Syst. Biol. 52(5):696–704, 2003

ISSN: 1063-5157 print / 1076-836X online
DOI: 10.1080/10635150390235520

A Simple, Fast, and Accurate Algorithm to Estimate Large Phylogenies by Maximum Likelihood

STÉPHANE GUINDON AND OLIVIER GASCUEL

LIRMM, CNRS, 161 Rue Ada, 34392, Montpellier Cedex 5, France; E-mail: gascuel@lirmm.fr (O.G.)

Abstract.—The increase in the number of large data sets and the complexity of current probabilistic sequence evolution models necessitates fast and reliable phylogeny reconstruction methods. We describe a new approach, based on the maximum-likelihood principle, which clearly satisfies these requirements. The core of this method is a simple hill-climbing algorithm that adjusts tree topology and branch lengths simultaneously. This algorithm starts from an initial tree built by a fast distance-based method and modifies this tree to improve its likelihood at each iteration. Due to this simultaneous adjustment of the topology and branch lengths, only a few iterations are sufficient to reach an optimum. We used extensive and realistic computer simulations to show that the topological accuracy of this new method is at least as high as that of the existing maximum-likelihood programs and much higher than the performance of distance-based and parsimony approaches. The reduction of computing time is dramatic in comparison with other maximum-likelihood packages, while the likelihood maximization ability tends to be higher. For example, only 12 min were required on a standard personal computer to analyze a data set consisting of 500 *rbc*L sequences with 1,428 base pairs from plant plastids, thus reaching a speed of the same order as some popular distance-based and parsimony algorithms. This new method is implemented in the PHYML program, which is freely available on our web page: http://www.lirmm.fr/w3ifa/MAAS/. [Algorithm; computer simulations; maximum likelihood; phylogeny; *rbc*L; RDPII project.]

The size of homologous sequence data sets has increased dramatically in recent years, and many of these data sets now involve several hundreds of taxa. Moreover, current probabilistic sequence evolution models (Swofford et al., 1996; Page and Holmes, 1998), notably those including rate variation among sites (Uzzell and Corbin, 1971; Jin and Nei, 1990; Yang, 1996), require an increasing number of calculations. Therefore, the speed of phylogeny reconstruction methods is becoming a significant requirement and good compromises between speed and accuracy must be found.

The maximum likelihood (ML) approach is especially accurate for building molecular phylogenies. Felsenstein (1981) brought this framework to nucleotide-based phylogenetic inference, and it was later also applied to amino acid sequences (Kishino et al., 1990). Several variants were proposed, most notably the Bayesian methods (Rannala and Yang 1996; and see below), and the discrete Fourier analysis of Hendy et al. (1994), for example. Numerous computer studies (Huelsenbeck and Hillis, 1993; Kuhner and Felsenstein, 1994; Huelsenbeck, 1995; Rosenberg and Kumar, 2001; Ranwez and Gascuel, 2002) have shown that ML programs can recover the correct tree from simulated data sets more frequently than other methods can. Another important advantage of the ML approach is the ability to compare different trees and evolutionary models within a statistical framework (see Whelan et al., 2001, for a review). However, like all optimality criterion–based phylogenetic reconstruction approaches, ML is hampered by computational difficulties, making it impossible to obtain the optimal tree with certainty from even moderate data sets (Swofford et al., 1996). Therefore, all practical methods rely on heuristics that obtain near-optimal trees in reasonable computing time. Moreover, the computation problem is especially difficult with ML, because the tree likelihood not only depends on the tree topology but also on numerical parameters, including branch lengths. Even computing the optimal values of these parameters on a single tree is not an easy task, particularly because of possible local optima (Chor et al., 2000).

The usual heuristic method, implemented in the popular PHYLIP (Felsenstein, 1993) and PAUP* (Swofford, 1999) packages, is based on hill climbing. It combines stepwise insertion of taxa in a growing tree and topological rearrangement. For each possible insertion position and rearrangement, the branch lengths of the resulting tree are optimized and the tree likelihood is computed. When the rearrangement improves the current tree or when the position insertion is the best among all possible positions, the corresponding tree becomes the new current tree. Simple rearrangements are used during tree growing, namely "nearest neighbor interchanges" (see below), while more intense rearrangements can be used once all taxa have been inserted. The procedure stops when no rearrangement improves the current best tree. Despite significant decreases in computing times, notably in fastDNAml (Olsen et al., 1994), this heuristic becomes impracticable with several hundreds of taxa. This is mainly due to the two-level strategy, which separates branch lengths and tree topology optimization. Indeed, most calculations are done to optimize the branch lengths and evaluate the likelihood of trees that are finally rejected.

New methods have thus been proposed. Strimmer and von Haeseler (1996) and others have assembled four-taxon (quartet) trees inferred by ML, in order to reconstruct a complete tree. However, the results of this approach have not been very satisfactory to date (Ranwez and Gascuel, 2001). Ota and Li (2000, 2001) described

NJML, an algorithm that is a the continuation of that of Adachi and Hasegawa (1996), combining neighbor joining (NJ) and ML. NJML first builds a tree by the fast distance-based NJ algorithm (Saitou and Nei, 1987); the unreliable branches of this initial tree are then detected using the bootstrap procedure (Felsenstein, 1985) and resolved by computing the branch lengths and the likelihood of alternative resolutions. In this way, NJML avoids spending time on the well-supported parts the tree. Ranwez and Gascuel (2002) proposed another combination of distance-based and ML approaches, with the computation speed of the former and topological accuracy midway between both. This method relies on triplets of taxa, sharing a divide-and-conquer strategy with the quartet approach.

Stochastic optimization is another attractive strategy. Markov chain Monte Carlo (MCMC) algorithms are widely used by Bayesian methods (Li, 1996; Mau, 1996; Rannala and Yang, 1996; Simon and Larget, 2000; Huelsenbeck and Ronquist, 2001). This approach has several advantages. It generates a number of trees (instead of a single one) and allows estimation of their posterior probabilities, as well as that of various (e.g., clade) hypotheses. Moreover, these posteriors are based on the integrated likelihood, i.e., the likelihood averaged over branch lengths and model parameters values. In this way, Bayesian approaches take into account sources of uncertainty due to numerical values that the standard ML methods do not. Salter and Pearl (2001) described a simulated annealing algorithm that is substantially faster than DNAML (Felsenstein, 1993) and PAUP*. This algorithm simultaneously perturbs branch lengths and tree topology and then accelerates the computations in comparison with standard hill climbing. The genetic algorithms recently proposed by Lewis (1998) and Lemmon and Milinkovitch (2002) are specially efficient. These algorithms navigate in the tree space, randomly perturbing a population of trees by modifying their branch lengths and topology, combining these trees to obtain better trees, and selecting the best trees until an optimum is reached. In this way, large phylogenies containing hundreds of taxa are obtained in few hours on a standard computer (Lemmon and Milinkovitch, 2002). Moreover, due to the fact that these approaches build a number of trees, they permit approximation of the posterior probabilities of trees or clades. Finally, an efficient parallel implementation of the genetic optimization approach has been described by Brauer et al. (2002).

The hill-climbing principle is usually considered faster than stochastic optimization and sufficient for numerous combinatorial optimization problems (Aarts and Lenstra, 1997), particularly when the function to be optimized is a simplification of the overall reality of the problem at hand. This is clearly the case with phylogenetic reconstruction because we do not know the real substitution process that occurred. Moreover, despite the ever increasing size of databases, the length of sequences used to build phylogenies is not limitless. Sampling variations in the likelihood are inevitable, even when the chosen evolutionary model fits the sequences well.

Here, we present a new, simple hill-climbing algorithm that avoids the limits of the previous ones. The tree topology and branch lengths of a unique tree are simultaneously and progressively modified so that the tree likelihood increases at each step until an optimum is reached. During this process, we can also adjust the model parameters, such as the transition/transversion ratio or the gamma shape parameter accounting for rate variation among sites. This algorithm is implemented in the PHYML package, which is faster than other existing ML programs, including MetaPIGA (Lemmon and Milinkovitch, 2002). Here, we present this algorithm and then compare PHYML with other packages using extensive computer simulations and two large data sets comprising 218 and 500 taxa. It is shown that PHYML is at least equivalent to other ML programs, both in terms of topological accuracy and likelihood maximization, while having a speed similar to that of some popular distance-based and parsimony methods. Such a speed not only makes possible the inference of very large trees, but also greatly facilitates the building of multiple trees in bootstrap analysis.

Method and Algorithms

The principle is to start from an initial tree constructed by a fast distance-based algorithm and to improve it. We first present how every branch can be adjusted independently of the other branches, to maximize tree likelihood. We then show how this process extends to tree swapping, without much more computation. Branch-length optimization and tree swapping define possible modifications of the current tree, and we explain how these modifications are selected and combined. Finally, we present the whole method, including model parameter optimization. For the sake of simplicity, the method is described for nucleotide sequences, but it can also be applied to proteins.

Branch-Length Optimization

Let e be the branch under consideration and l its current length. U and V are the subtrees at the two extremities of e, with roots u and v (Fig. 1a). We assume that these subtrees are fixed. We then define the conditional likelihood $L(i = h|U)$ as the probability of observing the data at site i at the tips of U, given that node u has nucleotide h. When U is reduced to taxon u, $L(i = h|U)$ is equal to 1 if site i of taxon u has nucleotide h, and 0 otherwise. $L(i = h|V)$ has the same meaning when V (and v) replaces U (and u). We also assume, as usual, that the sequence substitution model is homogeneous, stationary, and time reversible. The a priori probability of nucleotide h is then denoted as π_h, and $P_{hh'}(l)$ is the probability for the nucleotide h to become h' in interval l. With this notation, and assuming that all sites independently evolve at the same rate, the likelihood of the whole tree is equal to the product of the site likelihoods and may be written

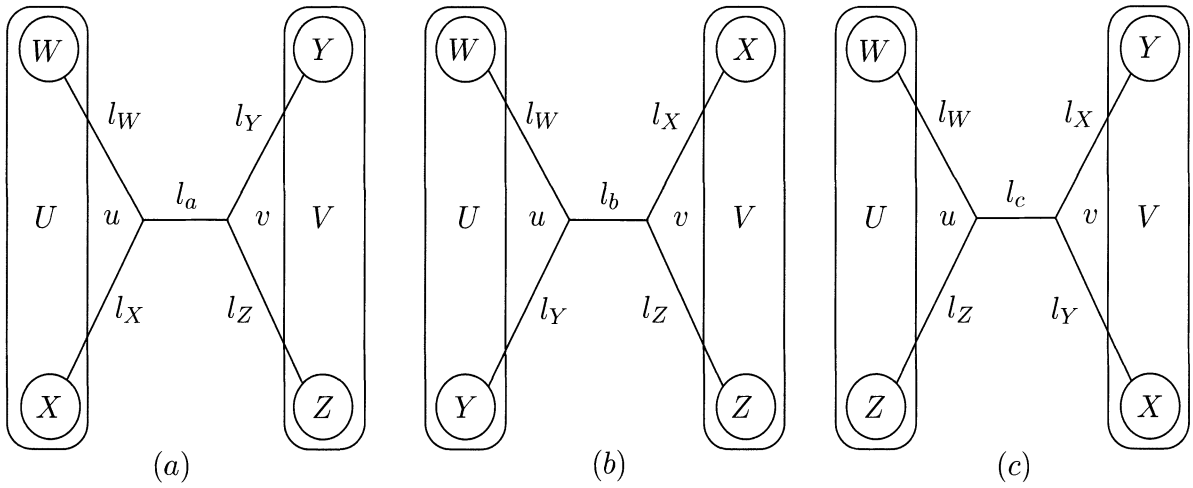

FIGURE 1. The three alternative topological configurations around an internal branch. W, X, Y, and Z are four subtrees, and l_w, l_x, l_y, and l_z are the lengths of the four branches connected to the roots of W, X, Y, and Z, respectively. These lengths are the same in the three topological configurations. U and V are the subtrees on the left and right, respectively, of the internal branch, and l_a, l_b, and l_c are the internal branch lengths that maximize the likelihoods of the corresponding phylogenies.

(Adachi and Hasegawa, 1996) as

$$L = \prod_i \sum_{h,h' \in \{A,C,G,T\}} \pi_h L(i = h \mid U) L(i = h' \mid V) P_{hh'}(l). \quad (1)$$

This equation is easily adapted to incorporate site-to-site variation using a discrete rate (e.g., gamma) distribution. The full likelihood of a given site is then obtained by summing over rate categories the likelihoods of the site according to each rate weighted by the probability that the site is drawn from each category (Yang, 1994). Equation 1 applies to both internal and external branches. In the latter case, either U or V is reduced to a single extant taxon. When the conditional likelihoods $L(i = h|U)$ and $L(i = h|V)$ are known for every site, computation of Equation 1 is fast and requires $\mathcal{O}(s)$ time, where s denotes the sequence length, i.e., a time proportional to s.

Because U and V are fixed, likelihood L in Equation 1 only depends on l, and we adjust l by maximizing L. This optimization of one parameter function is achieved using Brent's (1973) method. This very simple method does not require function derivatives and in our experiments the computational speed was similar to that of the Newton–Raphson method (Olsen et al., 1994; Felsenstein and Churchill, 1996; Yang, 2000). The optimal length of e is denoted as l_a, and the tree likelihood increases when the current length l is changed into l_a. We denote as L_a the likelihood of tree a when e has length l_a, while subtrees U and V remain identical.

Tree Swapping

Let e now be an internal edge; e then defines four subtrees (Fig. 1). When swapping these subtrees, which corresponds to a nearest neighbor interchange, the initial configuration a is changed into b or c (Fig. 1). Consider configuration b. Subtree U now contains subtrees W and Y, and V contains subtrees X and Z. However, we assume that W, X, Y, and Z are unchanged from configuration a, and branch lengths l_W, l_X, l_Y, and l_Z remain the same. The conditional likelihood of U for any given site i (Felsenstein, 1981) is then equal to

$$L(i = h \mid U) = \left(\sum_{g \in \{A,C,G,T\}} L(i = g \mid W) P_{hg}(l_W) \right) \times \left(\sum_{g \in \{A,C,G,T\}} L(i = g \mid Y) P_{hg}(l_Y) \right), \quad (2)$$

and the conditional likelihood of V is obtained by symmetry from X and Z. Once the conditional likelihoods of U and V have been computed, we adjust the length of e by Equation 1. We thus obtain the likelihood of configuration b, denoted as L_b, and l_b is the corresponding branch length of e. L_c and l_c are defined and obtained in the same way for configuration c. When the conditional likelihoods of W, X, Y, and Z are known, the computation of Equation 2 for all sites is fast and requires $\mathcal{O}(s)$. So computing L_b and L_c has essentially the same cost as computing L_a.

If L_b is larger than L_a and L_c, then configuration b is more likely than the two other configurations. Moreover, the larger the gap between L_b and L_a, the more confident we are in the swap from the current configuration a to configuration b. When L_b is larger than L_a and L_c, we say that e defines b as a possible swap, with score $S = L_b - L_a$. The same holds by symmetry when L_c is larger than L_a and L_b.

Selecting and Combining These Modifications

Changing l into l_a or performing a possible swap increases the tree likelihood. However, edges are not independent, and when simultaneously modifying two edges with values computed as described above we cannot be sure that the tree likelihood will increase. The standard approach is to perform one modification at a time; after each modification, the conditional likelihoods (and even all branch lengths in case of branch swapping) are updated. However, this process is slow because conditional likelihood (and branch length) updating is time consuming.

Our approach involves first independently computing all modifications, i.e., the optimal lengths of all branches and possible swaps around all internal branches, and then simultaneously applying "most" of these modifications to the current tree. This latter step is performed as follows:

1. The possible swaps are ranked according to their scores S. When two possible swaps correspond to two adjacent branches, they have one subtree in common and only the best swap is conserved. We then apply a proportion λ of the remaining swaps to the current tree, starting from the higher values of S. However, the best possible swap is always performed, even for very low λ value.
2. For external branches and internal branches that do not correspond to a possible swap (or that have not been retained in the previous selection), we change the current branch length l into $l + \lambda(l_a - l)$, i.e., we apply a proportion λ of the change that has been computed using Equation 1.
3. Having $\lambda = 1$ would simultaneously apply all possible modifications, whereas $\lambda = 0$ would leave the current tree unchanged. We start with a high λ value but check that the tree likelihood increases. In the (rare) cases where the likelihood decreases, λ is divided by 2, the tree is modified accordingly, and we again check the likelihood. If the likelihood still decreases, λ is divided by 2 again, and the process is repeated until we get a tree with higher likelihood than the current tree. When there are possible swaps and when λ becomes very small, the best swap is the only one to be selected and all branch lengths (except the new one) remain identical, resulting in a tree that is better than the current tree, thus ensuring convergence. When no more possible swaps remain, only branch lengths have to be optimized, and our algorithm becomes close to the global Newton–Raphson algorithm described by Felsenstein and Churchill (1996). This algorithm uses the first and second derivatives of the log-likelihood function, assumes that the Hessian is diagonal, and therefore independently computes the changes for every branch. Moreover, it uses the same λ "safeguard" to ensure convergence and was a source of inspiration when we designed our algorithm. The main differences between both algorithms are that we use the Brent method and then do not compute the derivatives and, most importantly, that our algorithm not only optimizes the branch lengths but also the tree topology. When only branch length optimization is concerned, both algorithms have similar convergence guarantees and can be trapped in local optima, which fortunately are very sparse with real data (Rogers and Swofford, 1999).

In practice, the backward movement by dividing λ is rare. For example, with the first 1,000 data sets in our simulation, λ is divided by 2 only 71 times and only when there are possible swaps; with the 218-taxon ribosomal data set, the backward movement occurs only once. PHYML uses 0.75 as the initial λ value and resets λ to this value after each refinement stage. The initial λ value is not a sensitive parameter; nearly identical trees (but different run times) are obtained with λ in the [0.1, 1.0] range.

Whole Method

We have seen in the previous sections how the possible modifications are computed and how they are combined to refine the current tree. We detail here how these components are incorporated into the complete method, which is described step by step.

1. A pairwise evolutionary distance matrix is computed from the sequences, by an algorithm analogous to DNADIST (Felsenstein, 1993). This step necessitates comparing all sequence pairs and then requires $\mathcal{O}(n^2 s)$ time, where n is the number of taxa.
2. An initial tree is built from this matrix, using BIONJ (Gascuel, 1997). Tests with other distance-based methods led to identical results, so the main criterion at this step is computational speed. BIONJ is just as fast as NJ but is slightly more accurate and requires $\mathcal{O}(n^3)$ time.
3. The conditional likelihoods $L(i = h|U)$ are computed for all sites and every subtree U, as well as the likelihood of the whole tree, using Equations 1 and 2, respectively. These computations are achieved using an algorithm similar to that of Adachi and Hasegawa (1996), which requires $\mathcal{O}(ns)$ time.
4. The values of the free parameters of the substitution model (i.e., the transition/transversion ratio(s) and the gamma shape parameter measuring the variability of substitution rates among sites) are adjusted to increase the likelihood of the starting phylogeny. This adjustment is achieved independently for each parameter using the golden section numerical optimization method (Press et al., 1988). The parameter estimates so obtained are dependent on the starting tree. However, this dependency is slight (Yang, 1996). Moreover, the free parameters are periodically reestimated during the refinement process (every four stages in the current version of PHYML).
5. The current tree is iteratively refined until convergence, as described in the previous section. Each refinement stage involves (a) computing the possible modifications of every branch, (b) applying a λ proportion of these modifications to the current tree, and

(c) checking that the tree likelihood increases and, if necessary, returning to step b with a lower λ value. Step a requires $\mathcal{O}(s)$ time per branch and then $\mathcal{O}(ns)$ for the whole tree. Step b is fast, basically in $\mathcal{O}(n)$. Step c performs likelihood computations as described for step 3, i.e., requiring $\mathcal{O}(ns)$ time. Moreover, after step c all conditional likelihoods have been updated, and then a new refinement stage can start.

6. Tree refinement stops when there are no more possible swaps and when the branch lengths are stable. The current tree is then returned.

The time complexity of model parameter, topology, and branch length optimization (steps 3–6) is then $\mathcal{O}(pns)$, where p basically represents the number of refinement stages that have been performed. Even when this analysis does not clarify some (bounded but significant) parameters, e.g., the number of iterations required by the Brent method to optimize branch lengths, it reveals why our method is so fast. With the 218-taxon data set, p is equal to only 15, and in practice p is always much smaller than n (see Table 1). This explains why our $\mathcal{O}(pns)$ ML optimization has computing time in the same range as distance methods such as NJ, BIONJ, and Weighbor, which require $\mathcal{O}(n^2s + n^3)$ time, including distance estimation.

RESULTS

Computer Simulations

We generated 5,000 random phylogenies, each comprising 40 taxa, using the standard speciation process described by Kuhner and Felsenstein (1994). This process makes the trees molecular clock-like, so we created a deviation from this model by multiplying every branch length by $(1 + X)$, where X followed an exponential distribution with expectation μ. The μ value represents the extent of deviation and was identical within each tree but different from tree to tree and equal to $0.2/(0.001 + U)$, where U was uniformly drawn from [0, 1]. The smaller the U, the larger the μ and the larger the deviation from the molecular clock. Tree length was rescaled by multiplying every branch length by $(0.4 + 8.6V)/T$, where T is the total tree length and V was identical within each tree but different from tree to tree and uniform in [0, 1]. This scaling made the tree length uniformly distributed in the [0.4, 9.0] range.

Phylogenies generated in this way have a broad variety of deviations from the molecular clock and various evolutionary rates. The branch length mean is equal to 0.06 substitutions/site, with the 5%, 25%, 50%, 75%, and 95% quantiles about equal to 0.0015, 0.01, 0.03, 0.07, and 0.20, respectively. The ratio of the length of the longest to the length of the shortest lineages measures the deviation from the molecular clock, with the perfect molecular clock having a ratio of 1. The mean of this ratio, among the 5,000 phylogenies, is equal to 3.4, with the 5%, 25%, 50%, 75%, and 95% quantiles about equal to 1.3, 2.3, 3.2, 4.2, and 6.4, respectively. These values come from an analysis of substitution rates in various organisms (Page and Holmes, 1998) and of numerous recently published phylogenies; they should then cover the features of almost all real data sets, even when the extreme values, notably the highest divergence rates, are likely rare.

Sequences 500 base pairs (bp) in length were generated from these phylogenies using Seq-Gen (Rambaut and Grassly, 1997) under the Kimura two-parameter (K2P) model (Kimura, 1980), with a transition/transversion ratio of 2.0. The 5,000 data sets (phylogenies and sequences) obtained in this way are available on our web page.

These 5,000 data sets were generated without rate heterogeneity across sites, even when this is clearly an important parameter for accurate phylogeny estimation from most sequence sets. Indeed, the ML programs we tested (see below) deal with rate heterogeneity in a different way or simply do not take this parameter into account, which makes comparison impossible. In fact, PAUP* is the only program using the same discrete gamma distribution (Yang, 1994) as PHYML. To compare these two programs in this setting and to check the properties of PHYML, we then generated 1,000 other data sets from the first 1,000 trees, with the same sequence length and Kimura model, plus a four-category discrete gamma distribution of parameter 1.0, which corresponds to moderate heterogeneity (Yang, 1996).

Topological Accuracy

Using these data sets, we compared PHYML with numerous other packages: NJ, Weighbor 1.2 (Bruno et al., 2000), DNAPARS 3.5 (Felsenstein, 1993), NJML+ (Ota and Li, 2001), fastDNAml (Olsen et al., 1994), PAUP* 4.0beta (Swofford, 1999), and MrBayes 2.01 (Huelsenbeck and Ronquist, 2001). NJ and Weighbor are distance-based methods and were combined with DNADIST 3.6 (Felsenstein, 1993), DNAPARS uses the parsimony principle, and the other programs implement ML approaches. We did not test MetaPIGA (Lemmon and Milinkovitch, 2002) at this stage because no batch version allowing for multiple data sets was available. Moreover, for computing time reasons, PAUP* and MrBayes were only run on the first 1,000 of the 5,000 data sets, and only nearest neighbor interchanges were used in PAUP*. MrBayes was run with a random starting tree, 30,000 generations, a sampling frequency of 10, and the resulting consensus phylogeny was built from the last 1,500 trees. The options for NJML were bootstrap threshold = 90% and composite mode for likelihood computation. Other packages were used with default options, supplying the simulation settings (e.g., the sequence length or the transition/transversion ratio) when required.

The topological accuracy of these various methods was measured on the 5,000 data sets (without rate heterogeneity) by the standard Robinson and Foulds (1979) distance between the inferred tree and the true tree. This distance corresponds to the proportion of internal branches that are found in one tree and not in the other one. Its value ranges from 0.0 (both topologies are identical) to 1.0 (they do not share any branch in common). The value

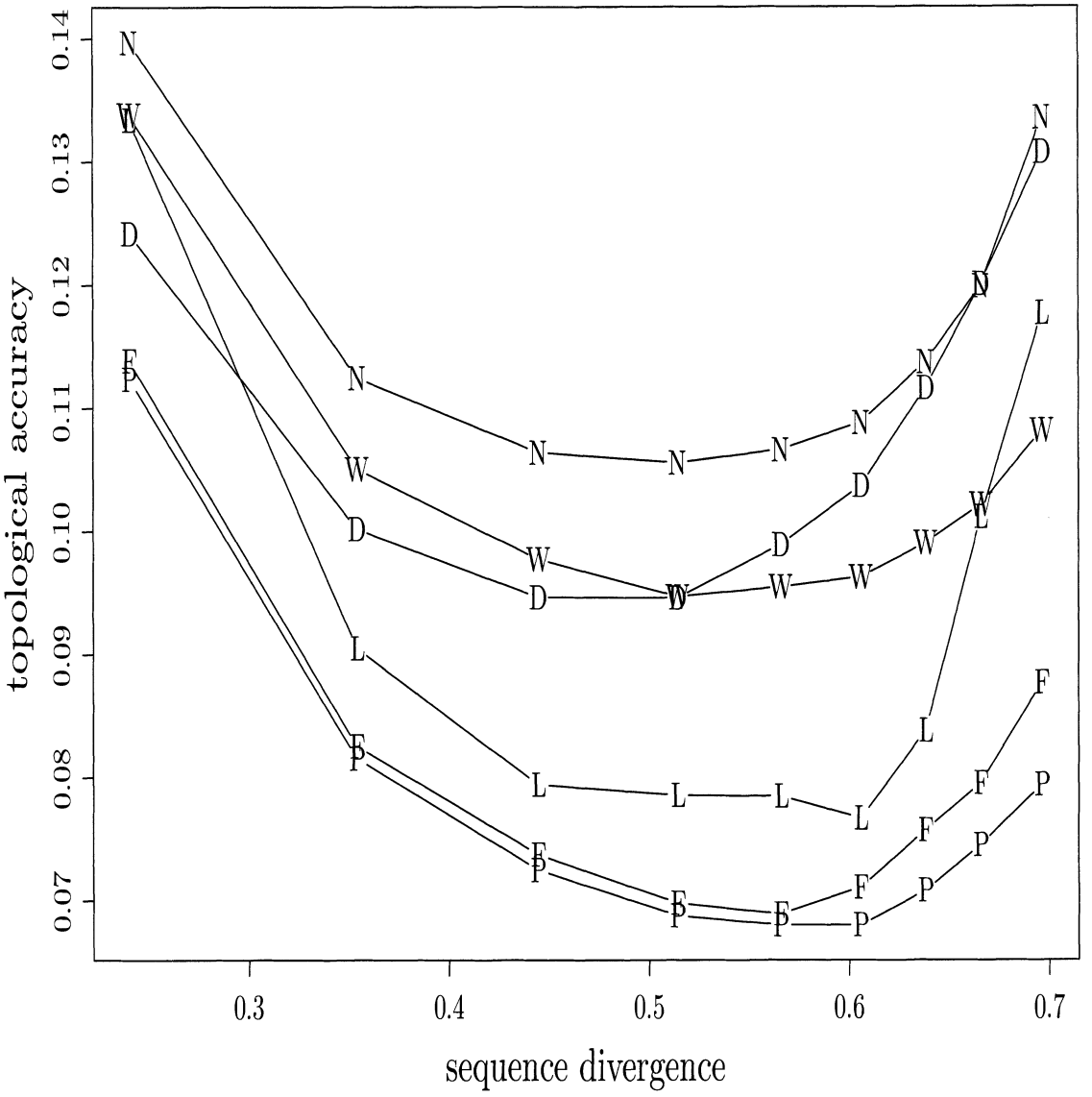

FIGURE 2. Topological accuracies of various tree building methods as a function of the divergence between sequences. N = NJ; W = Weighbor, L = NJML; D = DNAPARS; F = fastDNAml; P = PHYML. BIONJ (used to build the starting tree in PHYML) is midway between NJ and Weighbor.

of this distance was plotted against the maximum pairwise divergence in the data set under consideration, with the (uncorrected) divergence between two sequences being simply the proportion of sites where both sequences differ. The results are displayed in Figure 2, where the 5,000 original points corresponding to each method are smoothed by averaging over a sliding window of length 1,000.

These results are in accordance with expectations and with previously published simulations (Huelsenbeck and Hillis, 1993; Kuhner and Felsenstein, 1994; Huelsenbeck, 1995; Rosenberg and Kumar, 2001; Ranwez and Gascuel, 2002). When the divergence rate is low, phylogeny reconstruction is hard because there is not enough information in the data to estimate the short internal edges. With a high divergence rate, saturation corrupts the phylogenetic signal and reconstruction is again hard. This explains why all methods perform better with medium divergence rates. The best region for parsimony corresponds to low rates, as expected since it assumes that multiple substitutions are rare, while distance-based methods (which account for multiple substitutions) tend to perform better than parsimony when substitution rates are high. The performance of NJML, which combines distance-based and ML approaches, is midway between both. However, the best approach is clearly ML. Both fastDNAml and PHYML outperform all other methods, and PHYML even tends to improve fastDNAml with high substitution rates. Indeed, PHYML and fastDNAml are very close concerning likelihood optimization, except with high substitution rates, where PHYML is slightly better (results not shown). Moreover, for about 95% of the data sets, both programs infer trees with likelihoods identical to or higher than the likelihood

of the true tree, which indicates that there is little room for accuracy improvement by further optimizing the tree likelihood. This indication is confirmed by the average accuracy of the various ML programs on the first 1,000 data sets, i.e., 0.086, 0.086, 0.081, and 0.081 for PAUP*, fastDNAml, MrBayes and PHYML, respectively. So we do not expect to achieve major improvements on these data sets by any ML method, including MetaPIGA.

For the 1,000 data sets incorporating rate heterogeneity, both PHYML and PAUP* were given the true value (1.0) of the gamma distribution parameter. Adjusting this parameter in PAUP* was (too) time consuming: about 1 hr 25 min for a single data set, instead of 4 min without adjustment, and 43 sec for PHYML including adjustment. Results confirm above findings: PHYML is slightly more accurate than PAUP*; their average topological accuracies are 0.101 and 0.105, respectively.

Computing Times and Likelihood Optimization

We compared the computing time of the various methods using 30 data sets comprising 40 taxa and 30 data sets with 100 taxa, both sets being generated as described above (available on our web page). We also used two large real data sets. The first set contains 218 prokaryotic sequences with 4,182 bp from the small ribosomal subunit and was downloaded from the RDPII project web page (http://rdp.cme.msu.edu/download/SSU_rRNA/alignments). The second set includes 500 *rbc*L sequences with 1,428 bp from plant plastids and was obtained from http://www.cis.upenn.edu/~krice/treezilla/record.nex.

The computing time was measured on a PC Pentium IV 1.8 GHz (1 Go RAM) running with Linux. Basically, all methods were run as described above, while MetaPIGA was run by hand using one run and four metapopulations, which is the default option to build a single tree. To approximate posterior probabilities, Lemmon and Milinkovitch (2002) used 10 runs and 10 metapopulations, which makes running times much longer than those reported here. The speed of PHYML is partly explained by the fact that it starts with a reasonably good distance-based tree. Therefore, we also tested PAUP* and MetaPIGA using the option they give of starting from the NJ tree, denoted here as PAUP*+NJ and MetaPIGA+NJ. In principle, the same approach can be used to accelerate MCMC approaches, but we observed that it does not fit well with MrBayes, providing no significant gain in convergence time. Moreover, starting with random trees is the recommended strategy for evaluating convergence of MCMCs and then obtaining reliable estimation of posterior probabilities. FastDNAml and PAUP* were not run with the two larger data sets because even the 218-taxon set required more than 2 days of computations. PAUP*+NJ did not output any tree on the 500-taxon set for numerical reasons. With the two larger data sets, we were not able to obtain any result with NJML, seemingly for memory size reasons, while MrBayes was stopped after 1,000,000 generations without having reached stable likelihood values. With the 100-taxon data sets, MrBayes was run with 200,000 generations and a consensus tree was built from the last 10,000 trees. These parameters were chosen to converge on stable likelihood values, but we did not explore a large range of settings, first preferring computation speed. In fact, this criterion was used for MrBayes and for the other packages. So it is likely that other relevant speed/performance compromises could be found for any of these programs, and our results must therefore not be overinterpreted. Finally, the results (computing time, likelihood, inferred tree) of stochastic methods vary from one run to another, and the choice of Linux/Windows also has an influence. MetaPIGA (written in JAVA) is twice faster with Windows as with Linux, while the speed of other programs remains nearly identical.

The results are displayed in Table 1. PHYML is faster than all other ML programs. For example, with 100 taxa PHYML requires 12 sec and fastDNAml requires about 25 min. With 500 taxa, PHYML requires only about 12 min and MetaPIGA requires more than 9 hr. MetaPIGA+NJ is remarkably fast as well, being basically equivalent to PHYML with 40 and 100 taxa but still requiring 3 hr for 500 taxa. PAUP*+NJ is also much faster than PAUP*, which indicates that starting from a distance-based tree, as with PHYML, makes a significant difference with respect to computing time (see also results below concerning likelihood optimization).

TABLE 1. Average run times for various methods. The computing times were measured on a 1.8-GHz (1 Go RAM) PC with Linux. For PHYML, the number in parentheses is the average number of refinement stages.

	Simulations		Real data	
Method	40 taxa (500 bp)	100 taxa (500 bp)	218 taxa (4,182 bp)	500 taxa (1,428 bp)
DNADIST+ NJ/BIONJ	0.3 sec	2.3 sec	50 sec	2 min, 19 sec
DNADIST+ Weighbor	1.5 sec	22 sec	4 min, 52 sec	58 min, 40 sec
DNAPARS	0.5 sec	6 sec	4 min, 4 sec	13 min, 12 sec
PAUP*	3 min, 21 sec	1 hr, 4 min		
PAUP*+ NJ	1 min, 10 sec	22 min	10 hr, 50 min	
MrBayes	2 min, 6 sec	32 min, 37 sec		
fastDNAml	1 min, 13 sec	26 min, 31 sec		
NJML	15 sec	6 min, 4 sec		
MetaPIGA	21 sec	3 min, 27 sec	4 hr, 45 min	9 hr, 4 min
MetaPIGA+ NJ	6 sec	23 sec	1 hr, 40 min	3 hr
PHYML	2.7 sec (6.4)	12 sec (8.3)	8 min, 13 sec (15)	11 min, 59 sec (13)

Finally, it appears from Table 1 that the computing time of PHYML is in the same range as that of NJ, Weighbor, and DNAPARS (Table 1).

We also checked that the speed of PHYML is not offset by lower performance in optimizing the tree likelihood. For a fair comparison, the branch lengths of trees inferred by the various ML packages were reoptimized using the same Newton-Raphson procedure, and the tree likelihood was recomputed. Because all the methods used the same Kimura model to infer trees and because the branch lengths of the inferred trees were reoptimized by the same procedure, the results of the various methods were then fully comparable. Therefore, we sorted the (ML) methods with respect to their log-likelihood values and computed the mean of their rank by averaging over the 30 data sets analyzed. For the 40-taxon data sets, the mean ranks were 4.2, 4.2, 3.9, 3.0, 2.9, and 2.8 for MetaPIGA, NJML, MrBayes, PAUP*, PHYML, and fastDNAml, respectively, and with 100 taxa, the results were 4.8, 4.6, 4.1, 2.8, 2.5, and 2.4, respectively. Using this comparison method, PHYML is the second best program with 40 taxa and the best with 100 taxa, but these results illustrate the fact no method is systematically better than the others. Results for MetaPIGA+NJ and PAUP*+NJ were very close to those for MetaPIGA and PAUP* alone, with PAUP*+NJ being even slightly better than PAUP*.

We also computed the average log likelihood of every method for the 30 data sets corresponding to each tree size. The results basically confirmed the above ordering but with lower contrast. For 40 taxa, log likelihood values were −6196.745 for NJML, −6193.817 for MetaPIGA, −6193.688 for fastDNAml and PAUP*, −6193.626 for MrBayes, and −6193.569 for PHYML. PHYML is then best, and NJML is clearly last, although it was better than MetaPIGA when considering average ranking. This poor ranking is due to the fact that in a few cases NJML performs poorly and is far behind the other programs, which likely explains its relatively weak results regarding topological accuracy. While MrBayes performs relatively poorly when ranks are compared (see above), the trees that are inferred with this method are generally very likely even if they are not the most likely trees. This result could be explained by the fact that MrBayes tends to maximize the integrated likelihood, while we used the standard likelihood in our comparisons. MrBayes has then little chance to find the best tree regarding the standard likelihood, but because of its extensive search of the tree space it always finds good trees. Finally, it is worth noting that the ordering with average log likelihood is the same as that with topological accuracy on the first 1,000 data sets (see above).

For the large real data sets, PAUP*+NJ, MetaPIGA and PHYML were run with the Hasegawa–Kishino–Yano (HKY; Hasegawa et al., 1985) model, and the three programs adjusted the transition/transversion ratio. We did not account for rate heterogeneity because this is dealt with in a very different way by MetaPIGA and PHYML. With the 218-taxon set, the log likelihoods (after branch length reoptimization) were 156,881, −156,860, and −156,727, for PAUP*+NJ, PHYML, and MetaPIGA, respectively, and those for the 500-taxon set were −100,631 and −100,208 for MetaPIGA and PHYML, respectively. MetaPIGA is then best with the 218-taxon set, and PHYML is best with the 500-taxon set. These findings illustrate (again) the fact that no method is systematically better than the others and seem to indicate that further improvements could to be made for such very large sets, possibly by combining both approaches.

Therefore, it appears from the above results that PHYML is not only fast but also finds trees with high likelihood, being at least as good on average as the other methods we tested.

CONCLUSION

PHYML is freely available on our web page. The current version implements several models of nucleotide sequence evolution: JC69 (Jukes and Cantor, 1969), F81 (Felsenstein, 1981), K2P (Kimura, 1980), F84 (Felsenstein, 1993), HKY (Hasegawa et al., 1985) and TN93 (Tamura and Nei, 1993). The Dayhoff (Dayhoff et al., 1978) and JTT (Jones et al., 1992) models for proteins are also available and run quickly, requiring about 3 min to analyze a data set comprising 50 mammalian sequences and 1,729 sites (F. Delsuc, pers. com.). A discrete gamma distribution (Yang, 1994) can be used to account for variable substitution rates among sites. The parameters of these models can be either user defined or fitted to the data by likelihood maximization. PHYML can also be used to refine a user-supplied tree.

In regard to its simplicity, the performance of our algorithm is quite surprising. It is not only much faster than the standard approach but also slightly better in terms of topological accuracy and likelihood maximization. In fact, it seems that adjusting the branch lengths and the tree topology together appears to keep the program from getting trapped too early in local optima. The algorithm does not follow the slope corresponding to a unique branch or a unique swap but moves in a direction that improves the whole tree and, by striding this way, avoids getting lost in local irregularities of the likelihood landscape. However, testing more intense topological rearrangements or introducing some randomness in the search are interesting directions for future research.

ACKNOWLEDGMENTS

We thank Bruce Rannala for his help and comments, Michel C. Milinkovitch for providing us with useful advice, notably about MetaPIGA, David Bryant, Nicolas Galtier, and Marc Robinson-Rechavi for reading the preliminary versions of the paper, Franck Le Thiec for his help in implementing the protein version of PHYML, and Frédéric Delsuc for providing us with protein data sets. This work was supported by Montpellier Genopole and the InterEPST Bioinformatics Program.

REFERENCES

AARTS, E., AND J. K. LENSTRA. 1997. Local search in combinatorial optimization. Wiley, Chichester, U.K.

ADACHI, J., AND M. HASEGAWA. 1996. Molphy, version 2.3. Programs for molecular phylogenetics based on maximum likelihood. *In* Computer science monographs 28 (M. Ishiguro, G. Kitagawa, Y. Ogata, H. Takagi, Y. Tamura, and T. Tsuchiya, eds.). Institute of Statistical Mathematics, Tokyo.

BRAUER, M., M. HOLDER, L. DRIES, D. ZWICKL, P. LEWIS, AND D. HILLIS. 2002. Genetic algorithms and parallel processing in maximum-likelihood phylogeny inference. Mol. Biol. Evol. 19:1717–1726.

BRENT, R. 1973. Algorithms for minimization without derivatives. Prentice-Hall, Englewood Clifts, New Jersey.

BRUNO, W., N. SOCCI, AND A. HALPERN. 2000. Weighted neighbor joining: A likelihood-based approach to distance-based phylogeny reconstruction. Mol. Biol. Evol. 17:189–197.

CHOR, B., M. HENDY, B. HOLLAND, AND D. PENNY. 2000. Multiple maxima of likelihood in phylogenetic trees: An analytic approach. Mol. Biol. Evol. 17:1529–1541.

DAYHOFF, M., R. SCHWARTZ, AND B. ORCUTT. 1978. A model of evolutionary change in proteins. Pages 345–352 *in* Atlas of protein sequence and structure, Volume 5 (M. Dayhoff, ed.). National Biomedical Research Foundation, Washington, D.C.

FELSENSTEIN, J. 1981. Evolutionary trees from DNA sequences: A maximum likelihood approach. J. Mol. Evol. 17:368–376.

FELSENSTEIN, J. 1985. Confidence limits on phylogenies: An approach using the bootstrap. Evolution 39:783–791.

FELSENSTEIN, J. 1993. PHYLIP (phylogeny inference package), version 3.6a2. Distributed by the author, Department of Genetics, Univ. Washington, Seattle.

FELSENSTEIN, J., AND G. CHURCHILL. 1996. A hidden Markov model approach to variation among sites in rate of evolution. Mol. Biol. Evol. 13:93–104.

GASCUEL, O. 1997. BIONJ: An improved version of the NJ algorithm based on a simple model of sequence data. Mol. Biol. Evol. 14:685–695.

HASEGAWA, M., H. KISHINO, AND T. YANO. 1985. Dating of the human–ape splitting by a molecular clock of mitochondrial DNA. J. Mol. Evol. 22:160–174.

HENDY, M., D. PENNY, AND M. A. STEEL. 1994. A discrete Fourier analysis for evolutionary trees. Proc. Natl. Acad. Sci. USA 91:3339–3343.

HUELSENBECK, J. P. 1995. Performance of phylogenetic methods in simulation. Syst. Biol. 44:17–48.

HUELSENBECK, J. P., AND D. M. HILLIS. 1993. Success of phylogenetic methods in the four-taxon case. Syst. Biol. 42:247–264.

HUELSENBECK, J. P., AND F. RONQUIST. 2001. MrBayes: Bayesian inference of phylogeny. Bioinformatics 17:754–755.

JIN, L., AND M. NEI. 1990. Limitations of the evolutionary parsimony method of phylogenetic analysis. Mol. Biol. Evol. 7:82–102.

JONES, D., W. TAYLOR, AND J. THORNTON. 1992. The rapid generation of mutation data matrices from protein sequences. Comput. Appl. Biosci. 8:275–282.

JUKES, T., AND C. CANTOR. 1969. Evolution of protein molecules. Pages 21–132, *in* Mammalian protein metabolism, Volume III (H. Munro, ed.). Academic Press, New York.

KIMURA, M. 1980. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. J. Mol. Evol. 16:111–120.

KISHINO, H., T. MIYATA, AND M. HASEGAWA. 1990. Maximum likelihood inference of protein phylogeny and the origin of chloroplasts. J. Mol. Evol. 31:151–160.

KUHNER, M. K., AND J. FELSENSTEIN. 1994. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. Mol. Biol. Evol. 11:459–468.

LEMMON, A., AND M. MILINKOVITCH. 2002. The metapopulation genetic algorithm: An efficient solution for the problem of large phylogeny estimation. Proc. Natl. Acad. Sci. USA 99:10516–10521.

LEWIS, P. 1998. A genetic algorithm for maximum likelihood phylogeny inference using nucleotide sequence data. Mol. Biol. Evol. 15:277–283.

LI, S. 1996. Phylogenetic tree construction using Markov chain Monte Carlo. Ph.D. Thesis, Ohio State Univ., Columbus.

MAU, B. 1996. Bayesian phylogenetic inference via Markov chain Monte Carlo methods. Ph.D. Thesis, Univ. Wisconsin, Madison.

OLSEN, G., H. MATSUDA, R. HAGSTROM, AND R. OVERBEEK. 1994. FastDNAml: A tool for construction of phylogenetic trees of DNA sequences using maximum likelihood. Comput. Appl. Biosci. 10:41–48.

OTA, S., AND W.-H. LI. 2000. NJML: A hybrid algorithm for the neighbor-joining and maximum-likelihood methods. Mol. Biol. Evol. 17:1401–1409.

OTA, S., AND W.-H. LI. 2001. NJML+: An extension of the NJML method to handle protein sequence data and computer software implementation. Mol. Biol. Evol. 18:1983–1992.

PAGE, R., AND E. HOLMES. 1998. Molecular evolution: A phylogenetic approach. Blackwell, Osney Mead, Oxford, U.K.

PRESS, W., B. FLANNERY, S. TEUKOLSKY, AND W. T. VETTERLING. 1988. Numerical Recipes in C. Press Syndicate, Univ. Cambridge, Cambridge, U.K.

RAMBAUT, A., AND N. GRASSLY. 1997. Seq-Gen: An application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees. Comput. Appl. Biosci. 13:235–238.

RANNALA, B., AND Z. YANG. 1996. Probability distribution of molecular evolutionary trees: A new method of phylogenetic inference. J. Mol. Evol. 43:304–311.

RANWEZ, V., AND O. GASCUEL. 2001. Quartet-based phylogenetic inference: Improvements and limits. Mol. Biol. Evol. 18:1103–1116.

RANWEZ, V., AND O. GASCUEL. 2002. Improvement of distance-based phylogenetic methods by a local maximum likelihood approach using triplets. Mol. Biol. Evol. 19:1952–1963.

ROBINSON, D., AND L. FOULDS. 1979. Comparison of weighted labeled trees. Pages 119–126, *in* Lectures notes in mathematics, Volume 748. Springer, Berlin.

ROGERS, J., AND D. SWOFFORD. 1999. Multiple local maxima for likelihoods of phylogenetic trees: A simulation study. Mol. Biol. Evol. 16:1079–1085.

ROSENBERG, M., AND S. KUMAR. 2001. Traditional phylogenetic reconstruction methods reconstruct shallow and deep evolutionary relationship equally well. Mol. Biol. Evol. 19:1823–1827.

SAITOU, N., AND M. NEI. 1987. The neighbor-joining method: A new method for reconstructing phylogenetic trees. Mol. Biol. Evol. 4:406–425.

SALTER, L., AND D. PEARL. 2001. Stochastic search strategy for estimation of maximum likelihood phylogenetic trees. Syst. Biol. 50: 7–17.

SIMON, D., AND B. LARGET. 2000. Bayesian analysis in molecular biology and evolution (BAMBE), version 2.03beta. Department of Mathematics and Computer Science, Duquesne Univ., Pittsburgh, Pennsylvania.

STRIMMER, K., AND A. VON HAESELER. 1996. Quartet puzzling: A quartet maximum-likelihood method for reconstructing tree topologies. Mol. Biol. Evol. 13:964–969.

SWOFFORD, D. 1999. PAUP*: Phylogenetic analysis using parsimony (*and other methods). Sinauer, Sunderland, Massachusetts.

SWOFFORD, D., G. OLSEN, P. WADDEL, AND D. M. HILLIS. 1996. Phylogenetic inference. Pages *in* (Molecular systematics, 2nd edition (D. M. Hillis, C. Moritz, and B. K. Mable, eds.). Sinauer, Sunderland, Massachusetts.

TAMURA, K., AND M. NEI. 1993. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. Mol. Biol. Evol. 10:512–526.

UZZELL, T., AND K. CORBIN. 1971. Fitting discrete probability distributions to evolutionary events. Science 172:1089–1096.

WHELAN, S., P. LIÒ, AND N. GOLDMAN. 2001. Molecular phylogenetics: State-of-the art methods for looking into the past. Trends Genet. 17:262–272.

YANG, Z. 1994. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: Approximate methods. J. Mol. Evol. 39:306–314.

YANG, Z. 1996. Among-site rate variation and its impact on phylogenetic analyses. Trends Ecol. Evol. 11:367–372.

YANG, Z. 2000. Maximum likelihood estimation on large phylogenies and analysis of adaptative evolution in human influenza virus A. J. Mol. Evol. 51:423–432.

First submitted 10 January 2003; reviews returned 31 March 2003; final acceptance 11 June 2003
Associate Editor: Bruce Rannala