Supporting information for Lemmon and Milinkovitch (2002) *Proc. Natl. Acad. Sci. USA*, 10.1073/pnas.162224399

Supporting Materials and Methods

The Cycle of Evaluation-Selection-Mutation. ***Evaluation.*** At the beginning of each generation, each tree is evaluated (without optimization of branch lengths or other model parameters) and its log likelihood (hereafter called score) is recorded. We use minimal updating (MU), i.e., a specific application of the pruning algorithm of Felsenstein (1), when calculating the score of each tree. The pruning algorithm takes advantage of the fact that likelihood calculation at a particular node depends only on the values at the two nodes connected to it in the direction of the terminal nodes. MU extends this by taking advantage of the fact that perturbation of the tree topology and branch lengths during the GA search (see below) forces recomputation of the likelihood only at nodes along the path from the changed part(s) of the tree to the center.

Selection. Once all individuals (trees) have been scored, selection is imposed on the population. Four types of selection are implemented: rank, tournament, replacement, and improve. In rank selection, individuals are assigned a probability of leaving an offspring (i.e., a copy of themselves) as a function of their position in a list in which they are ranked by their score. In METAPIGA, we implement a rank selection identical to that described in Lewis (2), i.e., in a population of n individuals ranked by their lnL, the probability for the ith individual of leaving an offspring to the next generation is equal to

$$\frac{2}{n(n+1)}(n-i+1)$$

In tournament selection, two individuals are drawn randomly from the population of n individuals, and one offspring is produced from the individual with the higher score. Both trees are then placed back into the mating population, and the whole process is repeated until n offspring have been generated. In replacement selection, two individuals are drawn randomly from the population of n individuals and two copies of

the better individual are returned to the mating pool (parents are discarded). The process is repeated *sn* times, where *s* is the strength of selection, then the offspring population is generated as an exact copy of the postselection parent population. All three of the above-mentioned selection regimes tolerate the maintenance of poor trees in the evolving populations, an effect that may, in some cases, allow escape from local optima but generally lowers the efficiency with which the GA searches for the optimal tree(s). The decreased efficiency is especially noticeable toward the end of the search when beneficial mutations are extremely rare and should never be ignored. The improve selection method avoids this problem by allowing only those individuals that have scores better than that of the best tree from the previous generation to produce an offspring. Each individual that fails this test is discarded and replaced by a copy of the current best individual. The latter selection scheme greatly reduces the intra-population variability after each selection step. Local optima are avoided, however, through the metapopulation procedures described in the main text.

Mutation. Each individual, with the exception of the current best individual from each population, is then subjected to a single mutation. Besides SPR and NNI, which are performed as described (3), we implemented two other types of topological mutations: taxa swap and subtree swap (STS). The former involves the exchange of two randomly selected terminal branches while the latter corresponds to a swap between two small subtrees. As in the GA of Lewis (2), we also allow branch length mutations (BLM): the length of a random branch in the tree is multiplied by a factor drawn from a gamma distribution with a gamma shape parameter α specified by the user and a mean equal to 1. Our exploratory analyses (data not shown) indicated that branch length optimization yields external branch lengths that are very similar to those obtained through topology-constrained neighbor joining (NJ), both on a NJ topology and a ML topology. Hence, we allow in our metaGA implementation an original operator: BLMint, i.e., BLM is allowed to act on internal branches only (whose initial length is set to the average of all NJ-estimated internal branch lengths) while external branch lengths are fixed throughout the search to the NJ-estimated values. This method typically allows for more rapid exploration of search space.

Finally, we incorporated an enhanced version of the recombination operator implemented in GAML (2). In Lewis' recombination strategy, individuals (trees) *i* and

j are recombined as follows: the subtree defined by a random branch in *i* is removed and grafted onto a random branch in individual *j* after pruning from *j* all taxa represented in the subtree. We found the use of this recombination procedure in our GA to be ineffective. This procedure is indeed unlikely to yield a recombined tree with a better score than any of the two parents because it essentially performs SPR on a tree that has already been modified through pruning of some of the taxa. We then defined a new recombination operator (RCM) that involves the identification of a consensus region between two trees from different populations (see the main text), followed by the swapping of the subtrees defined by that consensus region. By definition, this procedure does not require any pruning of taxa because the consensus branch defines, in both trees, the same partitions. Hence, RCM is more comparable to real DNA recombination (with homologous regions being exchanged).

Dynamic Operators. In addition to the possibility of assigning to the operators (NNI, SPR, taxa swap, STS, recombination operator, BLM/BLMint) fixed probabilities, we allow the probabilities to be assigned dynamically. Under this option, the probability assigned to each operator is relative to the average contribution that operator has made to improving the population, within the last G generations. To prevent an operator from being switched off while it could become efficient later in the search, a lower probability bound (specified by the user) is placed on each operator.

Starting Trees. To allow for variation across populations, initial topologies are either random (each of the I individuals in each of the P populations is a different random tree) or constructed through procedures we call jackknifed NJ (JNJ) and noisy NJ (NNJ). JNJ consists of generating all starting trees with the NJ algorithm, but on a different subset of the data set for each population. We implemented two variations: nonoverlapping and overlapping JNJ. The former consists of randomly assigning each character to one of the P populations, such that the original data set is eventually divided into P nonoverlapping sets of characters. On the other hand, the overlapping JNJ consists of randomly assigning a proportion *p* of the characters to each of the populations. Because the process is independent (and performed from the original data set) for each population, the P sets of characters are typically overlapping and each character may be assigned to 0, 1, ..., or N of the populations. The second procedure, NNJ, uses the full data set but with a modified method for joining nodes: at each step,

suboptimal nodes (i.e., nodes that are never joined in classical NJ because they exhibit a nonminimal value in the current distance matrix) are joined with a probability proportional to the inverse of their pairwise distance.

The JNJ and NNJ procedures allow the search to start with trees whose scores are much better than those of random trees while still keeping enough variation among the initial populations for the metaGA to be effective in finding the ML topology. Initial internal branch lengths are set either to those specified by the NJ algorithm or an arbitrary value specified by the user.

The Software. METAPIGA is written in the Java programming language and supports Windows, Unix/Linux, and Macintosh operating systems. Java was chosen to allow for cross-platform compatibility and easy implementation of a user-friendly interface. Preliminary tests suggested that coding the software in Java instead of C++ would not have an appreciable (i.e., >15%) effect on the speed of the software. METAPIGA (and a supporting user's manual) is distributed at http://dbm.ulb.ac.be/ueg.

Results and Discussion

Population Size. For small (<60 taxa) data sets, varying the population size from two to 32 individuals in single population GA runs decreases accuracy but does not significantly change the mean running time (data not shown). On the other hand, for large data sets (>60 taxa), increasing the population size increases run time but has no significant effect on accuracy (data not shown). In short, small populations yield both the quickest and most accurate results, regardless of the size of the data set.

Relative Efficiencies of the Starting Tree Options. Although a random tree will generally exhibit a score vastly worse than that of a StepAdd (3) or a NJ (4) tree, the former constitutes an unbiased starting point for a search, such that different random trees are less likely (on average) to become systematically trapped in the same local optimum. One could therefore argue that the most conservative strategy for obtaining a globally optimal solution would be to repeat the heuristic search beginning with different starting trees (e.g., different random trees) and test whether most of these searches yield the same topology. Obviously, this approach has a drawback: computing

time. Preliminary analyses indicate (data not shown) that our JNJ/NNJ starting tree strategies (described above) are both vastly faster than starting with random trees and efficient at avoiding local optima. Although more extensive analyses need to be performed, these methods seem to be effective compromises between speed and efficiency because they allow both for high starting scores and some random variation.

1. Felsenstein, J. (1981) *J. Mol. Evol.* **17,** 368–376.

2. Lewis, P. O. (1998) *Mol. Biol. Evol.* **15,** 277-283.

3. Swofford, D. L., Olsen, G. J., Waddell, P. J. & Hillis, D. M. (1996) in *Molecular Systematics*, eds. Hillis, D. M., Moritz, C. & Mable, B. K. (Sinauer, Sunderland, MA), pp. 482–485.

4. Saitou, N. & Nei, M. (1987) *Mol. Biol. Evol.* **4,** 406–425.