Clustering and Data Mining in R

Introduction

Thomas Girke

December 7, 2012

Outline

What is Clustering?

- Clustering is the classification of data objects into similarity groups (clusters) according to a defined distance measure.
- It is used in many fields, such as machine learning, data mining, pattern recognition, image analysis, genomics, systems biology, etc.

Why Clustering and Data Mining in R?

- Efficient data structures and functions for clustering
- Reproducible and programmable
- Comprehensive set of clustering and machine learning libraries
- Integration with many other data analysis tools

Useful Links

- Cluster Task Views Link
- Machine Learning Task Views Link
- UCR Manual Link

Outline

Data Transformations

Choice depends on data set!

- Center & standardize
  1. Center: subtract from each vector its mean
  2. Standardize: devide by standard deviation

  $$\Rightarrow Mean = 0 \text{ and } STDEV = 1$$
- Center & scale with the `scale()` fuction
  1. Center: subtract from each vector its mean
  2. Scale: divide centered vector by their root mean square (rms)

  $$x_{rms} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} {x_i}^2}$$

  $$\Rightarrow Mean = 0 \text{ and } STDEV = 1$$
- Log transformation
- Rank transformation: replace measured values by ranks
- No transformation

Distance Methods

List of most common ones!

- Euclidean distance for two profiles X and Y

$$d(X, Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Disadvantages: not scale invariant, not for negative correlations

- Maximum, Manhattan, Canberra, binary, Minowski, ...
- Correlation-based distance: $1 - r$
  - Pearson correlation coefficient (PCC)

$$r = \frac{n \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{(\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2)(\sum_{i=1}^{n} y_i^2 - (\sum_{i=1}^{n} y_i)^2)}}$$

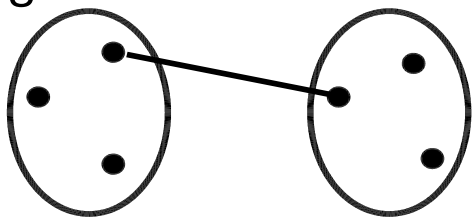
Disadvantage: outlier sensitive

  - Spearman correlation coefficient (SCC)

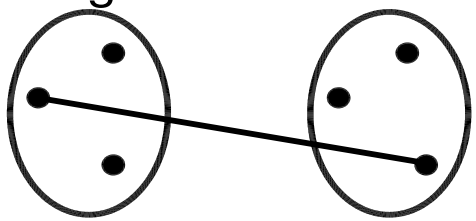
Same calculation as PCC but with ranked values!

Cluster Linkage

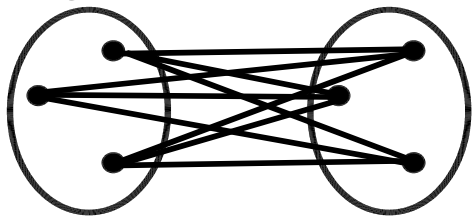
Single Linkage



Complete Linkage



Average Linkage



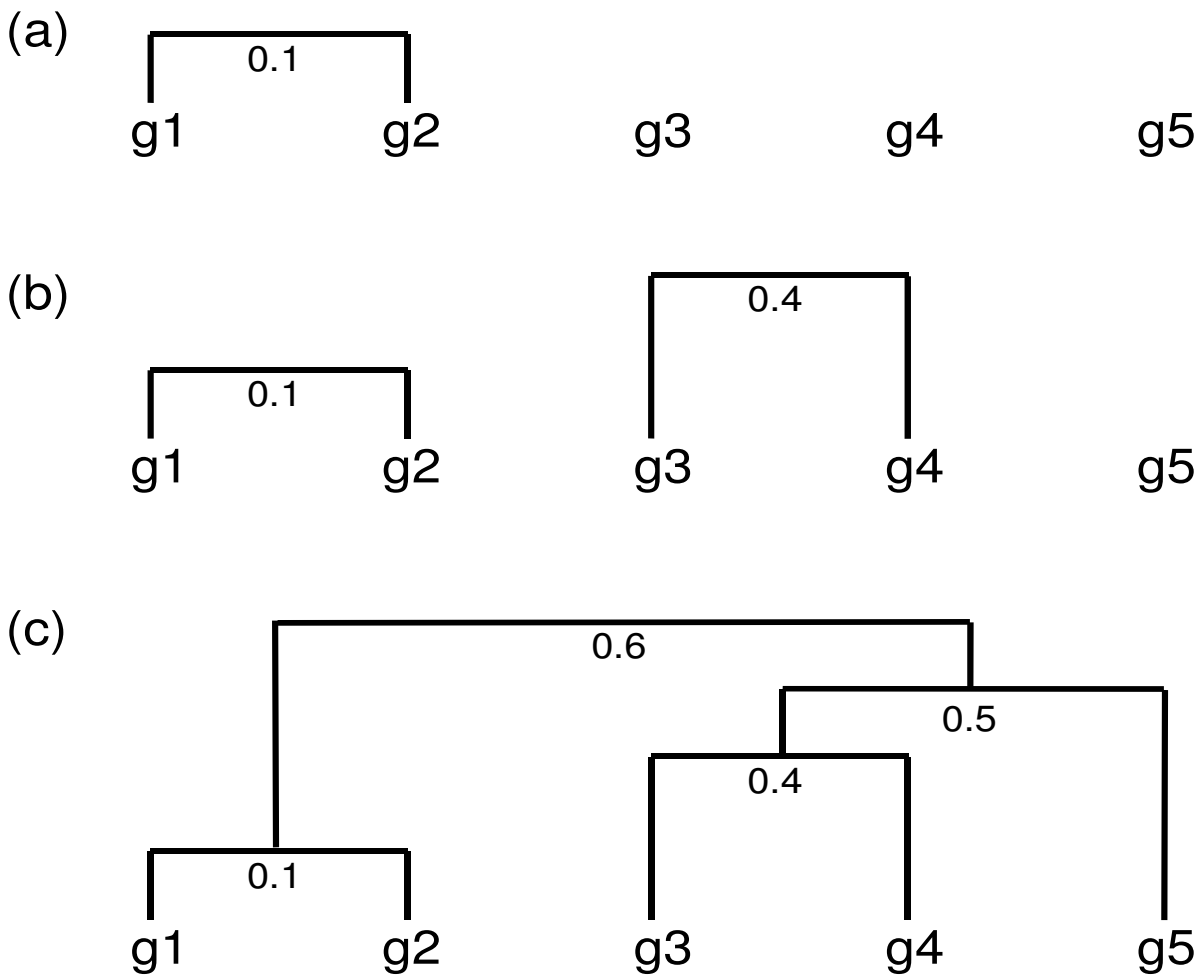
Outline

Hierarchical Clustering Steps

1. Identify clusters (items) with closest distance
2. Join them to new clusters
3. Compute distance between clusters (items)
4. Return to step 1

Hierarchical Clustering

Agglomerative Approach



Hierarchical Clustering Approaches

1. Agglomerative approach (bottom-up)
   `hclust()` and `agnes()`
2. Divisive approach (top-down)
   `diana()`

Tree Cutting to Obtain Discrete Clusters

1. Node height in tree
2. Number of clusters
3. Search tree nodes by distance cutoff

Outline

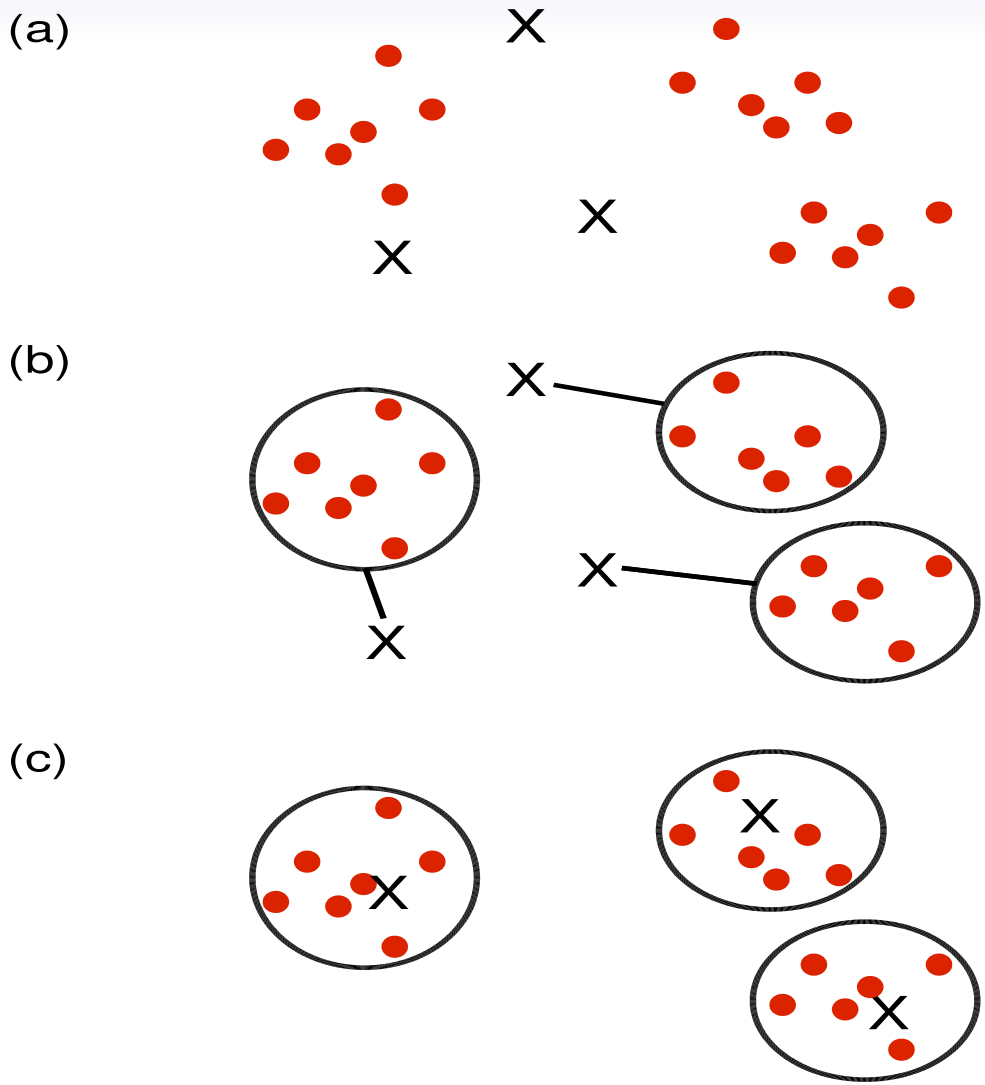
Non-Hierarchical Clustering

Selected Examples

K-Means Clustering

1. Choose the number of k clusters
2. Randomly assign items to the k clusters
3. Calculate new centroid for each of the k clusters
4. Calculate the distance of all items to the k centroids
5. Assign items to closest centroid
6. Repeat until clusters assignments are stable

K-Means



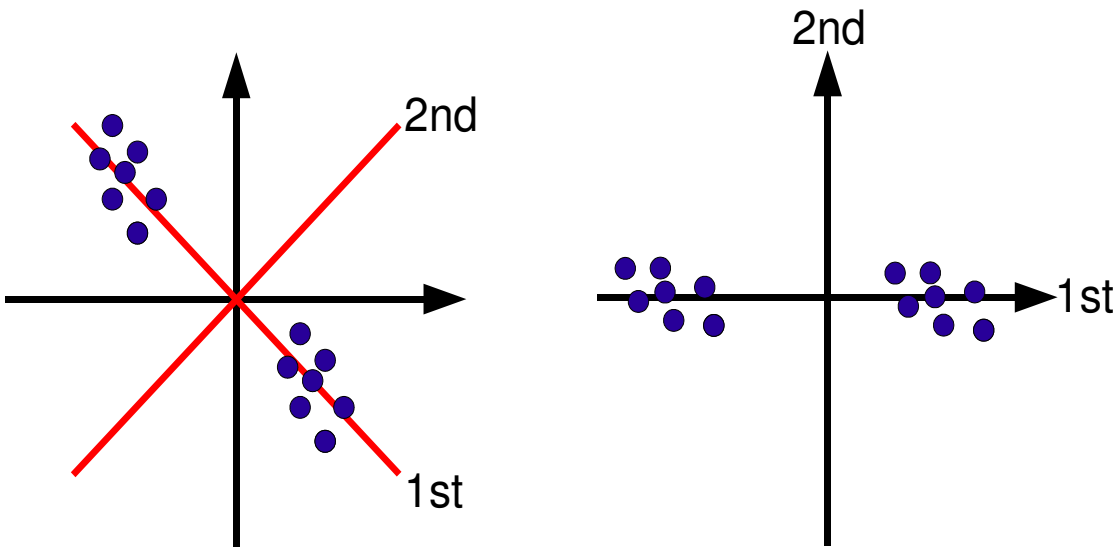
Principal Component Analysis (PCA)

Principal components analysis (PCA) is a data reduction technique that allows to simplify multidimensional data sets to 2 or 3 dimensions for plotting purposes and visual variance analysis.

Basic PCA Steps

- Center (and standardize) data
- First principal component axis
  - Accross centroid of data cloud
  - Distance of each point to that line is minimized, so that it crosses the maximum variation of the data cloud
- Second principal component axis
  - Orthogonal to first principal component
  - Along maximum variation in the data
- 1^{st} PCA axis becomes x-axis and 2^{nd} PCA axis y-axis
- Continue process until the necessary number of principal components is obtained

PCA on Two-Dimensional Data Set



Identifies the Amount of Variability between Components

Example

Principal Component	**1st**	**2nd**	**3rd**	**Other**
Proportion of Variance	62%	34%	3%	rest

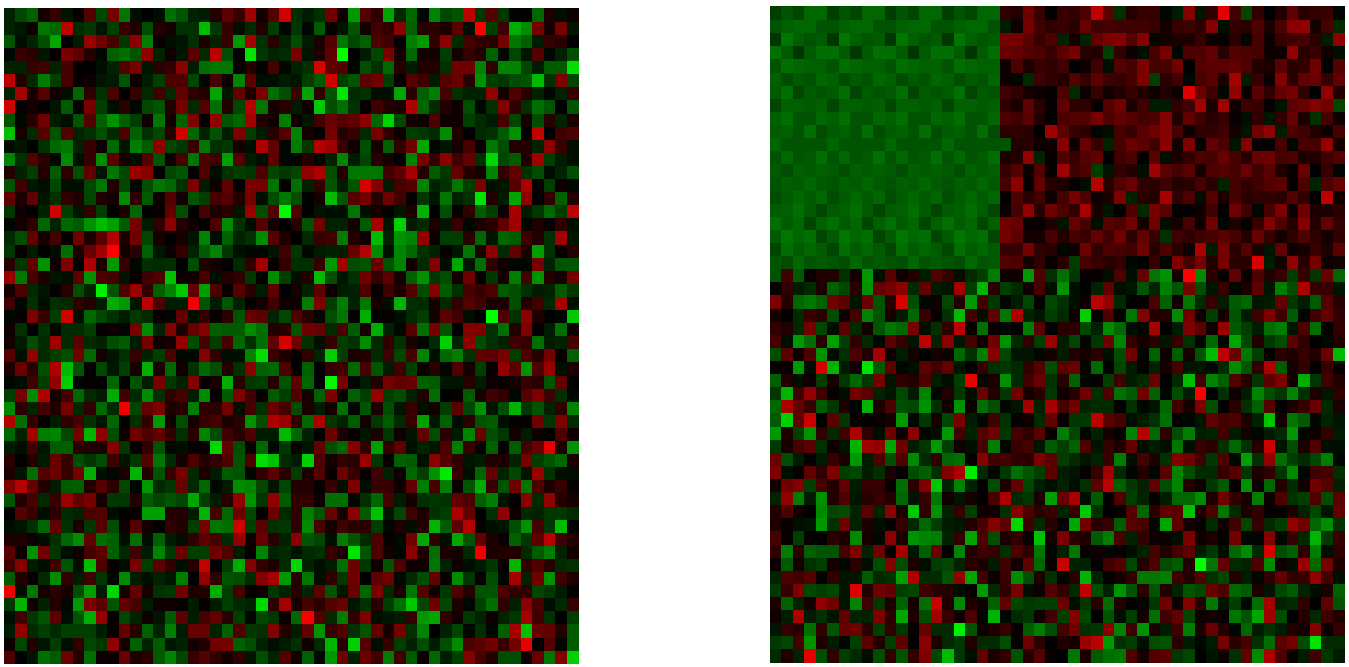
1st and 2nd principal components explain 96% of variance.

Multidimensional Scaling (MDS)

- Alternative dimensionality reduction approach
- Represents distances in 2D or 3D space
- Starts from distance matrix (PCA uses data points)

Biclustering

Finds in matrix subgroups of rows and columns which are as similar as possible to each other and as different as possible to the remaining data points.



Unclustered $\Rightarrow$ Clustered

Remember: There Are Many Additional Techniques!

Additional details can be found in the Clustering Section of the R/Bioc Manual Link

Outline

Data Preprocessing

Scaling and Distance Matrices

```
> ## Sample data set
> set.seed(1410)
> y <- matrix(rnorm(50), 10, 5, dimnames=list(paste("g", 1:10, sep=""),
+             paste("t", 1:5, sep="")))
> dim(y)

[1] 10  5

> ## Scaling
> yscaled <- t(scale(t(y))) # Centers and scales y row-wise
> apply(yscaled, 1, sd)

 g1  g2  g3  g4  g5  g6  g7  g8  g9 g10
  1   1   1   1   1   1   1   1   1   1

> ## Euclidean distance matrix
> dist(y[1:4,], method = "euclidean")

         g1       g2       g3
g2 4.793697
g3 4.932658 6.354978
g4 4.033789 4.788508 1.671968
```

Correlation-based Distances

Correlation matrix

```
> c <- cor(t(y), method="pearson")
> as.matrix(c)[1:4,1:4]
```

```
            g1         g2          g3         g4
g1  1.00000000 -0.2965885 -0.00206139 -0.4042011
g2 -0.29658847  1.0000000 -0.91661118 -0.4512912
g3 -0.00206139 -0.9166112  1.00000000  0.7435892
g4 -0.40420112 -0.4512912  0.74358925  1.0000000
```

Correlation-based distance matrix

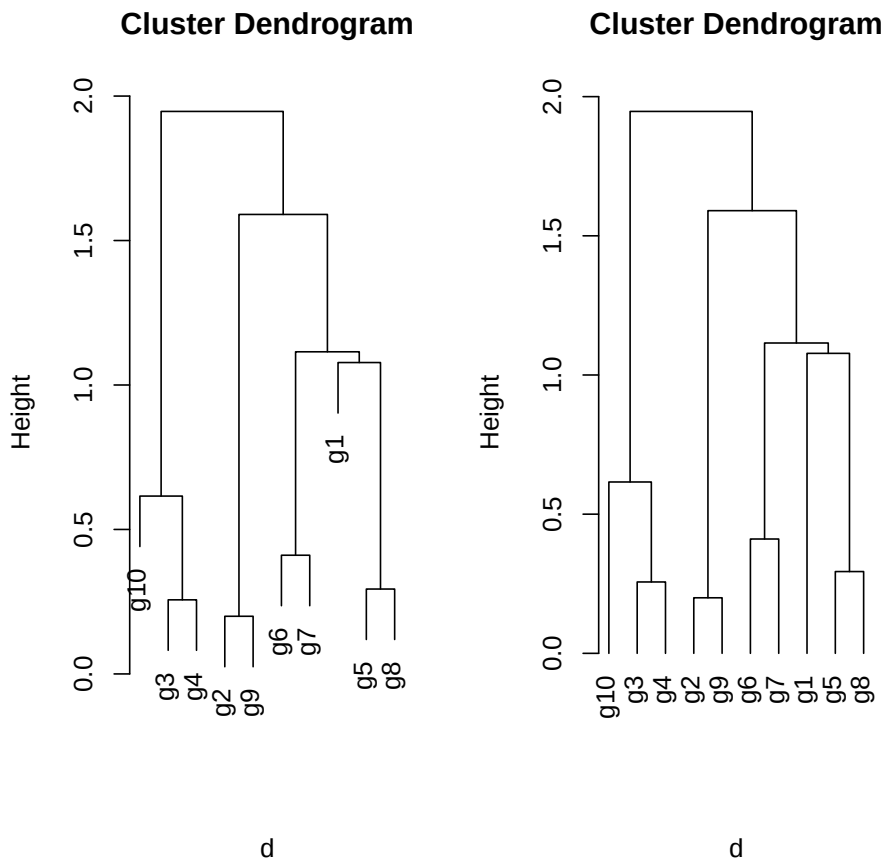
```
> d <- as.dist(1-c)
> as.matrix(d)[1:4,1:4]
```

```
         g1       g2        g3        g4
g1 0.000000 1.296588 1.0020614 1.4042011
g2 1.296588 0.000000 1.9166112 1.4512912
g3 1.002061 1.916611 0.0000000 0.2564108
g4 1.404201 1.451291 0.2564108 0.0000000
```

Hierarchical Clustering with `hclust` I

Hierarchical clustering with complete linkage and basic tree plotting

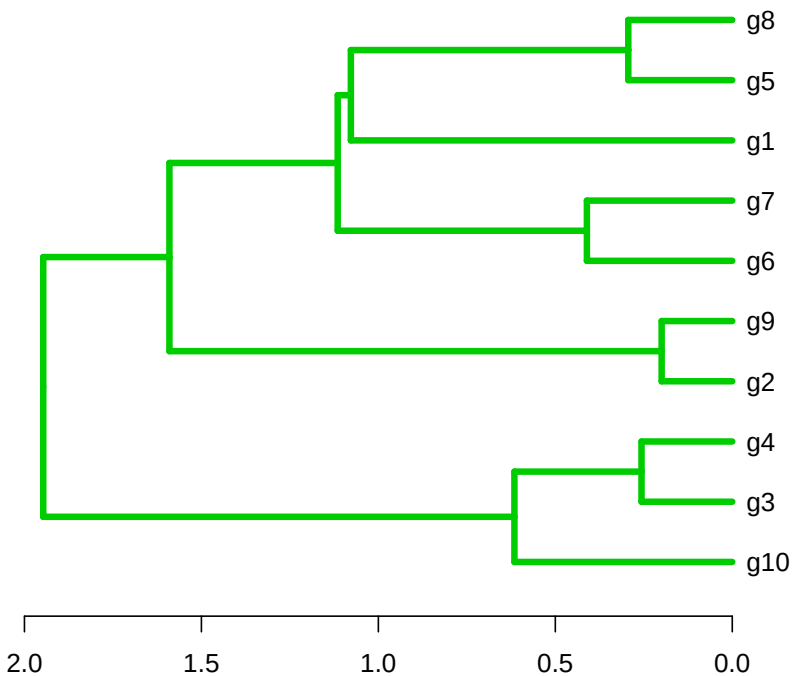
```
> hr <- hclust(d, method = "complete", members=NULL)
> names(hr)
[1] "merge"       "height"      "order"       "labels"      "method"
[6] "call"        "dist.method"
> par(mfrow = c(1, 2)); plot(hr, hang = 0.1); plot(hr, hang = -1)
```



Tree Plotting I

Plot trees horizontally

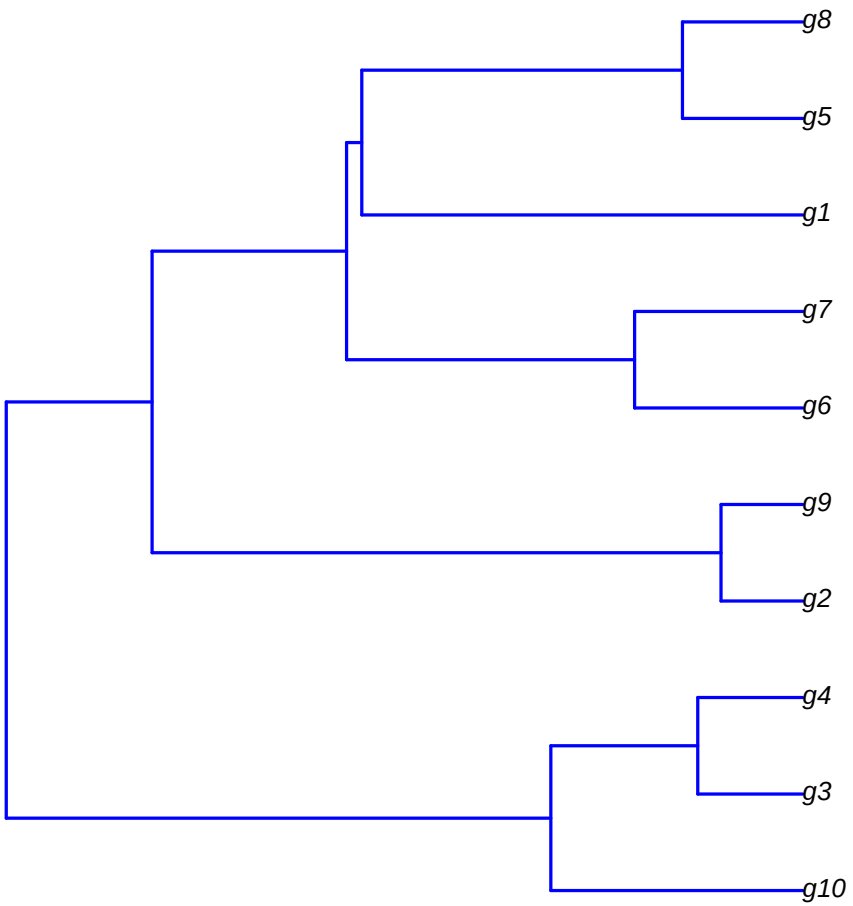
```
> plot(as.dendrogram(hr), edgePar=list(col=3, lwd=4), horiz=T)
```



Tree Plotting II

The ape library provides more advanced features for tree plotting

```
> library(ape)
> plot.phylo(as.phylo(hr), type="p", edge.col=4, edge.width=2,
+            show.node.label=TRUE, no.margin=TRUE)
```



Tree Cutting

Accessing information in hclust objects

```
> hr

Call:
hclust(d = d, method = "complete", members = NULL)

Cluster method   : complete
Number of objects: 10

> ## Print row labels in the order they appear in the tree
> hr$labels[hr$order]

 [1] "g10" "g3"  "g4"  "g2"  "g9"  "g6"  "g7"  "g1"  "g5"  "g8"
```

Tree cutting with cutree

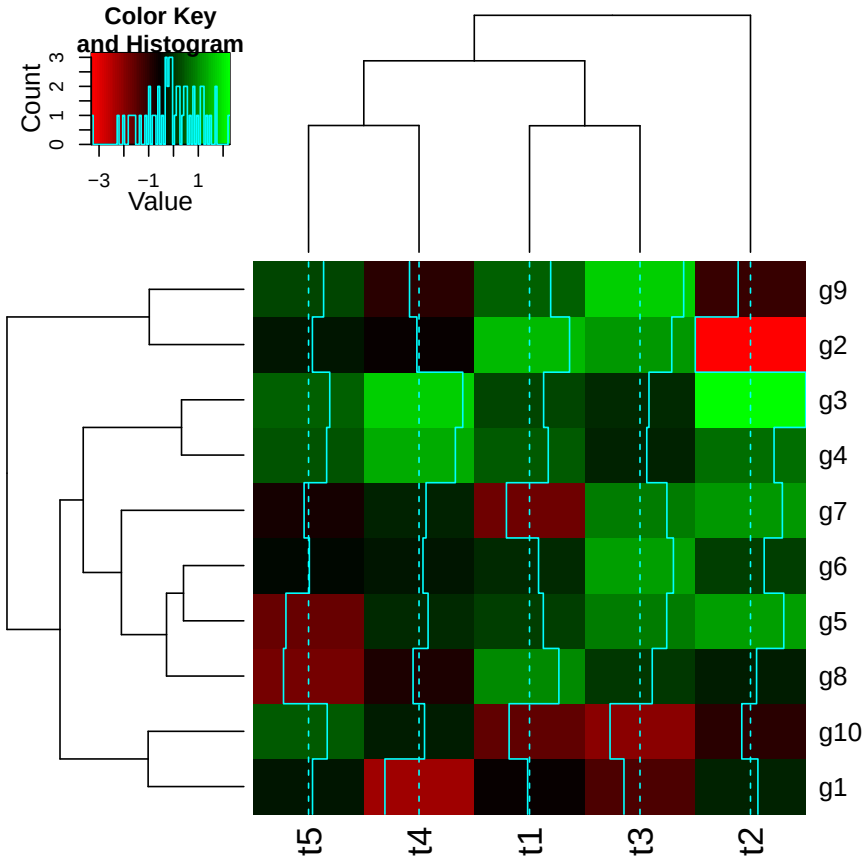
```
> mycl <- cutree(hr, h=max(hr$height)/2)
> mycl[hr$labels[hr$order]]

g10  g3  g4  g2  g9  g6  g7  g1  g5  g8
  3   3   3   2   2   5   5   1   4   4
```

Heatmaps

All in one step: clustering and heatmap plotting

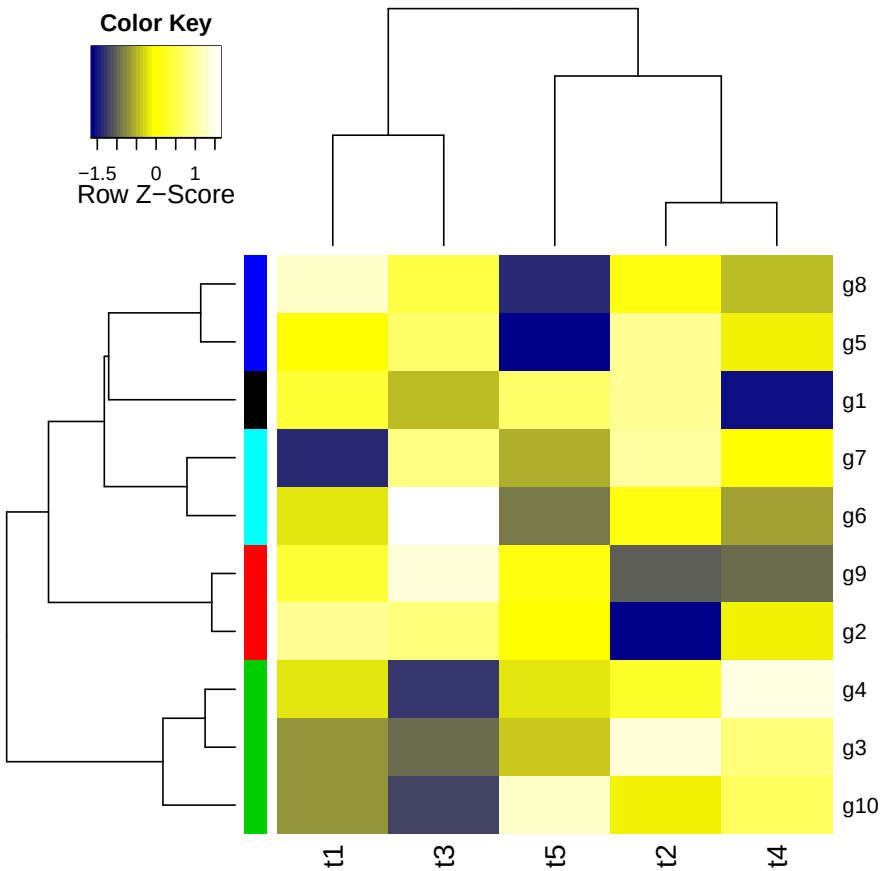
```
> library(gplots)
> heatmap.2(y, col=redgreen(75))
```



Customizing Heatmaps

Customizes row and column clustering and shows tree cutting result in row color bar.
Additional color schemes can be found here Link

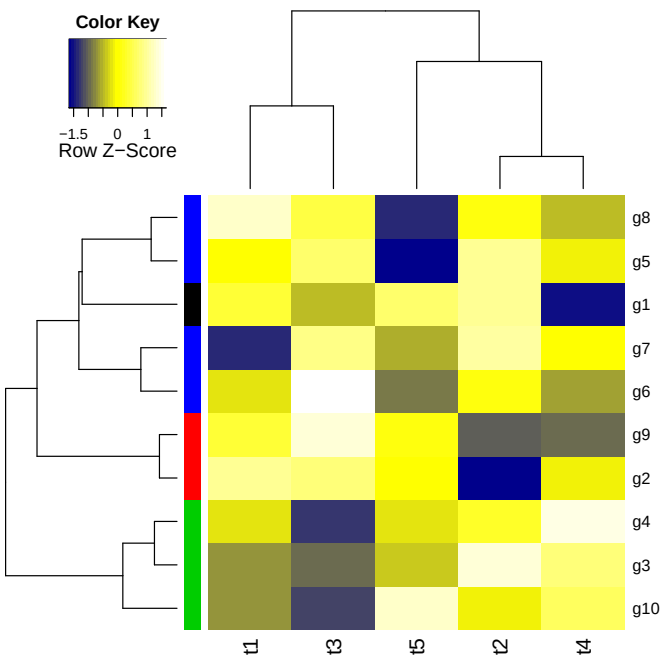
```
> hc <- hclust(as.dist(1-cor(y, method="spearman")), method="complete")
> mycol <- colorpanel(40, "darkblue", "yellow", "white")
> heatmap.2(y, Rowv=as.dendrogram(hr), Colv=as.dendrogram(hc), col=mycol,
+           scale="row", density.info="none", trace="none",
+           RowSideColors=as.character(mycl))
```



K-Means Clustering with PAM

Runs K-means clustering with PAM (partitioning around medoids) algorithm and shows result in color bar of hierarchical clustering result from before.

```
> library(cluster)
> pamy <- pam(d, 4)
> (kmcol <- pamy$clustering)
 g1  g2  g3  g4  g5  g6  g7  g8  g9 g10
  1   2   3   3   4   4   4   4   2   3
> heatmap.2(y, Rowv=as.dendrogram(hr), Colv=as.dendrogram(hc), col=mycol,
+           scale="row", density.info="none", trace="none",
+           RowSideColors=as.character(kmcol))
```



K-Means Fuzzy Clustering

Runs k-means fuzzy clustering

```
> library(cluster)
> fannyy <- fanny(d, k=4, memb.exp = 1.5)
> round(fannyy$membership, 2)[1:4,]
```

```
   [,1] [,2] [,3] [,4]
g1 1.00 0.00 0.00 0.00
g2 0.00 0.99 0.00 0.00
g3 0.02 0.01 0.95 0.03
g4 0.00 0.00 0.99 0.01
```

```
> fannyy$clustering
```

```
 g1  g2  g3  g4  g5  g6  g7  g8  g9 g10
  1   2   3   3   4   4   4   4   2   3
```

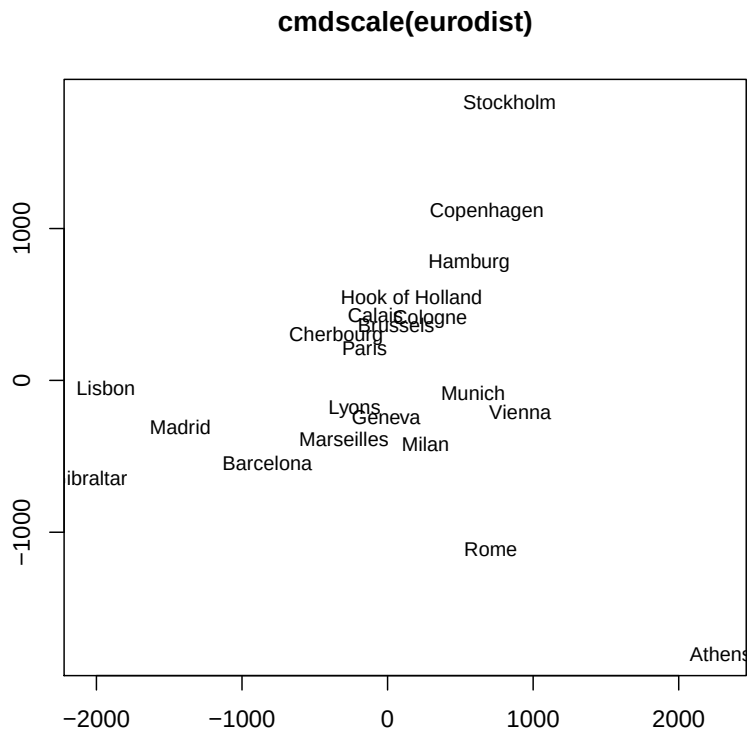
```
> ## Returns multiple cluster memberships for coefficient above a certain
> ## value (here >0.1)
> fannyyMA <- round(fannyy$membership, 2) > 0.10
> apply(fannyyMA, 1, function(x) paste(which(x), collapse="_"))
```

```
   g1    g2    g3    g4    g5    g6    g7    g8    g9   g10
  "1"   "2"   "3"   "3"   "4"   "4"   "4" "2_4"   "2"   "3"
```

Multidimensional Scaling (MDS)

Performs MDS analysis on the geographic distances between European cities

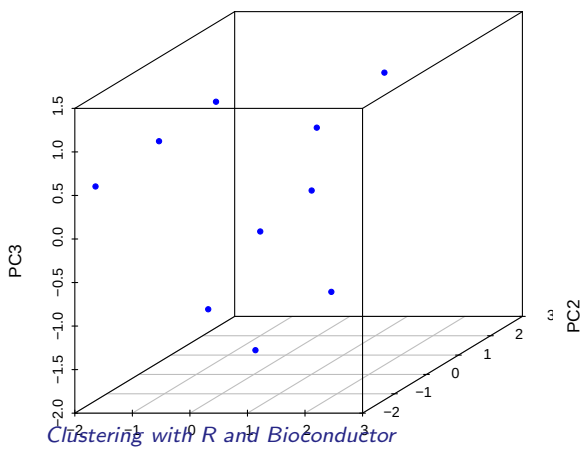
```
> loc <- cmdscale(eurodist)
> ## Plots the MDS results in 2D plot. The minus is required in this example to
> ## flip the plotting orientation.
> plot(loc[,1], -loc[,2], type="n", xlab="", ylab="", main="cmdscale(eurodist)")
> text(loc[,1], -loc[,2], rownames(loc), cex=0.8)
```



Principal Component Analysis (PCA)

Performs PCA analysis after scaling the data. It returns a list with class `prcomp` that contains five components: (1) the standard deviations (sdev) of the principal components, (2) the matrix of eigenvectors (rotation), (3) the principal component data (x), (4) the centering (center) and (5) scaling (scale) used.

```
> library(scatterplot3d)
> pca <- prcomp(y, scale=TRUE)
> names(pca)
[1] "sdev"     "rotation" "center"   "scale"    "x"
> summary(pca) # Prints variance summary for all principal components.
Importance of components:
                          PC1    PC2    PC3     PC4    PC5
Standard deviation     1.3611 1.1777 1.0420 0.69264 0.4416
Proportion of Variance 0.3705 0.2774 0.2172 0.09595 0.0390
Cumulative Proportion  0.3705 0.6479 0.8650 0.96100 1.0000
> scatterplot3d(pca$x[,1:3], pch=20, color="blue")
```



Additional Exercises

See here Link

Session Information

```
> sessionInfo()

R version 2.15.2 (2012-10-26)
Platform: x86_64-apple-darwin9.8.0/x86_64 (64-bit)

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] grid      stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
 [1] scatterplot3d_0.3-33 cluster_1.14.3       gplots_2.11.0
 [4] MASS_7.3-22          KernSmooth_2.23-8    caTools_1.13
 [7] bitops_1.0-4.2       gdata_2.12.0         gtools_2.7.0
[10] ape_3.0-6

loaded via a namespace (and not attached):
[1] gee_4.13-18     lattice_0.20-10 nlme_3.1-105    tools_2.15.2
```