

Phylogeny

MRCA Most Recent Common Ancestor (Root of Tree)

Internal Nodes Ancient hypothetical species

Leaf Nodes Extant species

Neighbour Two species that are most closely related on tree

Clade Group of organisms with common ancestor

Outgroup Set of organisms that are not in the “ingroup”, and are distant in the tree.

Homolog Descendant of common ancestor

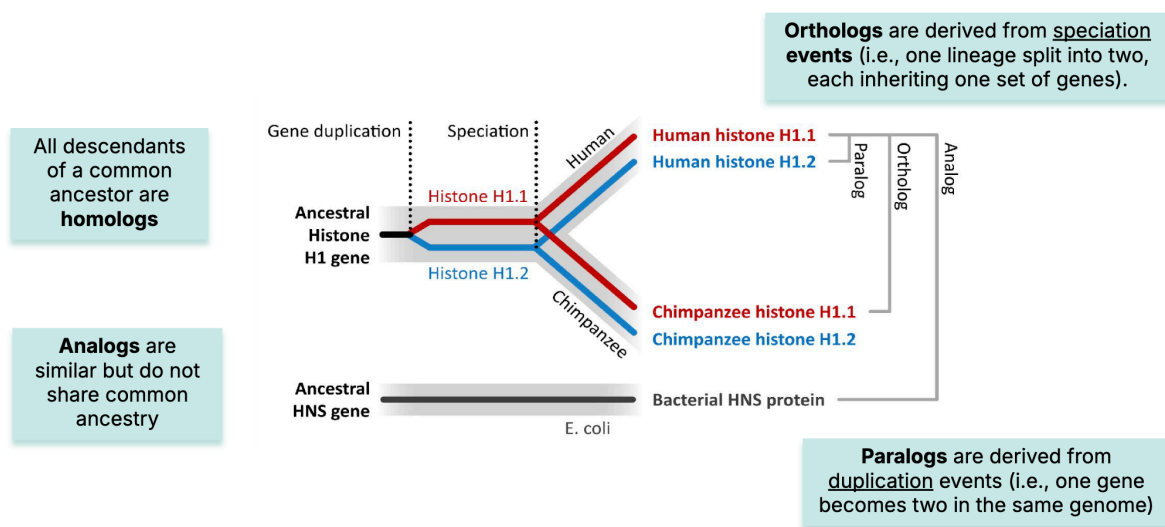
Analog Similar but don't share ancestry

Ortholog Speciation event, where common ancestors break into two species

Paralog One gene breaks into two separate versions in same genome

Phylogeny Construction of evolutionary tree by evolutionary relationship

Morphology Construction of evolutionary tree by physical features

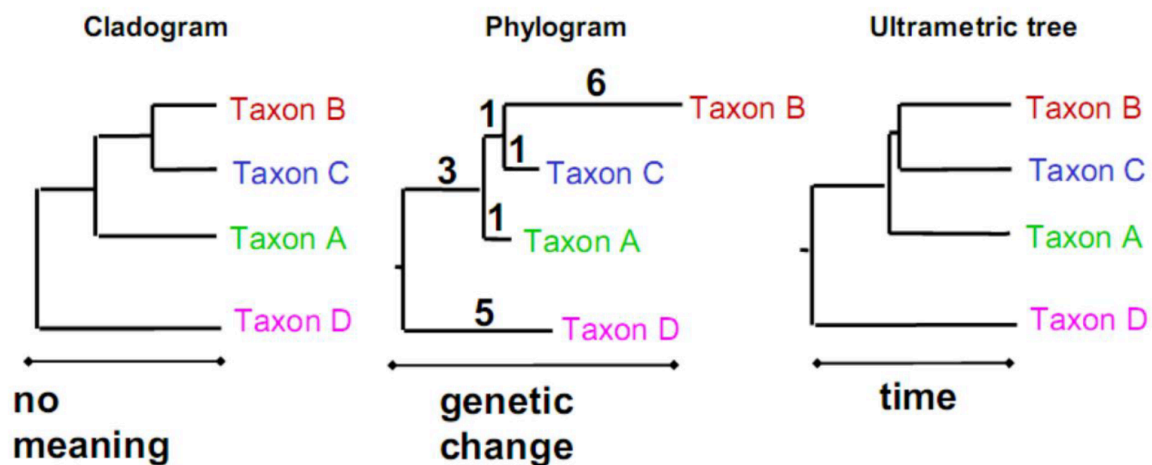


3 problems: **Topology**, **Root**, **Branch Lengths**

Cladogram Tree where branch length has no meaning

Phylogram Tree where branch length conveys genetic change

Ultrametric tree Tree where branch length conveys time



Hierarchical Clustering Arrange sequences based on pairwise distance in hierarchical manner

Well-behaved tree All nodes have equal distance from root to leaves (ultrametric), the distance is defined as “time”. Also, the three-point condition is met, where for all distances, x, y, z ,
 $d(x, y) \leq \max(d(x, z), d(z, y))$

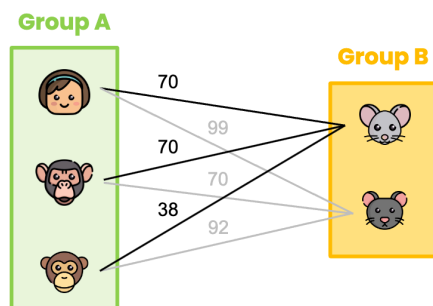
Distance-based methods

UPGMA

UPGMA Unweighted Pair Group Method with Arithmetic Mean

Linkage Criterion Distance metric between two groups (not just sequences, but 2 separate groups of sequences)














UPGMA (unweighted pair group method with arithmetic mean)

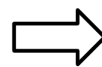













UPGMA: The average pairwise distance

$$\text{UPGMA} = \frac{\text{Sum of pairwise distances}}{\text{Number of pairs}} = (70 + 70 + 38 + 99 + 70 + 92) / 6 = 73$$

	0	14	25	73	70	99	83
	14	0	25	73	70	70	83
	25	25	0	73	38	92	83
	73	73	73	0	83	92	92
	70	70	38	83	0	25	99
	99	70	92	92	25	0	92
	83	83	83	92	99	92	0

							
	0	25	73	70	85	83	
	25	0	73	38	92	83	
	73	73	0	83	92	92	
	70	38	83	0	25	99	
	85	92	92	25	0	92	
	83	83	92	99	92	0	



						
	0	73	54	88	83	
	73	0	83	92	92	
	54	83	0	25	99	
	88	92	25	0	92	
	83	92	99	92	0	

Note that distance from node to leaf is half of distance value in the matrix.

Given distance matrix, find closest pair of species, then combine them to a “new node” and recalculate the values for it and the remaining sequences by $\text{UPGMA} = \frac{\text{Sum of pairwise distances}}{\text{Number of pairs}}$ and continue until you’re done.

This best describes topology and distance of species.

Ultrametric trees are represented by symmetric 0-diagonal matrices, where indices represent divergence events. Given ultrametric matrix, an ultrametric tree for that matrix exists if:

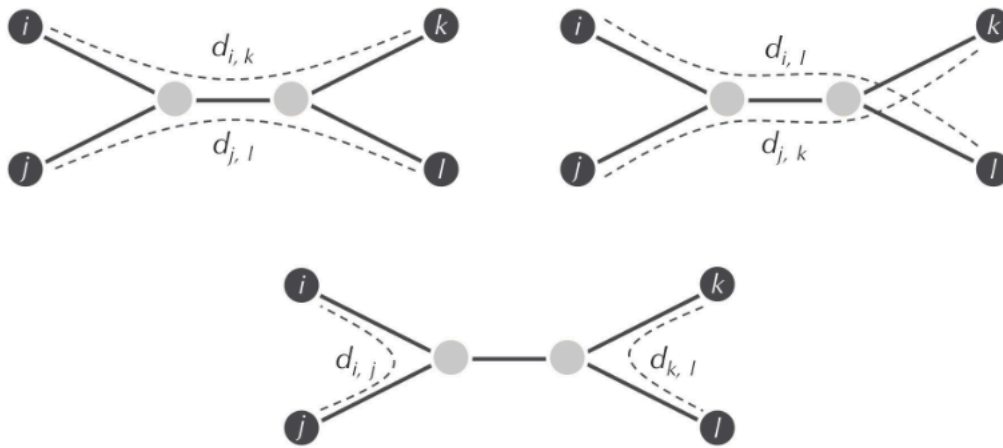
- There are n leaves, one for each row and column of matrix
- Each internal node is labeled by a time in D and has exactly two-children
- Along any path from root to a leaf, the divergence times at the internal nodes strictly decrease.
- For any triplet of nodes, the distances $D_{i,j}$, $D_{j,k}$, $D_{i,k}$ are either equal, or two are equal and the remaining one is smaller. (By 4-point theorem)

Neighbour Joining

Additive Matrix There exists an additive tree fitting the matrix

Additive Tree Given symmetric $n \times n$ 0-diagonal matrix, for any neighbouring leaves i, j , with parent m , the distance from m to any other leaf k is $d_{k,m} = \frac{d_{i,k} + d_{j,k} - d_{i,j}}{2}$

Four-Point Theorem Distance matrix is additive iff two of the sums are equal and third is less than or equal to other sums:



Procedure

To avoid the previous limitations, the trick is to subtract the averaged distances to all other leaves to compensate for long edges. We can define a new matrix such that:

$$D_{ij} = d_{ij} - (r_i + r_j), \quad r_i = \frac{1}{|L| - 2} \sum_{k \in L} d_{ik},$$

Algorithm: Neighbour-joining

Initialisation:

Define T to be the set of leaf nodes, one for each given sequence, and put $L = T$.

Iteration:

Pick a pair i, j in L for which D_{ij} , defined by (7.4), is minimal.

Define a new node k and set $d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$, for all m in L .

Add k to T with edges of lengths $d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$, $d_{jk} = d_{ij} - d_{ik}$, joining k to i and j , respectively.

Remove i and j from L and add k .

Termination:

When L consists of two leaves i and j add the remaining edge between i and j , with length d_{ij} .

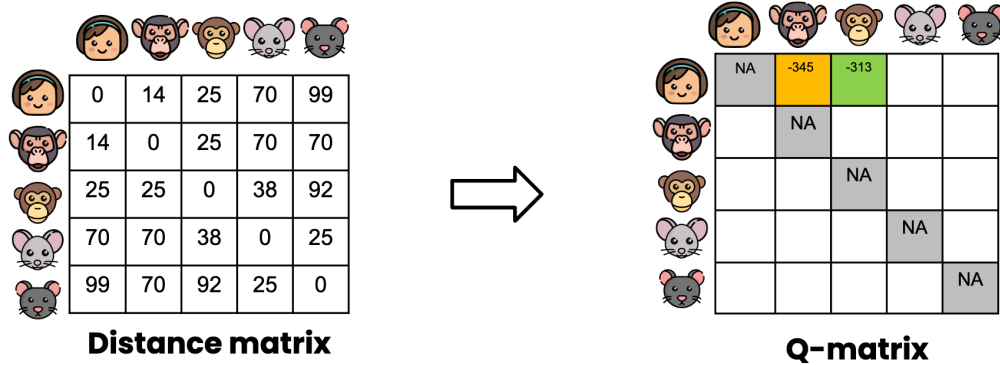
Procedure:

1. Create auxiliary Q-matrix where $Q(i, j) = (n - 2)d_{i,j} - \sum_{k \in L} d_{i,k} - \sum_{k \in L} d_{j,k}$.
2. Merge closest pair wrt Q-matrix (lowest number in matrix)
3. Estimate branch length between chosen node and new node. $l(i, u) = \frac{1}{2}d(i, j) + \frac{1}{2(n-2)} \left[\sum_{k=1}^n d(i, k) - \sum_{k=1}^n d(j, k) \right]$. Note that $l(i, u) + l(j, u) = d(i, j)$. Sometimes these

branch lengths can be negative, so we can add a fixed constant to all branch lengths for non-negativity.

4. Update unrooted tree with new parent node between neighbours, and have respective lengths between them.
5. Update distance matrix so $d(u, k) = \frac{1}{2}[d(i, k) + d(j, k) - d(i, j)]$

Step 1: Build a Q-matrix to choose the closest pair to merge

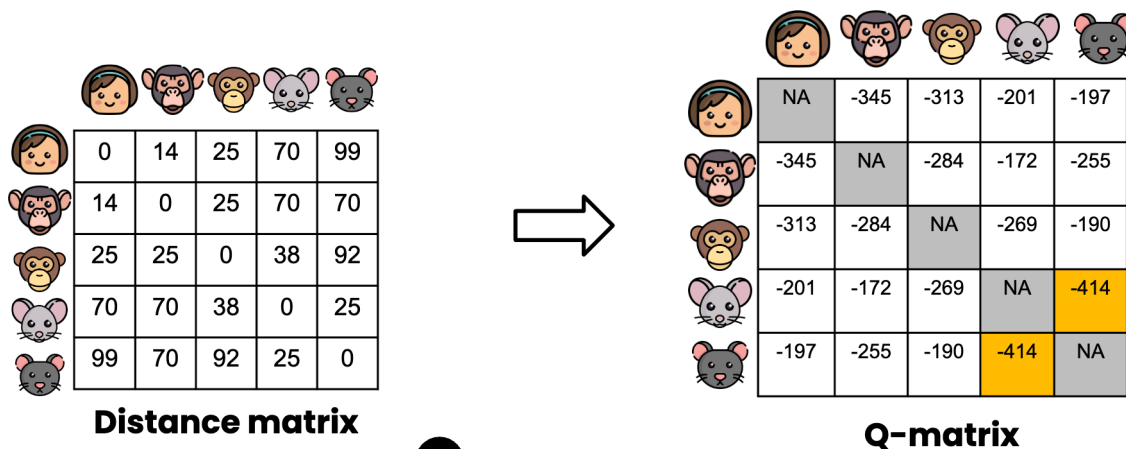


- The Q-metric considers the distance relative to other species

$$Q(i, j) = (n - 2)d(i, j) - \sum_{k=1}^n d(i, k) - \sum_{k=1}^n d(j, k)$$

$$= (5 - 2) \times 14 - (14 + 25 + 70 + 99) - (14 + 25 + 70 + 70) = -345$$

$$= (5 - 2) \times 25 - (14 + 25 + 70 + 99) - (25 + 25 + 38 + 92) = -313$$



1

Mouse and rat are the closest pair

	0	14	25	70	99
	14	0	25	70	70
	25	25	0	38	92
	70	70	38	0	25
	99	70	92	25	0

Distance matrix

- Create a new internal node u :
- Estimate the branch length between the chosen node and the new node:

$$\begin{aligned} \text{➤ } l(i, u) &= \frac{1}{2} d(i, j) + \frac{1}{2(n-2)} [\sum_{k=1}^n d(i, k) - \sum_{k=1}^n d(j, k)] \\ \text{➤ } l(i, u) + l(j, u) &= d(i, j) \end{aligned}$$



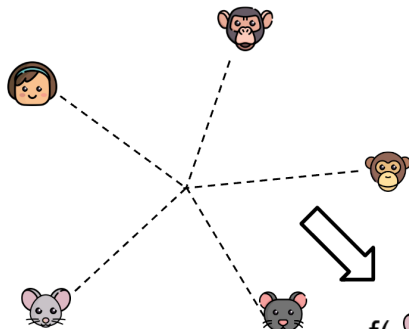
$$f(\text{mouse}, \text{mouse, monkey}) \approx -1 + 2$$

$$f(\text{monkey}, \text{mouse, monkey}) \approx 26 + 2$$

The branch length can be negative in neighbor joining

In practice, we can add a fixed constant to all branch lengths for nonnegativity!

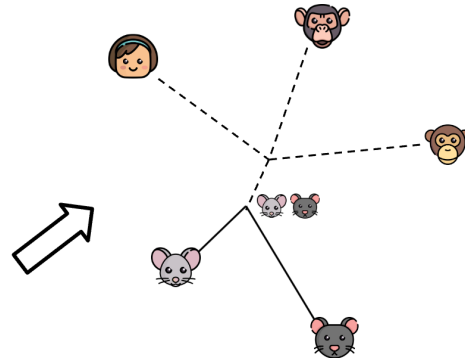
Initial tree (A random star graph)



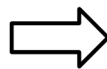
$$f(\text{mouse}, \text{mouse, monkey}) = 1$$

$$f(\text{monkey}, \text{mouse, monkey}) = 28$$

Updated tree after merging mouse/rat



	0	14	25	70	99
	14	0	25	70	70
	25	25	0	38	92
	70	70	38	0	25
	99	70	92	25	0



	0	14	25	72	
	14	0	25	58	
	25	25	0	53	
	72	58	53	0	
					0

Updated Distance matrix

The distance is designed to be additive:

$$\begin{aligned} \text{➤ } d(u, k) &= \frac{1}{2} [d(i, k) + d(j, k) - d(i, j)] \\ \text{➤ } &= \frac{1}{2} [70 + 99 - 25] = 72 \\ \text{➤ } &= \frac{1}{2} [70 + 70 - 25] \approx 58 \end{aligned}$$

Jukes-Cantor

Jukes-Cantor is a tool to recompute distances based on back-mutations for more accurate phylogenetic trees.

Assumptions:

1. All nucleotide bases occur with equal probability
2. Each base has equal probability of mutating into any other with $r = 0.25$
3. Substitutions occur independently at each site over time

Let $p_{A(t)}, p_{C(t)}, p_{G(t)}, p_{T(t)}$ be probabilities that a given site contains nucleotide A, C, G, T at time t .

Since we assume all nucleotides are equally likely, $p_{X(t)}$ is probability that a site is $X(C, G, T)$ at time t

$\sum_{\delta \in A, C, G, T} p_{\delta}(t) = 1$ and if a nucleotide is A at time $t = 0$, then $p_{A(0)} = 1, p_{C(0)} = p_{G(0)} = \dots = 0$

Define total mutation velocity away from A as α , then mutaiton velocity from A to C is $\frac{\alpha}{3}$, same with all other nucleotides.

For infinitesimal time step dt , $p_{A(t+dt)}$ = probability of staying A = $p_{A(t)} \cdot -p_{A(t)}\alpha dt + (p_{C(t)} + p_{G(t)} + p_{T(t)}) \cdot (\frac{\alpha}{3})dt$

Similarly, $p_{X(t+dt)}$ = probability of becoming C, G, or T = $p_{X(t)} - p_{X(t)}\alpha dt + p_{A(t)}(\frac{\alpha}{3})dt + 2p_{X(t)}(\frac{\alpha}{3})dt$

Taking the limit as $dt \rightarrow 0$, $\frac{dp_A}{dt} = -\frac{4\alpha}{3}p_A + \frac{\alpha}{3}$ and $\frac{dp_X}{dt} = \frac{\alpha}{3}p_A - \frac{\alpha}{3}p_X$

Then we solve to get $p_{A(t)} = \frac{1}{4} + \frac{3}{4}e^{-\frac{4\alpha}{3}t}$ and $p_{X(t)} = \frac{1}{4} - \frac{1}{4}e^{-\frac{4\alpha}{3}t}$

Thus, the expected number of substitutions per site is: $d = -\frac{3}{4} \ln(1 - \frac{4}{3}p)$ which is our new distance function. $p = 3P_x$ is the observed fraction of nucleotide differences between sequences, and d is assumed to be αt to agree with p

Character-based Method

Maximum Parsimony

Score trees based on the minimum number of substitutions required to convey the found tree.

Parsimony Simplest explanation is usually best.

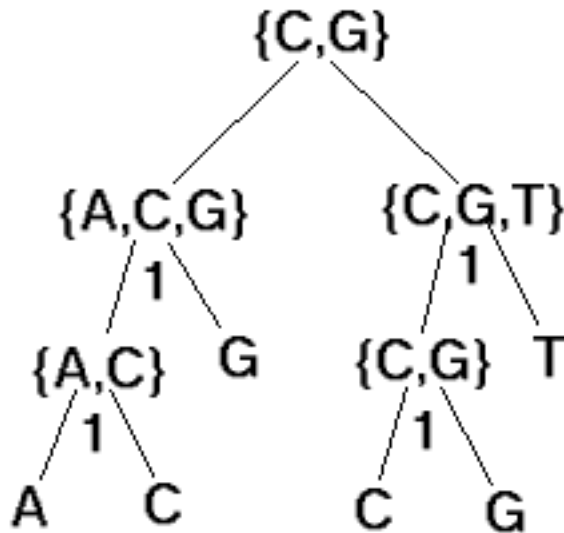
Pseudocode: Given tree and alignment column u , label internal nodes to minimize substitutions.

Initialization: Set $C = 0$ and root node $k = 2N - 1$ **Iteration:** If k is a leaf, set $R_k = \{x_{k(u)}\}$ If k is an internal node with children i, j : If $R_i \cap R_j \neq \emptyset$, set $R_k = R_i \cap R_j$ Else $R_k = R_i \cup R_j$ and increment C

To score a tree using maximum parsimony we do the following:

1. Assign nucleotide to root node:
 - Choose arbitrarily from R_{2N-1}
2. For each internal node k , assign nucleotides recursively
 - if parent k has state r and $r \in R_i$, assign r to descendant i
 - Otherwise choose arbitrarily from R_j

Note that $x_{k(u)}$ means the u 'th base pair for node k . Then, for internal nodes, you take the intersection as the list of possible values that wouldn't require a substitution. If there is nothing, then join them, so that you only require one substitution.



Note this doesn't build a tree, but scores it.

Maximum Likelihood

The likelihood of a tree T with branch lengths t and given sequence alignment is $L(T, t) = P(x_1, \dots, x_n \mid T, t)$

We want to find the most likely tree given sequence alignments. Note, $L(x_1, x_2) = \sum_X \pi_X P(x_1 \mid X, t_1) P(x_2 \mid X, t_2)$

π_X is stationary probability of nucleotide X , X is possible ancestral states at root. $P(x_i \mid X, t_i)$ is probability of transition from X to x_i in time t_i

$L_{k(X)}$ means likelihood of subtree rooted at k , given it has state X $P(X \rightarrow Y, t)$ is probability that state X at a node changes to Y over time t π_x is stationary probability of nucleotide X at the root.

Pseudocode: Initialization: At each leaf node, define $L_{i(X)} = 1$ if $X = x_i$, 0 otherwise.

Recursion: For each internal node with children i, j $L_{k(X)} = \sum_{Y,Z} P(X \rightarrow Y, t_i) L_{i(Y)} \cdot P(X \rightarrow Z, t_j) L_{j(Z)}$ **Final Step:** $L(T) = \sum_X \pi_X L_{\text{root}}(X)$

Bayesian Inference

Like maximum likelihood, based on probabilities, but tree topology is not single tree, but probability distribution of all trees. Uses Bayes' theorem to calculate posterior probability of trees based on prior probability of trees and observed data. Samples from probability distribution and updates model states iteratively

Tree Space Search

Stepwise Addition Start from minimum tree (3 taxa) and add taxa, then optimize topology.

Use starting tree A tree with all taxa but is less accurate. Either use a cheap method for construction or randomly create.

Summary

Methods	Optimality Criterion
Distance Methods	Distance
Maximum Parsimony	Parsimony
Maximum Likelihood	Likelihood

Note top-to-bottom is increasing accuracy and decreasing efficiency.

Newick Tree Format:

Newick Tree format is a text file like this:

```
(( (monkey:0.01572,chimp:0.01038):0.27040,chicken:0.37215):0.12011,  
(pig:0.08412,(rat:0.03239,mouse:0.03430):0.19210):0.06343,human:0.03265);
```

- **()** -- group (hierarchy)
- **,** -- siblings
- **:** -- branch length (optional)
- **;** -- end of tree

- **For example:** **((A,B),C);**

