Fast and Accurate Phylogeny Reconstruction Algorithms Based on the Minimum-Evolution Principle

Richard Desper, Olivier Gascuel

▶ **To cite this version:**

Richard Desper, Olivier Gascuel. Fast and Accurate Phylogeny Reconstruction Algorithms Based on the Minimum-Evolution Principle. WABI 2002 - 2nd International Workshop on Algorithms in Bioinformatics, Sep 2002, Roma, Italy. pp.357-374, 10.1007/3-540-45784-4_27 . lirmm-00269513

HAL Id: lirmm-00269513

https://hal-lirmm.ccsd.cnrs.fr/lirmm-00269513

Submitted on 17 Oct 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Fast and Accurate Phylogeny Reconstruction Algorithms Based on the Minimum-Evolution Principle

Richard Desper[1] and Olivier Gascuel[2]

[1] National Center for Biotechnology Information, National Library of Medicine, NIH
Bethesda, MD USA
desper@ncbi.nlm.nih.gov
[2] Dept. Informatique Fondamentale et Applications
LIRMM, Montpellier, France
gascuel@lirmm.fr
http://www.lirmm.fr/~w3ifa/MAAS/

Abstract. This paper investigates the standard ordinary least-squares version [24] and the balanced version [20] of the minimum evolution principle. For the standard version, we provide a greedy construction algorithm (GME) which produces a starting topology in $O(n^2)$ time, and a tree swapping algorithm (FASTNNI) that searches for the best topology using nearest neighbor interchanges (NNIs), where the cost of doing p NNIs is $O(n^2 + pn)$, i.e. $O(n^2)$ in practice because p is always much smaller than n. The combination of GME and FASTNNI produces trees which are fairly close to Neighbor Joining (NJ) trees in terms of topological accuracy, especially with large trees. We also provide two closely related algorithms for the balanced version, called BME and BNNI, respectively. BME requires $O(n^2 \times \mathrm{diam}(T))$ operations to build the starting tree, and BNNI is in $O(n^2 + pn \times \mathrm{diam}(T))$, where $\mathrm{diam}(T)$ is the topological diameter of the output tree. In the usual Yule-Harding distribution on phylogenetic trees, the diameter expectation is in $\log(n)$, so our algorithms are in practice faster that NJ. Furthermore, BNNI combined with BME (or GME) has better topological accuracy than NJ, BIONJ and WEIGHBOR (all three in $O(n^3)$), especially with large trees.

Minimum evolution (ME) was proposed by several authors as a basic principle of phylogenetic inference [28]. Given the matrix of pairwise evolutionary distances between the taxa being studied, this principle involves first estimating the length of any given topology and then selecting the topology with shortest length. Numerous variants exist, depending on how the edge lengths are estimated and how the tree length is calculated from these edge lengths [14]. Several definitions of tree length have been proposed, differing from one another by the treatment of negative edge lengths. The most common solution [24,25] simply defines the tree length as the sum of all edge lengths, regardless of whether they are positive or negative. Edge lengths are usually estimated within the least-squares framework. When all distance estimates are supposed to be independent and to have the same variance, we use ordinary least-squares (OLS).

The weighted least-squares corresponds to the case were distance estimates are independent but (possibly) with different variances, while the generalized least-squares approach does not impose any restriction and is able to take benefit from the covariances of the distance estimates. Distance estimates obtained from sequences do not have the same variance, large distances are much more variable than the shortest ones [11], and are mutually dependent when they share a common history (or path) in the true phylogeny [19]. Therefore, to estimate edge lengths from evolutionary distances, using generalized least-squares is theoretically superior to using weighted least-squares, which is in turn more appropriate than ordinary least-squares [4].

The ME principle has been shown to be statistically consistent when combined with ordinary least-squares [7,24]. However, surprisingly (and unfortunately) we recently demonstrated [14] that the combination of this principle with weighted least-squares might be inconsistent in some (likely unrealistic) cases, while the combination with generalized least-squares is deeply inconsistent and cannot be used for phylogenetic inference.

The ME principle, combined with a form of OLS length estimation, constitutes the basis of the popular Saitou and Nei's [25] Neighbor Joining (NJ) algorithm, which greedily refines a star tree by agglomerating taxa pairs in order to minimize the tree length at each step. The traditional approach to using ME is then to start with the NJ tree, and then do a topological search from that starting point. The first stage requires $O(n^3)$, where n is the number of taxa, while the current implementations [24] of the second are in $O(pn^3)$, where p is the number of swaps performed by the program. Another approach, available in FITCH 3.6 [10], is to greedily construct a tree by iterative addition of taxa to a growing tree, and then to perform swaps from this initial tree. But the time complexity is $O(n^4)$ or more.

This paper further investigates the ME principle. First, we demonstrate that its combination with OLS, even when not fully optimal in terms of topological accuracy, has the great advantage to lead to very fast $O(n^2)$ algorithms, able to build huge trees as envisaged in biodiversity studies. Second, we show that a new version of this principle, first introduced by Pauplin [20] to simplify tree length computation, is more appropriate than the OLS version. In this new version, sibling subtrees have equal weight, as opposed to the standard unweighted OLS, where all taxa have the same weight so that the weight of a subtree is equal to the number of its taxa. Algorithms to deal with this new "balanced" version are faster than NJ, although not as fast as their OLS counter-part, and have a better topological accuracy than NJ, but also than BIONJ [12] and WEIGHBOR [2]. The rest of this paper is organized as follows: we first provide the notation, definitions and formulae, we describe the tree swapping algorithms for both OLS and balanced ME versions, we explain how these algorithms are modified to greedily construct a tree by iterative taxa addition, we provide simulation results to illustrate the gain in topological accuracy and run times, and we then conclude by a brief discussion. We have omitted some of the details of

the algorithms and some mathematical proofs, which will appear in the journal version of the paper.

1 Notation, Definitions, and Formulae

A tree is made of nodes (or vertices) and of edges. Among the nodes we distinguish the internal (or ancestral) nodes and the leaves (or taxa), The leaves are denoted as i, j or k, the internal nodes as u, v or w, while an edge e is defined by a pair of nodes and a length $l(e)$. We shall be considering various length assignments of the same underlying shape. In this case, we shall use the word "topology", while "tree" will be reserved for an instance of a topology with given edge lengths associated. We use the letter $\mathcal{T}$ to refer to a topology and T to refer to a tree. A tree is also made of subtrees typically denoted as A, B, C or D. For the sake of simplicity, we shall use the same notation for the subtrees and for the sets of taxa they contain. Accordingly, T also represents the set of taxa being studied, and n is the number of these taxa. Moreover, we shall use lowercase letters, e.g. a, b, c or d, to represent the subtree roots. If A and B are two disjoint subtrees, with roots a and b respectively, we'll say that A and B are distant-k subtrees if there are k edges in the path from a to b.

Δ is the matrix of pairwise evolutionary distance estimates, with Δ_{ij} being the distance between taxa i and j. Let A and B be two non-intersecting subtrees from a tree T. We define the average distance between A and B as:

$$\Delta_{A|B} = \frac{1}{|A||B|} \sum_{i \in A, j \in B} \Delta_{ij}. \tag{1}$$

$\Delta_{A|B}$ may also be defined recursively as:

- If A and B are singleton sets, i.e. $A = \{a\}$ and $B = \{b\}$, then $\Delta_{A|B} = \Delta_{ab}$,
- Else, without loss of generality let $B = B_1 \cup B_2$, so we then have

$$\Delta_{A|B} = \frac{|B_1|}{|B|} \Delta_{A|B_1} + \frac{|B_2|}{|B|} \Delta_{A|B_2}. \tag{2}$$

It is easily seen that Equations (1) and (2) are equivalent. Equation (2) follows the notion that the weight of a subtree is proportional to the number of its taxa. Thus every taxon has the same weight, and the same holds for the distances as shown by Equation (1). Thus, this average is said to be unweighted (Sneath and Sokal 1973). It must be noted that the unweighted average distance between subtrees does not depend on their topologies, but only of the taxa they contain.

Δ^T is the distance induced by the tree T, i.e., Δ^T_{ij} is equal to the length of the path connecting i to j in T, for every taxon pair (i, j). Given a topology $\mathcal{T}$ and a distance matrix Δ, the OLS edge length estimation produces the tree T with topology $\mathcal{T}$ minimizing the sum of squares

$$\sum_{i,j \in T} (\Delta^T_{ij} - \Delta_{ij})^2.$$

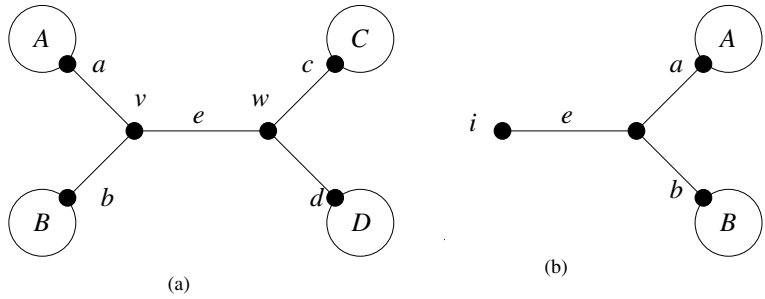

Fig. 1. Corner subtrees used to estimate the length of e, (a) for e an internal edge; (b) for e an external edge

Vach [29], Rzhetsky and Nei [24] and others showed analytical formulae for the proper OLS edge length estimation, as functions of the average distances. Suppose e is an internal edge of T, with the four subtrees A, B, C and D defined as depicted in Figure 1(a). Then, the OLS length estimate of e is equal to:

$$l(e) = \frac{1}{2}\left[\lambda(\Delta_{A|C} + \Delta_{B|D}) + (1-\lambda)(\Delta_{A|D} + \Delta_{B|C}) - (\Delta_{A|B} + \Delta_{C|D})\right], \quad (3)$$

where

$$\lambda = \frac{|A||D| + |B||C|}{(|A| + |B|)(|C| + |D|)}.$$

Suppose e is an external edge, with i, A and B as represented in Figure 1(b). Then we have:

$$l(e) = \frac{1}{2}(\Delta_{A|i} + \Delta_{B|i} - \Delta_{A|B}). \quad (4)$$

Following Saitou and Nei [25] and Rzhetsky and Nei [24], we define the tree length $l(T)$ of T to be the sum of the edge lengths of T. The OLS minimum evolution tree is then that tree with topology $\mathcal{T}$ minimizing $l(T)$, where T has the OLS edge length estimates for $\mathcal{T}$, and $\mathcal{T}$ ranges over all possible tree topologies for the taxa being studied.

Now, suppose that we are interested in the length of the tree T shown in Figure 1(a), depending on the configuration of the corner subtrees. We then have (proof deferred until journal publication):

$$\begin{aligned} l(T) = \frac{1}{2}\left[\lambda(\Delta_{A|C} + \Delta_{B|D}) + (1-\lambda)(\Delta_{A|D} + \Delta_{B|C}) + \Delta_{A|B} + \Delta_{C|D}\right] \quad (5) \\ + l(A) + l(B) + l(C) + l(D) - \Delta_{a|A} - \Delta_{b|B} - \Delta_{c|C} - \Delta_{d|D}, \end{aligned}$$

where λ is defined as in Equation (3). The advantage of Equation (5) is that the lengths $l(A), l(B), l(C)$ and $l(D)$ of the corner subtrees as well as the average root/leaf distances, $\Delta_{a|A}, \Delta_{b|B}, \Delta_{c|C}$ and $\Delta_{d|D}$, do not depend of the configuration of A, B, C and D around e. Exchanging B and C or B and D might change the length of the five edges shown in Figure 2(a), and then the length of T, but

not the lengths of A, B, C and D. This simply comes from the fact that the edge e is within the corner subtrees associated to any of the edges of A, B, C and D. As we shall see in the next section, this property is of great help in designing fast OLS tree swapping algorithms.

Let us now turn our attention toward the balanced version of minimum evolution, as defined by Pauplin [20]. The tree length definition is the same. Formulae for edge length estimates are identical to Equations (3) and (4), with λ replaced by $1/2$, but using a different definition of the average distance between subtrees, a definition which depends on the topology under consideration. Letting $\mathcal{T}$ be this topology, the balanced average distance between two non-intersecting subtrees A and B is then recursively defined by:

- If A and B are singleton sets, i.e. $A = \{a\}$ and $B = \{b\}$, then $\Delta^{\mathcal{T}}_{A|B} = \Delta_{ab}$,
- Else, without loss of generality let $B = B_1 \cup B_2$. We then have

$$\Delta^{\mathcal{T}}_{A|B} = \frac{1}{2}(\Delta^{\mathcal{T}}_{A|B_1} + \Delta^{\mathcal{T}}_{A|B_2}). \tag{6}$$

The change from Equation (2) is that the sibling subtrees B_1 and B_2 now have equal weight, regardless of the number of taxa they contain. Thus taxa do not have the same influence depending on whether they belong to a large clade or are isolated, which can be seen as consistent in the phylogenetic inference context [26]. Moreover, a computer simulation comparing variants of the NJ algorithm [13] showed that this "balanced" (or "weighted") approach is more appropriate than the unweighted one for reconstructing phylogenies with evolutionary distances estimated from sequences. Therefore, it was tempting to test the performance of this balanced version of the minimum evolution principle.

Unfortunately, this new version does not have all of the good properties as the OLS version: the edge length estimates given by Equations (3) and (4) now depend on the topology of the corner subtrees, simply because the weighted average distances between these subtrees depend on their topologies. As we shall see, this makes the algorithms more complicated and more expensive in computing time than with OLS.

However, the same tree length Equation (5) holds with Δ being replaced by $\Delta^{\mathcal{T}}$ and λ by $1/2$, and, fortunately, we still have the good property that tree lengths $l(A), l(B), l(C)$ and $l(D)$ as well as average root/leaves distances $\Delta^{\mathcal{T}}_{a|A}, \Delta^{\mathcal{T}}_{b|B}, \Delta^{\mathcal{T}}_{c|C}$ and $\Delta^{\mathcal{T}}_{d|D}$ remain unchanged when B and C or B and D are swapped. Edge lengths within the corner subtrees may change when performing a swap, but their (balanced) sums remain identical (proof deferred until journal publication).

2 FASTNNI & BNNI Tree Swapping Algorithms

This paper presents two algorithmic schemes for phylogenetic inference. The first constructs an initial tree by the stepwise addition of taxa to a growing tree, while the second improves this tree by performing local rearrangements (or swapping)

of subtrees. Both follow a greedy approach and seek, at each step, to improve the tree according to the minimum evolution criterion. This approach does not guarantee that the global optimum will be reached, but only a local optimum. However, this kind of approach has proven to be effective in many optimization problems [5], and we shall see that further optimizing the minimum evolution criterion would not yield significant improvement in terms of topological accuracy. Moreover, such a combination of heuristic and optimization algorithms is used in numerous phylogenetic reconstruction methods, for example those from the PHYLIP package [10].

In this section, we consider *tree swapping* to minimize the minimum evolution criterion, when edge lengths are assigned to the topology either by OLS or the balanced version. The algorithm iteratively exchanges subtrees of an initial tree, in order to minimize the length of the tree as described in Equation (5). There are many possible definitions of "subtree exchange". Since the number of combinations is high, we usually only consider exchanging neighboring subtrees, and, at least initially, we restrict ourselves to the exchange of subtrees separated by three edges, for example B and C in Figure 1(a). Such a procedure is called a "nearest neighbor interchange", since exchanging subtrees separated by one or two edges does not yield any modification of the initial tree. We adopted this approach because it allows a fast algorithm, and it is sufficient to reach a good topological accuracy.

2.1 The FASTNNI Tree Swapping Algorithm

Assume that OLS edge lengths are used, and that the swap between B and C in Figure 1(a) is considered. Let T be the initial tree and T' the swapped tree. According to Equation (5) and our remarks, we have:

$$L(T) - L(T') = \frac{1}{2}\left[(\lambda - 1)(\Delta_{A|C} + \Delta_{B|D}) - (\lambda' - 1)(\Delta_{A|B} + \Delta_{C|D}) \right.$$
$$\left. -(\lambda - \lambda')(\Delta_{A|D} + \Delta_{B|C})\right], \tag{7}$$

where λ is as defined in Section 1, and

$$\lambda' = \frac{|A||D| + |B||C|}{(|A| + |C|)(|B| + |D|)}.$$

If $L(T) - L(T') > 0$, T is not optimal, and we greedily choose the tree swap which corresponds to the largest difference between $L(T)$ and $L(T')$ (there are 2 such possible swaps for each internal edge of T). Moreover, assuming that the average distances between the corner subtrees have already been computed, $L(T) - L(T')$ can be obtained in $O(1)$ time via Equation (7). Instead of computing the average distances between the corner subtrees (which change when swaps are realized), we compute the average distances between every pair of non-intersecting subtrees. This takes place before evaluating the swaps and requires $O(n^2)$ time, using an algorithm that is based on the recursive Equation (2) and is completely explained in the journal version of the paper. The whole algorithm can be summarized as follows:

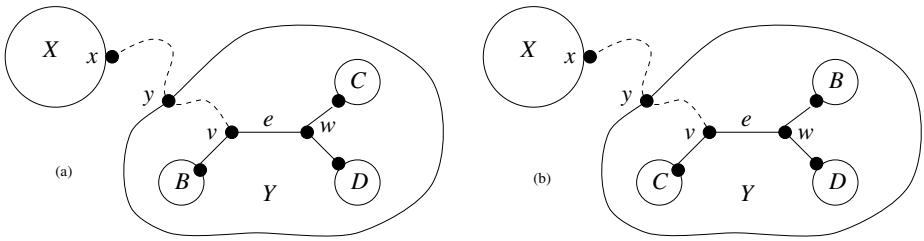

Fig. 2. T' is obtained from T by swapping B and C

1. Precompute the average distances between non-intersecting subtrees;
2. Run over all internal edges and select the best swap using Equation (7);
3. If the best swap is not relevant ($L(T) - L(T') \leq 0$) stop and return the current tree, else perform the swap, compute the average distances between the newly created subtrees ($A \cup C$ and $B \cup D$ in our example above) and the other non-intersecting subtrees using Equation (2), and go to Step 2.

Step 1 requires $O(n^2)$ time, Step 2 requires $O(n)$ time and Step 3 also requires $O(n)$ time because the total number of new averages is $O(n)$. Thus, the total complexity of the algorithm is $O(n^2 + pn)$, where p is the number of swaps performed. In practice, p is much smaller than n, as we shall see in Section 4, so this algorithm has a practical time complexity of $O(n^2)$. It is very fast, able to improve trees with thousands of taxa, and we called it FASTNNI (Fast Nearest Neighbor Interchanges).

Rzhetsky and Nei [24] describe a procedure that requires $O(n^2)$ to compute every edge length. In one NNI, five edge lengths are changed, so evaluating a single swap is in $O(n^2)$, searching for the best swap in $O(n^3)$, and their whole procedure in $O(pn^3)$. This can be improved using Bryant and Waddell [3] results, but the implementation in the PAUP environment of these ideas is still in progress (David Bryant, personal communication). In any case, our $O(n^2)$ complexity is optimal since it is identical to the data size.

2.2 The BNNI Tree Swapping Algorithm

The BNNI algorithm is modeled after FASTNNI, with a simple change to each step. Step 1 is changed by replacing Equation (2) with Equation (6) to calculate subtree average distances. Step 2 is changed in the balanced setting by using the equation:

$$L(T) - L(T') = \frac{1}{4}\left[(\Delta^{\mathcal{T}}_{A|B} + \Delta^{\mathcal{T}}_{C|D}) - (\Delta^{\mathcal{T}}_{A|C} + \Delta^{\mathcal{T}}_{B|D})\right]$$

instead of Equation (7). There is a major difference within Step 3, where average distances between subtrees are updated. Consider the swap of subtrees B and C as depicted in Figure 2. To maintain an accurate table of average distances,

BNNI must recalculate $\Delta^{\mathcal{T}'}_{X|Y}$. Suppose there are $l-2$ edges in the path from y to u in T. We then compute

$$\Delta^{\mathcal{T}'}_{X|Y} = \Delta^{\mathcal{T}}_{X|Y} + 2^{-l}\Delta^{\mathcal{T}}_{X|C} - 2^{-l}\Delta^{\mathcal{T}}_{X|B}.$$

We'll need to do such a recomputation for every pair (x, y) in the same subtree (here A, see Figure 1(a) and Figure 2) of the tree swapping step. Since there are at most $n \times \mathrm{diam}(T)$ such pairs for any tree T, the time complexity of the updating step is $O(n \times \mathrm{diam}(T))$. The upper bound is worst when T is a chain. In this case, the diameter is proportional to n and both the number of distances to update and the bound are proportional to n^2. However, the diameter is usually much lower. Assuming, as usual, a Yule-Harding speciation process [15,30], the expected diameter is $O(\log(n))$ [8], which implies an average complexity of the updating step in $O(n \log(n))$. Other (e.g. uniform) distributions on phylogenetic trees are discussed in [1,18], with expected diameter at most $O(\sqrt{n})$. Therefore, the total time complexity of the algorithm is $O(pn^2)$ in the worst case, and $O(n^2 + pn\log(n))$ (or $O(n^2 + pn\sqrt{n})$) in practice. As we shall see, BNNI is a bit slower than FASTNNI, but is still able to improve thousand-taxa trees.

Neither FASTNNI nor BNNI explicitly computes the edge lengths until all tree swaps are completed. This is done in $O(n)$ time by using the edge length formulae (3) and (4) for FASTNNI, while for BNNI, we use

$$l(e) = \Delta^{\mathcal{T}}_{A\cup B|C\cup D} - \frac{1}{2}(\Delta^{\mathcal{T}}_{A|B} + \Delta^{\mathcal{T}}_{C|D})$$

and

$$l(e) = \frac{1}{2}(\Delta^{\mathcal{T}}_{i|A} + \Delta^{\mathcal{T}}_{i|B} - \Delta^{\mathcal{T}}_{A|B}),$$

for the external and internal edges, respectively (see Figure 1 for the notation).

3 The GME and BME Addition Algorithm

The tree swapping algorithms of Section 2 are very fast at moving from an initial topology to a local minimum for their respective minimum evolution criterion. Rzhetsky and Nei [23] proposed the Neighbor-Joining tree as a good initial starting point. However, Neighbor-Joining requires $O(n^3)$ computations, more than FASTNNI or BNNI would require. In this section, we will consider an insertion algorithmic scheme, which iteratively adds taxa to partial trees by using tree swapping techniques similar to those from Section 2 to find the optimal insertion point. This simple method provides an $O(n^2)$ (resp. $O(n^2 \times \mathrm{diam}(T))$ algorithm for finding a good initial topology for the OLS (resp. balanced) version of minimum evolution.

3.1 The GME Greedy Addition Algorithm

Given an ordering on the taxa, denoted as $(1, 2, 3, \ldots, n)$, for $k = 4$ to n we create a tree T_k on the taxa set $(1, 2, 3, \ldots, k)$. We do this by testing each edge

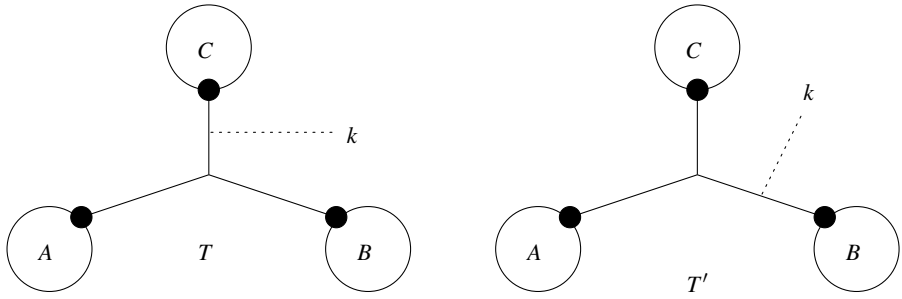

Fig. 3. T' is obtained from T by swapping A and k

of T_{k-1} as a possible insertion point for k, and the different insertion points are compared by the minimum evolution criterion. Inserting k on any edge of T_{k-1} removes that edge, changes the length of every other already existing edge, and requires the computation of the length of the three newly created edges. For each edge $e \in T_{k-1}$, define T_e to be tree created by inserting k along e, and set $c(e)$ to be the OLS size of T_e. Computing $c(e)$ for each e would seem to be computationally expensive. However, a much simpler approach using tree swapping exists to determine the best insertion point.

Consider the tree T of Figure 3, where k is inserted between subtrees C and $A \cup B$, and assume that we have the length $l(T) = L$ of this new tree. If we knew the size of T, we could quickly calculate the size of T', using a special version of Equation (5):

$$l(T') = L + \frac{1}{2}\left[(\lambda - \lambda')(\Delta_{k|A} + \Delta_{B|C}) + (\lambda' - 1)(\Delta_{A|B} + \Delta_{k|C}) + (1 - \lambda)(\Delta_{A|C} + \Delta_{k|B})\right]. \tag{8}$$

In this situation,

$$\lambda = \frac{|A| + |B||C|}{(|A| + |B|)(|C| + 1)},$$

and

$$\lambda' = \frac{|A| + |B||C|}{(|A| + |C|)(|B| + 1)}.$$

Suppose we were to first root the tree at some leaf r, let $e_0 = (r, s)$ be the edge incident to r, and L to be $c(e_0)$. We could then calculate $c(e_1)$ and $c(e_2)$ where e_1 and e_2 are the two edges incident to e_0, and in fact we could find $c(e)$ for each edge $e \in T_{k-1}$ merely by repeated applications of Equation (8). To simplify matters, we could write $c(e) = L + f(e)$. Because we only seek to determine the best insertion point, we need not calculate the actual value of L, as it is sufficient to minimize $f(e)$ with $f(e_0) = 0$.

To carry out the $(2k - 5)$ calculations of Equation (8), we need only precalculate

1. all average distances $\Delta_{k|S}$ between k and any subtree S from T_{k-1};

2. all average distances between subtrees of T_{k-1} separated by two edges, for example A and B in Figure 3;
3. the number of leaves of every subtree.

The algorithm can be summarized as follows:

- For $k = 3$, initialize the matrix of average distances between distant-2 subtrees and the array counting the number of taxa per subtree. Form T_3 with leaf set $\{1, 2, 3\}$.
- For $k = 4$ to n,
  1. Compute all $\Delta_{k|S}$ average distances;
  2. Starting from an initial edge e_0 of T_{k-1}, set $f(e_0) = 0$, and recursively search each edge e to obtain $f(e)$ from Equation (8).
  3. Select the best edge by minimizing f, insert k on that edge to form T_k, and update the average distance between every pair of distant-2 subtrees as well as the number of taxa per subtree.
- Return T_n.

To achieve Step 1, we recursively apply Equation (2), which requires $O(k)$ computing time. Step 2 is also done in $O(k)$ time, as explained above. Finally, to update the average distance between any pair A, B of distant-2 subtrees, if k is inserted in the subtree A,

$$\Delta_{\{k\}\cup A|B} = \frac{1}{1+|A|}\Delta_{k|B} + \frac{|A|}{1+|A|}\Delta_{A|B}. \tag{9}$$

Step 3 is also done in $O(k)$ time, because there are $O(k)$ pairs of distant-2 subtrees, and all the quantities in the right-hand side of Equation (9) have already been computed. So we build T_k from T_{k-1} in $O(k)$ computational time, and thus the entire computational cost of the construction of T, as we sum over k, is $O(n^2)$. This is much faster than NJ-like algorithms which require $O(n^3)$ operations, and the FITCH [9] program which requires $O(n^4)$ operations. As we shall see, this allows trees with 4,000 taxa to be constructed in few minutes, while this task requires more than an hour for NJ, and is essentially impossible for FITCH on any existing and imaginable machine. This algorithm is called GME (Greedy Minimum Evolution algorithm).

3.2 The BME Addition Algorithm

This algorithm is very similar to GME. Equation (8) simplifies into:

$$L(T') = L + \frac{1}{4}\left[(\Delta^{\mathcal{T}}_{A|C} + \Delta^{\mathcal{T}}_{k|B}) - (\Delta^{\mathcal{T}}_{A|B} + \Delta^{\mathcal{T}}_{k|C})\right]. \tag{10}$$

Step 1 is identical and provides all $\Delta^{\mathcal{T}}_{k|S}$ distances by recursively applying Equation (6). Step 2 simply uses Equation (10) instead of Equation (8) to calculate the relative cost of each possible insertion point. The main difference is the updating performed in Step 3. Equation (10) only requires the balanced average

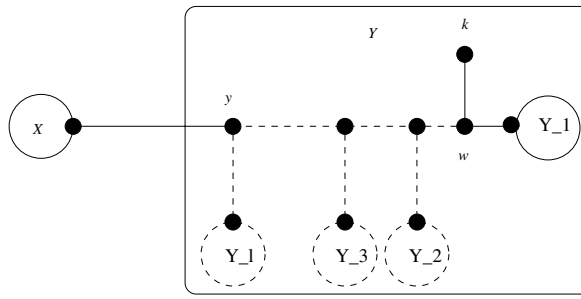

Fig. 4. Calculating balanced average $\Delta^{\mathcal{T}}_{A|B}$ when k is inserted into A

distances between distant-2 subtrees, but to iteratively update these distances, we use (and update) a data structure that contains all distances between every pair of non-intersecting subtrees (as for FASTNNI).

When k is inserted into T_{k-1}, we must calculate the average $\Delta^{\mathcal{T}}_{A|B}$ for any subtree A of T_k which contains k, and any subtree B disjoint from A. We can enumerate all such pairs by considering their respective roots. Let a be the root of A and b the root of B. Regardless of the position of k, any node of T_k could serve as the root b of B. Then, considering a fixed b, any node in the path from b to k could serve as the root a. Thus there are $O(k \times \mathrm{diam}(T_k))$ such pairs.

Given such a pair, A, B, let us consider how we may quickly calculate $\Delta^{\mathcal{T}}_{A|B}$ from known quantities. Consider the situation as depicted in Figure 4. Suppose k is inserted by creating a new node w which pushes the subtree A_1 farther away from B. Suppose there are $(l-1)$ edges in the path from w to a, and the subtrees branching off this path are, in order from w to a, $A_2, A_3, \ldots, A_l$. Then

$$\Delta^{\mathcal{T}}_{A\cup k|B} = 2^{-l}(\Delta^{\mathcal{T}}_{k|B} + \Delta^{\mathcal{T}}_{A_1|B}) + \sum_{i=2}^{l} 2^{-(l+1-i)}\Delta^{\mathcal{T}}_{A_i|B}.$$

However, we already know the value of

$$\Delta^{\mathcal{T}}_{A|B} = 2^{-(l-1)}\Delta^{\mathcal{T}}_{A_1|B} + \sum_{i=2}^{l} 2^{-(l+1-i)}\Delta^{\mathcal{T}}_{A_i|B}.$$

Thus

$$\Delta^{\mathcal{T}}_{A\cup k|B} = \Delta^{\mathcal{T}}_{A|B} + 2^{-l}(\Delta^{\mathcal{T}}_{k|B} - \Delta^{\mathcal{T}}_{A_1|B}).$$

The upper bound on the number of pairs is worst when T_k is a chain, with k inserted at one end. In this case, the diameter is proportional to k and both the number of distances to update and the bound are proportional to k^2. However, the diameter is usually much lower. We repeat the arguments of Section 2.2: if we assume a Yule-Harding speciation process, the expected diameter is $O(\log(k))$ [8], which implies an average complexity of the updating step in $O(k\log(k))$, while a uniform distribution on phylogenetic trees yields an expected diameter of $O(\sqrt{k})$.

Therefore, the complexity of the whole insertion algorithm is $O(n^3)$ in the worst case, and $O(n^2 \log(n))$ (or $O(n^2\sqrt{n})$) in practice. As we shall see in the next section, this is still less than NJ and allows trees with thousands of taxa to be constructed within a reasonable amount of time.

4 Results

4.1 Protocol

We used simulations based on random trees. Parameter values were chosen so as to cover the features of most real data sets, as revealed by the compilation of the phylogenies published in the journal *Systematic Biology* during the last few years. This approach induces much smaller contrasts between the tested methods than those based on model trees (e.g., [2,12]). Indeed, model trees are generally used to emphasize a given property of the studied methods, for example their performance when the molecular clock is strongly violated. Thus, model trees are often extreme and their use tends to produce strong and possibly misleading differences between the tested methods. On the other hand, random trees allow comparisons with a large variety of tree shapes and evolutionary rates, and provide a synthetic and more realistic view of the average performances.

We used 24- and 96-taxon trees, and 2000 trees per size. For each of these trees, a true phylogeny, denoted as T, was first generated using the stochastic speciation process described by Kuhner and Felsenstein [17], which corresponds to the usual Yule-Harding distribution on trees [15,30]. Using this generating process makes T ultrametric (or molecular clock-like). This hypothesis does not hold in most biological data sets, so we created a deviation from the molecular clock. Each edge length of T was multiplied by $1.0 + \mu X$, where X followed the standard exponential distribution ($P(X > \eta) = e^{-\eta}$) and μ was a tuning factor to adjust the deviation from molecular clock; μ was set to 0.8 with 24 taxa and to 0.6 with 96 taxa. The average ratio between the mutation rate in the fastest evolving lineage and the rate in the slowest evolving lineage was then equal to about 2.0 with both tree sizes. With 24 taxa, the smallest value (among 2000) of this ratio was equal to about 1.2 and the largest to 5.0 (1.0 corresponds to the strict molecular clock), while the standard deviation was approximately 0.5. With 96 taxa, the extreme values became 1.3 and 3.5, while the standard deviation was 0.33.

These (2×2000) trees were then rescaled to obtain "slow", "moderate" and "fast" evolutionary rates. With 24 taxa, the branch length expectation was set to 0.03, 0.06 and 0.15 mutations per site, for the slow, moderate and fast conditions, respectively. With 96 taxa, we had 0.02, 0.04 and 0.10 mutations per site, respectively. For both tree sizes, the average maximum pairwise divergence was of about 0.2, 0.4 and 1.0 substitutions per site, with a standard deviation of about 0.07, 0.14 and 0.35 for 24 taxa, and of about 0.04, 0.08 and 0.20 for 96 taxa. These values are in good accordance with real data sets. The maximum pairwise divergence is rarely above 1.0 due to the fact that multiple alignment from highly divergent sequences is simply impossible. Moreover, with such distance value, any correction formula, e.g. Kimura's (1980), becomes very doubtful,

due to our ignorance of the real substitution process and to the fact that the larger the distance the higher the gap between estimates obtained from different formulae. The medium condition ($\sim$0.4) corresponds to the most favorable practical setting, while in the slow condition (from $\sim$0.1 to $\sim$0.3) the phylogenetic signal is only slightly perturbed by multiple substitutions, but it can be too low, with some short branches being not supported by any substitution.

SeqGen [21] was used to generate the sequences. For each tree T (among $3 \times 2 \times 2000$), these sequences were obtained by simulating an evolving process along T according to the Kimura [16] two parameter model with a transition/transversion ratio of 2.0. The sequence length was set to 500 sites. Finally, DNADIST from the PHYLIP package [10] was used to compute the pairwise distance matrices, assuming the Kimura model with known transition/transversion ratio. The data files are available on our web page (see title page).

Every inferred tree, denoted as $\widehat{T}$, was compared to the true phylogeny T (i.e., that used to generate the sequences and then the distance matrix) with a topological distance equivalent to Robinson and Foulds' [22]. This distance is defined by the proportion of internal edges (or bipartitions) which are found in one tree and not in the other one. This distance varies between 0.0 (both topologies are identical) and 1.0 (they do not share any internal edge). The results were averaged over the 2000 test sets for each tree size and evolutionary rate. Finally, to compare the various methods to NJ, we measured the relative error reductions $(P_M - P_{NJ})/P_{NJ}$, where M is any tested method different from NJ and P_X is the average topological distance between $\widehat{T}$ and T when using method X. This measure highlights the performance difference between the related, distance-based, tested methods.

4.2 Phylogeny Estimation Algorithm Comparison

We used a variety of different algorithms to try to reconstruct the original tree, given the matrix of estimated distances. We used the NJ [25] program from the PAUP package [27], with and without the BIONJ [12] flag; WEIGHBOR version 1.2 [2], available at `http://www.t10.lanl.gov/billb/weighbor/`, the FITCH [9] program from the PHYLIP package [10]; the Harmonic Greedy Triplet (HGT/FP) program, provided by Miklos Csűrös [6]; GME and BME. Also, we used output topologies from other programs as input for FASTNNI and BNNI. GME, BME, FASTNNI, and BNNI will be available in the near future at our web page, or via ftp at `ftp://ncbi.nlm.nih.gov/~pub/desper/ME`.

We also measured how far from the true phylogeny one gets with NNIs. This served as a measure of the limitation of each of the minimum evolution frameworks, as well as a performance index for evaluating our algorithms. Ordinarily, the (OLS or balanced) minimum evolution criterion will not, in fact, observe a minimum value at the true phylogeny. So, starting from the true phylogeny and running FASTNNI or BNNI, we end up with a tree with a significant proportion of false edges. When this proportion is high, the corresponding criterion can be seen as poor regarding topological accuracy. This proportion represents the best possible topological accuracy that can be achieved by optimizing the considered

Table 1. Topological accuracy for 24-taxon trees at various rates of evolution

	slow rate			moderate rate		
	w/o NNIs	FASTNNI	BNNI	w/o NNIs	FASTNNI	BNNI
True Tree		.109 -1.6%	.104 -6.2%		.092 3.7%	.083 -5.8%
FITCH	.109 -1.9%	.113 2.0%	.107 -3.4%	.085 -4.9%	.094 6.0%	.085 -4.0%
WEIGHBOR	.109 -1.8%	.112 1.7%	.107 -3.0%	.085 -4.3%	.094 6.2%	.085 -4.0%
BIONJ	.111 -0.3%	.113 2.0%	.107 -3.6%	.087 -2.0%	.094 6.5%	.085 -4.2%
NJ	.111 0%	.113 2.0%	.107 -3.5%	.088 0%	.094 6.6%	.085 -4.0%
BME	.118 7.1%	.113 1.9%	.107 -2.8%	.100 13%	.094 6.3%	.084 -4.9%
GME	.122 10%	.113 2.1%	.107 -3.4%	.107 21%	.095 7.1%	.084 -4.8%
HGT/FP	.334 202%	.112 1.1%	.107 -2.9%	.326 268%	.095 7.5%	.088 -0.2%

	fast rate		
	w/o NNIs	FASTNNI	BNNI
True Tree		.088 6.5%	.076 -8.3%
FITCH	.076 -7.8%	.090 8.8%	.077 -7.0%
WEIGHBOR	.077 -6.8%	.089 8.2%	.077 -6.9%
BIONJ	.079 -3.6%	.090 9.0%	.077 -6.6%
NJ	.082 0%	.090 9.1%	.077 -6.9%
BME	.098 19%	.090 9.1%	.076 -7.1%
GME	.105 28%	.090 9.8%	.076 -7.6%
HGT/FP	.329 300%	.090 9.8%	.083 0.8%

criterion, since we would not expect any algorithm optimizing this criterion in the whole tree space to find a tree closer from the true phylogeny than the tree that is obtained by "optimizing" the true phylogeny itself.

Results are displayed in Table 1 and Table 2. The first number indicates the average topological distance between the inferred tree and the true phylogeny. The second number provides the percentage of relative difference in topological distance between the method considered and NJ (see above); the more negative this percentage, the better the method was relative to NJ.

The performances of the basic algorithms (1st columns) are strongly correlated with the number of computations that they perform. Both $O(n^2)$ algorithms are clearly worse than NJ. BME (in $O(n^2 \log(n))$) is still worse than NJ, but becomes very close with 96 taxa, which indicates the strength of the balanced minimum evolution framework. BIONJ, in $O(n^3)$ like NJ and having identical computing times, is slightly better than NJ in all conditions, while WEIGHBOR, also in $O(n^3)$ but requiring complex numerical calculations, is better than BIONJ. Finally, FITCH, which is in $O(n^4)$, is the best with 24 taxa, but was simply impossible to evaluate with 96 taxa (see the computing times below).

After FASTNNI (2nd columns), we observe that the output topology does not depend much on the input topology. Even the deeply flawed HGT becomes close to NJ, which indicates the strength of NNIs. However, except for the true phylogeny in some cases, this output is worse than NJ. This confirms our previous results [13], which indicated that the OLS version of minimum evolution is reasonable but not excellent for phylogenetic inference. But this phenomenon

Table 2. Topological accuracy for 96-taxon trees at various rates of evolution

	slow rate			moderate rate		
	w/o NNIs	FASTNNI	BNNI	w/o NNIs	FASTNNI	BNNI
True Tree		.172 -5.6%	.167 -8.8%		.132 -3.0%	.115 -15.4%
WEIGHBOR	.178 -2.5%	.181 -0.7%	.173 -5.2%	.129 -5.4%	.137 0.5%	.118 -13.0%
BIONJ	.180 -0.9%	.182 -0.3%	.173 -5.1%	.134 -1.9%	.138 1.3%	.118 -13.0%
NJ	.183 0%	.182 -0.2%	.173 -5.2%	.136 0%	.139 1.8%	.119 -12.9%
BME	.186 1.9%	.181 -0.6%	.173 -5.3%	.137 1.0%	.138 1.1%	.118 -13.2%
GME	.199 8.8%	.183 0.3%	.173 -5.3%	.158 16%	.140 2.7%	.118 -13.2%
HGT/FP	.512 185%	.185 1.5%	.175 -4.3%	.480 253%	.143 5.2%	.123 -.9.3%

	fast rate		
	w/o NNIs	FASTNNI	BNNI
True Tree		.115 0.6%	.088 -23.4%
WEIGHBOR	.103 -10%	.119 3.8%	.091 -21.0%
BIONJ	.112 -2.5%	.121 5.1%	.090 -21.7%
NJ	.115 0%	.121 5.5%	.090 -21.3%
BME	.117 1.8%	.120 4.4%	.090 -21.4%
GME	.144 25%	.122 6.3%	.091 -21.1%
HGT/FP	.465 306%	.126 9.4%	.098 -14.7%

is much more visible with 24 than with 96 taxa, so we expect the very fast GME+FASTNNI $O(n^2)$ combination to be equivalent to NJ with large n.

After BNNI (3rd columns), all initial topologies become better than NJ. With 24 taxa, the performance is equivalent to that of FITCH, while with 96 taxa the results are far better than those of WEIGHBOR, especially in the Fast condition where BNNI improves NJ by 21%, against 10% for WEIGHBOR. These results are somewhat unexpected, since BNNI has a low $O(n^2 + pn\log(n))$ average time complexity. Moreover, as explained above, we do not expect very high contrast between any inference method and NJ, due to the fact that our data sets represent a large variety of realistic trees and conditions, but do not contain extreme trees, notably concerning the maximum pairwise divergence. First experiments indicate that maximum parsimony is close to BNNI in the Slow and Moderate conditions, but worse in the Fast condition, the contrast between these methods being in the same range as those reported in Table 1 and 2. The combination of BNNI with BME or GME can thus be seen as remarkably efficient and accurate. Moreover, regarding the true phylogeny after BNNI, it appears that little gain could be expected by further optimizing this criterion, since our simple optimization approach is already close to these upper values.

4.3 Computing Times

To compare the actual running speeds of these algorithms, we tested them on a Sun Enterprise E4500/E5500, with ten 400 MHz processors and 7 GB of memory, running the Solaris 8 operating system. Table 3 summarizes the average computational times (in hours:minutes:seconds) required by the various programs to

Table 3. Average computational time (HH:MM:SS) for reconstruction algorithms

Algorithm	24 taxa	96 taxa	1000 taxa	4000 taxa
GME	.02522	.07088	8.366	3:11.01
GME + FASTNNI	.02590	.08268	9.855	3:58.79
GME + BNNI	.02625	.08416	11.339	6:02.10
HGT/FP	.02518	.13491	13.808	3:33.14
BME	.02534	.08475	19.231	12:14.99
BME + BNNI	.02557	.08809	19.784	12:47.45
NJ	.06304	.16278	21.250	20:55.89
BIONJ	.06528	.16287	21.440	20:55.64
WEIGHBOR	.42439	26.88176	*****	*****
FITCH	4.37446	*****	*****	*****

Table 4. Average number of NNIs performed

Algorithm	24 taxa	96 taxa	1000 taxa	4000 taxa
GME + FASTNNI	1.244	8.446	44.9	336.50
GME + BNNI	1.446	11.177	59.1	343.75
BME + BNNI	1.070	6.933	29.1	116.25

build phylogenetic trees. The leftmost two columns were averaged over two thousand 24- and 96-taxon trees, the third over ten 1000-taxon trees, and the final column over four 4000-taxon trees. Stars indicate entries where the algorithm was deemed to be feasible

FITCH would take approximately 25 minutes to make one 96-taxon tree, a speed which makes it impractical for real applications, because they often include bootstrapping which requires the construction of a large number of trees. As WEIGHBOR's running time increased more than 60-fold when moving from 24-taxon trees to 96-taxon trees, we judged it infeasible to run WEIGHBOR on even one 1000-taxon tree.

Table 3 confirms that all of our algorithms are faster than NJ, with the improvement becoming notably impressive when thousands of taxa are dealt with. Furthermore, we suspect that implementation refinements such as those used in PAUP's NJ could be used to make our algorithms still much faster.

Table 4 contains the number of NNIs performed by each of the three combinations which appear in Table 3. Not surprisingly, the largest number of NNIs was consistently required when the initial topology was made to minimize the OLS ME criterion, but NNIs were chosen to minimize the balanced ME criterion (i.e., GME + BNNI). This table shows the overall superiority of the BME tree over the GME tree, when combined with BNNI. In all of the cases considered, the average number of NNIs considered for each value of n was considerably less than n itself.

5 Discussion

We have presented a new distance-based approach to phylogeny estimation which is faster than most distance algorithms currently in use. The current most popu-

lar fast algorithm is Neighbor-Joining, an $O(n^3)$ algorithm. Our greedy ordinary least-squares minimum evolution tree construction algorithm (GME) runs at $O(n^2)$, the size of the input matrix. Although the GME tree is not quite as accurate as the NJ tree, it is a good starting point for nearest neighbor interchanges (NNIs). The combination of GME and FASTNNI, which achieves NNIs according to the ordinary least-squares criterion, also in $O(n^2)$ time, has a topological accuracy close to that of NJ, especially with large numbers of taxa and moderate or slow rates.

However, the balanced minimum evolution framework appears much more appropriate for phylogenetic inference than the ordinary least-squares version. This is likely due to the fact that it gives less weight to the topologically long distances, i.e. those containing numerous edges, while the ordinary least-squares method puts the same confidence on each distance, regardless of its length. Even when the usual and topological lengths are different, they are strongly correlated. The balanced minimum evolution framework is thus conceptually closer to weighted least-squares [11], which is more appropriate than ordinary least-squares for evolutionary distances estimated from sequences.

The balanced NNI algorithm (BNNI) achieves a good level of performance, superior to that of NJ, BIONJ and WEIGHBOR. BNNI is an $O(n^2 + np \times \mathrm{diam}(T))$ algorithm, where $\mathrm{diam}(T)$ is the diameter of the inferred tree, and p the number of swaps performed. With $p \times \mathrm{diam}(T) = O(n)$ for most data sets, the combination of GME and BNNI effectively gives us an $O(n^2)$ algorithm with high topological accuracy.

Acknowledgements

Special thanks go to Stephane Guindon, who generated the simulated data sets used in Section 4.

References

1. Aldous, D.J.: Stochastic models and descriptive statistics for phylogenetic trees from Yule to today. Statist. Sci. **16** (2001) 23–34
2. Bruno, W.J., Socci, N.D., Halpern, A.L.: Weighted neighbor joining: A likelihood-based approach to distance-based phylogeny reconstruction. Mol. Biol. Evol. **17** (2000) 189–197
3. Bryant, D., Waddell, P.: Rapid evaluation of least-squares and minimum-evolution criteria on phylogenetic trees. Mol. Biol. Evol. **15** (1998) 1346–1359
4. Bulmer, M.: Use of the method of generalized least squares in reconstructing phylogenies from sequence data. Mol. Biol. Evol. **8** (1991) 868–883
5. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to algorithms. MIT Press, Cambridge, MA (2000)
6. Csűrós, M.: Fast recovery of evolutionary trees with thousands of nodes. Journal of Computational Biology **9** (2002) 277–297
7. Denis, F., Gascuel, O.: On the consistency of the minimum evolution principle of phylogenetic inference. Discr. Appl. Math. **In press** (2002)

8. Erdös, P.L., Steel, M., Székély, L., Warnow, T.: A few logs suffice to build (almost) all trees: Part II. Theo. Comp. Sci. **221** (1999) 77–118
9. Felsenstein, J.: An alternating least-squares approach to inferring phylogenies from pairwise distances. Syst. Biol. **46** (1997) 101–111
10. Felsenstein, J.: PHYLIP — Phylogeny Inference Package (Version 3.2). Cladistics **5** (1989) 164–166
11. Fitch, W.M., Margoliash, E.: Construction of phylogenetic trees. Science **155** (1967) 279–284
12. Gascuel, O.: BIONJ: an improved version of the NJ algorithm based on a simple model of sequence data. Mol. Biol. Evol. **14** (1997) 685–695
13. Gascuel, O.: On the optimization principle in phylogenetic analysis and the minimum-evolution criterion. Mol. Biol. Evol. **17** (2000) 401–405
14. Gascuel, O., Bryant, D., Denis, F.: Strengths and limitations of the minimum evolution principle. Syst. Biol. **50** (2001) 621–627
15. Harding, E.: The probabilities of rooted tree-shapes generated by random bifurcation. Adv. Appl. Probab. **3** (1971) 44–77
16. Kimura, M.: A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. J. Mol. Evol. **16** (1980) 111,120
17. Kuhner, M.K., Felsenstein, J.: A simulation comparison of phylogeny algorithms under equal and unequal rates. Mol. Biol. Evol. **11** (1994) 459–468
18. McKenzie, A., Steel, M.: Distributions of cherries for two models of trees. Math. Biosci. **164** (2000) 81–92
19. Nei, M., Jin, L.: Variances of the average numbers of nucleotide substitutions within and between populations. Mol. Biol. Evol. **6** (1989) 290–300
20. Pauplin, Y.: Direct calculation of a tree length using a distance matrix. J. Mol. Evol. **51** (2000) 41–47
21. Rambaut, A., Grassly, N.C.: Seq-Gen: An application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees. Computer Applications in the Biosciences **13** (1997) 235–238
22. Robinson, D., Foulds, L.: Comparison of phylogenetic trees. Math. Biosci. **53** (1981) 131–147
23. Rzhetsky, A., Nei, M.: A simple method for estimating and testing minimum-evolution trees. Mol. Biol. Evol. **9** (1992) 945–967
24. Rzhetsky, A., Nei, M.: Theoretical foundation of the minimum-evolution method of phylogenetic inference. Mol. Biol. Evol. **10** (1993) 1073–1095
25. Saitou, N., Nei, M.: The neighbor-joining method: A new method for reconstructing phylogenetic trees. Mol. Biol. Evol. **4** (1987) 406–425
26. Sneath, P.H.A., Sokal, R.R. In: Numerical Taxonomy. W.K. Freeman and Company, San Francisco (1973) 230–234
27. Swofford, D.: PAUP—Phylogenetic Analysis Using Parsimony (and other methods), Version 4.0 (1996)
28. Swofford, D.L., Olsen, G.J., Waddell, P.J., Hillis, D.M.: Phylogenetic inference. In Hillis, D., Moritz, C., Mable, B., eds.: Molecular Systematics. Sinauer, Sunderland, MA (1996) 407–514
29. Vach, W.: Least squares approximation of addititve trees. In Opitz, O., ed.: Conceptual and numerical analysis of data. Springer-Verlag, Berlin (1989) 230–238
30. Yule, G.: A mathematical theory of evolution, based on the conclusions of Dr. J. C. Willis. Philos. Trans. Roy. Soc. London Ser. B, Biological Sciences **213** (1925) 21–87