Class 12: RNA-Seq mini project

Mirte Ciz Marieke Kuijpers

25/02/2022

Outline of mini-project

1. Input our counts and metadata files.
   - Check the format and fix if necessary
2. Run differential expression analysis
   - Setup the object required by deseq()
   - Run deseq()
3. Add annotation
   - Gene names and entrez IDs
4. Make volcano plot
   - Make sure to have it in log format
5. Pathway analysis
   - Potentially look at multiple pathways
6. Save results

```
# Load libraries
## for differential expression analysis
library("DESeq2")

## For plotting
library("ggplot2")
library("EnhancedVolcano")

## For annotation
library("AnnotationDbi")
library("org.Hs.eg.db")

## For pathway analysis
library("gage")
library("gageData")
library("pathview")

# Data for pathway analysis
data(kegg.sets.hs)
data(sigmet.idx.hs)
data(go.sets.hs)
data(go.subs.hs)
```

1: Input counts and metadata

The data is from the knock-down of a Hox gene. As Hox genes have many functions this will effect many pathways in the cell. First load this data into r objects. Set the counts row.names as the gene identifiers using `row.names = 1`.

```
# Load data
countData <- read.csv("GSE37704_featurecounts.csv", header = TRUE, row.names
= 1)
metadata <- read.csv("GSE37704_metadata.csv", header = TRUE)

# Check data
str(countData)

## 'data.frame':    19808 obs. of  7 variables:
##  $ length   : int  918 718 1982 939 939 3214 5539 3395 2833 3424 ...
##  $ SRR493366: int  0 0 23 0 0 124 1637 120 24 4 ...
##  $ SRR493367: int  0 0 28 0 0 123 1831 153 48 9 ...
##  $ SRR493368: int  0 0 29 0 0 205 2383 180 65 16 ...
##  $ SRR493369: int  0 0 29 0 0 207 1226 236 44 14 ...
##  $ SRR493370: int  0 0 28 0 0 212 1326 255 48 16 ...
##  $ SRR493371: int  0 0 46 0 0 258 1504 357 64 16 ...

str(metadata)

## 'data.frame':    6 obs. of  2 variables:
##  $ id       : chr  "SRR493366" "SRR493367" "SRR493368" "SRR493369" ...
##  $ condition: chr  "control_sirna" "control_sirna" "control_sirna"
"hoxa1_kd" ...
```

At this stage it is important to check the order and contents of the metadata and counts match, as if this is not the case there will be issues later. We can already see from the output of the `str()` that there is a length column that will not match up with the metdata and so will need to be removed.

```
# If we do not remove the column the objects will not match
if(all(colnames(countData) == metadata$id)){
  print("The conditions in the two objects are in matching order and analysis
can begin.")
}else{
  print("The conditions in the two objects do not match or are not in
matching order, please do not continue to analysis until all conditions are
in both objects and are in matching order.")
}

## Warning in colnames(countData) == metadata$id: longer object length is not
a
## multiple of shorter object length
```

```
## [1] "The conditions in the two objects do not match or are not in matching
order, please do not continue to analysis until all conditions are in both
objects and are in matching order."
```

```
# Remove the length column
countDat <- countData[, -1]

# Check this was successful
str(countDat)
```

```
## 'data.frame':    19808 obs. of  6 variables:
##  $ SRR493366: int  0 0 23 0 0 124 1637 120 24 4 ...
##  $ SRR493367: int  0 0 28 0 0 123 1831 153 48 9 ...
##  $ SRR493368: int  0 0 29 0 0 205 2383 180 65 16 ...
##  $ SRR493369: int  0 0 29 0 0 207 1226 236 44 14 ...
##  $ SRR493370: int  0 0 28 0 0 212 1326 255 48 16 ...
##  $ SRR493371: int  0 0 46 0 0 258 1504 357 64 16 ...
```

```
# Do the conditions match
if(all(colnames(countDat) == metadata$id)){
  print("The conditions in the two objects are in matching order and analysis
can begin.")
}else{
  print("The conditions in the two objects do not match or are not in
matching order, please do not continue to analysis until all conditions are
in both objects and are in matching order.")
}
```

```
## [1] "The conditions in the two objects are in matching order and analysis
can begin."
```

The two objects now match and so analysis could begin. However, first we should remove rows that have no data (i.e. all columns have 0 at that row).

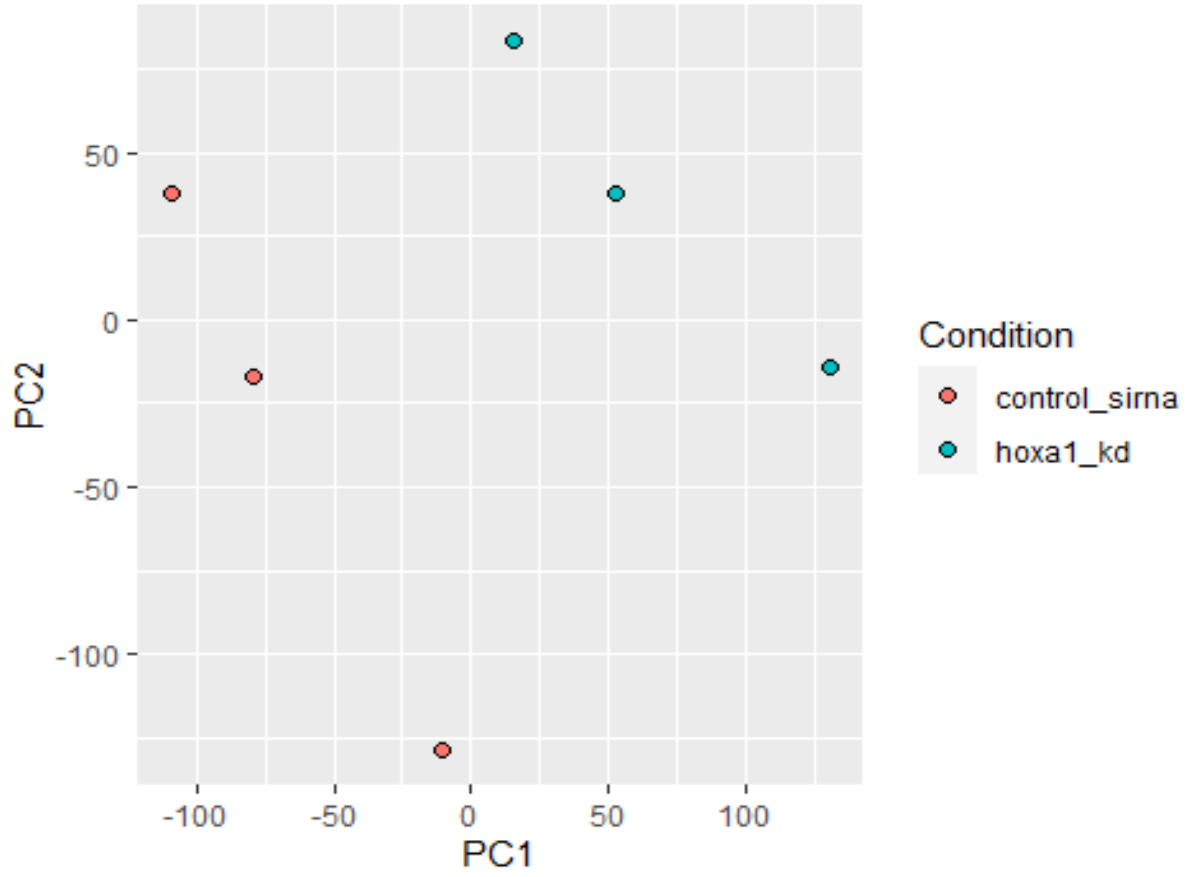
```
# Remove no data rows from countDat and asign to a new variable counts
counts <- countDat[rowSums(countDat) != 0, ]
```

Finally, we could do a Principle Component Analysis for quality control before beginning analysis.

```
# PCA
pca <- prcomp(t(counts), scale. = TRUE) # transpose counts because it is a
symmetric matrix

# Check pca results
summary(pca)
```

```
## Importance of components:
##                             PC1     PC2      PC3      PC4      PC5
PC6
## Standard deviation     87.7211 73.3196 32.89604 31.15094 29.18417 6.648e-
```

```
13
## Proportion of Variance  0.4817  0.3365  0.06774  0.06074  0.05332
0.000e+00
## Cumulative Proportion   0.4817  0.8182  0.88594  0.94668  1.00000
1.000e+00

# Plot pca results
## Create a data.frame for ggplot
df <- as.data.frame(pca$x)
df$meta <- metadata$condition

## Make a scatter plot colored by condition
ggplot(df, aes(PC1, PC2, fill = meta)) +
  geom_point(pch = 21, size = 2) +
  labs(fill = "Condition")
```



As expected, the control and knock down groups are clearly separated. If this were not the case it might not be worth continuing, as if there are no clear differences between control and condition of interest something may be amiss with our data. With quality control performed, it is now safe to proceed to analysis.

2: Run differential expression analysis

Before analysis can truly begin we need to put data into a format that can be read by the DESeq2 package functions. It requires data in an ordered object, which can be achieved by using functions provided in the package.

```
# Convert data into a format the DESeq2 package can use
dds <- DESeqDataSetFromMatrix(countData = counts, colData = metadata, design
= ~condition)

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables
in
## design formula are characters, converting to factors

# Check this object
dds

## class: DESeqDataSet
## dim: 15975 6
## metadata(1): version
## assays(1): counts
## rownames(15975): ENSG00000279457 ENSG00000187634 ... ENSG00000276345
##   ENSG00000271254
## rowData names(0):
## colnames(6): SRR493366 SRR493367 ... SRR493370 SRR493371
## colData names(2): id condition
```

We can then run DESeq to obtain results including statistics.

```
# Obtain results
dds.r <- DESeq(dds)

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

res <- results(dds.r)

# Inspect results
res

## log2 fold change (MLE): condition hoxa1 kd vs control sirna
## Wald test p-value: condition hoxa1 kd vs control sirna
## DataFrame with 15975 rows and 6 columns
```

```
##                  baseMean log2FoldChange     lfcSE       stat      pvalue
##                 <numeric>      <numeric> <numeric>  <numeric>   <numeric>
## ENSG00000279457   29.9136      0.1792571 0.3248216   0.551863 5.81042e-01
## ENSG00000187634  183.2296      0.4264571 0.1402658   3.040350 2.36304e-03
## ENSG00000188976 1651.1881     -0.6927205 0.0548465 -12.630158 1.43990e-36
## ENSG00000187961  209.6379      0.7297556 0.1318599   5.534326 3.12428e-08
## ENSG00000187583   47.2551      0.0405765 0.2718928   0.149237 8.81366e-01
## ...                   ...            ...       ...        ...         ...
## ENSG00000273748  35.30265       0.674387  0.303666   2.220817 2.63633e-02
## ENSG00000278817   2.42302      -0.388988  1.130394  -0.344117 7.30758e-01
## ENSG00000278384   1.10180       0.332991  1.660261   0.200565 8.41039e-01
## ENSG00000276345  73.64496      -0.356181  0.207716  -1.714752 8.63908e-02
## ENSG00000271254 181.59590      -0.609667  0.141320  -4.314071 1.60276e-05
##                      padj
##                 <numeric>
## ENSG00000279457 6.86555e-01
## ENSG00000187634 5.15718e-03
## ENSG00000188976 1.76549e-35
## ENSG00000187961 1.13413e-07
## ENSG00000187583 9.19031e-01
## ...                   ...
## ENSG00000273748 4.79091e-02
## ENSG00000278817 8.09772e-01
## ENSG00000278384 8.92654e-01
## ENSG00000276345 1.39762e-01
## ENSG00000271254 4.53648e-05
```

Before we inspect or plot this data we should first annotate the results to add gene identifiers that are more readable ("human-friendly").

3: Annotate results

As pathway analysis with the `pathview()` function requires Entrez results it makes sense to annotate our results with these, and perhaps also gene names and synbols for readability.

```
# Inspect which databases we could annotate with
columns(org.Hs.eg.db)
```

```
##  [1] "ACCNUM"       "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"
```

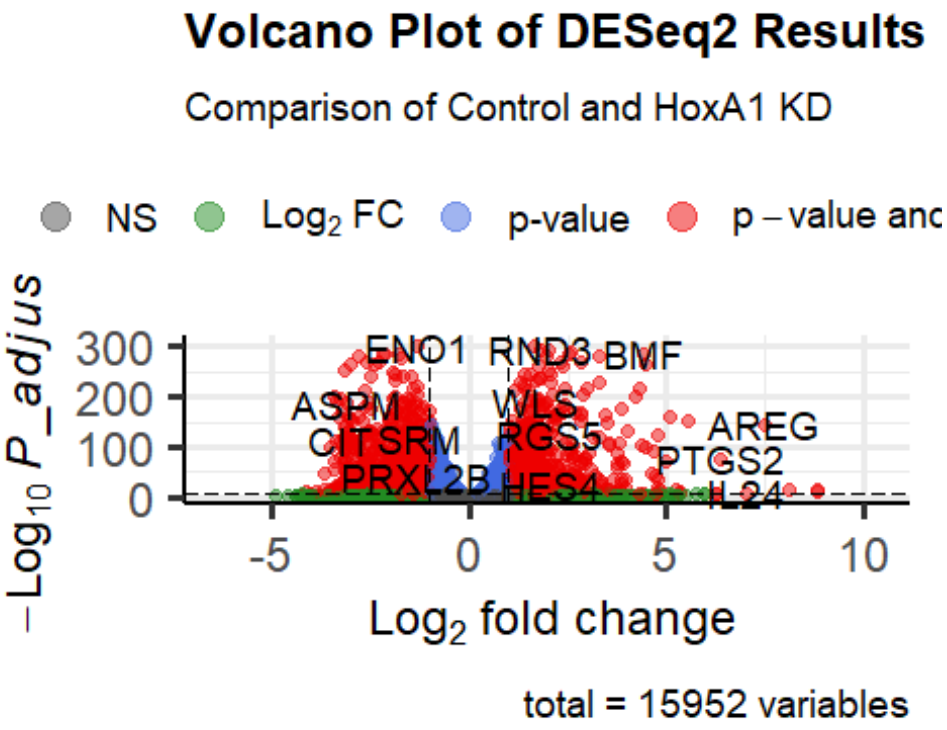
```
# Map Entrez IDs and gene names to gene identifiers
res$entrez <- mapIds(org.Hs.eg.db,
```

```
                    keys=row.names(res), # Our gene identifiers
                    keytype="ENSEMBL",        # The format of the gene
identifiers from keys
                    column="ENTREZID",          # The new format we want to
add
                    multiVals="first")     # There can be multiple
transcripts per gene, we only want the first (the highest frequency version),
note this one to many mapping

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

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

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

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

# Check this was successful
head(res)

## log2 fold change (MLE): condition hoxa1 kd vs control sirna
## Wald test p-value: condition hoxa1 kd vs control sirna
## DataFrame with 6 rows and 9 columns
##                  baseMean log2FoldChange     lfcSE       stat      pvalue
##                 <numeric>      <numeric> <numeric>  <numeric>   <numeric>
## ENSG00000279457   29.9136      0.1792571 0.3248216   0.551863 5.81042e-01
## ENSG00000187634  183.2296      0.4264571 0.1402658   3.040350 2.36304e-03
## ENSG00000188976 1651.1881     -0.6927205 0.0548465 -12.630158 1.43990e-36
## ENSG00000187961  209.6379      0.7297556 0.1318599   5.534326 3.12428e-08
## ENSG00000187583   47.2551      0.0405765 0.2718928   0.149237 8.81366e-01
## ENSG00000187642   11.9798      0.5428105 0.5215598   1.040744 2.97994e-01
##                       padj      entrez             gene.name      symbol
##                  <numeric> <character>           <character> <character>
## ENSG00000279457 6.86555e-01   102723897 WAS protein family h..      WASH9P
## ENSG00000187634 5.15718e-03      148398 sterile alpha motif ..      SAMD11
## ENSG00000188976 1.76549e-35       26155 NOC2 like nucleolar ..       NOC2L
## ENSG00000187961 1.13413e-07      339451 kelch like family me..      KLHL17
## ENSG00000187583 9.19031e-01       84069 pleckstrin homology ..     PLEKHN1
## ENSG00000187642 4.03379e-01       84808 PPARGC1 and ESRR ind..       PERM1
```

4: Plot results

With annotations in place we can now do some exploratory data analysis, by plotting our annotated data.

```
# Plot with this package
x <- as.data.frame(res)
x2 <- x[x$padj != 0,]

EnhancedVolcano(x2, lab = x2$symbol, x = 'log2FoldChange', y = 'padj', ylab =
bquote(~-Log[10] ~ italic(P_adjusted)), title = "Volcano Plot of DESeq2
Results", subtitle = "Comparison of Control and HoxA1 KD")
```



5. Pathway analysis

At this point, with the results annotated, we can perform pathway analysis to see which pathways are most effected by knock down of the HoxA1 gene. Gage requires a named vector of fold changes, and requires names to be of the Entrez gene IDs type, so the first step is to manipulate data into this format.

```
# Make a fold change vector
foldchanges <- res$log2FoldChange

# Set names of each value to the entrez gene ID
names(foldchanges) <- res$entrez
```

```
# Check this has worked
head(foldchanges)
```

```
##   102723897      148398       26155      339451       84069       84808
##  0.17925708  0.42645712 -0.69272046  0.72975561  0.04057653  0.54281049
```

Now that the data is in an appropriate format we can run the analysis.

```
# Focus on signaling and metabolic pathways only
kegg.sets.hs <- kegg.sets.hs[sigmet.idx.hs]

# Run KEGG analysis
keggres <- gage(foldchanges, gsets=kegg.sets.hs, same.dir=TRUE)

# Check results
attributes(keggres)
```

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

```
str(keggres)
```

```
## List of 3
##  $ greater: num [1:177, 1:6] 0.00282 0.0052 0.00726 0.01011 0.01875 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:177] "hsa04640 Hematopoietic cell lineage" "hsa04630
Jak-STAT signaling pathway" "hsa00140 Steroid hormone biosynthesis" "hsa04142
Lysosome" ...
##   .. ..$ : chr [1:6] "p.geomean" "stat.mean" "p.val" "q.val" ...
##  $ less   : num [1:177, 1:6] 9.00e-06 9.42e-05 1.25e-03 3.07e-03 3.78e-03
...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:177] "hsa04110 Cell cycle" "hsa03030 DNA replication"
"hsa03013 RNA transport" "hsa03440 Homologous recombination" ...
##   .. ..$ : chr [1:6] "p.geomean" "stat.mean" "p.val" "q.val" ...
##  $ stats  : num [1:177, 1:2] 2.83 2.59 2.53 2.34 2.11 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:177] "hsa04640 Hematopoietic cell lineage" "hsa04630
Jak-STAT signaling pathway" "hsa00140 Steroid hormone biosynthesis" "hsa04142
Lysosome" ...
##   .. ..$ : chr [1:2] "stat.mean" "exp1"
```

We can also look within the these attributes, for example, we can look at the pathways which have a lot of down-regulated and up-regulated genes.

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

```
##                            p.geomean stat.mean        p.val       q.val
## hsa04110 Cell cycle     8.995727e-06 -4.378644 8.995727e-06 0.001448312
```

```
## hsa03030 DNA replication 9.424076e-05 -3.951803 9.424076e-05 0.007586381
## hsa03013 RNA transport   1.246882e-03 -3.059466 1.246882e-03 0.066915974
##                          set.size         exp1
## hsa04110 Cell cycle           121 8.995727e-06
## hsa03030 DNA replication       36 9.424076e-05
## hsa03013 RNA transport        144 1.246882e-03
```

```
# Look at the first three up (greater) pathways
head(keggres$greater, 3)
```

```
##                                       p.geomean stat.mean       p.val
## hsa04640 Hematopoietic cell lineage   0.002822776  2.833362 0.002822776
## hsa04630 Jak-STAT signaling pathway   0.005202070  2.585673 0.005202070
## hsa00140 Steroid hormone biosynthesis 0.007255099  2.526744 0.007255099
##                                          q.val set.size        exp1
## hsa04640 Hematopoietic cell lineage   0.389357       55 0.002822776
## hsa04630 Jak-STAT signaling pathway   0.389357      109 0.005202070
## hsa00140 Steroid hormone biosynthesis 0.389357       31 0.007255099
```

The data from this analysis can be plotted using the `pathview()` function from the pathview package. To simplify the initial test of this package, I will initially view a single pathway, using `pathway.id` and choosing the cell cycle pathway for down regulated and the Jak-STAT signaling pathway for up regulated pathway.

```
# Diagram of cell cycle pathway up and down regulation of genes
pathview(gene.data=foldchanges, pathway.id="hsa04110")
```

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

```
## Info: Working in directory C:/Users/mirte/Documents/UCSD/Foundations of
Bioinformatics/Week.08/Class.12
```

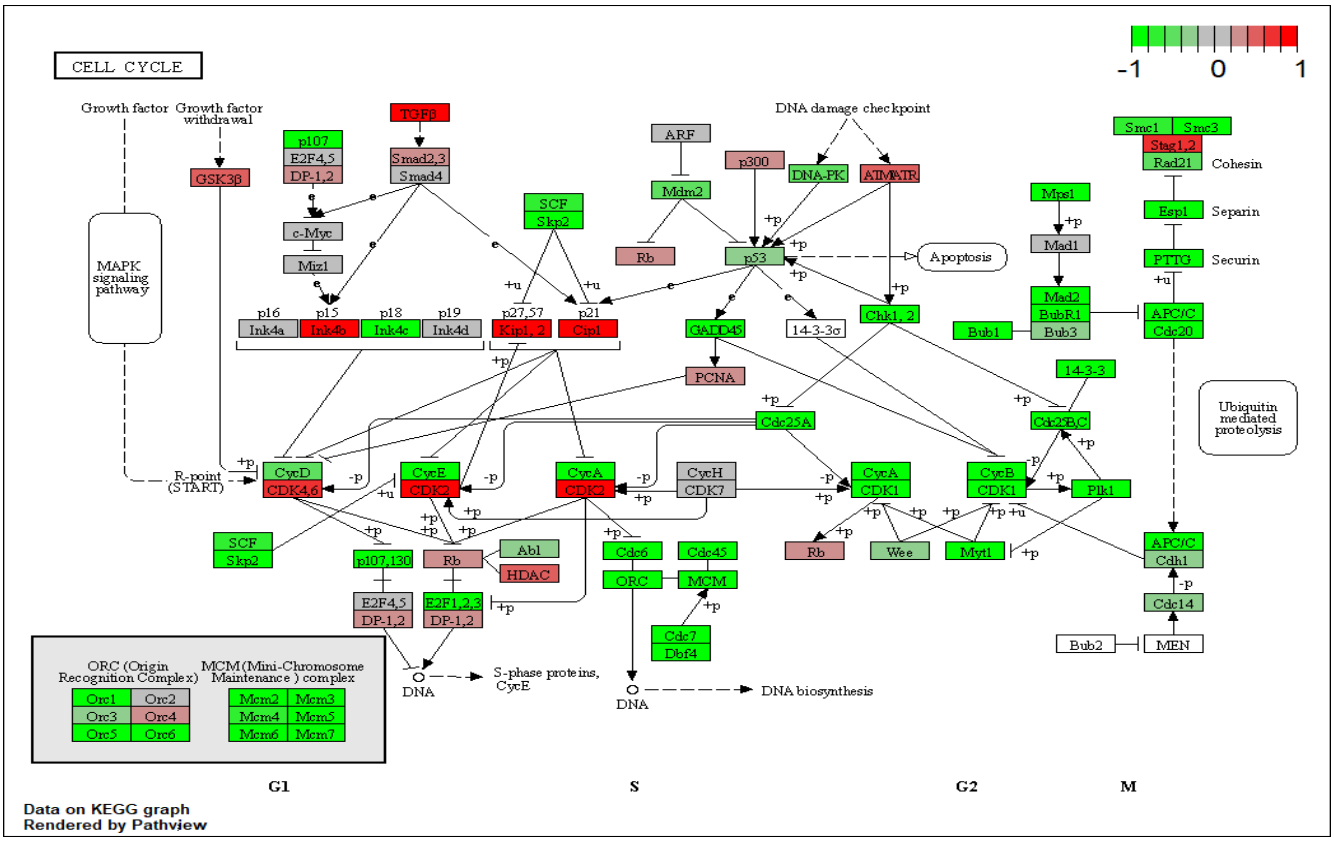
```
## Info: Writing image file hsa04110.pathview.png
```

```
# Diagram of cytokine-cytokine receptor interaction pathway up and down
regulation of genes
pathview(gene.data=foldchanges, pathway.id="hsa04630")
```

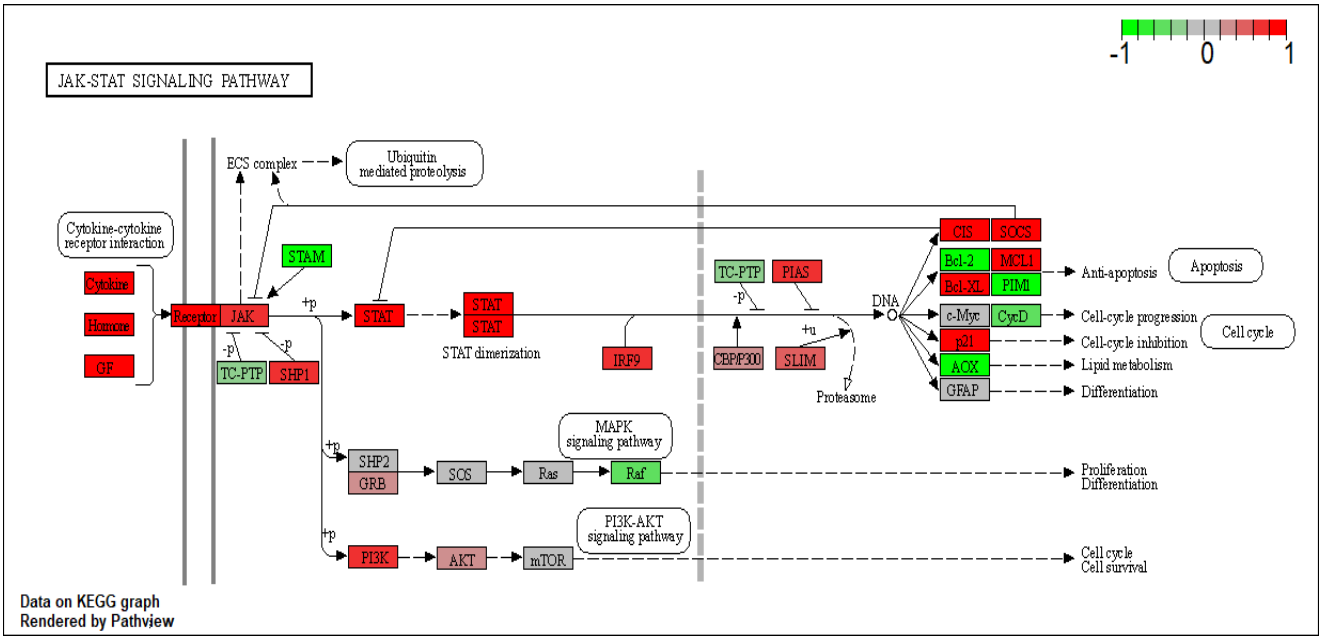
```
## 'select()' returned 1:1 mapping between keys and columns
```

```
## Info: Working in directory C:/Users/mirte/Documents/UCSD/Foundations of
Bioinformatics/Week.08/Class.12
```

```
## Info: Writing image file hsa04630.pathview.png
```



The image created by the code above



The image created by the code above

We can also use the GO database and/or many other databases for pathway analysis. To use the GO (Gene Ontology) database we just pass the GO genesets to the gage function in place of KEGG(that we used above).

```
# Focus on Biological Process subset of GO
gobpsets <-  go.sets.hs[go.subs.hs$BP]

# Use gage for pathway analysis
gobpres <-  gage(foldchanges, gsets=gobpsets)

# View top results for up and down regulated genes
head(gobpres$less, 3)
```

```
##                                 p.geomean stat.mean        p.val
q.val
## GO:0048285 organelle fission 1.536227e-15 -8.063910 1.536227e-15
5.841698e-12
## GO:0000280 nuclear division  4.286961e-15 -7.939217 4.286961e-15
5.841698e-12
## GO:0007067 mitosis           4.286961e-15 -7.939217 4.286961e-15
5.841698e-12
##                                 set.size         exp1
## GO:0048285 organelle fission         376 1.536227e-15
## GO:0000280 nuclear division          352 4.286961e-15
## GO:0007067 mitosis                   352 4.286961e-15
```

```
head(gobpres$greater, 3)
```

```
##                                            p.geomean stat.mean
p.val
## GO:0007156 homophilic cell adhesion       8.519724e-05  3.824205
8.519724e-05
## GO:0002009 morphogenesis of an epithelium 1.396681e-04  3.653886
1.396681e-04
## GO:0048729 tissue morphogenesis           1.432451e-04  3.643242
1.432451e-04
##                                                q.val set.size         exp1
## GO:0007156 homophilic cell adhesion       0.1951953      113 8.519724e-05
## GO:0002009 morphogenesis of an epithelium 0.1951953      339 1.396681e-04
## GO:0048729 tissue morphogenesis           0.1951953      424 1.432451e-04
```

Unfortunately, these results cannot be fed to `pathview()` as it only excepts KEGG inputs. It is nevertheless interesting to examine the top GO hits. Furthermore, the GO database is particularly computer friendly and can probably used for many other analyses.

6. Save results

The results can also be ordered for better viewing and then should also be saved for future use.

```
# Order the results by p value and place the index of these orders in a
vector
ord <- order( res$padj )

# View the results ordered by the index vector of ordered p values
head(res[ord,])
```

```
## log2 fold change (MLE): condition hoxa1 kd vs control sirna
## Wald test p-value: condition hoxa1 kd vs control sirna
## DataFrame with 6 rows and 9 columns
##                  baseMean log2FoldChange     lfcSE      stat    pvalue
##                 <numeric>      <numeric> <numeric> <numeric> <numeric>
## ENSG00000117519   4483.63       -2.42272 0.0600016  -40.3776         0
## ENSG00000183508   2053.88        3.20196 0.0724172   44.2154         0
## ENSG00000159176   5692.46       -2.31374 0.0575534  -40.2016         0
## ENSG00000150938   7442.99       -2.05963 0.0538449  -38.2512         0
## ENSG00000116016   4423.95       -1.88802 0.0431680  -43.7366         0
## ENSG00000136068   3796.13       -1.64979 0.0439354  -37.5504         0
##                     padj      entrez             gene.name      symbol
##                <numeric> <character>           <character> <character>
## ENSG00000117519         0        1266            calponin 3        CNN3
## ENSG00000183508         0       54855 terminal nucleotidyl..      TENT5C
## ENSG00000159176         0        1465 cysteine and glycine..       CSRP1
## ENSG00000150938         0       51232 cysteine rich transm..       CRIM1
## ENSG00000116016         0        2034 endothelial PAS doma..       EPAS1
## ENSG00000136068         0        2317             filamin B        FLNB
```

```
# Save these results for later
write.csv(res[ord,], "deseq_results.csv")
```

Session info

```
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22000)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.1252
## [2] LC_CTYPE=English_United Kingdom.1252
## [3] LC_MONETARY=English_United Kingdom.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.1252
##
## attached base packages:
```

```
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] pathview_1.34.0             gageData_2.32.0
##  [3] gage_2.44.0                 org.Hs.eg.db_3.14.0
##  [5] AnnotationDbi_1.56.2        EnhancedVolcano_1.12.0
##  [7] ggrepel_0.9.1               ggplot2_3.3.5
##  [9] DESeq2_1.34.0               SummarizedExperiment_1.24.0
## [11] Biobase_2.54.0              MatrixGenerics_1.6.0
## [13] matrixStats_0.61.0          GenomicRanges_1.46.1
## [15] GenomeInfoDb_1.30.1         IRanges_2.28.0
## [17] S4Vectors_0.32.3            BiocGenerics_0.40.0
##
## loaded via a namespace (and not attached):
##  [1] bitops_1.0-7           bit64_4.0.5            ash_1.0-15
##  [4] RColorBrewer_1.1-2     httr_1.4.2             Rgraphviz_2.38.0
##  [7] tools_4.1.2            utf8_1.2.2             R6_2.5.1
## [10] KernSmooth_2.23-20     vipor_0.4.5            DBI_1.1.2
## [13] colorspace_2.0-2       withr_2.4.3            tidyselect_1.1.2
## [16] ggrastr_1.0.1          ggalt_0.4.0            bit_4.0.4
## [19] compiler_4.1.2         extrafontdb_1.0        graph_1.72.0
## [22] cli_3.2.0              DelayedArray_0.20.0    labeling_0.4.2
## [25] KEGGgraph_1.54.0       scales_1.1.1           proj4_1.0-11
## [28] genefilter_1.76.0      stringr_1.4.0          digest_0.6.29
## [31] rmarkdown_2.11         XVector_0.34.0         pkgconfig_2.0.3
## [34] htmltools_0.5.2        extrafont_0.17         highr_0.9
## [37] fastmap_1.1.0          maps_3.4.0             rlang_1.0.1
## [40] rstudioapi_0.13        RSQLite_2.2.10         farver_2.1.0
## [43] generics_0.1.2         BiocParallel_1.28.3    dplyr_1.0.8
## [46] RCurl_1.98-1.6         magrittr_2.0.2         GO.db_3.14.0
## [49] GenomeInfoDbData_1.2.7 Matrix_1.3-4           Rcpp_1.0.8
## [52] ggbeeswarm_0.6.0       munsell_0.5.0          fansi_1.0.2
## [55] lifecycle_1.0.1        stringi_1.7.6          yaml_2.2.2
## [58] MASS_7.3-54            zlibbioc_1.40.0        grid_4.1.2
## [61] blob_1.2.2             parallel_4.1.2         crayon_1.5.0
## [64] lattice_0.20-45        Biostrings_2.62.0      splines_4.1.2
## [67] annotate_1.72.0        KEGGREST_1.34.0        locfit_1.5-9.4
## [70] knitr_1.37             pillar_1.7.0           geneplotter_1.72.0
## [73] XML_3.99-0.8           glue_1.6.1             evaluate_0.15
## [76] png_0.1-7              vctrs_0.3.8            Rttf2pt1_1.3.10
## [79] gtable_0.3.0           purrr_0.3.4            cachem_1.0.6
## [82] xfun_0.29              xtable_1.8-4           survival_3.2-13
## [85] tibble_3.1.6           beeswarm_0.4.0         memoise_2.0.1
## [88] ellipsis_0.3.2
```