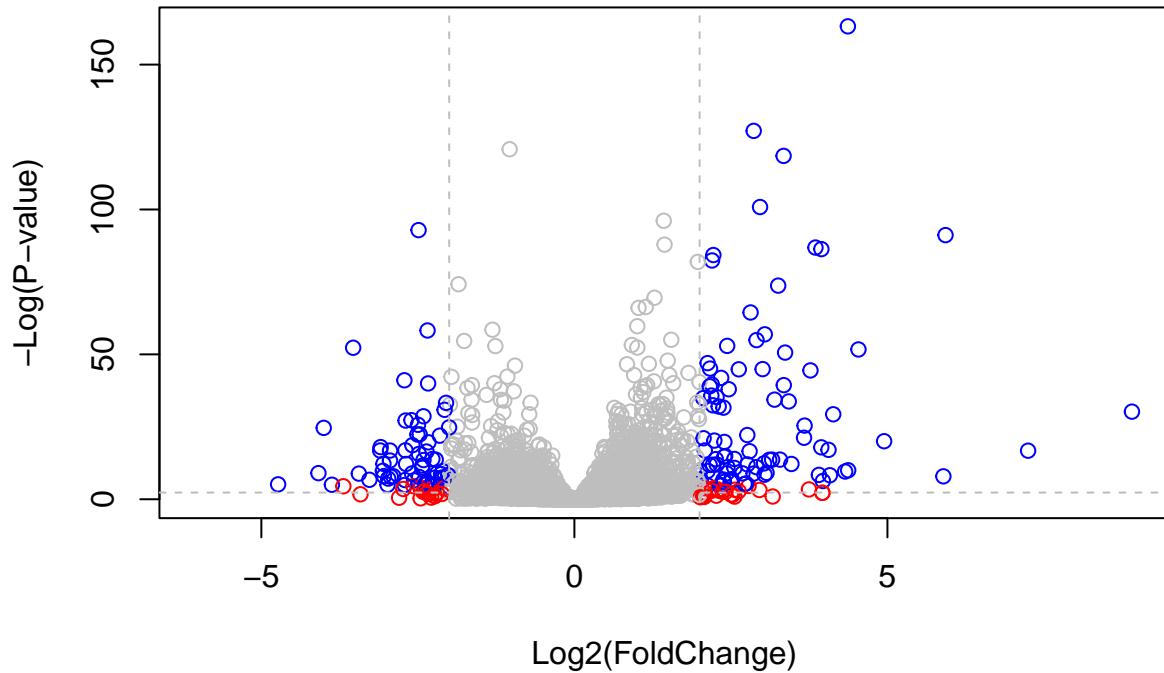
```

#Finalized plot

```

plot( res$log2FoldChange, -log(res$padj),
      col=mycols, ylab="-Log(P-value)", xlab="Log2(FoldChange)" )
abline(v=c(-2,2), col="gray", lty=2)
abline(h=-log(0.1), col="gray", lty=2)

```



```
#Enhanced volcano plot package
library(EnhancedVolcano)

## Loading required package: ggplot2
## Warning: package 'ggplot2' was built under R version 4.1.3
## Loading required package: ggrepel
## Warning: package 'ggrepel' was built under R version 4.1.3

## Registered S3 methods overwritten by 'ggalt':
##   method           from
##   grid.draw.absoluteGrob  ggplot2
##   grobHeight.absoluteGrob ggplot2
##   grobWidth.absoluteGrob ggplot2
##   grobX.absoluteGrob     ggplot2
##   grobY.absoluteGrob     ggplot2

x <- as.data.frame(res)

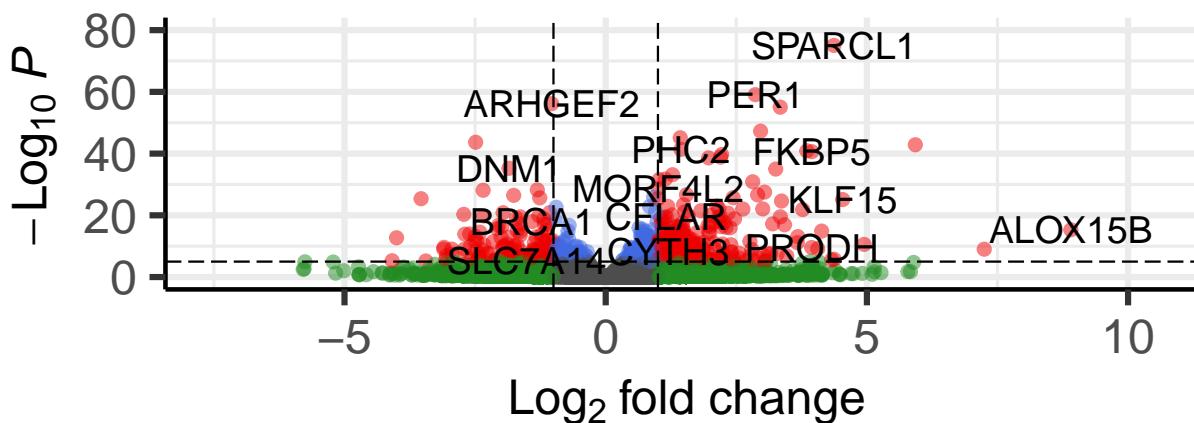
EnhancedVolcano(x,
                 lab = x$symbol,
                 x = 'log2FoldChange',
                 y = 'pvalue')
```

```
## Warning: Ignoring unknown parameters: xlim, ylim
```

## Volcano plot

### EnhancedVolcano

● NS ● Log<sub>2</sub> FC ● p-value ● p-value and log<sub>2</sub> FC



total = 38694 variables

#Pathway analysis

```
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 the human kegg set
```

```
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"
```

```
#Vector of fold changes
```

```
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)
attributes(keggres)
```

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

```
#First 3 down reg 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
```

```
#Pathview
```

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

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

```
## Info: Working in directory C:/Users/laryb/Documents/BIMM 143/Class12 RNA Seq
```

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

```

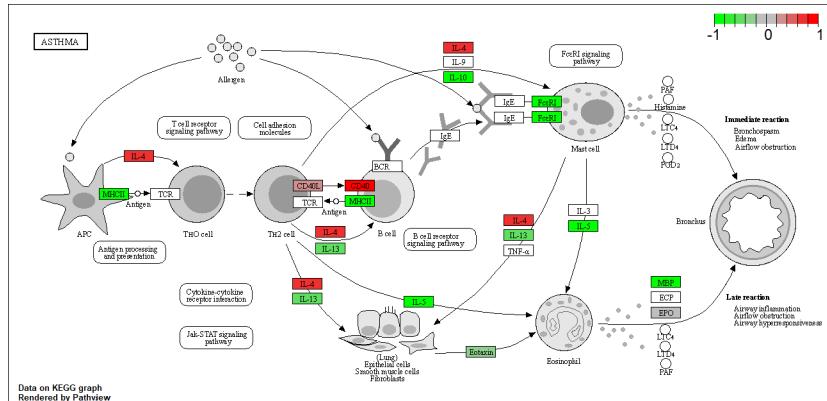
pathview(gene.data=foldchanges, pathway.id="hsa05310", kegg.native=FALSE)

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

## Info: Working in directory C:/Users/laryb/Documents/BIMM 143/Class12 RNA Seq

## Info: Writing image file hsa05310.pathview.pdf

```



#Q12. Can you do the same procedure as above to plot the pathview figures for the top 2 down-regulated pathways?

```
head(keggres$greater, 2)
```

	p.geomean	stat.mean	p.val	
## hsa00500 Starch and sucrose metabolism	0.002822007	2.825461	0.002822007	
## hsa00330 Arginine and proline metabolism	0.012317455	2.280002	0.012317455	
##		q.val	set.size	exp1
## hsa00500 Starch and sucrose metabolism	0.6010875		54	0.002822007
## hsa00330 Arginine and proline metabolism	0.7774866		54	0.012317455

```
pathview(gene.data=foldchanges, pathway.id="hsa00500")
```

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

```
## Info: Working in directory C:/Users/laryb/Documents/BIMM 143/Class12 RNA Seq
```

```
## Info: Writing image file hsa00500.pathview.png
```

```
## Info: some node width is different from others, and hence adjusted!
```

```
pathview(gene.data=foldchanges, pathway.id="hsa00330")
```

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

```
## Info: Working in directory C:/Users/laryb/Documents/BIMM 143/Class12 RNA Seq
```

## Info: Writing image file hsa00330.pathview.png

