# Clustering

Dick de Ridder

6/10/2018

In these exercises, you will continue to work with the Arabidopsis ST vs. HT RNAseq dataset. First, you will select a subset of the data and inspect it; then cluster the data using hierarchical clustering and K-means clustering; and finally inspect one of the clusters you found for enriched functional annotations.

## Data selection, visualisation and exploration

Start RStudio, and load the CSV data file you used earlier:

```
counts =
read.table("http://www.bioinformatics.nl/courses/RNAseq/ST_vs_HT.csv",
sep=",", row.names=1, header=TRUE, stringsAsFactors=FALSE)
```

Next, try to select a number of interesting genes, e.g. `ngenes = 1000`:

```
# Calculate MAD for each gene
m = apply(counts,1,mad)
# Sort genes by decreasing MAD
ind = order(m,decreasing = TRUE,na.last = NA)
data = counts[ind[1:ngenes],]
```

Can you find out what definition of "interesting" is used here? (hint: `help(mad)`).

Now inspect the data. First, convert the data into a matrix:

```
dm = as.matrix(data)
```

Then try one or more of the following:

```
image(dm)
image(log(dm))
image(cor(t(dm)))
image(cor(dm))
```

Note that `t()` transposes the data matrix, allowing you to select genes resp. samples as columns. What do you notice in the figures?

You can also create a heatmap, using

```
heatmap(dm)
```

What does this tell you about the genes you selected? Compare the output to

```
heatmap(log(dm+1))
```

You can inspect some gene profiles, e.g.

```
plot(dm[1,])
plot(dm[2,])
plot(dm[3,])
```

or the expression ranges of some genes, for example using

```
boxplot(t(dm[1:10,]))
```

Next, you can perform a principal component analysis on the *samples*, using

```
p <- prcomp(t(data))
summary(p)
```

How much variance is explained by the first principal component? If you plot the samples on the principal components, using

```
biplot(p)
```

What does it look like the first component corresponds to? Note that the black items are the points (samples) projected on the first two principal components, and the red arrows indicate what combinations of original dimensions (genes) lead to the principal components.

Similarly, you can perform a principal component analysis on the *genes*, using

```
p <- prcomp(data)
summary(p)
biplot(p)
```

Can you spot "outlying" genes in the biplot? Make a plot of their values to inspect the expression values.

## *Optional*

You can also inspect the data (both genes and samples) using t-SNE. You will need to install and use the library first (`install.packages("tsne")`), then

```
library(tsne)
t <- tsne(t(data), perplexity=1); plot(t); text(t, labels=colnames(data))
t <- tsne(data, perplexity=10); plot(t); text(t, labels=rownames(data))
```

Play with the perplexity parameter, the number of neighbours considered, to obtain different visualisations (do not set it too high).

## *Optional*

Repeat the analyses above for a dataset containing time series measurements on the Arabidopsis apical meristem, two replicates for 10 stages M1, …, M10 (Klepikova et al., BMC Genomics 2015). The data is available as:

```
counts =
read.table("http://www.bioinformatics.nl/courses/RNAseq/time_series_klepikova
_2015_DESeq2.csv", sep=",", row.names=1, header=TRUE, stringsAsFactors=FALSE)
```

## Hierarchical clustering

Next, you will attempt to find clusters in the ST vs. HT data. Start by calculating the Euclidean distance matrix between the genes in the data and use this to calculate a hierarchical clustering with complete linkage:

```
D <- dist(data)
hcgc <- hclust(D)
```

You can visualise the clustering and highlight 5 clusters as follows:

```
plot(hcgc, labels=FALSE)
rect.hclust(hcgc, k=12, border="red")
```

What do you think of the resulting clustering?

Now replace the Euclidean distance by a correlation-based distance and repeat the hierarchical clustering:

```
D <- as.dist(1-cor(t(data)))
hcgc <- hclust(D)
plot(hcgc, labels=FALSE)
rect.hclust(hcgc, k=12, border="red")
```

Do the clusterings improve?

Try to vary the linkage (try “single” and “average”) and see whether the dendrograms change:

```
hcgs <- hclust(D, method="single")
plot(hcgs, labels=FALSE)
hcga <- hclust(D, method="average")
plot(hcga, labels=FALSE)
```

Try to explain the results you get.

Next, repeat the above to cluster *samples* rather than *genes*:

```
D <- dist(t(data))
hcs <- hclust(D)
plot(hcs)
```

Cut the gene dendrogram into k (e.g. 12) clusters, select one of these (of a reasonable size, look at the output of `table`), say group m (e.g. 2), and write the genes it contains to a file as follows:

```
groups <- cutree(hcgc,k)
table(groups)
write.table(colnames(t(data))[groups==m], file="hc.txt", quote=FALSE,
row.names=FALSE, col.names=FALSE)
```

where you replace `k` and `m` by your chosen values. Please save the file "hc.txt" in a safe place (such as your desktop), as you will need it for further analysis tomorrow.

## K-means clustering

Calculate the total within-scatter when using 2 ... 15 clusters and plot it as a function of `k`:

```
wss = c()
for (i in 2:15)
  wss[i] <- kmeans(data,centers=i)$tot.withinss
plot(1:15,wss)
```

What do you think the optimal number of clusters is? Rerun the above commands a few times. Do you get the same results each time? Why (not)?

Cluster the data into `k` (e.g. 7) clusters, select one of these (of a reasonable size, look at the output of `table`), say `m`, and write the genes it contains to a file (also in a safe place):

```
km <- kmeans(data,k)
table(km$cluster)
write.table(colnames(t(data))[km$cluster==m], file="km.txt", quote=FALSE,
row.names=FALSE, col.names=FALSE)
```

To find a reasonable size cluster, you can check the sizes

where you replace `k` and `m` by the numbers you chose.

## Coexpression networks: WGCNA

First, install and load the WGCNA library:

```
source("https://bioconductor.org/biocLite.R")
biocLite("WGCNA")

library(WGCNA)
```

Note that this is a complex package with many routines, and a long list of options and parameters. Here you will use pre-selected parameters which give reasonable results, but you may have to choose different settings on other datasets.

Load the Arabidopsis apical meristem time series data and select a number of interesting genes, as above:

```
counts =
read.table("http://www.bioinformatics.nl/courses/RNAseq/time_series_klepikova
_2015_DESeq2.csv", sep=",", row.names=1, header=TRUE, stringsAsFactors=FALSE)
m = apply(counts,1,mad)
ind = order(m,decreasing = TRUE,na.last = NA)
data = counts[ind[1:ngenes],]
```

Now create a module network on the log-transformed data

```
net = blockwiseModules(log2(t(data)+1), power=12,
                       TOMType="unsigned",
                       minModuleSize=30, reassignThreshold=0,
                       mergeCutHeight=0.1,numericLabels=TRUE,
                       pamRespectsDendro=FALSE,
                       saveTOMs=FALSE, verbose=3)
```

Each gene is now assigned a "color" corresponding to its module. Check how many modules there are and how large they are:

```
table(net$colors)
```

Now visualise the network in the form of a clustering:

```
mergedColors = labels2colors(net$colors)
plotDendroAndColors(net$dendrograms[[1]], mergedColors[net$blockGenes[[1]]],
                    "Module colors", dendroLabels=FALSE, hang=0.03,
                    addGuide=TRUE, guideHang=0.05)
```

What do you think of the module sizes?

WGCNA also summarizes the expression in each module as an "eigengene", i.e. the first principal component of the gene expression values in that module. You can inspect these as follows:

```
me = moduleEigengenes(log2(t(data)+1),colors=net$colors)$eigengenes
dev.off()
for (i in 1:8)
  plot(me[,i])
```

You can also inspect the relation between the modules, by clustering the eigengenes:

```
D <- 1-(1+cor(me,use="p"))/2
hcm = hclust(as.dist(D), method="average")
plot(hcm)
```

Can you make sense of the expression patterns in the modules (M1…M10 correspond to 10 different development stages)? And of the relation between the modules? Note that the modules are numbered 0…N-1, where the eigengene plots are numbered 1…N.

Finally, save one of the modules (e.g. the 6th) to a file, for use tomorrow:

```
TOM = TOMsimilarityFromExpr(log2(t(data)+1), power = 6);
inmod = which(net$colors==6)
modgenes = rownames(data)
colnames(TOM) <- modgenes; rownames(TOM) <- modgenes;
exportNetworkToCytoscape(TOM[inmod,inmod],
                         edgeFile='wgcna_edges.txt',
                         nodeFile='wgcna_nodes.txt',
                         weighted=TRUE, threshold = 0.02,
                         nodeNames=modgenes);
```

If you want to learn more about WGCNA, you can find tutorials and examples at https://labs.genetics.ucla.edu/horvath/CoexpressionNetwork/Rpackages/WGCNA/Tutorials/.