Distance methods.

Optimality criteria based on distances

General idea:

1) Calculate a measure of a distance between each pair in a group of observations (sequences)
2) Find a tree that predicts the observed set of distances as closely as possible

Note: we leave out all information from higher-order combinations of character states, reducing the data matrix to a simple table of pairwise distances.

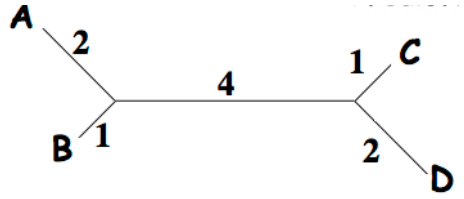
We can consider distances as estimates of the branch length separating a pair of species. Then each distance represents the best unrooted tree for that pair and we need to find the full n-species tree that does the best job of approximating these individual two-species trees.

Note: branch lengths are not simply a function of time; they reflect amounts of evolution in different branches of the tree (b=r*t), where *b* is branch length, *r* is rate of evolution and *t* is time.

Let's say we have an alignment and we've counted the number of differences between individual sequences (observed set of distances $\boldsymbol{D_{ij}}$) and presented them as a matrix (the matrix is symmetrical so I've skipped the lower half):

	A	B	C	D
A	-	3	7	8
B	-	-	6	7
C	-	-	-	3
D	-	-	-	-

Can we draw a tree that would predict this set of distances? We can:



We can calculate a matrix of predicted distances ($\boldsymbol{d_{ij}}$) from this tree, which will be equal to the matrix of observed distances (but only because I made up this example). In most cases predicted distances (from a tree you made) won't match exactly the observed

pairwise distances you calculated from the data so we need to calculate a measure of disparity between them.

The sum of squares is a nice statistical measure of such a disparity:

$Q = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}(D_{ij} - d_{ij})^2$, where w_{ij} are optional weights (can be 1, 1/D, etc).

Now we have re-adjust all the branches on the tree to find a tree that predicts our observed distances as closely as possible.

The second step is the same as in parsimony: we need to search in the tree space for the best tree. But what optimality criterion should we use? One possibility is to use the same optimality criterion as for tree optimization: the least sum of squares. This approach is known as The Least Squares method, a popular version of which was introduced by *Fitch and Margoliash* in 1967. Another possibility is to choose the tree that has the shortest total lengths. This approach is known as the *Minimum Evolution* method and has been introduced by Rzhetsky and Nei in 1992.

Clustering algorithms:
Distance data can also be used in clustering algorithms:

- We start with a distance matrix
- Cluster the least different pair of sequences
- Repeat until all notes are linked
- *Voilà*, we have our tree

Note, however, that there is no measure of tree-goodness. It may be a good tree; it may be a poor tree.

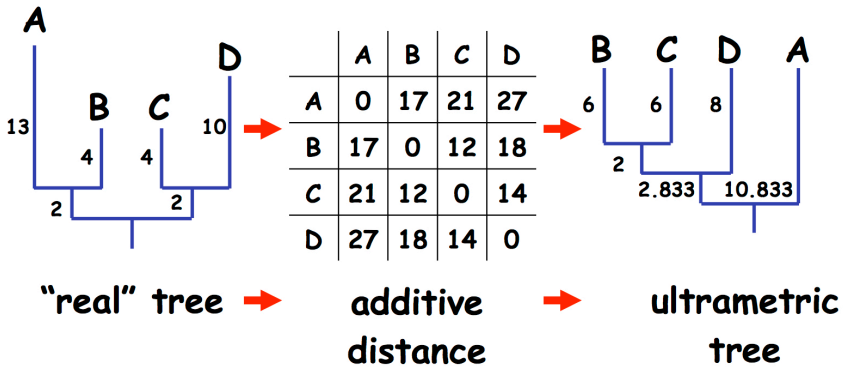
UPGMA = Unweighted Pair Group Method with Arithmetic means
We used the following algorithm to build an UPGMA tree:

1. Find the *i* and *j* that have the smallest distance, D_{ij}.
2. Create a new group, (*ij*), which has $n_{(ij)} = n_i + n_j$ members.
3. Connect *i* and *j* on the tree to a new node [which corresponds to the new group (*ij*)]. Give the two branches connecting *i* to (*ij*) and *j* to (*ij*) each length $D_{ij}/2$.
4. Compute the distance between the new group and all the other groups (except for *i* and *j*) by using:

$$D_{(ij),k} = \left(\frac{n_i}{n_i + n_j}\right) D_{ik} + \left(\frac{n_j}{n_i + n_j}\right) D_{jk}$$

5. Delete the columns and rows of the data matrix that correspond to groups *i* and *j*, and add a column and row for group (*ij*).
6. If there is only one item in the data matrix, stop. Otherwise, return to step 1.

The problem:



What's the problem?

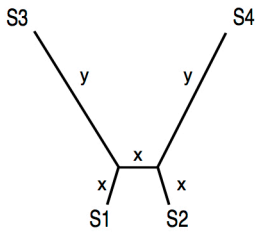


Figure 5.9: The true relationships of taxa S1, S2, S3, S4

	S1	S2	S3	S4
S1		$3x$	$x+y$	$2x+y$
S2			$2x+y$	$x+y$
S3				$x+2y$

Table 5.8: Distances between taxa in Figure 5.9

The essential problem here is a conflict between (dis)similarity of taxa and their closeness in the graph theoretic or evolutionary sense.

The solution: NJ = Neighbor Joining Algorithm

First need to define neighbors or a *cherry* on a tree: two leaves that are separated by one internal node (or that are graph-theoretic distance 2 apart).

Note, that for the two cherries above, $d_{13} + d_{24} < d_{12}+d_{34} = d_{14}+d_{23}$

This leads us to the following theorem: A metric quartet tree relating taxa *a, b, c, d* which as positive edge lengths has cherries a,b and c,d if, and only if, any (and hence all) of the three inequalities/equalities in the tree matrix that follow hold:

$d(a,b) + d(c,d) < d(a,c) + d(b,d) = d(a,d) + d(b,c)$

… and to the definition of the *four-point condition*:
A dissimilarity map δ on X satisfies the four-point condition if for every choice of four taxa

a, b, c, d ∈ X, δ(a,b)+ δ(c,d) ≤ max{δ(a,c)+ δ(b,d), δ(a,d) + δ(b,c)}

Now, we will use this idea for the NJ algorithm, but because our tree usually has more than 4 taxa, we will calculate the average-like distance for each leaf of the tree:

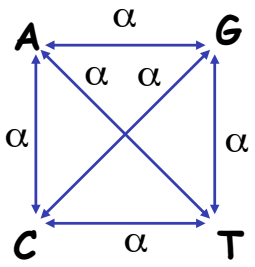
1. For each tip, compute $u_i = \sum_{j:j\neq i}^{n} d_{ij} /(N-2)$. Note, that this is similar to the average distance, but that the denominator is (deliberately) not the number of items summed (see the textbook for the explanation).
2. Compute the quantity $M_{ij} = d_{ij} - u_i - u_j$ for each pair of taxa *i*, *j*, which is like a measure of "neighborliness". A small M_{ij} indicates *i* and *j* are relatively close to each other and relatively distant from all other taxa.
3. Identify the pair of taxa *i*, *j* with the smallest M_{ij}
4. Join taxa *i* and *j.* Compute the branch length from *i* to the new node (v_i) and from j to the new node (v_j) as:
$v_i = \frac{1}{2}d_{ij} + \frac{1}{2}(u_i - u_j)$
$v_j = \frac{1}{2}d_{ij} + \frac{1}{2}(u_j - u_i)$
5. Compute the distance between the new node (*ij*) and each of the remaining tips as
$d_{(ij),k} = (d_{ik} + d_{jk} - d_{ij})/2$
6. Delete tips *i* and *j* from the tables and replace them by the new node, (*ij*), which is now treated as a tip.
7. If more than two nodes remain, go back to step 1. Otherwise, connect the two remaining nodes (say, *l* and *m*) by a branch of length d_{lm} .

Substitution models (we'll talk about them later!).
The use of distance methods is based on two main assumptions:

- Each distance is measured independently from the others (usually violated)
- Distances are proportional to the to the total branch length between the species (distances are additive).

The simple approach to measure the divergence between two strands of aligned DNA sequences is to count the number of differences between them. The proportion of sites where two sequences differ is called *observed distance*, or *p-distance.* Although *p-distances* are an intuitive measure of divergence, they are not additive and so violate the second assumption of distance methods.

To estimate the "true" evolutionary distance between two sequences we need to calculate expected distances based on observed distances and a substitution model. The simplest substitution model was introduced by Jukes and Cantor in 1969 and includes only one parameter: the rate of substitution, which is assumed to be the same for any pair of nucleotides.



Based on this simple model, we can calculate the probabilities that a particular nucleotide will remain the same at a given position over time t or change to a different nucleotide:

$P_{ii}(t) = 1/4 + (3/4)e^{-4\alpha t}$

$P_{ij}(t) = 1/4 - (1/4)e^{-4\alpha t}$

We can use these equations to calculate the expected number of matches and mismatches between two sequences after time *t*:

$I_{(t)}=P^2_{AA(t)} + P^2_{AT(t)} + P^2_{AC(t)} + P^2_{AG(t)}$

$D_{(t)}=p=1- I_{(t)}=3/4(1-e^{-8\alpha t})$ or $8\alpha t = -\ln(1-4/3p)$

Why do we want to do it if we can simply count the number of mismatches? We can use this formula to estimate αt from *p.* And if we know αt, we can use it to calculate the expected genetic distance (d) between the two sequences: $d=2(3\alpha t)$, where $3\alpha t$ is the number of substitutions per site in a single lineage. Hence, we get a formula that estimates *d* from *p.*

$d=-3/4 \ln(1-4/3p)$

We can also calculate the variance as $V(d) = \frac{p-p^2}{L\left(1-\frac{4}{3}p\right)^2}$ (where L is the length of sequences)