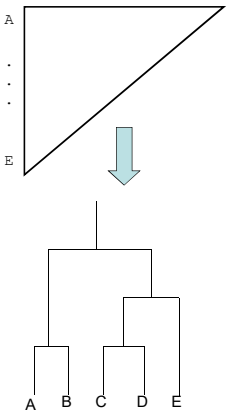
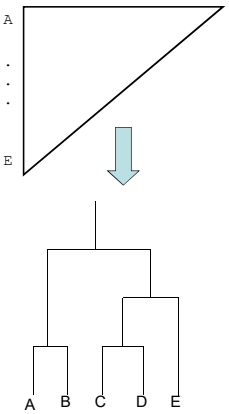
Clustering

- Method to categorize multivariate data
- Start with a triangular distance matrix
- Put samples into groups and display groups as a dendrogram
- Dendrograms represent hierarchical grouping of all samples as bifurcating tree.
- R package: cluster



Clustering

- Clustering is largely a visualization technique. Also used for assigning groups.
- Information is lost in an attempt to simplify for the sake of grouping
- Since we start with a distance matrix, the link back to original data is also lost
- Assumption is that meaningful clusters exist, analyses find them based on distances.

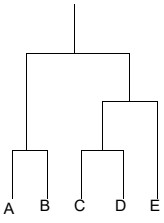


Clustering basics

- **Hierarchical**:
  - **Agglomerative**: start with no groups, form groups by combining the most similar pairs.
    - UPGMA
  - **Divisive** : start with one large group, divide into groups based on dissimilarity.
- **Non Hierarchical**:
  - Have a predefined number of clusters, place samples into groups.
  - K-means cluster or Partition around medoids, not based on similarity matrix
- **Assumption of both** – all samples will be put into a group regardless of any biological significance.

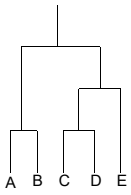
Output from Hierarchical cluster

- List of the pairwise clusters (n-1)
- Height (dissimilarity) of each cluster (n-1)
- Order of clustered objects (n)
- Dendrogram
- How many clusters?



Cophenetic Distance and Cophenetic Correlation

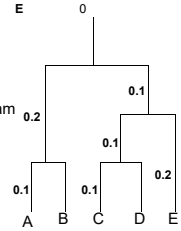
- Once clustered, similarities among groups can be expressed as branch lengths.
  - **Cophenetic distance** – pairwise distance among all samples. The height of connecting branch among any two samples.
  - **Cophenetic correlation** – correlation between original distance and cophenetic distance. Higher = clustering better represents structure in data



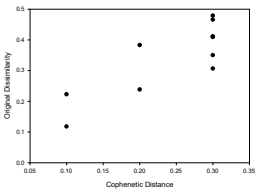
Dissimilarity

	E	D	C	B	A
A	0.349898	0.408985	0.411381	0.222683	0
B	0.4788	0.306381	0.465702	0	
C	0.382983	0.117582	0		
D	0.238272	0			
E	0				

Dendrogram



Cophenetic Correlation

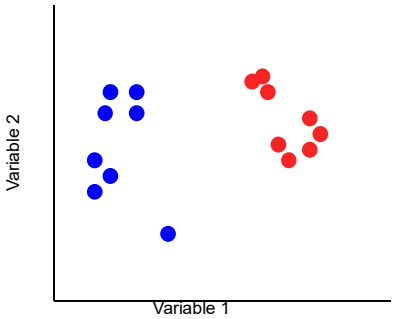


Cophenetic Dissimilarity

	E	D	C	B	A
A	0.3	0.3	0.3	0.1	0
B	0.3	0.3	0.3	0	
C	0.2	0.1	0		
D	0.2	0			
E	0				

Agglomerative Clustering

At each stage the two nearest groups/clusters are grouped.



Agglomerative Coefficient

$$ac = \frac{\sum_{i=1}^{n} m_i}{n}$$

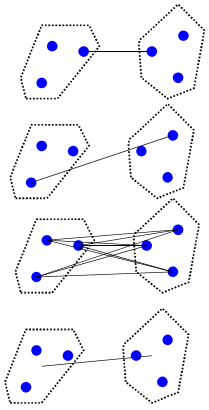
Where m_i is the dissimilarity of each sample/individual to it's nearest cluster divided by the dissimilarity of the most distant sample/individual.

N=number of clusters (sample size)

Unbounded number, increases with sample size.

Agglomerative Clustering

- Multiple methods differ only in how successive clusters are formed – what defines similarity among clusters.
- Methods
  - UPGMA (average)
  - Single
  - Complete
  - Ward's (sums of squares within cluster)
  - WPGMA (weighted)
  - Unweighted Centroid



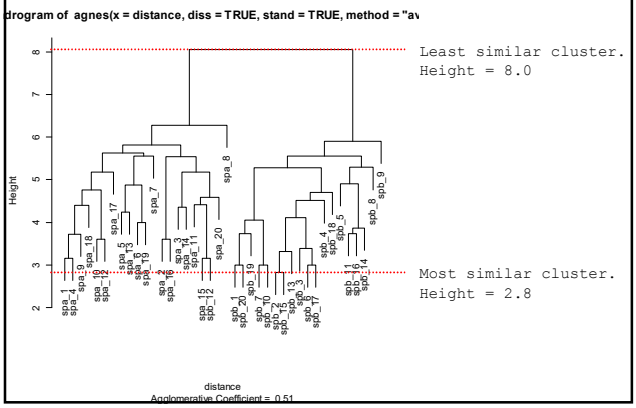
Agglomerative Clustering

- Code:
  - `cluster<-hclust(distance, method="average")`
  - `summary(cluster)`
  - `plot(cluster)`
- Summary
  - order of samples (as in dendrogram)
  - list of clusters with height
  - Agglomerative coefficient from function coef.hclust
- Plot
  - Dendrogram – visual summary of clusters
- **Install fastClust to replace hclust with a faster alternative. You will need this for the assignment**.

Cophenetic Distances and Correlation

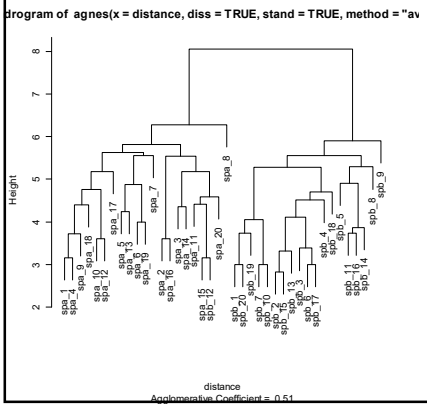
- Function cophenetic calculates cophenetic distances for a tree
  - `tree_dist<-cophenetic(UPGMA)`
- The cophenetic correlation is then the relationship between the original distance matrix and the copenetic distance matrix
  - `plot(tree_dist, distance)`
- The correlation between these two measures the quality of the tree (1.0=perfect tree)
  - `cor(tree_dist, distance)`

Dendrogram and heights



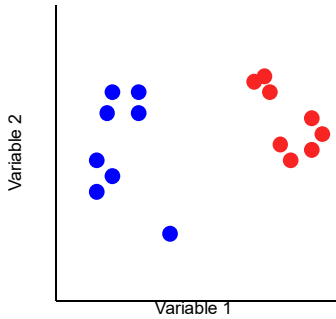
Cutree function

```
cut_tree<-cutree(cluster,k=2:5)
```




```
      2 3 4 5
 [1,] 1 1 1 1
 [2,] 1 1 1 2
 [3,] 1 1 1 2
 [4,] 1 1 1 1
 [5,] 1 1 1 1
 [6,] 1 1 1 1
 [7,] 1 1 1 1
 [8,] 1 2 2 3
 [9,] 1 1 1 1
[10,] 1 1 1 1
[11,] 1 1 1 2
[12,] 1 1 1 1
[13,] 1 1 1 1
[14,] 1 1 1 2
[15,] 1 1 1 2
[16,] 1 1 1 2
[17,] 1 1 1 1
```

Divisive Clustering



Divisive Analysis Clustering – keep dividing until each observation in its own cluster
Partitioning – divide into a pre-defined number of groups

Divisive Coefficient

- Similar to agglomerative coefficient.
- d_i = dissimilarity within cluster divided by the maximum dissimilarity in the dataset

$$dc = \frac{\sum_{i=1}^{n} d_i}{n}$$

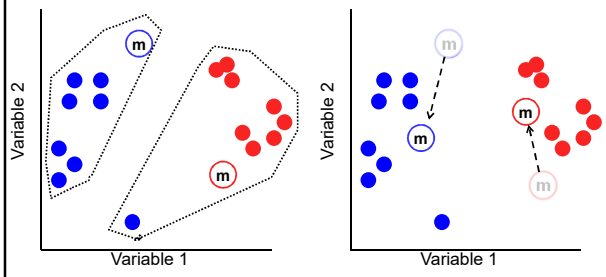
Divisive Analysis Clustering

- Code:
  - `cluster<-diana(morphology, diss=FALSE, metric="euclidian", stand=TRUE, keep.diss=TRUE)`
  - `summary(cluster)`
  - `plot(cluster)`
- Accepts a dissimilarity matrix or raw data (diss=FALSE, specify which metric to use)
- At each iteration divides where there is the largest mean dissimilarity
- Output similar to agglomerative methods

K-means and Partitioning Around Medoids

- Indicate number of clusters (but not membership) *a priori*
- Samples divided into clusters to minimize intra-cluster variability
- Can indicate starting group **centroids** or the analyses will guess
- Iterative process
- Not hierarchical, no dendogram

Partitioning Around Medoids

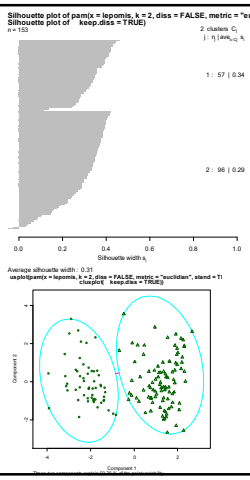


Partitioning Around Medoids

- Code
  - `cluster<-pam(morphology, 6, diss=FALSE, metric="euclidian", stand=TRUE, keep.diss=TRUE)`
  - `summary(cluster)`
  - `plot(cluster)`
- Output
  - No dendrogram
  - Representative objects of each cluster
  - Clustering list (vector assigning each sample to a cluster)
  - Cluster summary statistics
    - Number of members
    - Mean and max dissimilarity within each
  - Silhouette width for each sample – ratio of mean dissimilarity within the same to mean dissimilarity in the next nearest cluster

Plots

- **Silhouette plot** – silhouette numbers plotted in order for each cluster, low or negative numbers indicate questionable clustering.
- **Cluster plot** – actually a principle components analysis with clusters circled.



K-means

- Code
  - `cluster<-kmeans(morphology, 6)`
  - `Cluster`
- Output
  - Cluster means – mean values for all original variables within the two clusters
  - Cluster assignment
  - Within cluster sums of squares
- Similar to pam, but not as flexible because it works only with sums of square differences. Pam will work with any distance metric.

Summary of clustering functions

- **hclust** – various types of agglomerative hierarchical clustering. flashClust package replaces this with a more computationally efficient algorithm.
- **diana** – divisive hierarchical clustering
- **pam** – partitioning around medoids, non-hierarchical, must specify number of groups
- **kmeans** – k means clustering, non-hierarchical, must specify number of groups

Assignment

- Reading
  - Weigelt, P. and W. Jetz. 2013. Bioclimatic and physical characterization of the world's islands. Proceedings of the National Academy of Science. 110: 15307-15312.
- Chapter 4 in text
- Assignment
  - Download the raw data from the Weigelt and Jetz paper (follow datadryad link on the first page of the paper, download the islanddata.csv file).
  - Use the sample function to select 2500 random islands
    - Hint: this code will sample an array from 1-17883 without replacement
      - `sample(seq(1:17883),2500,replace=FALSE)`
  - Follow their methods to produce a UPGMA classification of islands with all variables except Area and Elevation (Fig. S6).
    - The PCA scores needed for this are in the dataset (PCAnEAPC1…)
    - You will need to weight variables by the square root of the eigenvalues from table S3 (B). Each of these values (in the last row) is multiplied with the corresponding PCA axes (e.g. 3.748^0.5*PCAnEAPC1).
    - Bind the resulting weighted variables together with cbind.
    - Use flashClust to do a UPGMA on a Euclidean distance matrix.
    - Use the table function to check your results against theirs (UPGMAnoAE variable in dataset, and vector from cutree() function).
    - Calculate the cophenetic correlation.
    - Use the intCriteria function (Calinski Harbasz metric) to determine the optimal number of clusters.