Previous Lecture: Neighbor Joining

Reminder:

In previous lecture we learn about Neighbor Joining, an algorithm for reconstructing of non-ultrametric trees. We define a new distance matrix $Q$, that for each pair of leaves determine their score not by their absolute distance alone $(d(a,b))$ but also consider their common distance from the rest of the tree. For doing so we define a vector $R$, $R \approx$Average distance for one leave to all the others, and calculate $Q$ considering both $R$ and $D$.

$$R(i) = \frac{\sum_{k}^{n} d(i,k)}{n-2} \text{ s.t } n \text{ is the number of leaves in } T \text{ and } k \in \text{leaves}(T)$$

$$Q(i,j) = R(i) + R(j) + d(i,j).$$

The algorithm pick two leaves, $i$ and $j$, such that $i,j = \arg\max_{i,j} Q(i,j)$ and merge them to a new leave $e$. For merging the next pair the algorithm must recalculate $R$ and $Q$ (With $e$ and without $i$ and $j$) for doing so he must find:

$$d(e,k) = \frac{1}{2}(d(i,k) + d(j,k) - d(i,j))$$
$$d(i,e) = d(i,k) - d(e,k)$$
$$d(j,e) = d(i,j) - d(i,k)$$
NJ example:

\[
\begin{align*}
D &= \begin{pmatrix}
a & b & c & d \\
a & 0 & 6 & 7 & 11 \\
b & 0 & 5 & 9 \\
c & 0 & 6 \\
d & 0 \\
\end{pmatrix},
R &= \begin{pmatrix}
a & b & c & d \\
a & 12 \\
b & 10 \\
c & 9 \\
d & 13 \\
\end{pmatrix},
Q &= \begin{pmatrix}
a & b & c & d \\
a & * & 16 & 14 & 14 \\
b & * & 14 & 14 \\
c & * & 16 \\
d & * \\
\end{pmatrix}
\end{align*}
\]

\((a, b) = \text{argmax}(Q), e := \text{the father of } a \text{ and } b,\)

\[
d_{ae} = \frac{1}{2}(6 + 12 - 10) = 4
\]

\[
d_{be} = 6 - 4 = 2
\]

\[
d_{ec} = \frac{1}{2}(7 + 5 - 6) = 3
\]

\[
d_{ed} = \frac{1}{2}(11 + 9 - 6) = 7
\]

**There is only one way to built a tree with three leaves (Tree without a root):**
Running NJ on non-additive matrix:

It is possible to run NJ on non-additive matrix, for doing so we have to modify the definition of $d_{ie}$. Instead of $d_{ie} = d_{im} - d_{em}$, that based upon the additivity property, we define $d_{ie} = \frac{1}{2}(d_{ij} + r_i - r_j)$.

Running modify NJ on non-additive matrix will create an additive distance matrix $D'$ s.t $D' \approx D$.

Evolutionary distance between two sequences

We will expect from the distance function to have the following properties:

1. $d(x, y) = d(y, x)$
2. $d(x, x) = 0$
3. $d(x, z) \leq d(x, y) + d(y, z)$
4. The distance function graph should be
We will try two kind of distance functions that we saw in previous lessons:

1. $\sigma_{ab} = LLR = \log\left(\frac{P_{\alpha}(a,b)}{P_{0}(a)P_{0}(b)}\right)$

2. Percentage of similarity

Those two methods do not satisfy the necessary properties of an evolution tree, for example we examine the sequence $S_0 = 000$ during the evolution.

\[ S_0 = 000 \]
\[ \downarrow \]
\[ S_1 = 001 \]
\[ \downarrow \]
\[ S_2 = 011 \]
\[ \downarrow \]
\[ S_3 = 001 \]

In evolution tree we expect $d(S_3, S_0) = 3$, but the previous methods gives us $d(S_3, S_0) = 1$. 
Probability tools for constructing evolutionary tree

Given a set of sequences \( \{s^j\} \), find \( t \) (tree structure) and \( l \) (length of branches) that maximize \( Pr(t, l|\{s^j\}) \).

\[
\arg\max_{t, l} Pr(t, l|\{s^j\}) = \arg\max_{t, l} \frac{Pr(t, l, \{s^j\})}{Pr(\{s^j\})} = \arg\max_{t, l} \frac{Pr(\{s^j\}|t, l) \cdot Pr(t, l)}{Pr(\{s^j\})} = \arg\max_{t, l} Pr(\{s^j\}|t, l)
\]

*By the assumption that probability for a tree is uniformly distributed.

**It is important the notice that:**

1. We do not know the sequence of parents node \((S_4, S_5)\), only the sequence of leaves \((S_1, S_2, S_3)\).

2. We will assume Markov property by the assumption that a sequence of a leaf is dependent only on his father sequence and the length of his branch.

\[
Pr(s^1, s^2, s^3|t, l) = \sum_{s^4, s^5} Pr(s^1, s^2, s^3, s^4, s^5|t, l) \cdot Pr(s^1|s^4, l_1) \cdot Pr(s^2|s^4, l_2) \cdot Pr(s^3|s^5, l_3) \cdot Pr(s^4|s^5, l_4) \cdot Pr(s^5)
\]

*By the Markov assumption

\[
Pr(s^1|s^4, l_1) = \prod_{i} Pr(s^1_i|s^4_i, l_1)
\]

*By the assumption that there is alignment between the sequences, There are no gaps and there is no dependencies between positions.

The probability to have a letter \( b \) from a letter \( a \) after time \( h \), \( Pr(b|a, h) = Pr(s^1_i|s^4_i, l_1) \), can be described by Markov chain.
From the Markov chain we can get the length of a branch

\[ \hat{l} = \arg\max_l Pr(s^1|s^4, l) \]

**Continuous Time Markov Model**

\[ P_t : \text{ transition matrix for time } t. \]

\[ P_t(i, j) = \text{ Probability that letter } i \text{ become } j \text{ after time } t \]

\[
\begin{align*}
t = 0, & \quad P_0(a \rightarrow b) = \begin{cases} 0 & a \neq b \\ 1 & a = b, \end{cases} \quad P_0 = I \\

P_t = & \begin{bmatrix}
P_t(a \rightarrow a) & \ldots & \ldots & P_t(a \rightarrow z) \\
\vdots & & \vdots & \\
\vdots & & \vdots & \\
P_t(z \rightarrow a) & \ldots & \ldots & P_t(z \rightarrow z)
\end{bmatrix} \\
\forall i, j: P_t(i, j) \geq 0, \quad \sum_k P_t(i, k) = 1
\end{align*}
\]

**There are infinity number of possible matrix for } P_t.**

We will notice that \[ P_{t_1 + t_2}(a \rightarrow b) = \sum_{c \in C} P_{t_1}(a \rightarrow c)P_{t_2}(c \rightarrow b), \] when \( C := \) all legal characters.
This is exactly the result of $P_{t_1} \cdot P_{t_2}(a \to b)$, we conclude that $P_{t_1+t_2} = P_{t_1} \cdot P_{t_2}$ and in particular $P_{2t} = P_t^2$ and $P_{nt} = P_t^n$.

Giving $\epsilon$ and $P_\epsilon$ s.t $t = n \cdot \epsilon$ then $P_t = P_{n \cdot \epsilon} = P_\epsilon^n$.

\[
\frac{\partial P_t}{\partial t} = \lim_{\epsilon \to 0} \frac{P_{t+\epsilon} - P_t}{\epsilon} = \lim_{\epsilon \to 0} \frac{P_{t+\epsilon} - P_t}{\epsilon} = \lim_{\epsilon \to 0} \frac{P_t(P_{\epsilon} - I)}{\epsilon} = P_t \cdot \lim_{\epsilon \to 0} \frac{P_{\epsilon} - I}{\epsilon} \quad \text{The derivative around 0}
\]

We define $R = \frac{\partial P_\epsilon}{\partial \epsilon}$ and we get $\frac{\partial P_t}{\partial t} = P_t \cdot R$.

We know that $\frac{\partial e^{ax}}{\partial x} = ae^x$ so we can conclude that $P_t = e^{t \cdot R}$.