Algorithms in Computational Biology - Scribes
The Neighbor Joining Algorithm by Saitou and Nei

Amitai Frey

April 23, 2023

1 Introduction

Phylogenetic trees are used in biology to represent the evolutionary relationships among a group of taxa. The Neighbor Joining (NJ) algorithm is a popular method for reconstructing such trees, and was introduced by Saitou and Nei in 1987 [9]. The NJ algorithm is based on a distance matrix that provides the pairwise distances between all the taxa in the group. In this paper, we will describe the ideas behind the NJ algorithm and least squares methods, and show how they relate to the Balanced Minimum Evolution (BME) algorithm introduced by Desper and Gascuel in [2].

2 The Neighbor Joining Algorithm

2.1 The Algorithm

The NJ algorithm is a hierarchical clustering algorithm that constructs a phylogenetic tree by iteratively joining the closest pairs of nodes. The algorithm starts with a distance matrix D that provides the pairwise distances between all taxa. The matrix D is assumed to satisfy a few conditions. First of all, the distance between a taxon and itself is zero. Secondly the distances are symmetric, i.e. $\forall i, j : D_{ij} = D_{ji}$. Finally, all distances must be non-negative.

The NJ algorithm constructs a binary tree that represents the evolutionary relationships among the taxa. The leaves of the tree correspond to the taxa, and the internal nodes correspond to hypothetical ancestors that are inferred to have existed at various points in the past. The length of each branch in the tree represents the amount of evolutionary change that has occurred along that branch. The NJ algorithm is a fast and efficient method for constructing phylogenetic trees, and has been shown to perform well on a wide range of data sets. The full details of the algorithm can be seen in Algorithm 1. Note that the formula that is being used is that of Studier and Keppler [11], which was shown in [3] to be equivalent to that of Saitou and Nei [9].

Algorithm 1 Neighbour Joining Algorithm

```
Let $D$ be the $n \times n$ distance matrix
Initialize $T$ as a star tree with $n$ leaves
$s_i \leftarrow \sum_{j=1}^{n} d_{ij}$ for all $i \in [n]$
while $T$ has more than 2 leaves do
    Find the pair of nodes $i$ and $j$ in $T$ with the smallest $Q_{ij}$ value, where $Q_{ij} = (n-2)d_{ij} - s_i - s_j$
    Create a new node $k$ and add edges $(k, i)$ and $(k, j)$ to $T$
    Update the distances $d_{km}$ for all $m \notin \{i, j\}$: $d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$
    Update the distance of $d_{ik}$: $d_{ik} = \frac{1}{2}d_{ij} + \frac{1}{2(n-2)}(\sum_{l=1}^{n} d_{il} - \sum_{l=1}^{n} d_{jl})$. Similarly for $d_{jk}$
    Remove nodes $i$ and $j$ from $T$
    Update $s_k \leftarrow \sum_{j=1}^{n} d_{kj}$
end while
Let the remaining node in $T$ be the root
```

The NJ algorithm has time complexity $O(n^3)$, where n is the number of taxa. The computation of the $n \times n$ matrix Q requires $O(n^2)$ operations. The search for the smallest value of Q_{ij} requires $O(n^2)$ operations, and the computation of the new distances requires $O(n^2)$ operations. The algorithm must be repeated $n-2$ times, so the total time complexity is $O(n^3)$.

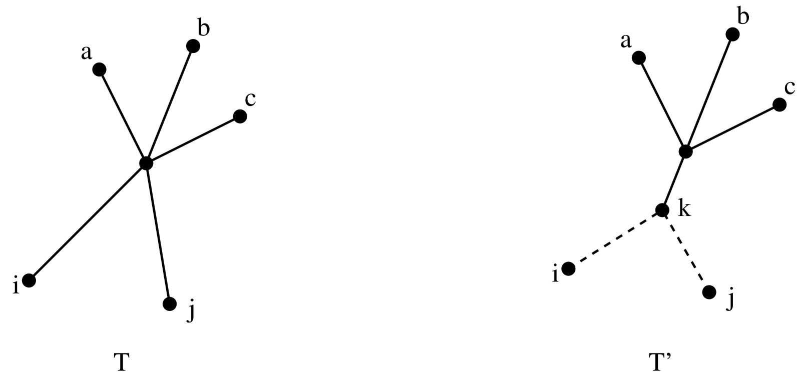

Figure 1: Illustration of one step of the NJ algorithm

2.2 Consistency of the Algorithm

After the algorithm's publication, a fundamental question had arisen as to its consistency. Specifically, the consistency of the algorithm is the question if it can accurately reconstruct a tree given that the distances are perfect. A few proofs have been given, and we will present an elegant one that was provided by Bryant [1].

Theorem 2.1. *If D is a distance matrix of T and i, j minimize Q_{ij}, then i and j are neighbours in T.*

Proof. First note that $\forall i \in [n]$, if we add some constant amount K to each distance d_{ij} then the only effect on Q is that we subtract $\frac{2K}{n-2}$ from Q_{uv} for all $u, v \in T$. This does not change the ordering of pairs with respect to Q, and also allows us to assume that every external edge in T has weight zero.

Now, for each internal nodes u, v we will have that $s_u > s_v$ if and only if there are more taxa closer to v than u. Thus, the maximum of all s's would be for some vertex v^* that is adjacent to exactly one internal edge. Since all external edges have weight 0, all leaves i adjacent to v^* have $s_i = s_{v^*}$.

Now to the proof: Suppose i, j are not neighbours, and let a, b be leaves adjacent to v^*. Since i, j are not neighbours, some internal edges separates them and we have $d_{ij} > 0 = d_{ab}$. Additionally, by the way we chose v^* we get $s_i \le s_{v^*} = s_a$ and $s_j \le s_{v^*} = s_b$. Therefore, $Q_{ij} > Q_{ab}$, and i, j do not minimize Q. □

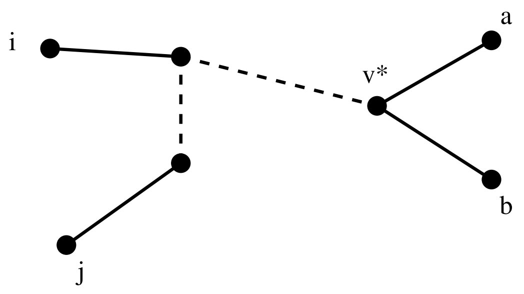

Figure 2: Illustration of the proof

3 The Least Squares Methoed

3.1 Ordinary Least Squares

Another approach for phylogenetic tree reconstruction is the least squares method [6]. The goal is to find a tree that minimizes the sum of squared differences between the observed distances and the expected distances under a given model of evolution. The ordinary least squares (OLS) criterion can then be formulated as:

$$\text{minimize} \sum_{i<j} (d_{ij} - e_{ij})^2$$

where e_{ij} is the actual distance between taxa i and j in the original tree.

However, this approach has a few disadvantages. Firstly, OLS is sensitive to outliers, and errors or other factors that lead to distances much larger or smaller than expected can result in inaccurate or misleading results. Secondly, OLS assumes that evolutionary rates are constant across all branches of the tree, which is often not the case in practice. Neglecting this variation can lead to biased or inconsistent estimates of the tree topology and branch lengths. Lastly, OLS requires a complete distance matrix with no missing values, which can be difficult to obtain in practice. OLS also does not offer a principled way to handle missing data, which can again lead to biased or inconsistent estimates of the tree topology and branch lengths.

3.2 Weighted and Constrained Least Squares

Weighted Least Squares (WLS) is a modification of the least squares method that assigns weights to the observed distances based on their estimated variances. The weighted least squares criterion can be expressed as

$$\text{minimize} \sum_{i<j} \frac{(d_{ij} - e_{ij})^2}{d_{ij}}$$

Weighted least squares can reduce the influence of observations with large variances and can improve the accuracy of the reconstructed tree.

A common problem of both OLS and WLS is that they can lead to negative weights on the branches of the constructed trees. Therefore, the constrained least squares method incorporates additional constraints on the branch lengths - that they must be non-negative. For this formulation we get the following criterion:

$$\text{minimize} \sum_{i<j} (d_{ij} - e_{ij})^2 \quad \text{subject to } e_{ij} \geq 0$$

Constrained least squares can improve the accuracy of the reconstructed tree by incorporating additional information or assumptions about the evolutionary process. However, the constraints can make the optimization problem more difficult to solve and can lead to biased or inconsistent estimates if the constraints are not appropriate for the data.

4 The Relationship Between NJ and LS

4.1 Tree Length and Balanced Minimum Evolution

All least squares approaches assume that the evolutionary relationships among the sequences can be represented by a tree with the fewest number of changes in the data. This assumption is called “minimum evolution” (ME), and is not assumed a-priori in the NJ algorithm. At first it was thought that the NJ algorithm minimizes the OLS criterion, and therefore is also an ME method. However, simulations such as in [8, 5] showed that this was not the case, as shorter trees (in OLS terms) exist for some runs of NJ, even though they are less accurate.

Additionally, Gascuel and Steel describe in [4] that a run of the NJ algorithm minimizes the OLS criterion on the original tree, but not on an intermediate tree during the run of the NJ algorithm, where there are some agglomerated nodes. These discoveries led researches to believe that despite NJ’s good performance, it does not optimize a natural property of phylogenetic trees such as maximum likelihood or parsimony (which OLS is closely related to). Understanding the criteria that are being optimized is vital in these applications, and so a deeper understanding of the NJ algorithm was required to properly justify its use as well as understand its good performance.

It turns out that the NJ algorithm actually minimizes a different tree length property, which we will now introduce. As an example, consider the following tree:

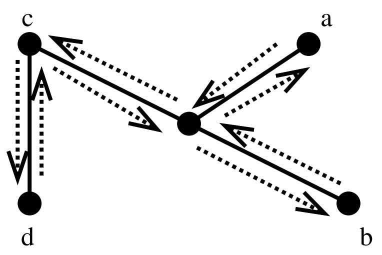

Figure 3: A tree with circular tree length estimate

We can calculate the tree's length by summing its branch lengths along a cyclical traersal of the tree. For this tree, the length will be:

$$\ell = \frac{1}{2}\left(d_{ab} + d_{bd} + d_{db}\right)$$

We traversed the tree in a cyclic manner, so we counted all of the edges twice and therefore multiply by $\frac{1}{2}$. However, since there may be a few ways to describe the "same" tree, this estimate isn't good enough. Therefore, Semple and Steel [10] suggest the following general formula:

$$\ell(T) = \sum_{\{i,j\}} w_{ij} d_{ij} \tag{1}$$

where the weight w_{ij} is calculated as follows: consider all interior nodes on the path from i to j. Count the number of outgoing branches from these nodes, and divide 1 by that number.

This is the general case of Pauplin's [7] formula for binary trees, where w_{ij} is equal to $\frac{1}{2}$ to the power of the number of interior nodes between i and j. Desper and Gascuel [2] show that the NJ algorithm actually minimizes the tree length in the sense of Pauplin's formula. They also design an algorithm that utilizes this approach, which they call Balanced Minimum Evolution (BME). Finally, they show that this algorithm achieves better results than both NJ and the least squares approaches. We will now prove that NJ acts as a greedy algorithm for minimizing tree length.

4.2 Neighbour Joining Minimizes Total Tree Length

Theorem 4.1 (from [2])**.** *The Neighbour Joining algorithm selects at each step as neighbors that pair of current taxa which most decreases the whole tree length, as computed using the generalized Pauplin formula.*

Proof. Consider trees like T and T' in Figure 1. Each leaf is a subtree that is either a single taxon in the original data set or was created by a previous step of the algorithm. Denote by A, B the two subtrees that are joined in the step of the algorithm between T and T', and by X, Y two other subtrees that are connected to the central node. Let r be the degree on the central node, and a, b, x, y be original taxa in the respective subtrees A, B, X, Y.

From Equation (1) we get:

$$\ell(T) - \ell(T') = \sum_{\{i,j\}} (w_{ij} - w'_{ij}) d_{ij}$$

where w_{ij} and w'_{ij} are computed in T and T', respectively. These differ only if i and j are not within a single subtree A, B, X or Y. So the previous equation becomes:

$$\ell(T) - \ell(T') = \sum_{\{a,b\}} (w_{ab} - w'_{ab}) d_{ab} + \sum_{\{a,x\}} (w_{ax} - w'_{ax}) d_{ax} + \\ \sum_{\{b,x\}} (w_{bx} - w'_{bx}) d_{bx} + \sum_{\{x,y\}} (w_{xy} - w'_{xy}) d_{xy}$$

As we stated before, since this is a binary tree we can use Pauplin's formula and get:

$$\ell(T) - \ell(T') = ((r-1)^{-1} - 2^{-1}) D^T_{AB} + ((r-1)^{-1} - (2(r-2))^{-1}) \sum_X (D^T_{AX} + D^T_{BX}) \\ + ((r-1)^{-1} - (r-2)^{-1}) \sum_{\{X,Y\}} D^T_{XY}$$

where D^T_{AB} is the distance matrix for the nodes in A, B, and similarly for $D^T_{AX}, D^T_{BX}, D^T_{XY}$.

Now denote by I, J subtrees that are leaves, and we get:

$$\ell(T) - \ell(T') = \frac{1}{2} D^T_{AB} + \frac{1}{2(r-2)} \left(\sum_{I \neq A} D^T_{AI} + \sum_{I \neq B} D^T_{BI} \right) \\ + ((r-1)^{-1} + (r-2)^{-1}) \sum_{\{I,J\}} D^T_{IJ}$$

The last term is indpendent of A, B and the other terms should look familiar - they correspond to the criterion that appears in Studier and Keppler's version of the NJ algorithm. Thus we can see that at each step of the algorithm, NJ greedily minimizes this notion of tree length. □

5 Conclusion

In summary, the Neighbor Joining (NJ) algorithm is a popular algorithm for reconstructing phylogenetic trees from distance matrices. The algorithm iteratively builds a binary tree by joining pairs of nodes with the smallest values of a certain criterion. The NJ algorithm is fast and produces reasonably accurate trees, but has been shown to have some limitations. On the other hand, least squares approaches have also been used to reconstruct phylogenetic trees, and for some time these approaches where thought to be equivalent.

Pauplin [7] derived a formula for the length of a balanced binary tree in terms of the pairwise distances between its leaves, and it was shown that the NJ algorithm can be viewed as an iterative algorithm for finding the balanced tree with minimum length. The Balanced Minimum Evolution (BME) algorithm, introduced by Desper and Gascuel [2], is a variant of the minimum evolution principle that takes into account the balanced nature of phylogenetic trees. The BME algorithm uses the formula derived by Pauplin to compute the length of a balanced tree, and has been shown to converge to the balanced tree with minimum length that is consistent with the input distance matrix. Additionally, they have shown that the NJ algorithm actually optimizes the criterion of tree length, and not that of least squares.

References

[1] David Bryant. On the uniqueness of the selection criterion in neighbor-joining. *Journal of Classification*, 22(1):3–15, 2005.

[2] Richard Desper and Olivier Gascuel. The minimum evolution distance-based approach of phylogenetic inference., 2007.

[3] Olivier Gascuel. A note on sattath and tversky's, saitou and nei's, and studier and keppler's algorithms for inferring phylogenies from evolutionary distances. *Molecular biology and evolution*, 11(6):961–963, 1994.

[4] Olivier Gascuel and Mike Steel. Neighbor-joining revealed. *Molecular biology and evolution*, 23(11):1997–2000, 2006.

[5] Sudhir Kumar. A stepwise algorithm for finding minimum evolution trees. *Molecular biology and evolution*, 13(4):584–593, 1996.

[6] Masatoshi Nei, Sudhir Kumar, et al. *Molecular evolution and phylogenetics*. Oxford University Press, USA, 2000.

[7] Yves Pauplin. Direct calculation of a tree length using a distance matrix. *Journal of Molecular Evolution*, 51(1), 2000.

[8] Naruya Saitou and Tadashi Imanishi. Relative efficiencies of the fitch-margoliash, maximum-parsimony, maximum-likelihood, minimum-evolution, and neighbor-joining methods of phylogenetic tree construction in obtaining the correct tree. 1989.

[9] Naruya Saitou and Masatoshi Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4):406–425, 1987.

[10] Charles Semple and Mike Steel. Cyclic permutations and evolutionary trees. *Advances in Applied Mathematics*, 32(4):669–680, 2004.

[11] John A Studier and Keith J Keppler. A note on the neighbor-joining algorithm of saitou and nei. *Molecular biology and evolution*, 5(6):729–731, 1988.