Class13 RNA Seq Mini Project

Julia Ainsworth

Background

We will do all of the following:

-Read countData and colData -Check and fix contData if required -DESeq Anaysis -Visualization -Gene annotations -Pathway analysis

About the data:

The data for for hands-on session comes from GEO entry: GSE37704, which is associated with the following publication:

Trapnell C, Hendrickson DG, Sauvageau M, Goff L et al. “Differential analysis of gene regulation at transcript resolution with RNA-seq”. Nat Biotechnol 2013 Jan;31(1):46-53. PMID: 23222703

1. Read contData and colData

We need at least two things for this type of analysis:

-contData -colData (aka Metadata)

```
colData <- read.csv("GSE37704_metadata.csv", row.names=1)
colData
```

```
              condition
SRR493366 control_sirna
SRR493367 control_sirna
SRR493368 control_sirna
SRR493369      hoxa1_kd
```

```
SRR493370      hoxa1_kd
SRR493371      hoxa1_kd
```

#2. Check and fix count data

```
countData <-read.csv("GSE37704_featurecounts.csv", row.names =1)
head(countData)
```

```
                length SRR493366 SRR493367 SRR493368 SRR493369 SRR493370
ENSG00000186092    918         0         0         0         0         0
ENSG00000279928    718         0         0         0         0         0
ENSG00000279457   1982        23        28        29        29        28
ENSG00000278566    939         0         0         0         0         0
ENSG00000273547    939         0         0         0         0         0
ENSG00000187634   3214       124       123       205       207       212
                SRR493371
ENSG00000186092         0
ENSG00000279928         0
ENSG00000279457        46
ENSG00000278566         0
ENSG00000273547         0
ENSG00000187634       258
```

The first column is “length”, which would mess us up. We want to get rid of it!

```
countData1 <-countData[, -1]
head(countData1)
```

```
                SRR493366 SRR493367 SRR493368 SRR493369 SRR493370 SRR493371
ENSG00000186092         0         0         0         0         0         0
ENSG00000279928         0         0         0         0         0         0
ENSG00000279457        23        28        29        29        28        46
ENSG00000278566         0         0         0         0         0         0
ENSG00000273547         0         0         0         0         0         0
ENSG00000187634       124       123       205       207       212       258
```

To check that the data lines up, we want to call on our old friend `==`, and use `all()` to make sure we aren’t missing anything

```
all(colnames(countData1) == rownames(colData))
```

```
[1] TRUE
```

Everything looks good so far except for zero count genes. We will want to remove them before we test anything. How to do that: Find anything that is equal to 0 across all the rows (rowSums), and then make that logical by doing an == to 0 (or the ones not equal to zero for what we want to keep), find the indices where this is true, and remove these indices

```
keep.inds <- (rowSums(countData1) != 0 )
counts <- countData1[keep.inds, ] #Data where the zero count genes have been removed
head(counts)
```

```
                SRR493366 SRR493367 SRR493368 SRR493369 SRR493370 SRR493371
ENSG00000279457        23        28        29        29        28        46
ENSG00000187634       124       123       205       207       212       258
ENSG00000188976      1637      1831      2383      1226      1326      1504
ENSG00000187961       120       153       180       236       255       357
ENSG00000187583        24        48        65        44        48        64
ENSG00000187642         4         9        16        14        16        16
```

```
nrow(counts)
```

```
[1] 15975
```

QC wit hPCA

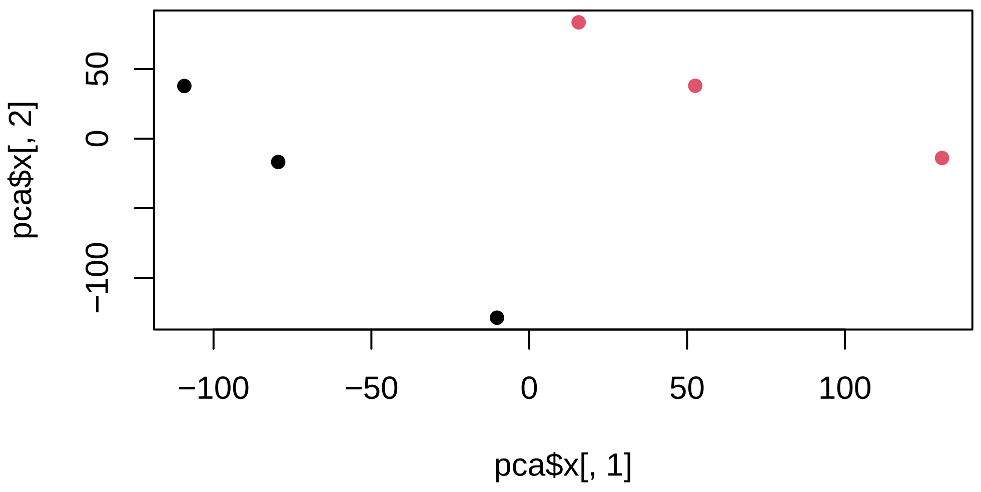
The `prcomp()` function in base R will let us do a PCA to look at our data; check that biggest differences in expression of genes is consistent with our experimental design

```
pca <- prcomp(t(counts), scale=T)
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
```

Our PCA score plot (aka PC1 vs PC2) PCA is in our pca$x component

```
plot(pca$x[,1], pca$x[,2], col=as.factor(colData$condition), pch =16)
```



3. DESeq

Next: DESeq! We have to load it, and then maek the objects it requires, run DE Seq, and plot our results.

```
library(DESeq2)
```

Next, the inputs required for DESeq

```
dds <- DESeqDataSetFromMatrix( countData = counts,
                               colData = colData,
                               design=~condition)
```

```
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
```

```r
res <- results(dds)
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 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
ENSG00000187642   11.9798      0.5428105 0.5215598   1.040744 2.97994e-01
                       padj
                  <numeric>
ENSG00000279457 6.86555e-01
ENSG00000187634 5.15718e-03
ENSG00000188976 1.76549e-35
ENSG00000187961 1.13413e-07
ENSG00000187583 9.19031e-01
ENSG00000187642 4.03379e-01
```

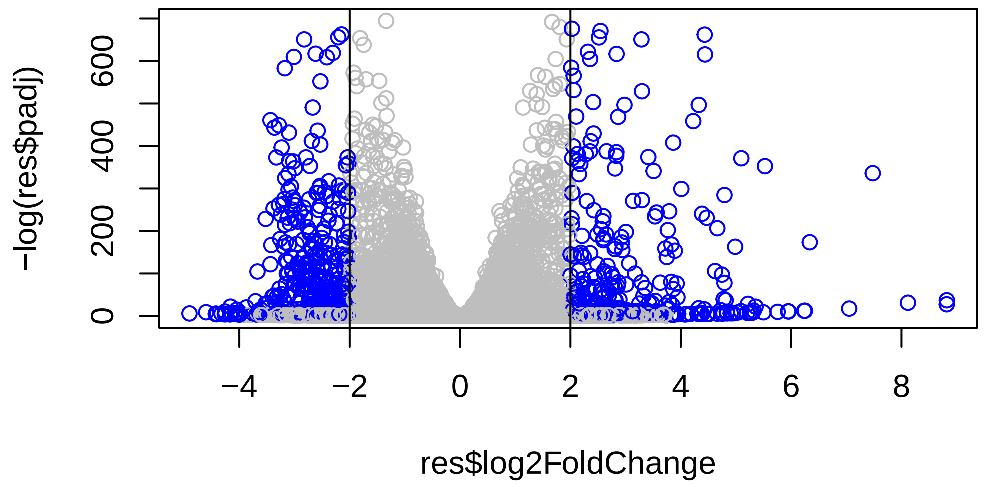
Plot a volcano

```r
# Making a color vector
mycols <- rep("gray", nrow(counts))
```

```r
mycols[abs(res$log2FoldChange) > 2] <- "blue"


mycols[res$padj > 0.05] <- "gray"

plot(res$log2FoldChange, -log(res$padj), col =mycols)
abline(v=c(-2,2))
```



4. Add gene annotation

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

Using the `mapIDs()` to add SYMBOL and ENTREZID annotations to our results

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

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

```
res$entrez <- mapIds(org.Hs.eg.db,
                       keys = rownames(counts),
                       keytype = "ENSEMBL",
                       column = "ENTREZID")
```

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

5. Pathway analysis (Gene set enrichment)

Using `gage()` again with KEGG and GO

```
library(gage)
library(gageData)
library(pathview)
```

What input `gage()` wants is a vector of importance - in our case that would be the log2 fold change values. This vector should have `names()` that are entrez IDs.

To get a fold change vector and set up our input so that names are set to input IDs

```
foldchange <- res$log2FoldChange
names(foldchange) <- res$entrez
```

```
data(kegg.sets.hs)
data(sigmet.idx.hs)
kegg.sets.hs = kegg.sets.hs[sigmet.idx.hs]
```

And now, run `gage()` with the KEGG human set

```
keggres = gage(foldchange, gsets =kegg.sets.hs)
head(keggres$less, 5)
```

```
                                    p.geomean stat.mean        p.val
hsa04110 Cell cycle              8.995727e-06 -4.378644 8.995727e-06
hsa03030 DNA replication         9.424076e-05 -3.951803 9.424076e-05
hsa03013 RNA transport           1.246882e-03 -3.059466 1.246882e-03
hsa03440 Homologous recombination 3.066756e-03 -2.852899 3.066756e-03
hsa04114 Oocyte meiosis          3.784520e-03 -2.698128 3.784520e-03
                                       q.val set.size         exp1
hsa04110 Cell cycle              0.001448312      121 8.995727e-06
hsa03030 DNA replication         0.007586381       36 9.424076e-05
hsa03013 RNA transport           0.066915974      144 1.246882e-03
hsa03440 Homologous recombination 0.121861535       28 3.066756e-03
hsa04114 Oocyte meiosis          0.121861535      102 3.784520e-03
```

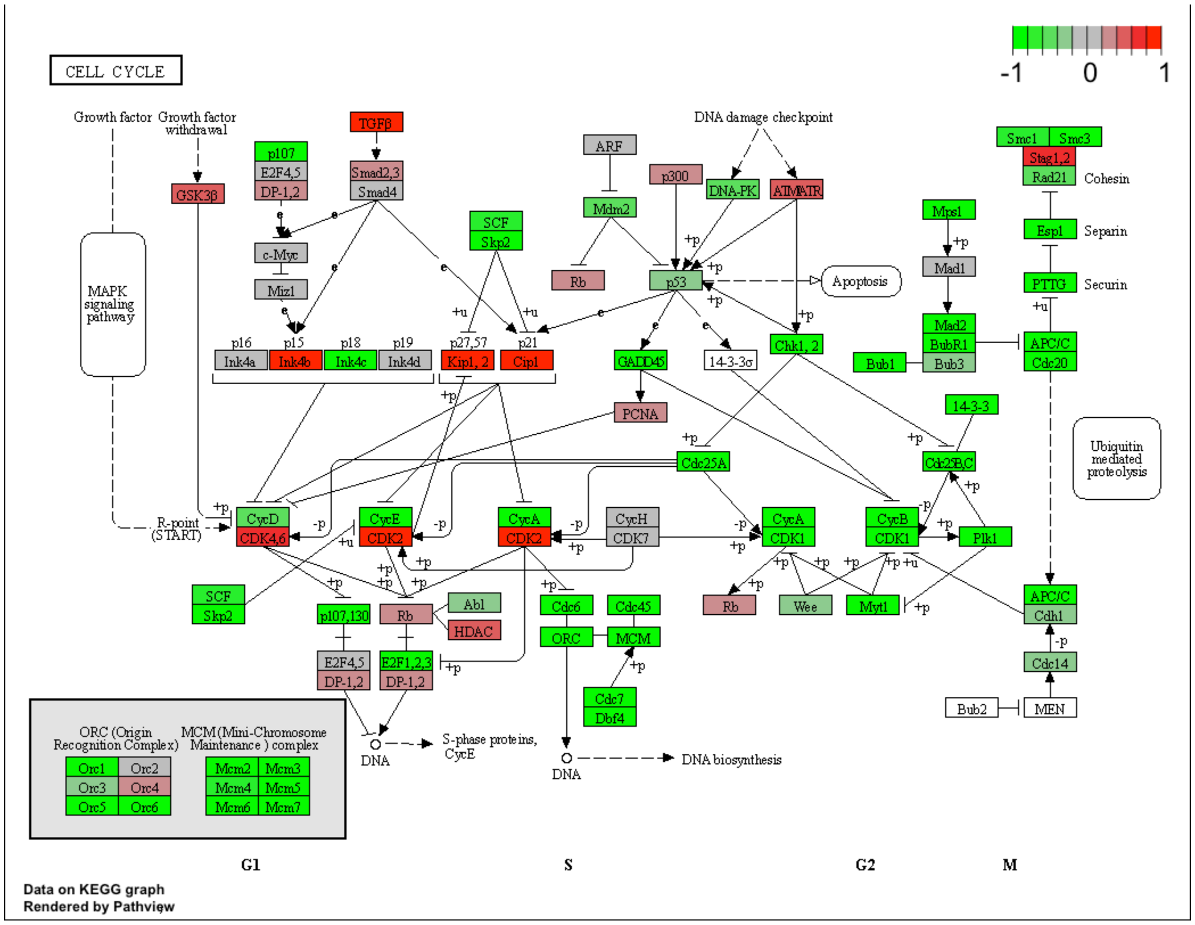
Then, to make a pretty image:

```
pathview(gene.data = foldchange, pathway.id = "hsa04110")
```

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

Info: Working in directory /Users/a1stmac/Desktop/BGGN213/Class13

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



# We can do the same thing with GO

```
data(go.sets.hs)
data(go.subs.hs)

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

gobpres = gage(foldchange, gsets=gobpsets)
head(gobpres$less)
```

```
                                          p.geomean stat.mean        p.val
GO:0048285 organelle fission           1.536227e-15 -8.063910 1.536227e-15
GO:0000280 nuclear division            4.286961e-15 -7.939217 4.286961e-15
GO:0007067 mitosis                     4.286961e-15 -7.939217 4.286961e-15
GO:0000087 M phase of mitotic cell cycle 1.169934e-14 -7.797496 1.169934e-14
GO:0007059 chromosome segregation      2.028624e-11 -6.878340 2.028624e-11
```

```
GO:0000236 mitotic prometaphase          1.729553e-10 -6.695966 1.729553e-10
                                                q.val set.size         exp1
GO:0048285 organelle fission             5.841698e-12      376 1.536227e-15
GO:0000280 nuclear division              5.841698e-12      352 4.286961e-15
GO:0007067 mitosis                       5.841698e-12      352 4.286961e-15
GO:0000087 M phase of mitotic cell cycle 1.195672e-11      362 1.169934e-14
GO:0007059 chromosome segregation        1.658603e-08      142 2.028624e-11
GO:0000236 mitotic prometaphase          1.178402e-07       84 1.729553e-10
```

Installing reactome in the console

```
sig_genes <- res[res$padj <= 0.05 & !is.na(res$padj), "symbol"]
write.table(sig_genes, file="significant_genes.txt", row.names=FALSE, col.names=FALSE, quo
```

Brought it into reactome website and explored