**Cluster Analysis**

Cluster Analysis is a methodology that attempts to form groups (clusters) of objects that are similar to each other but that differ among clusters (i.e., greater variation between than within clusters, or put another way, cluster analysis maximizes within-group similarity and minimizes between-group similarity). The exact definition of "similar" varies by algorithm. The methods of forming clusters also vary but follow a few general blueprints.

Cluster Analysis is used where pre-specified groups do not already exist. (If you already have groupings of data, then you’ll employ a Discriminant Function Analysis, the topic of the next lesson.)

Because clustering is based on similarity/dissimilarity, you start by making a distance matrix of your data.

**Exercises:**

Open a new RStudio session (with your class working directory) with the following libraries:

*cluster* – note that this is a package we haven’t used before, so you’ll have to install it first

*labdsv*

*MASS*

*MVA*

*optpart*

*picante*

*stats*

*vegan*

**Cluster algorithms:**

There are different forms of Cluster Analysis; I will present the two most commonly used ones today: hierarchical clustering and disjoint (also called non-hierarchical or partitioning) clustering.

**Hierarchical clustering:**

To assign entities (rows = sites) to groups and display relationships among groups as they form, a **hierarchical** algorithm is used. It produces a tree-like graph (called a **dendrogram**), with branching pattens that indicate similarity (clustering) of different entities. This can be done in either of two ways: by grouping things based on similarity (**agglomerative approach**) or by identifying distinctions (**divisive approach**). Agglomerative approaches are by far more common in ecology because they are more useful at identifying smaller clusters whereas divisive approaches are better at finding large clusters, which are generally already readily apparent.

In an agglomerative hierarchical cluster analysis, the two sites that are most similar (least dissimilar) to each are fused at what is called a node, forming the first cluster. This is repeated, with subsequent sites fused one-by-one in order of highest similarity to the cluster to which they are most similar. Eventually, all sites are accounted for. In the resulting dendrogram (also called a tree), the length of each branch indicates the similarity between variables and nodes.

Agglomerative cluster algorithms differ in how they calculate similarity. Commonly used ones include (as the argument in R and its more widely known name):

* single --- single linkage
* complete --- complete linkage
* average --- average linkage

In the single linkage method, similarity is determined by the maximum similarity of a site to any of the sites in a given cluster. This is akin to a nearest-neighbor approach. The name single linkage is because the site only needs to be similar to a single member of the cluster to join. Single linkage cluster dendrograms can be long and are subject to a phenomenon called chaining, whereby a single site is continually added to the tail of the biggest cluster. Thus, this method is not recommended for most ecological applications (McGarigal et al. 2000).

(Chaining is possible with the other methods; when it accounts for 15-25% of the branchings, you will need to decide whether the sample units are truly very distinctive [meaning that the dendrogram is appropriate] or perhaps if the distance measure you chose wasn’t the best for your dataset. Try transforming your data and/or using a difference distance metric; if those don’t improve performance/reduce chaining, try using another clustering method.)

In complete linkage, similarity is calculated as the minimum similarity of the site to any member of the cluster. Similar to the single linkage algorithm, the probability of a site joining a cluster is determined by a single other member of the cluster, but now it is the least similar (farthest-neighbor approach), not the most similar. Complete linkage cluster dendrograms tend to be compact, with tight clusters that overestimate the differences among groups.

In average linkage, similarity is defined by the mean similarity of the site to all the members of the cluster. In contrast to single linkage, a site needs to be relatively similar to *all* the members of the cluster to join, rather than just one as in the previous two methods. This is perhaps the most commonly used hierarchical clustering algorithm, although complete linkage is also a contender for most-used.

There are also some other agglomerative hierarchical clustering algorithms available in R that are less commonly used:

* ward --- Ward's minimum variance method
* mcquitty --- McQuitty's method (group size-weighted average distance method)
* median --- median similarity
* centroid --- geometric centroid (median linkage method)
* flexible --- flexible Beta (Lance-Williams formula)

All of these methods are sensitive to abundances, such that very abundant species tend to cluster together away from rare species even if they occur in the same sampling units (sites)! It is thus recommended that you relativize species abundance data before calculating a distance matrix and performing a Cluster Analysis. (One such way is via scale();  with default settings, this argument will calculate the mean and standard deviation of the entire vector, then "scale" each element by those values by subtracting the mean and dividing by the sd. If you use scale(x, scale=FALSE)it will only subtract the mean but not divide by the sd. Another approach is with the decostand() function you used in the “Site x environment” lesson.)

**Example of a hierarchical Cluster Analysis:**

We will use the hclust() function from *stats* to perform hierarchical cluster analysis. hclust() will perform a cluster analysis from either a similarity or dissimilarity matrix, but it plots better when working from a dissimilarity matrix. We can use any dissimilarity object from dist(), vegdist(), or dsvdis()(different packages’ arguments for calculating such a matrix).

Let’s use the grassland.community.csv file from the course website. Read in that file as an R object named comm (with header = TRUE and row.names = 1):

comm <- read.csv("grassland.community.csv", header = TRUE, row.names = 1)

Recall (or explore the data to refresh your memory) that there were replicate sampling locations in two habitat types (fescue and mixed grass). Let’s see whether sampling locations exhibit grouping by habitat type or otherwise.

Abundance data typically need to be relativized first so as to avoid problems with species that occurs at vastly different scales of abundance (100s or 1000s of individuals vs. singletons). But these are percent cover data, so they are already relativized!

Now you need to **construct a dissimilarity (distance) matrix. To decide which method to use to do so**, you should probably refresh your memory from the “Ordination” lesson. If you examine the data in grassland.community.csv, you’ll see:

-it is a homogeneous data matrix consisting of % cover data (numbers between 0 and 100) 🡪 that eliminates the Steinhaus and Hellenger

-there are lots of 0’s 🡪 that eliminates Euclidean, Manhattan, Canberra

-Mahalanobis isn’t needed here

-Chi-square de-emphasizes abundant species, of which there are some in this dataset, so that’s not the best choice

Therefore, Bray-Curtis is the best choice for these data and is one of the most widely used methods because it is so flexible.

We will use *vegan*’s vegdist() to calculate Bray-Curtis dissimilarity:

comm.bc.dist <- vegdist(comm, method = "bray")

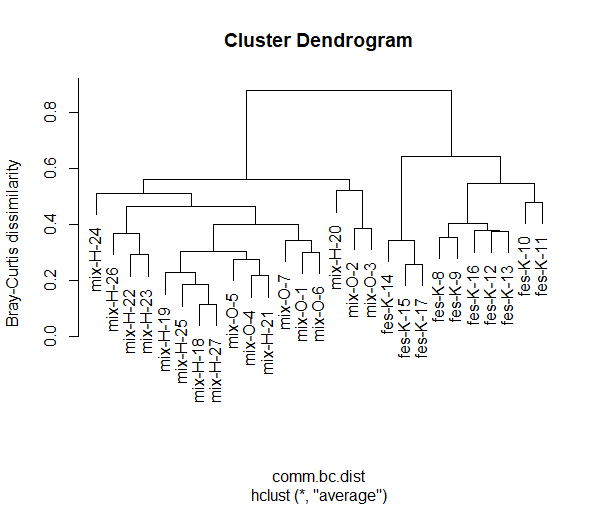
We will use the average linkage method in this example because it’s the most recommended clustering method in general.

In hclust(), the dissimilarity object is the first argument, and the method is the second:

comm.bc.clust <- hclust(comm.bc.dist, method = "average")

#Plot the dendrogram (in the form of an icicle plot):

plot(comm.bc.clust, ylab = "Bray-Curtis dissimilarity")



In this dendrogram, each fusion of sites into a cluster is shown as a horizontal line, and the sites are labeled at the bottom of the graph. You “read” the tree from the top down (in the case of an “icicle plot” like this one; you may also see cluster dendrograms that are horizontal and read those left to right). The vertical axis is similarity, so you can see how the “bottom” branches are individual units (similarity of 0 to any others).

**How would you interpret the output?**

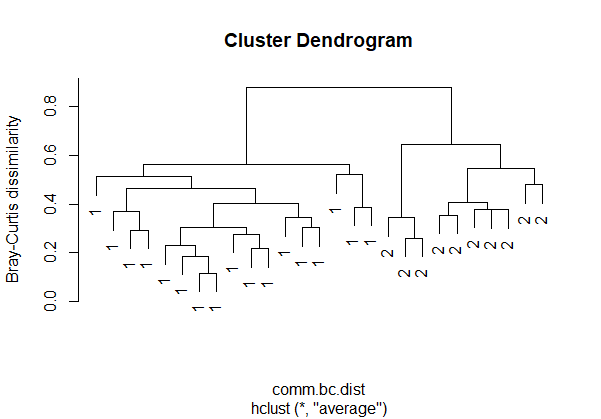
The cluster analysis can be "sliced" horizontally to produce unique clusters by specifying the number of clusters desired—this is the “hierarchical” part of hierarchical cluster analysis. For example, let’s specify that we want a dendrogram with 2 clusters (since the habitat types appear to separate; you can play around with different numbers, too):

comm.bc.clust.cut<-cutree(comm.bc.clust,k=2)

(By specifying the number of clusters [k], you are really doing a disjoint (non-hierarchical) form of clustering, which is what a later section is about.)

Then, to label the dendrogram with IDs, use:

plot(comm.bc.clust, labels = as.character(comm.bc.clust.cut))

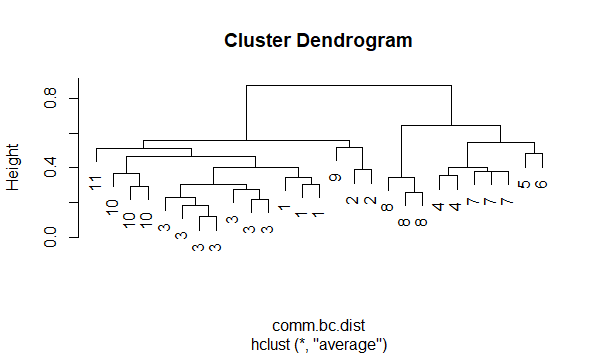


The appearance of the plot is the same as before, it’s just that the labels now fall into only two groups.

You can also partition the graph by selecting a “height” on the Y-axis and “slicing” the dendrogram horizontally. For example, let’s say you decided to form clusters at height = 0.4:

clustcut<-cutree(comm.bc.clust,h=0.4)

plot(comm.bc.clust, labels = as.character(clustcut))



Recall that there were 27 sites; by slicing at height = 0.4, those 27 sites formed 11 clusters.

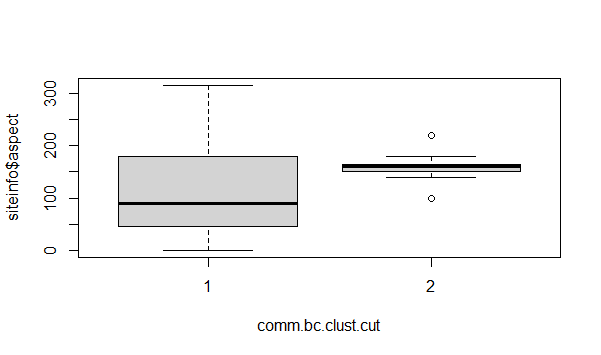
If you want to determine the number of members in each cluster:

table(comm.bc.clust.cut)

You can perform environmental analyses of the clusters using various other plotting techniques in R. For example, to examine differences in aspect between the clusters (since aspect is associated with temperature and moisture and, thus, likely to habitat type), we can do a boxplot (must read in site data from plot.metadata.csv first):

siteinfo <- read.csv("plot.metadata.csv", row.names=1)

boxplot(siteinfo$aspect~comm.bc.clust.cut)



**How would you interpret the resulting graph?**

To verify that the dendrogram reflects your data well:

One way to measure how well the dendrogram generated by hclust() reflects your data is to compute the correlation between the values on the vertical axis (height or dissimilarity/distance, which are more properly called cophenetic distances) and the distance matrix generated by the vegdist() function. If the clustering is valid, the linking of objects in the dendrogram should have a strong correlation with the distances between objects in your distance matrix. The closer the value of the correlation coefficient is to 1, the more accurately the clustering solution reflects your data. Values above 0.75 are felt to be good.

To do so, first use cophenetic()to compute the cophenetic distances for your hierarchical cluster dendrogram:

comm.coph <- cophenetic(comm.bc.clust)

Then do a correlation analysis between the cophenetic distances and the distance matrix we created with the Bray-Curtis distance metric:

cor(comm.bc.dist, comm.coph)

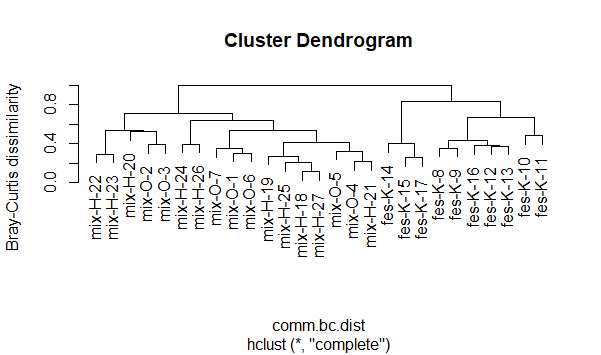
[1] 0.9316768

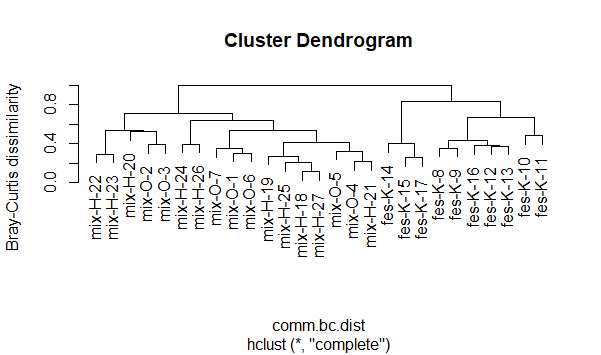
This indicates that our tree is a very good fit. But be advised that the average linkage method appears to produce high values of this statistic. (This may be one reason why this method is so popular.) Thus, don’t rely on just this correlation coefficient: also examine the structure of the tree to make sure it appears logical.

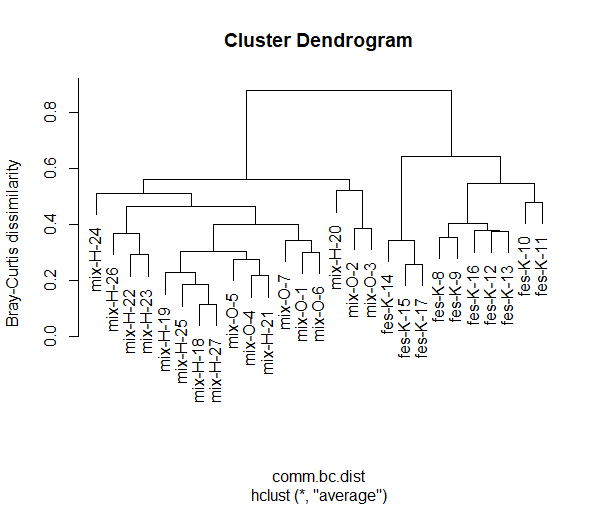
Now let’s compare the output above, generated with the average linkage method, to the complete linkage approach:

comm.bc.complete <- hclust(comm.bc.dist, method = "complete")

plot(comm.bc.complete, ylab = "Bray-Curtis dissimilarity")



For ease of comparison, I’ll copy/paste the dendrograms produced by the average method (left) and complete method (right), stretching them vertically to make the vertical axes roughly the same size:



It is readily apparent from these figures that the clustering patterns differ by method. But even so, both methods show that mixed grass and fescue separate into distinct clusters.

**Disjoint clustering:**

In contrast to the hierarchical approaches discussed above, non-hierarchical clustering separates all variables or objects into individual, mutually exclusive clusters. Thus, you must specify the number of clusters *a priori*. In practice, you usually don't know the number of clusters present, so the approach used is to examine a range of values. (Below, I’ll discuss a method for picking and evaluating the number of clusters.).

Unlike hierarchical clustering techniques, non-hierarchical ones are insensitive to outliers and can in fact be used to detect outliers because those values tend for form clusters of their own that are quite separated from other clusters. (We will do an example of that today.)

In disjoint clustering, the clusters are typically formed from random initial settings. All variables are then assigned to one of the pre-defined clusters based on the shortest distance of each variable to cluster centroids. The clusters are not hierarchical; they partition the data differently.

By far the most common procedure for doing so is k-means clustering, which partitions all of your variables into a pre-defined number (k) of clusters. The initial cluster reference profiles (variable’s value for each object) is typically generated randomly; all variables are then assigned to one of these pre-defined clusters based on the shortest distance of each variable to cluster centroids. Each cluster reference profile is then recalculated based on the mean of the variables in that cluster, and then all variables are repartitioned into clusters. This is repeated until a stable configuration is achieved. Because the initial cluster reference profiles are random, repeating your analysis on the same dataset and same number of clusters can lead to a different final partition; therefore, you should repeat this process many times and choose the clustering result that has the lowest total error sum of squares (an assay of how similar variables are within each cluster).

To do this in R we will use the pam() function (which stands for “partitioning around medoids,” with medoids being R-speak for cluster centers) from the *cluster* library (could also use kmeans() in *stats*) on our Bray-Curtis distance matrix comm.bc.dist. I will pick 5 as the number of clusters just as an example, but you should pick a number that is based on some reason (e.g. some pattern you can discern from your data that hints at how many clusters might exist):

demopam <- pam(comm.bc.dist,k=5)

attributes(demopam)

$names:

[1] "medoids" "id.med" "clustering" "objective" "isolation"

[6] "clusinfo" "silinfo" "diss" "call"

$class:

[1] "pam" "partition"

The sites that served as cluster centers (medoids) are indicated by demopam$medoids:

demopam$medoids

[1] "mix-H-18" "mix-O-3" "fes-K-12" "fes-K-11" "fes-K-17"

The cluster membership for each site is given by:

demopam$clustering

mix-O-1 mix-O-2 mix-O-3 mix-O-4 mix-O-5 mix-O-6 mix-O-7 fes-K-8 fes-K-9

1 2 2 1 1 1 1 3 3

fes-K-10 fes-K-11 fes-K-12 fes-K-13 fes-K-14 fes-K-15 fes-K-16 fes-K-17 mix-H-18

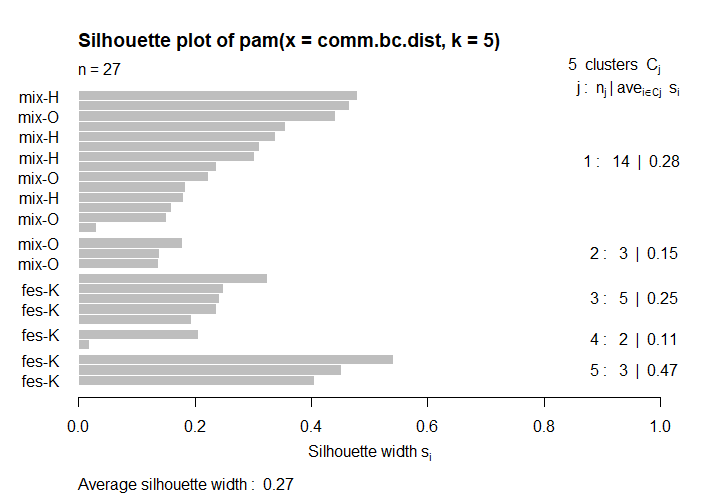
4 4 3 3 5 5 3 5 1

mix-H-19 mix-H-20 mix-H-21 mix-H-22 mix-H-23 mix-H-24 mix-H-25 mix-H-26 mix-H-27

1 2 1 1 1 1 1 1 1

Now let’s plot it:

plot(demopam)



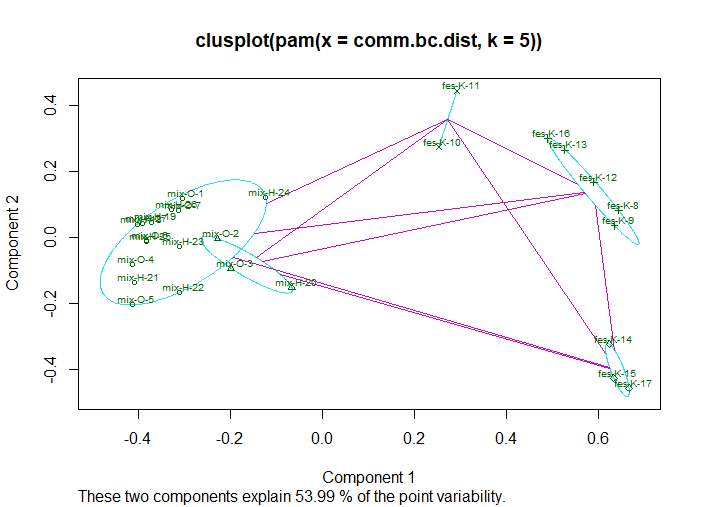
This kind of plot is called a **silhouette plot** and shows for each cluster:

* the number of sites per cluster = number of horizontal lines, also given in the right-hand column;
* the mean similarity of each site to its own cluster minus the mean similarity to the next most similar cluster (given by the length of the lines), with the mean in the right-hand column; and
* the average silhouette width.

Sites that fit well within their cluster have a large positive silhouette width; those that fit poorly have a small positive or even a negative silhouette width. I have seldom seen silhouette analysis in ecology, however, perhaps because “large positive silhouette width” is a relative measure (how large is large?).

There is also a clusplot() function in *cluster* that can be used to plot the results of pam():

clusplot(demopam, labels = 3, cex = 0.65)



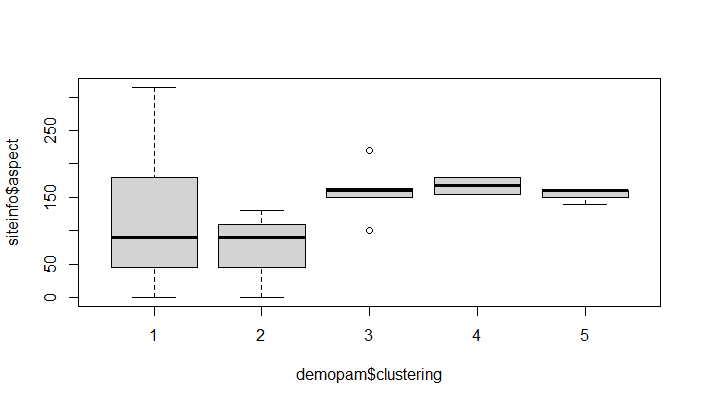
Notice the overlap in a couple of the ovals; this suggests that these aren’t two distinct clusters but perhaps are a single cluster.

The labels = 3 labels each point by site name and cex = 0.65 designates the text size. clusplot() has other plotting options regarding colors, lines, and labels; use ?clusplot for more info.

(If you have a very large dataset, there is a faster alternative to pam() in *cluster*, called clara(). It uses a simpler algorithm to determine clusters, which can save on computer memory and processing time. It cannot operate on a dissimilarity matrix, you must use a distance matrix (such as Euclidean or Manhattan).)

The $clustering values can be used to examine patterns with some other variable, like we did before with the hierarchical clustering example:

boxplot(siteinfo$aspect~demopam$clustering)



**How would you interpret this plot?**

**To try to pick the “best” value of k** (if you do not have a clear idea of the number of clusters *a priori*), then you need to go back to your distance matrix and create a hierarchical cluster object that you then cut into different numbers of clusters:

comm.clust <- hclust(comm.bc.dist, method = "complete")

cutree(comm.clust, k=2:5)

In this result, you can compare clustering from designating 2 to 5 clusters:

2 3 4 5

mix-O-1 1 1 1 1

mix-O-2 1 1 2 2

mix-O-3 1 1 2 2

mix-O-4 1 1 1 1

mix-O-5 1 1 1 1

mix-O-6 1 1 1 1

mix-O-7 1 1 1 1

fes-K-8 2 2 3 3

fes-K-9 2 2 3 3

fes-K-10 2 2 3 4

fes-K-11 2 2 3 4

fes-K-12 2 2 3 3

fes-K-13 2 2 3 3

fes-K-14 2 3 4 5

fes-K-15 2 3 4 5

fes-K-16 2 2 3 3

fes-K-17 2 3 4 5

mix-H-18 1 1 1 1

mix-H-19 1 1 1 1

mix-H-20 1 1 2 2

mix-H-21 1 1 1 1

mix-H-22 1 1 2 2

mix-H-23 1 1 2 2

mix-H-24 1 1 1 1

mix-H-25 1 1 1 1

mix-H-26 1 1 1 1

mix-H-27 1 1 1 1

Obviously the larger the value of k, the more clusters you will have. What you’re looking for is something logical and not choppy, so it looks like 2 clusters is still best. (However, be aware that this approach is subject to “confirmation bias.”)

The increasing numbers of clusters can allow you to **identify outliers**; for example, at k=3, sites fes-K-14, 15, and 17 are different from the other fescue sites (form their own cluster). Thus, Cluster Analysis can be used to identify subunits within datasets that should possibly be analyzed separately.

So let’s go back and designate 3 clusters:

comm3 <- cutree(comm.clust, k=3)

#Make plot labelled with cluster numbers:

plot(comm.clust, labels = as.character(comm3))

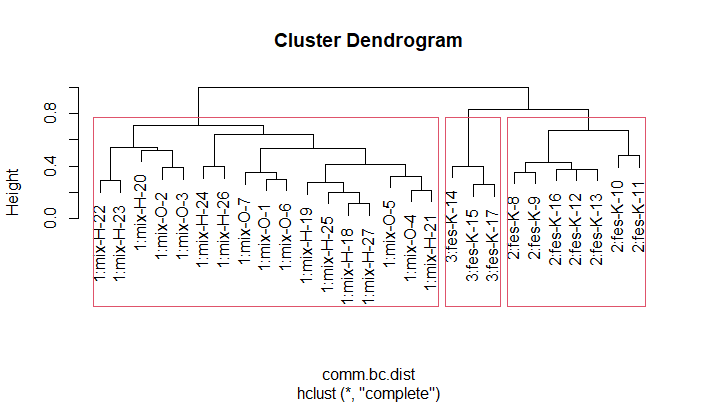
#Add cluster names to plot:

comm.lab <- paste(as.character(comm3), names(comm3), sep=":")

plot(comm.clust, labels=comm.lab)

#Highlight the clusters:

rect.hclust(comm.clust, k=3)



If you examine the validity of this dendrogram via the correlation analyses mentioned earlier, you’ll find that the result will be the same as for the dendrogram produced before changing the value of k. **Why do you think this might be so?**

**What if I only have presence/absence data?**

The grassland.community.csv data were abundance data in the form of % cover values. But sometimes all you have is presence/absence data. Let’s go back to our old friends the ground beetles (GBbiol.csv for a presence/absence site x species matrix). Read them in as gb.biol (with row.names = 1). Recall that there were 18 sites in three habitat types (Edge, Wood, Grass, with 6 sites in each type). We can use Cluster Analysis to determine whether any of the sites cluster together in terms of similarity, which can be especially useful for presence/absence data (go back to the “Diversity indices” lesson to refresh your memory on similarity/beta diversity). We will use *vegan*’s betadiver() to calculate similarity, specifying which of 24 different methods (similarity indices) to use. Whittaker’s w is one of the most common:

#Create a matrix of similarity values:

gb.beta <- betadiver(gb.biol, method = "w")

#Create a dendrogram via hierarchical clustering and plot it:

gb.clus <- hclust(gb.beta)

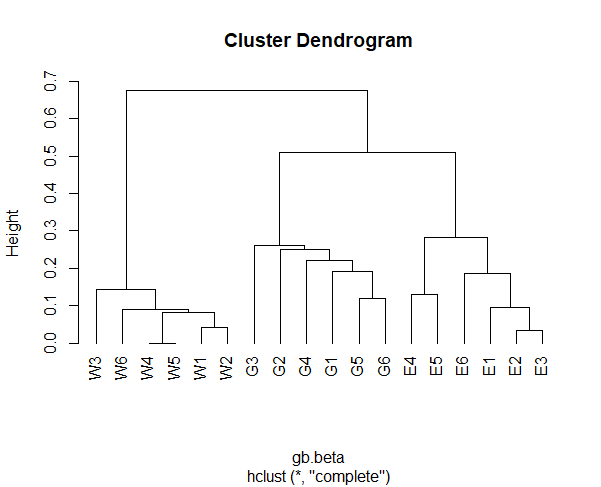
(Because no method was specified in the hclust() statement above, we will use hclust’s default method, which is complete linkage.)

plot(gb.clus)

#If you want a little less compressed dendrogram:

plot(gb.clus, hang = -2)

(The hang argument lets the labels hang down below the 0. However, it can obscure some patterns to identify outliers because it effectively “pulls” all of the labels down.)



From this dendrogram, you can see that sites from each of the three habitat types form distinct clusters. Wood sites are the most distinct whereas Grass and Edge are relatively more similar to each other than either is to Wood (at least in terms of beta diversity). (And if you look back at the lesson on diversity indices, you’ll see that this result is consistent with other analyses you did that compared these habitat types.)

There is also a form of cluster analysis called fuzzy cluster analysis; it’s called fuzzy because the algorithm can be altered to give more or less leeway in assigning an observation to a cluster. You designate the number of clusters as in k-means clustering. The fanny() command in *cluster* can be used for this; for more information, I recommend Gardener (2014).

**References:**

Gardener, M. 2014. *Community Ecology: Analytical Methods Using R and Excel*. Pelagic Publishing, Exeter, UK.

McGarigal, K., S. Cushman, and S. Stafford. 2000. *Multivariate Statistics for Wildlife and Ecology Research*. Springer, New York, NY.

**Assignment:** due 0800 Monday, April 26

Start a fresh RStudio session. Remember to set your working directory to your course folder and use the same package libraries as we used today.

**Q1. For the Ground\_beetles\_abundance.csv data, make a Euclidean dissimilarity matrix and then perform a hierarchical cluster analysis on it using the complete linkage method. (You can skip relativizing the data for this question.) Plot the dendrogram, verify it, and interpret it. As a hint, do not use the hang argument in your plot function. If the dendrogram indicates anything “odd,” then perform a k-means analysis to identify, plot, and label meaningful clusters.**

**Q2. Now use those same data to compare the complete linkage you did in Q1 to results from single and average clustering methods. Of the three hierarchical clustering methods, which would you choose to best represent these data?**

Make an RMarkdown Word file of your work and turn that in. Be sure to include your answers to the questions asked! Turn in your assignment as a Word document via email to [iroro.tanshi@ttu.edu](mailto:iroro.tanshi@ttu.edu) no later than 8:00 a.m. on Monday of next week. In your email, please include the following as the Subject line:

Assignment on cluster analysis