Lecture 12 - Phylogenetic trees

Gidon Rosalki

2025-11-30

1 Phylogenetic trees

Phylogenetic trees are graphical representations that show the evolutionary between a set of species or taxa over time. Generally in our case, they will be downward growing binary trees. We can do this since mathematically, if we allow very short branches, then any tree may be described by a binary tree. It should also be noted that there are also *non rooted tree*, which may also be considered to be *undirected* (commentary, the book does not refer to undirected trees, nor does it make particular logical sense. However, Nir did, so know this we must). In unrooted trees, the time direction is undetermined, and we know not the root node. In the rooted version, we will say that the parent node of 2 other nodes is the *last common ancestor*.

Building these trees is our objective. Given some sequences we can (using knowledge gained from the beginning of the course) create a matrix of distances between all the sequences. From there, we can use the UPGMA (Unweighted Pair Group Method using arithmetic Averages) algorithm to create the trees:
First we will define the distance d_{ij} to be the distance between two clusters C_i, C_j as follows:

$$d_{ij} = \frac{1}{|C_i|\,|C_j|} \sum_{p \in C_i,\ q \in C_j} d_{pq}$$

UPGMA 1
Distances matrix *input*
Tree *output*
1: **Initialisation:**
2: Assign each sequence i to a cluster C_i
3: Define one leaf T for each sequence, and place at height 0
4: **Iteration:**
5: Determine the two clusters i, j for which d_{ij} is minimal. If > 2 of equal weight, choose randomly
6: Define a new cluster k s.t. $C_k = C_i \cup C_j$
7: Define a node k, with daughters i, j, and place at height $\frac{d_{ij}}{2}$
8: Add k to the current clusters, and remove i, and j
9: **Termination:**
10: When only two clusters, i, j remain, place the root at height $\frac{d_{ij}}{2}$

However, UPGMA has its issues. If the nodes that are closest in length, are not those that share a common ancestor, then they will be misrepresented by UPGMA:

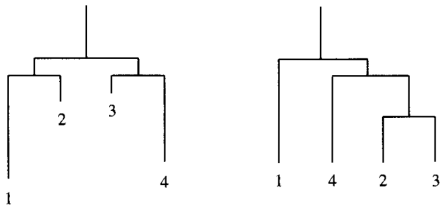

Figure 1: A tree (left) that is reconstructed incorrectly by UPGMA (right)

1.1 Neighbour joining

In describing the molecular clock property, we have implicitly assumed additivity. The edge lengths of a tree are said to be *additive* if the distance between any pair of leaves is the sum of the lengths of of the edges of the path connecting them. It is possible to for the molecular clock property to fail, but additivity to hold.
Given a tree T, with additive lengths $\{d\}$, we may try to reconstruct it from the pairwise distances of its leaves $\{d_{ij}\}$ as follows: Find a pair of *neighbouring leaves*, i.e. leaves that have the same parent node k. Suppose that their numbers

are i, j. Remove them from the list of leaf nodes, and add k to the current list of ndoes, defining its distance to leaf m by

$$d_{km} = \frac{1}{2}\left(d_{im} + d_{jm} - d_{ij}\right)$$

By additivity, the distances d_{km} are precisely those between the equivalent nodes in the original tree. In this way, we can strip away leaves, reducing the number by 1 at each operation, until we get down to a pair of leaves. If we could determine from distances alone a pair of neighbouring leaves, therefore, we could reconstruct a tree with additive length exactly. To avoid the problem that UPGMA encountered, of incorrectly reconstructing nodes based off the distances, we will first subtract the average distances to all other leaves, thus compensating for long edges. We will define

$$D_{ij} = d_{ij} - (r_i + r_j) \tag{1}$$

$$r_i = \frac{1}{|L| - 2}\sum_{k \in L} d_{ik} \tag{2}$$

Where $|L|$ is the size of the set of leaves L. We claim that leaves i, j, for which D_{ij} is minimal will be neighbouring leaves. Thus, we may construct the following algorithm for neighbour joining:

Neighbour-joining 2

1: **Initialisation:**
2: Define T to be the set of leaf nodes, one for each given sequence, and assign $L = T$
3: **Iteration:**
4: Pick a pair i, j in L for which D_{ij} is minimal
5: Define a new node k, and set $\forall m \in L\ d_{km} = \frac{1}{2}\left(d_{im} + d_{jm} - d_{ij}\right)$
6: Add k to T with edges of lengths $d_{ik} = \frac{1}{2}\left(d_{ij} + r_i - r_j\right),\ d_j k = d_{ij} - d_{ik}$ joining k to i, j respectively
7: Remove i, j from L, and add k
8: **Termination:**
9: When L consists of two leaves, i, j add the remaining edge between i, and j, with length d_{ij}