Tree of Life: phylogenetic tree

Input: set of sequences \( (n) \)

Distance Matrix \( n \times n \)

1. Distance Based

2. Parsimony-based
   - minimize the change
   - "simplest" solution

3. Probabilistic:

\[ P(T | \text{data}) = \frac{P(\text{data} | T) P(T)}{P(\text{data})} \]

\[ = \frac{P(\text{data} | T) P(T)}{\sum_{T'} P(\text{data} | T') P(T')} \]

Distance-Based

Data = set of sequences

\( \rightarrow \) hierarchical clustering with clock/time

Constant
**UPGMA:** Unweighted Pair Group Method using Averages

Given **n** sequences, create a **n x n** distance matrix:

\[ d_{ij} = \text{dist}(S_i, S_j) \]

\[ d_{ij} = d_{ji} \quad \text{Symmetric} \]

1. **Distance between two sequences**
   \[ d_{ij} = \frac{d_{ij}}{2} \text{ or } Hennings' distance \]

2. **Distance between two clusters**
   \[ C_a \quad C_b \]
   \[ n_a = |G| \quad n_b = |B| \]

   \[ d_{ab} = \text{dist}(C_a, C_b) = \frac{1}{n_a n_b} \sum_{i \in A} \sum_{j \in B} d_{ij} \]

   Average pairwise distance

\[ \left( d_{13} + d_{14} + d_{23} + d_{24} + d_{34} + d_{34} \right) \]

Given a cluster:

\[ s_{ij} \]

Distance between two sequences:

\[ d_{ij} \]

Distance between clusters:

\[ d_{ab} \]
\[ \frac{1}{6} \cdot \left( d_{i3} + d_{i4} + d_{i8} + d_{i2} + d_{i4} + d_{i8} \right) \]

If we merge \( C_i \) & \( C_j \) into \( C_k \)

\[ C_k = C_i \cup C_j \]

Recompute the distances from \( C_k \) to everyone in

\[ d_{kl} = \frac{d_{ij} \cdot |C_i| + d_{jk} \cdot |C_j|}{|C_i| + |C_j|} \]

before merging...
Neighbour-Joining method

\[ D_{ij} = d_{ij} - (\gamma_i + \gamma_j) \]

Pair with the smallest "adjusted" distance should be merged

\[ \gamma_1 = \frac{1.4}{2} = 0.7 \]
\[ \gamma_2 = 0.7 \]
\[ \gamma_3 = \frac{2.0}{2} = 1.0 \]
\[ \gamma_4 = \frac{2.3}{2} = 1.0 \]

\[ D_{12} = d_{12} - (\gamma_1 + \gamma_2) = 0.3 - (1.4) = -1.1 \]
\[ D_{13} = d_{13} - (\gamma_1 + \gamma_3) = 0.5 - (1.7) = -1.2 \]
\[ D_{14} = d_{14} - (\gamma_1 + \gamma_4) = 0.1 - (1.7) = -1.6 \]

\[ d_{ij} = \frac{1}{2} (d_{ij} + r_i r_j) \]
\[ d_{jk} = d_{ij} - d_{ik} \]
\[ \frac{1}{2} (0.5 + 0.7 - 1) \]
Parsimony

Given $T$, can we compute the score $(T)$

$$\text{score}(T) = \sum_{pos \ i} \text{score for}(f, i)$$

$$\Sigma = \{A, G\}$$

Dynamic Programming

$$\Sigma \times \Sigma = \begin{bmatrix} 1 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Simple

match = 0

mismatch = 1
\[ \sum = \{A, G\} \]

\[ S_0(A) = \min \left( \text{node } 5 \right) \]

\[ S_i(C) = \left( \frac{1}{2} \right)^i \]

\[ S_i(C) = \left( \frac{1}{2} \right)^i \]

\[ S_i(a) = \min_{b \in \sum} \left( S_j(b) + S(a, b) \right) + \min_{b \in \sum} \left( S_k(b) + S(a, b) \right) \]

**Score for symbol** \(a\) **at node** \(i\)

\[ \text{Cost for symbol } b \text{ at child} \]

**Enumerate** \(T\)

**Given a tree** \(T\)

1. \( \sum \) \( \min \) \# of changes for \(p_i, i = \text{root} (T) \)

1. **Greedy approach**

\[ S_i \quad S_j \]

**do for all pairs** \(s_i, s_j\)

**All we need sequence** \(s_k\)
try all extensions by branching from and visiting branch
do for remaining sequence
Close the leaf
repeat

2) Branch & Bound
guarantees optimality

Minimum cost so far

10
9

3) Nontree Cuts search

Search space over all trees
No guarantee of optimality.