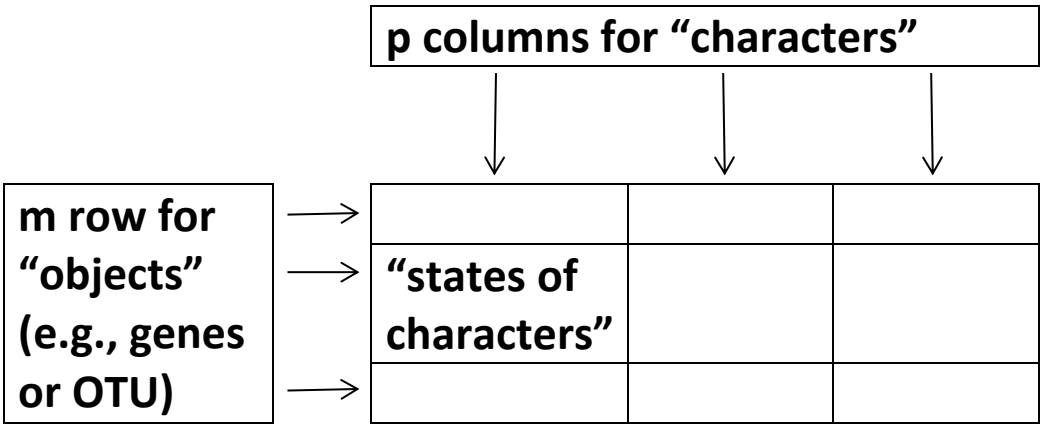
Chapter 10 Class Notes – Similarity, Distance and Clustering

10.1. The Biological Problem: focus on clustering (identifying groups of like objects) and classification (assigning objects into predetermined categories); OTU = operational taxonomic units (in evolutionary studies)



In the next chapter, we'll study spotted microarray & oligo-nucleotide array technologies: "the purpose of clustering in this case is to identify and group together (cluster) genes having similar expression patterns. Similar expression patterns may indicate that the genes participate in similar biological processes or that they respond to similar biological controls." Characters above are like "variables" in statistics; since we measure several characters on the same object, we will need multivariate statistical methods ('distance', etc.)

10.2. Characters: these can be:

- **Qualitative or categorical characters, which differ in type – the color of mice (2 = black, 1 = brown, 0 = white), or nucleotide at a certain position (1 = A, 2 = C, 3 = G, 4 = T); it doesn't make sense to average these**

- **Quantitative (discrete or continuous) characters, which are measured on a numerical scale – number of tail vertebrae in a dromaeosaur skeleton (discrete) or the length of this tail in cm (continuous)**
- **Dichotomous characters, which take one of two states – male/female, present/absent, or purine/pyrimidine. This last turns a categorical into a dichotomous character, and a quantitative becomes dichotomous via "low/high"**

Table 10.1. Comparisons between a selected region of cytochrome oxidase subunit II coding sequences (bp 121–180) for different pairs of primates. Hy: *Hylobates* (gibbon); Pa: *Pan* (chimpanzee); Go: *Gorilla*; Ho: *Homo*; Po: *Pongo* (orangutan).

Hy	GCCCTCTTCCTAACACTCACAACAAAACTAACCAACACTAACATTACGGATGCCCAAGAA
Pa	GCCCTTTTCCTAACACTCACAACAAAACTAACTAATACTAGTATTTCAGACGCCCAGGAA
Go	GCCCTTTTCCTAACACTCACAACAAAGCTAACTAGCACCAACATCTCAGACGCCCAAGAA
Ho	GCCCTTTTCCTAACACTCACAACAAAACTAACTAATACTAACATCTCAGACGCTCAGGAA
Po	GCCCTTTTCCTAACACTCACAACGAAACTCACCAACACTAACATCTCAGATGCCCAAGAG

Nucleotide strings *I* and *J* (such as 'Hy' and 'Pa' in Table 10.1 above: aligned portions of primate cytochrome oxidase subunit II DNA sequences) correspond to 2 objects. Here, 'similarity' can be measured by counting the number or percentage of nucleotide matches (this is done below for each pair). The edit distance, or Levenshtein distance, is the minimum number of indels or substitutions required to transform one string into another. Thus, the "distance" from 'Hy' to 'Pa' is 0.150 and that from 'Pa' to 'Ho' is 0.067. But this measure is not unique since we can also measure distance *at the amino acid (instead of nucleotide) level* – or even the protein level.

```
Hy    GCCCTCTTCCTAACACTCACAACAAAACTAACCAACACTAACATTACGGATGCCCAAGAA
Pa    *****T**************************T**T****GT***T*A**C*****G***
```
(9 differences; fractional difference = 0.150)

```
Hy    GCCCTCTTCCTAACACTCACAACAAAACTAACCAACACTAACATTACGGATGCCCAAGAA
Go    *****T*******************G*****T*G***C*****CT*A**C*********
```
(9 differences; fractional difference = 0.150)

```
Hy    GCCCTCTTCCTAACACTCACAACAAAACTAACCAACACTAACATTACGGATGCCCAAGAA
Ho    *****T**************************T**T********CT*A**C**T**G***
```
(9 differences; fractional difference = 0.150)

```
Pa    GCCCTTTTCCTAACACTCACAACAAAACTAACTAATACTAGTATTTCAGACGCCCAGGAA
Go    ************************G*******GC**C*AC**C***********A***
```
(8 differences; fractional difference = 0.133)

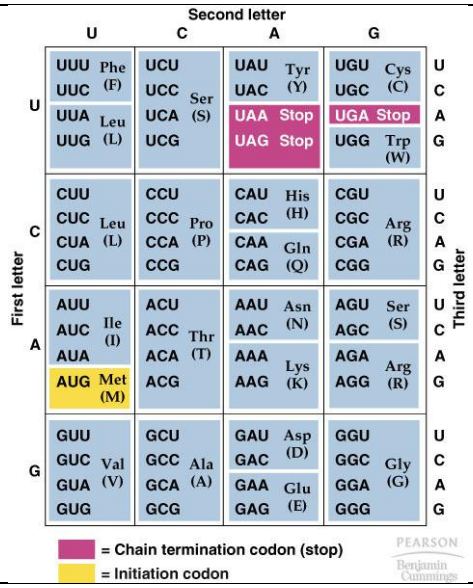
```
Pa    GCCCTTTTCCTAACACTCACAACAAAACTAACTAATACTAGTATTTCAGACGCCCAGGAA
Ho    ****************************************AC**C********T******
```
(4 differences; fractional difference = 0.067)

```
Go    GCCCTTTTCCTAACACTCACAACAAAGCTAACTAGCACCAACATCTCAGACGCCCAAGAA
Ho    ************************A*******AT**T**************T**G***
```
(6 differences; fractional difference = 0.100)

The corresponding amino acids are given below at left.

```
Hy    ALFLTLTTKLTNTNITDAQE
Pa    ALFLTLTTKLTNTSISDAQE
Go    ALFLTLTTKLTSTNISDAQE
Ho    ALFLTLTTKLTNTNISDAQE
Po    ALFLTLTTKLTNTSISDAQE
```

Strings of 20 amino acids corresponding to above 60 nucleotides



In comparing measured distance at the nucleotide (nt) versus amino acid (aa) level, notice that the percentage differences between the different pairs is not the same; comparing 'Hy' and 'Ho", the distance is 0.15 at the nt level but 0.05 at the aa level; selection tends to conserve the sequence of amino acids

Pairwise alignments:

```
Hy    ALFLTLTTKLTNTNITDAQE
Pa    **************S*S****
(2 differences = 0.10)

Hy    ALFLTLTTKLTNTNITDAQE
Go    ***********S***S****
(2 differences = 0.10)

Hy    ALFLTLTTKLTNTNITDAQE
Ho    ***************S****
(1 differences = 0.05)

Pa    ALFLTLTTKLTNTSISDAQE
Go    ***********S*N******
(2 differences = 0.10)

Pa    ALFLTLTTKLTNTSISDAQE
Ho    *************N******
(1 differences = 0.05)

Go    ALFLTLTTKLTSTNISDAQE
Ho    ***********N********
(1 differences = 0.05)
```

But "when dealing with protein sequences, we might elect to count not amino acid differences but the minimum number of base [nt] changes need to convert one residue to another." So, in comparing 'Hy' and 'Pa' above in residue 14 above, Asn (N) is changed to Ser (S), but this represents two nucleotide changes (AAC → AGT). Finally, note that when Asn replaces Ser or Ser replaces Thr, all three are uncharged polar amino acids, so "each of these protein regions has an identical string of physicochemical properties." So, before we can measure "distance," we need to clearly think through exactly how this is to be done.

10.3. Similarity and Distance: these metrics measure how close or distant objects are. Consider the example below at left (where 1 = present and 0 = absent).

OTU	Characters			
	Hair	Lungs	Egg-laying	Milk
Dog	1	1	0	1
Turtle	0	1	1	0
Canary	0	1	1	0
Goldfish	0	0	1	0

	OTU j	
	1	0
OTU i 1	a	b
0	c	d

	canary	
	1	0
goldfish 1	1	0
0	1	2

One measure of similarity between the OTUs is to count the number of matches and mismatches as above at right. Once we have the above table, we can then calculate the simple matching coefficient:

$$s_{ij} = \frac{a + d}{a + b + c + d}$$

In the case of Canary and Goldfish, $s_{34} = \frac{3}{4} = 0.75$. We then put these together into a similarity matrix:

	Dog	Turtle	Canary	Goldfish
Dog	---	---	---	---
Turtle	0.25	---	---	---
Canary	0.25	1.00	---	---
Goldfish	0	0.75	0.75	---

The above notwithstanding, sometimes negative matches convey no additional information about relationships, and an alternate similarity measure is Jaccard's coefficient:

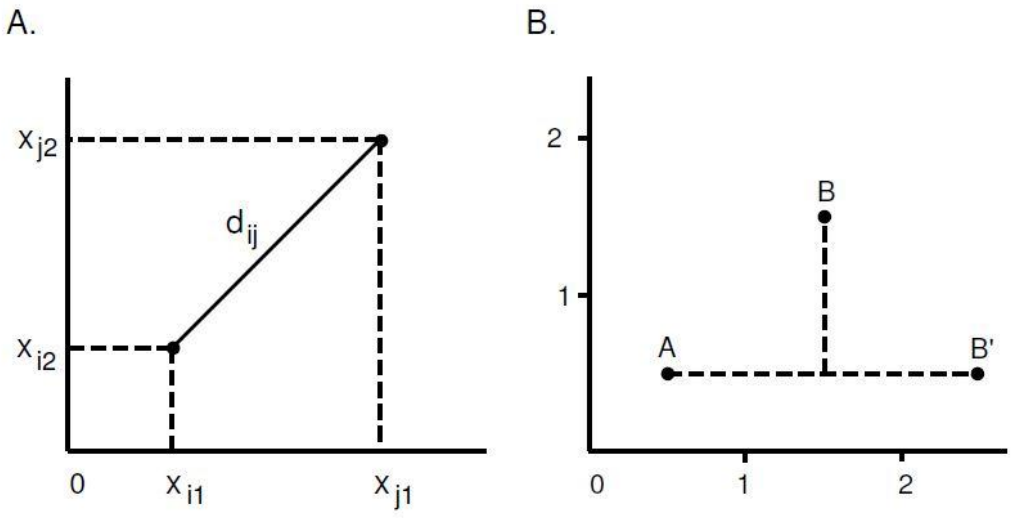
$$s_{ij} = \frac{a}{a + b + c}$$

In this case, dissimilarities are then measured as:

$$d_{ij} = \frac{b + c}{a + b + c} = 1 - s_{ij}$$

Dissimilarity measures or metrics may have properties:

- **Symmetry:** $d_{ij} = d_{ji}$ **for all** i, j
- **Distinguishability:** $d_{ij} \neq 0$ **if and only if** $i \neq j$
- **Triangle Inequality:** $d_{ij} \leq d_{ik} + d_{kj}$ **for all** i, j, k



Above at left is the Euclidean distance in two dimensions:

$$d_{ij} = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2}$$

When there are p dimensions (characters), the above is:

$$d_{ij} = \sqrt{\sum_{k=1}^{p} (x_{ik} - x_{jk})^2}$$

An alternative is the city-block (or Manhattan) metric:

$$d_{ij} = \sum_{k=1}^{p} |x_{ik} - x_{jk}|$$

This latter metric is illustrated above at right; notice the distances from A to B and from A to B' are the same here; in comparing nucleotide sequences, we usually use the city-block metric.

Before we start, in situations such as measuring mRNA expression levels with microarrays (next chapter), we need to scale the coordinate values – they then become dimensionless. Let x_{ik} be the value of character k for OTU i; then the standardized or scaled character values are:

$$x_{ik}^{*} = \frac{x_{ik}}{s_k}, k = 1, 2, \ldots, p$$

Here, s_k is the SD of column k (character k).

The above scaling is for each column. But sometimes (e.g. in the case of microarray experiments & rows are genes), the characters correspond to a time series for which the expression ratios as a function of time are measured. "The actual amplitude of the measurement at any time point may be less important than the pattern of values for all points

taken as a whole. In this case, the scaling should be over all time points (characters) for each gene (object). In other words, the scaling is applied to rows rather than columns."

10.4. Clustering: for hierarchical clustering (considered here – another type, involving optimization, is in §10.5), we can choose from three types -

$$\begin{aligned}
&\text{Single linkage:} && d_{\mathrm{IJ}} = \min\{d_{ij} : i \in \mathrm{I} \text{ and } j \in \mathrm{J}\},\\
&\text{Complete linkage:} && d_{\mathrm{IJ}} = \max\{d_{ij} : i \in \mathrm{I} \text{ and } j \in \mathrm{J}\},\\
&\text{Group average:} && d_{\mathrm{IJ}} = \textstyle\sum_{i\in \mathrm{I}} \sum_{j \in \mathrm{J}} d_{ij}/(N_{\mathrm{I}} N_{\mathrm{J}}),
\end{aligned}$$

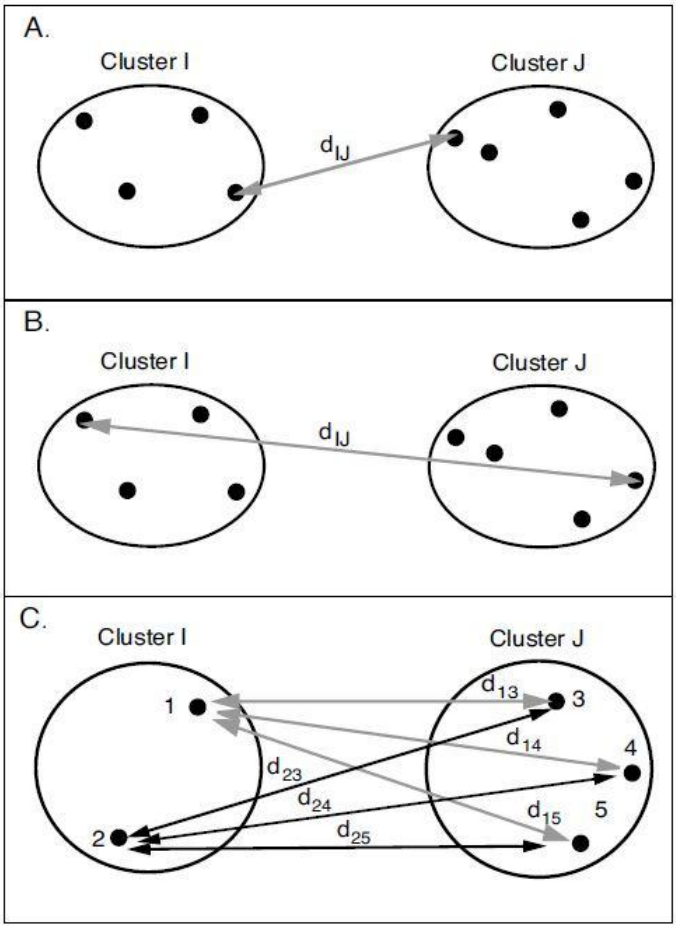


Fig. 10.2. Methods of defining the distance between clusters. Panel A illustrates single linkage, panel B complete linkage, and panel C group average linkage.

Here, we successively join together or split objects into groups based on some measure of distance. Using one of these distance measures, we first produce the $m \times m$ symmetric distance matrix $D_1 = [d_{ij}]$; this is done below at left for the primate data (at the nucleotide level). The second step is to find the OTUs which have the smallest distance, and collapse these two into one group, then determining the new distance matrix D_2; this is where the choice of the distance metric (single, complete, average) comes in. For the primate example, this produces the distance matrix below at right (top), and $d(Go, PaHo) = min\{0.133, 0.10\} = 0.10$. We then repeat the last step to completion; here, we next collapse Go and PaHo into one group, GoPaHo, and find the distance to Hy from D_1 by $min\{0.15, 0.15, 0.15\} = 0.15$.

$D_1 =$

OTU	Hy	Pa	Go	Ho
Hy	0			
Pa	0.150	0		
Go	0.150	0.133	0	
Ho	0.150	0.067	0.100	0

$D_2 =$

OTU	Hy	Go	(PaHo)
Hy	0		
Go	0.150	0	
(PaHo)	0.150	0.100	0

$D_3 =$

OTU	Hy	Go(PaHo)
Hy	0	
Go(PaHo)	0.15	0

This process produces the dendrogram given below (produced using R and the following code).

```
s1<-c(3,2,2,2,4,2,4,4,2,2,4,1,1,2,1,2,4,2,1,2,1,1,2,1,1,1,1,2,4,
1,1,2,2,1,1,2,1,2,4,1,1,2,1,4,4,1,2,3,3,1,4,3,2,2,2,1,1,3,1,1)
s2<-c(3,2,2,2,4,4,4,4,2,2,4,1,1,2,1,2,4,2,1,2,1,1,2,1,1,1,1,2,4,
1,1,2,4,1,1,4,1,2,4,1,3,4,1,4,4,4,2,1,3,1,2,3,2,2,2,1,3,3,1,1)
```

```
s3<-c(3,2,2,2,4,4,4,4,2,2,4,1,1,2,1,2,4,2,1,2,1,1,2,1,1,1,3,2,4,
1,1,2,4,1,3,2,1,2,2,1,1,2,1,4,2,4,2,1,3,1,2,3,2,2,2,1,1,3,1,1)
s4<-c(3,2,2,2,4,4,4,4,2,2,4,1,1,2,1,2,4,2,1,2,1,1,2,1,1,1,1,2,4,
1,1,2,4,1,1,4,1,2,4,1,1,2,1,4,2,4,2,1,3,1,2,3,2,4,2,1,3,3,1,1)
s5<-c(3,2,2,2,4,4,4,4,2,2,4,1,1,2,1,2,4,2,1,2,1,1,2,3,1,1,1,2,4,
2,1,2,2,1,1,2,1,2,4,1,1,2,1,4,2,4,2,1,3,1,4,3,2,2,2,1,1,3,1,3)
seqs<-rbind(s1,s2,s3,s4,s5)

seqdist<-function(x,n)
{
dmat<-matrix(nrow=n,ncol=n)
for(i in 1:n){
for (j in 1:n){
dmat[i,j]<-length(x[j,][x[j,]!=x[i,]])/length(x[1,])}}
return(dmat)
}
dapes<-seqdist(seqs,4)
dapes
          [,1]      [,2]      [,3]      [,4]
[1,] 0.000000 0.150000 0.150000 0.150000
[2,] 0.150000 0.000000 0.133333 0.066667
[3,] 0.150000 0.133333 0.000000 0.100000
[4,] 0.150000 0.066667 0.100000 0.000000

dapes1<-as.dist(dapes,diag=F,upper=F)
species=c("Hy","Pa","Go","Ho")
plclust(hclust(dapes1,"single"),labels=species,xlab="",sub="")
```

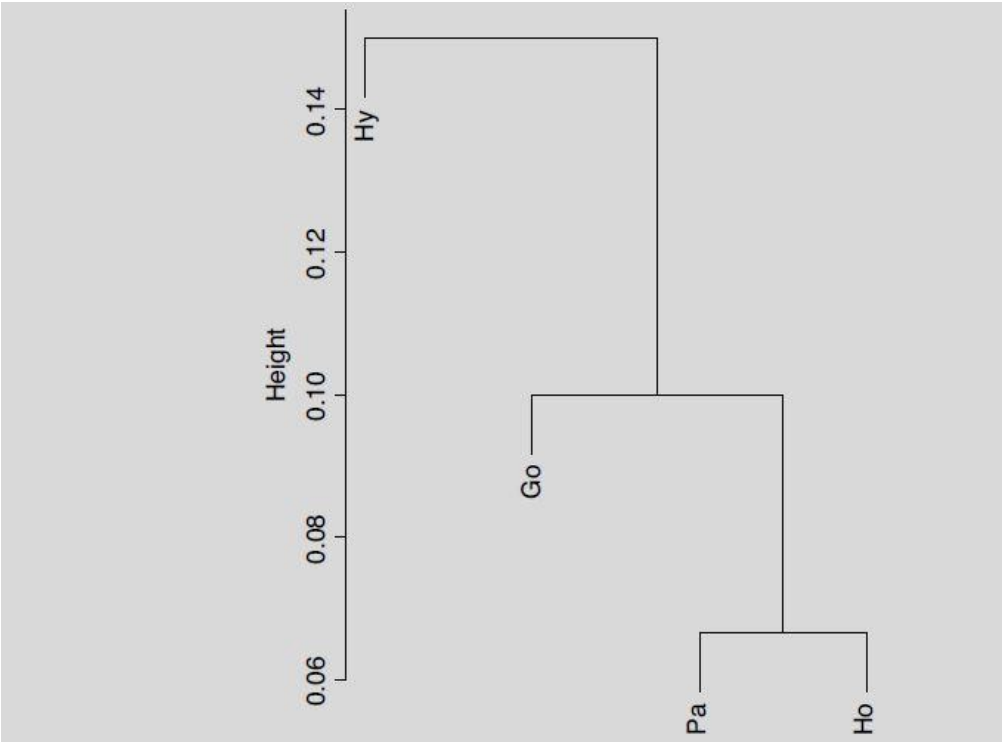
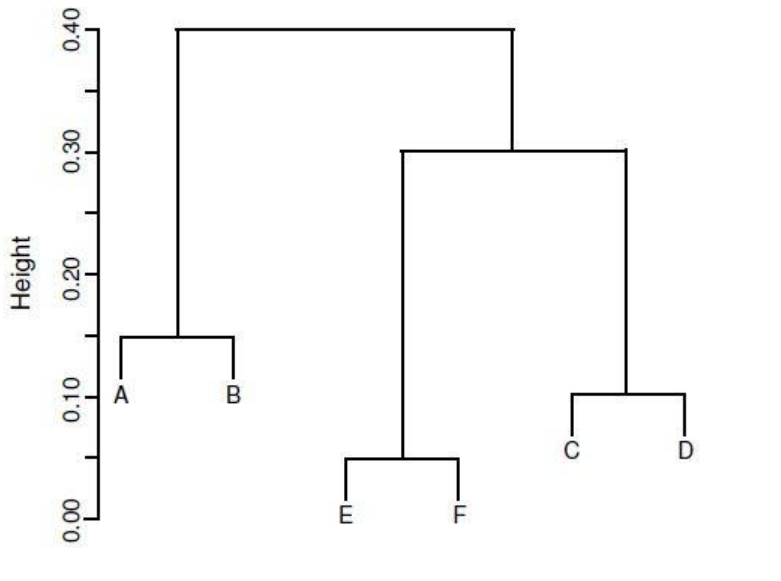


Fig. 10.3. Dendrogram of single-linkage clustering for primate species.

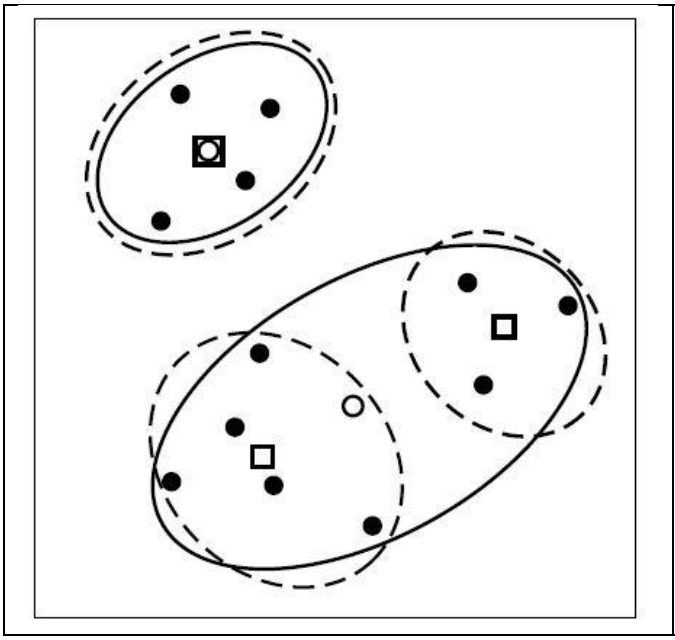
Now work through to find the dendrogram here using the complete linkage measure. In practice, we'll have much more data, and need to think through the number of clusters.

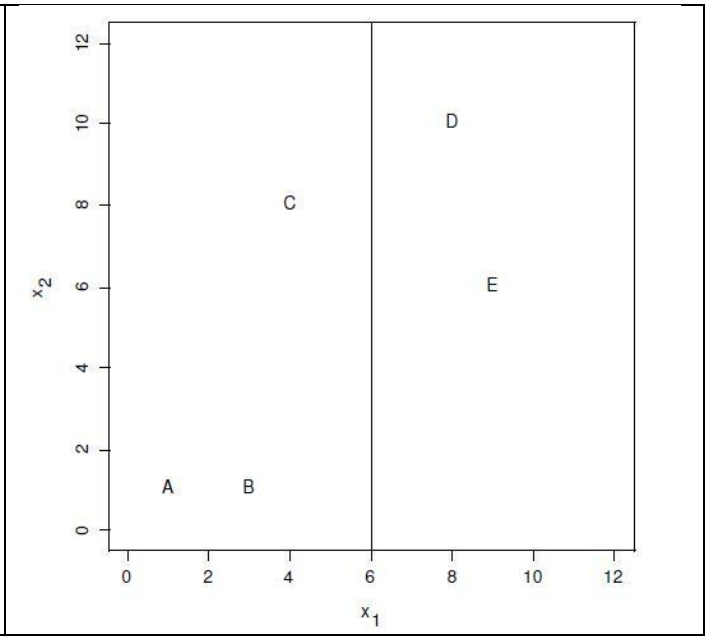
In looking at the dendrogram at right, most of us would identify three clusters: AB, CD, and EF. If we set the criterion at distances between 0.4 and 0.3, we would define two clusters. So, the number of clusters is therefore somewhat arbitrary.	

In producing dendrograms, here are some key considerations:

- **The choice of distance measure is important**
- **Errors can occur and can be carried forward**
- **No one inter-cluster distance measure is "best" – in head-to-head tests, single-linkage was least successful and average-linkage did well**
- **Important issues include: robustness of the grouping method (i.e., the method works well under a variety of input data and initial parameters), clustering methods need to be stable (p.278); we could use "jackknife" methods as well (leaving each of the m OTUs out in turn)**
- **"Building phylogenetic trees is procedurally similar to clustering but with biological models included."**

10.5. K-means: the # of groups/clusters (k) is prespecified. The relevant distance here is the distance of each OTU to the cluster center or centroid. Thus, OTUs are allocated into the k clusters such that the within-cluster sums of squares of distances from cluster centroids (within-ss), summed over all clusters, is minimized. Within cluster j, within ss = $\sum_{i=1}^{m_j} d_{ij}$, and total within-ss is $S = \sum_{j=1}^{k} \sum_{i=1}^{m_j} d_{ij}$. This is illustrated below at left for $k = 2$ (centroids are circles) and $k = 3$ (centroids are squares); increasing k from 2 to 3 splits the larger cluster at the bottom into two cluster (but with the top cluster unchanged), and the value of S for $k = 3$ will be much less than for $k = 2$. The K-means calculation involves trying different centroid positions and iteratively testing each OTU for its distance to one of the centroids.





The K-means steps are:

1. **Arbitrarily partition the OTUs into k clusters**
2. **Calculate the centroid of each cluster**
3. **Assign/reassign each OTU to that cluster whose centroid is the closest (using Euclidean distance)**
4. **Recalculate the new centroids**
5. **Repeat steps 3 & 4 until no change in memberships**

A simple example is plotted above at right. Suppose we start by putting A, B, C into cluster I and D, E into cluster II; then, the respective centroids are $(2.67, 3.33)$ and $(8.5, 8.0)$, and we get distances:

$d(A, I) = \mathbf{2.87}$	$d(A, II) = 10.26$
$d(B, I) = \mathbf{2.35}$	$d(B, II) = 8.90$
$d(C, I) = 4.86$	$d(C, II) = \mathbf{4.50}$
$d(D, I) = 8.54$	$d(D, II) = \mathbf{2.06}$
$d(E, I) = 6.87$	$d(E, II) = \mathbf{2.06}$

We then reassign C to cluster II, recalculate the new centroids and above distances, and find that this is optimal: the clusters are then {A,B} and {C,D,E}: new centroids are $I' = (2.0, 1.0)$ and $II' = (7.0, 8.0)$, and the distances are now:

$d(A, I') = 1.00$	$d(A, II') = 9.22$
$d(B, I') = 1.00$	$d(B, II') = 8.06$
$d(C, I') = 7.28$	$d(C, II') = 3.00$
$d(D, I') = 10.82$	$d(D, II') = 2.24$
$d(E, I') = 8.60$	$d(E, II') = 2.83$

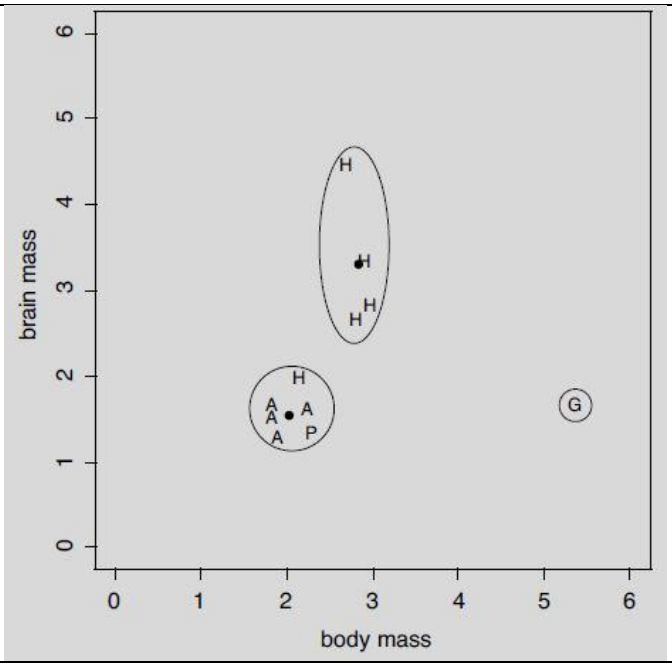
The second example involves living and fossil hominoid species and using characters brain mass and body mass; notice in the data below that the scales for body mass (kg) and brain mass (g) are very different – hence we need to scale the data first:

```
body.mass<-c(53,57,55,58,42,36,44,36,37,45,105)
brain.mass<-c(1355,1016,804,854,597,502,488,457,384,395,505)
raw.dat<-cbind(body.mass,brain.mass)
scaled.dat<-raw.dat
scaled.dat[,1]<-raw.dat[,1]/sqrt(var(raw.dat[,1]))
scaled.dat[,2]<-raw.dat[,2]/sqrt(var(raw.dat[,2]))
species<-c("H.sapiens","H.erectusL","H.erectusE","H.ergaster",
"H.habilis","A.robustus","A.boisei","A.africanus","A.afarensis",
"P.troglodytes","G.gorilla")
body<-scaled.dat[,1]
brain<-scaled.dat[,2]
h<-mean(c(4.425578,1.254186))
```

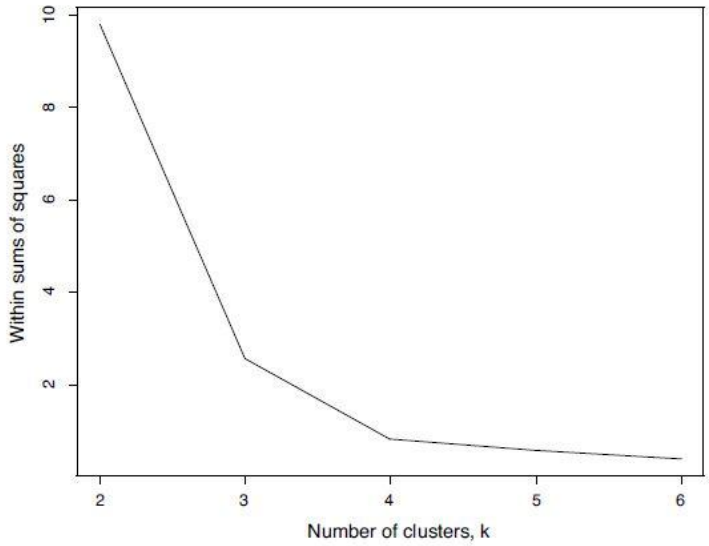
```
x1<-x2<-mean(body)
y1<-mean(brain[brain<h])
y2<-mean(brain[brain>h])
in.cent<-cbind(c(x1,x2),c(y1,y2))
in.cent
         [,1]         [,2]
[1,] 2.638996   1.809424
[2,] 2.638996   3.871972

k.dat2<-kmeans(scaled.dat,in.cent,iter.max=100)
k.dat2

K-means clustering with 2 clusters of sizes 6, 5

Cluster means:
  body.mass brain.mass
1  2.044293   1.536704
2  3.352640   2.961708

Clustering vector:
 [1] 2 2 2 2 1 1 1 1 1 1 2

Within cluster sum of squares by cluster:
[1]   0.5517516   9.2416832
 (between_SS / total_SS =  51.0 %)
```

```
k.dat3<-kmeans(scaled.dat,3,iter.max=100)
k.dat3

K-means clustering with 3 clusters of sizes 4, 6, 1

Cluster means:
  body.mass brain.mass
1  2.849233   3.289788
2  2.044293   1.536704
3  5.366268   1.649385

Clustering vector:
 [1] 1 1 1 1 2 2 2 2 2 2 3

Within cluster sum of squares by cluster:
[1]   2.0205711   0.5517516   0.0000000
 (between_SS / total_SS =  87.1 %)
```

For the $k = 2$ case, we provided a 2×2 matrix identifying initial cluster centers, $(2.64, 1.81)$ and $(2.64, 3.87)$. Here, since G.gorilla is combined with the first four $H.$ members, $S = 9.79$. For the $k = 3$ case, we see three distinct groups (see graph at right) and $S = 2.57$.



An important concern is how many clusters should there be. The number is somewhat arbitrary (so beware), but the plot below helps: note that the "elbow" occurs at $k = 3$, so most researchers would choose this value (i.e. choose $k = 3$).



10.6. Classification: the current chapter extends what we did in the last chapter, where we classified according to a score (derived from PWM/PSSM matrices or from Markov transition matrices), and determined a cut-off taking account of sensitivity and specificity.

Now, we consider an OTU not used in the classification rule, and we wish to determine to which cluster it should be assigned. For K-means, this is simple: find the Euclidean distance from this new OTU to the centroids of the k clusters, and add it to the nearest cluster. In the hierarchical case, we add the new OTU into the mix and redo the clustering. It is then telling both if the clusters remain unchanged (easy case) and when they do change (clustering may not be robust).