Scientific Computing for Biologists
Hands-On Exercises, Lecture 9

Paul M. Magwene

01 October 2011

Hierarchical Clustering in R

The function `hclust()` provides a simple mechanism for carrying out standard hierarchical clustering in R. The `method` argument determines the group distance function used (single linkage, complete linkage, average, etc.).

The input to `hclust()` is a dissimilarity matrix. The function `dist()` provides some of the basic dissimilarity measures (e.g. Euclidean, Manhattan, Canberra; see method argument of `dist()`) but you can convert an arbitrary square matrix to a distance object by applying the `as.dist()` function to the matrix.

```
> iris.data <- subset(iris, select=-Species)
> iris.cl <- hclust(dist(iris.data), method='single')
> plot(iris.cl) # plot a dendrogram
# let's improve the look a little bit
> plot(iris.cl, labels=iris$Species, cex=0.7)
> # use neg. values of hang to make labels on leaves line up
> plot(iris.cl, labels=iris$Species,  hang=-0.1, cex=0.7)
```

Other functions of interested related to dendrograms include `cuttree()` for cutting the tree at a specified height (or number of groups) and `identify()` for graphically highlighting a cluster of interest in a dendrogram.

```
> plot(iris.cl, labels=iris$Species, cex=0.7)
> interesting.cluster <- identify(iris.cl) # use left-mouse to choose, right-mouse to stop choosing
> interesting.cluster
# [output ommitted]
```

Fancy formatting of dendrogram plots in R is awkward. You need to use the `plot()` function in combination with the `as.dendrogram()` function to access many options. See the help for 'dendrogram' in R for a discussion of options and type `example(dendrogram)` to set some possibilities. A few of them are illustrated here:

```
> plot(as.dendrogram(iris.cl)) # contrast this with plot(iris.cl)
> plot(as.dendrogram(iris.cl), horiz=T) # draw horizontally
> # here's one way to change the labels
> iris.cl$labels <- iris$Species
> levels(iris.cl$labels) <- factor(c("S","Ve","Vi"))
> iris.dend <- as.dendrogram(iris.cl)
> plot(iris.dend)
```

The `heatmap()` function combines a false color image of a matrix with a dendrogram. Here's we apply it to the yeast-subnetwork data set from previous weeks.

```
> yeast <- read.delim('yeast-subnetwork-clean.txt')
> ymap <- heatmap(as.matrix(yeast), labRow=NA) # suppress the numerous row labels
> ymap <- heatmap(as.matrix(yeast),labRow=rownames(yeast)) # w/row labels, kinda messy
```

The R package `cluster` provides some slightly fancier clustering routines. The basic agglomerative clustering methods in `cluster` are accessed via the function `agnes()`

Compare the results of different hierarchical clustering methods (single linkage, complete linkage, etc.) as applied to the iris data set using the hclust() or agnes() functions. For single and average linkage use both Euclidean and Manhattan distance as the dissimilarity measures.

Neighbor joining in R

The package `ape` provides an implementation of neighbor joining in R (and many other useful phylogenetic methods). Here's a couple of examples of using neighbor joining taken from the `ape` documentation:

```
library(ape) # install ape if need be
### From Saitou and Nei (1987, Table 1):
x <- c(7, 8, 11, 13, 16, 13, 17, 5, 8, 10, 13, 10, 14, 5, 7, 10, 7, 11, 8, 11, 8, 12, 5, 6, 10, 9, 13, 8)
M <- matrix(0, 8, 8)
# create a symmetric matrix by filling upper and lower triangles
# of the matrix M
M[row(M) > col(M)] <- x
M[row(M) < col(M)] <- x
rownames(M) <- colnames(M) <- 1:8
tree <- nj(M)
plot(tree, "u")

### a less theoretical example
?woodmouse  # check out the info about the
            # woodmouse data set in the ape package
data(woodmouse)
dist <- dist.dna(woodmouse) # see the help on the dist.dna fxn
tree.mouse <- nj(dist)
plot(tree.mouse)
```

Multidimensional scaling in R

Metric MDS

The implementation of classic metric scaling in R is carried out using the `cmdscale()` function. Read the documentation for `cmdscale` and then work through the example showing the application of MDS to analysis of road distances between US cities available at the following link (but see notes below first):

http://personality-project.org/r/mds.html.

As you work through your example note the following:

- You can use the `source()` function not only with a local file but also with a URL. This is convenient but potentially a security issue so don't run code willy nilly without checking out what it does.
- You can download the code at http://personality-project.org/r/useful.r and check out the functions that it includes. I thought the `read.clipboard()` function was particularly nice.

Minimum Spanning Tree in R

The package `ape` has an `mst()` function. Several others packages, including vegan also have minimum spanning tree functions. The `mst()` function takes a dissimilarity matrix as its input and returns a square adjacency matrix, A, where $A_{ij} = 1$ if (i, j) is an edge in the MST or 0 otherwise.

Here's an application of the MST function to the cities example you completed above.

```
> library(ape) # install ape first if necessary
> city.mst <- mst(as.dist(cities))
> city.mst # see the adjacency matrix return by mst
```

If you want to create a nice looking plot you can use the `mat2listw()` function in the package `spdep`. `mat2listw` converts the adjacency matrix into a form that you can extract the neighbor information from:

```
> library(spdep) # install spdep first if necessary
> plot(city.location, type='n', xlab='PCoord1', ylab='PCoord2')
> text(city.location, labels=names(cities))

# note British spelling of 'neighbours'
> plot(mat2listw(city.mst)$neighbours, city.location, add=T)
```

Non-metric MDS

The `isoMDS()` function in the `MASS` package implements the Shepard-Krusal version of non-metric scaling, while the `sammon()` function in the same package use the criterion proposed by Sammon (1969). You will need to utilize these functions, along with `cmdscale` and the hierarchical clustering functions covered last week for the following assignment.

Assignment:

Harding and Sokal (1998; PNAS 85:9370-9372; see course wiki) used cluster analysis and non-metric MDS to explore the relationship between European language families as measured by genetic distances among the people who speak those languages. The classification they derived at large reflects geographic proximity but there are some language families that have distant genetic relationships to their geographic neighbors.

Harding and Sokal provide a table of genetic distances that they used in their analyses. Use R to reconstruct the cluster analysis they report (Fig. 1) and repeat this analysis using neighbor joining. In a similar manner use both metric scaling and the Shepard-Kruskal and Sammon criteria for non-metric scaling to do an MDS analysis (similar to Harding and Sokal's fig. 2). Try to also recreated the MST shown in their figure 2.

Submit your figures and a brief paragraph describing what differences, if any, you found in your re-analysis of Harding and Sokal data. Are these differences significant (i.e. do they change your interpretation of the data)?