Building Phylogenetic Trees from Quartets by Using Local Inconsistency Measures

Stephen J. Willson
Department of Mathematics and Iowa Computational Biology Laboratory, Iowa State University

A new quartet method is described for building phylogenetic trees, making use of a numerical measure of local inconsistency. For each quartet consisting of four species, the user chooses numbers indicating evidence for each of the three possible completely resolved trees. These numbers may be, for example, tree lengths or likelihoods. From these numbers, I describe how to measure the “local inconsistency” that results from placing a new species into a particular position in a phylogenetic tree. The best placements are those with low local inconsistency. A phylogenetic tree for a collection of taxa may be constructed by picking a random order of species and adding the species in this order, each time using the placement with the lowest local inconsistency. To summarize the results, one may select a majority-rule consensus tree or the tree most frequently obtained. Alternatively, taxa can be added in the order that maximizes the signal strength. Advantages of the method may include flexibility and better resolution. Studies are performed for artificial data sets for which long-branch attractions are a serious problem; comparisons show performance much superior to maximum parsimony and somewhat superior to quartet puzzling. A case study with real data also illustrates the method.

Introduction

Suppose that we are given a set $S$ of taxa, and for each species $i$ in $S$, we have a string $s_i$. The string $s_i$ may correspond to the nucleotides in common portions of DNA, or it may correspond to amino acids in proteins. We assume that all the strings have a known common alignment. We seek to construct a phylogenetic tree relating the species in $S$.

There are numerous methods for constructing a candidate tree, as may be seen from Felsenstein’s web page (http://evolution.genetics.washington.edu/phylip/software.html). It often happens that different methods yield different candidate trees. How are we to decide whether a candidate tree appears “correct”? Among the considerations we might utilize to accept a given tree are the following:

1. Does the method used to construct the tree appear to be biologically reasonable?
2. Is the criterion used to select the tree logically consistent?
3. Is the tree “robust”? Does the same tree arise if various parameters in the construction method are modified?
4. Do different methods give rise to the same tree?

In Willson (1998), I presented a critique of maximum parsimony as a criterion for selecting a candidate tree. The essential idea is one called “consistency” in that paper: If $T$ is the “correct” tree for a set $S$ of species, $T'$ is the “correct” tree for a set $S'$ of species, and $S \subseteq S'$, then we expect $T$ to be obtained from $T'$ by ignoring the species of $T'$ not in $T$. When this happens, $T$ and $T'$ are “consistent.” (This use of the word is slightly different from other usage in the literature. A standard use of “consistency” as given in Hillis, Moritz, and Mable [1996, pp. 426 and 527] is that the method yields the correct tree as more and more data become available.) It turns out that if $T$ is the maximum-parsimony tree for the species $S$ and $T'$ is the maximum-parsimony tree for the species $S'$, then $T$ and $T'$ need not be consistent in our sense. The same criticism holds for maximum-likelihood trees.

The fundamental idea of this paper is to seek a tree that exhibits a great deal of consistency. The method presented in the paper proceeds in two steps:

Step (1): Decide for each “quartet” $J$ consisting of four distinct species $J$ from $S$ (for example, $J = \{1, 2, 3, 4\}$) which of the fully resolved unrooted trees (“dogbones”) for the four species best describes their relationship. The possibilities are indicated in figure 1; the star-shaped tree with only one internal vertex is not considered explicitly, since it arises as a “tie” between the three trees in figure 1. Call the best tree $T(J)$.

Step (2): Fit together the trees $T(J)$ to obtain a supertree $T$ containing all the species of $S$.

Both of these steps can be done in several ways. Step (1) may be solved in different ways because a number of different methods might be utilized to decide for a quartet $J = \{1, 2, 3, 4\}$ which of the dogbones is the preferred tree $T(J)$. Among them, the simplest include the following:

1. Maximum parsimony. Given the strings $s_i$ for the four species, compute the tree length of each of the three possible trees $U$ in figure 1. This may easily be accomplished by Fitch’s (1971) algorithm. Select as the preferred tree $T(J)$ for the quartet $J$ the tree with the shortest tree length. More explicitly, suppose $w(U)$ denotes the tree length of a tree $U$. Select $T(J)$ to be the dogbone $U$ for which $w(U)$ is minimal. Since there are only three trees for each quartet $J$, it is easy to find $T(J)$ for each such $J$. 

Key words: phylogenetic, parsimony, likelihood, quartet, inconsistency.

Address for correspondence and reprints: Stephen J. Willson, Department of Mathematics and Iowa Computational Biology Laboratory, Iowa State University, Ames, Iowa 50011. E-mail: swillson@iastate.edu.
2. Maximum likelihood. Given the strings $s$, for the four species, compute the likelihood $L(U)$ of each of the possible trees $U$ for those species. Select as the preferred tree $T(J)$ for the quartet the tree with maximum likelihood; thus, $T(J)$ is the dogbone $U$ for which $L(U)$ is maximal. Different researchers may, of course, utilize different models for computing likelihoods. In any event, since there are only four species in each quartet, the method chosen may be carefully optimized to seek the correct tree for the four species.

Similarly, step (2) may be accomplished in several ways. Most of these involve adding species to a tree one at a time. A criterion is then needed for when the placement of a species appears “correct” so as to favor “consistency.” For both maximum parsimony and maximum likelihood, I will describe in this paper such a criterion.

1. Add the species in a random order using the criterion for placement. If this is done for a reasonable number of replications, we output a consensus tree for the various replicates or the most common tree among the replicates.

2. Add the species in an order to “maximize the signal strength” at each stage.

3. Find a “minimally inconsistent tree.”

There are other “quartet” methods which similarly start with step (1) but proceed differently for step (2). Strimmer and von Haeseler’s (1996) and Strimmer, Goldman, and von Haeseler’s (1997) “quartet puzzling” utilizes maximum likelihood for step (1) and adds species in a random order as in the first approach, but it does not utilize the measures in this paper for selecting the preferred placement. Willson (1998) focuses on the third approach. In a particular case, it may be possible to prove that a given tree is “minimally inconsistent” in the sense that the numerical measure of inconsistency is minimized by that tree. We have, however, no automated procedure other than exhaustive search for proving that a given tree is minimally inconsistent.

The basic philosophy in this paper is the same as that in Willson (1998). Here, however, I stress the computational advantages of the first and second approaches, rather than the third approach, for step (2). I also introduce a notion of “local inconsistency” which provides somewhat more discrimination and resolution in hard cases. These technical innovations may yield more natural heuristics and, possibly, greater reliability.

The heart of the method is the determination of the “correct” tree for four species at a time and a numerical measure of our confidence in the correctness of that tree. Figure 1 shows the three possible unrooted completely resolved trees for four distinct species 1, 2, 3, and 4. According to step (1), the first issue is to determine for each such set of four species which of the trees analogous to those in figure 1 is correct. The notation presented below each tree in figure 1 is convenient: ((1 2) (3 4)) indicates that the path joining species 1 and 2 is disjoint from the path joining species 3 and 4. There is no root, but in some sense, species 1 and 2 are grouped together while species 3 and 4 are grouped together.

Colomnius and Schultze (1981) and Bandelt and Dress (1986) show that there is at most one tree $T$ that is consistent with all the trees $T(J)$ for all the quartets $J$ from the set of species. Willson (1998) (as well as the other papers) describes a method for constructing such a tree $T$ when such a tree exists. Unfortunately, usually, such a tree does not exist when the strings arise from real data. The real issue is thus constructing a tree when no completely consistent tree exists.

For both maximum parsimony and maximum likelihood, there is additional information for each quartet $J$ besides the tree $T(J)$, and this information can provide a measure of confidence in the selection of $T(J)$. Consider the case of maximum parsimony. Suppose that we have two different quartets, $J = \{1, 2, 3, 4\}$ and $K = \{5, 6, 7, 8\}$. Suppose that their tree lengths $w$ are, respectively, $w((1 2) (3 4)) = 200$, $w((1 3) (2 4)) = 203$, and $w((1 4) (2 3)) = 202$. Then, $T(J) = ((1 2) (3 4))$, because that tree has the least steps, but the tree lengths are nearly tied, and “errors” in the data set, misalignments, or convergent evolution could easily have changed the result. Hence, we have comparatively little confidence in the correctness of $T(J)$. By contrast, suppose that $w((5 6) (7 8)) = 200$, $w((5 7) (6 8)) = 400$, and $w((5 8) (6 7)) = 500$. Now, $T(K) = ((5 6) (7 8))$, but we have considerable confidence in our identification of $T(K)$. In order for $T(K)$ to be incorrectly identified, there would need to be many errors in the data set and/or much convergence in the evolutionary history.

The idea of our method is to build the consensus tree by giving high priority to quartets such as $K$ in our first example, for which we have much confidence in the correctness of $T(K)$, and by giving low priority to quartets such as $J$, for which we have little confidence in the correctness. Formally, we proceed to build a tree by adding species one at a time, taking into account these priorities.

Similar considerations apply to the case of maximum likelihood. The likelihoods $L(U)$ are all available for each tree. Suppose that the maximum-likelihood tree is ((1 2) (3 4)). Then, $L((1 2) (3 4)) \geq L((1 3) (2 4))$, and $L((1 2) (3 4)) \geq L((1 4) (2 3))$. If, however, there is a tie or nearly a tie for maximum likelihood, then our confidence in the choice of tree is lower than it would be if the likelihoods were widely different. Again, the differences between the numbers $L((1 2) (3 4))$, $L((1 3)$
(2 4), and \(L((1 4) (2 3))\), or between their logarithms, may be utilized to help decide our confidence in \(T(J)\).

### Consistency and Measurement of Inconsistency

Suppose that \(T\) is a tree in which each leaf is labeled with the name of a species. Suppose \(J\) is a subset of the set \(S\) of leaves of \(T\). We form a tree \(T|J\) by removing all the leaves from \(T\) other than the species in \(J\), together with their attachments; we call \(T|J\) the tree “induced from” \(T\) by \(J\) (see fig. 2). The tree \(T|J\) summarizes the relationships in \(T\) among the species of \(J\) only. We say that \(T\) and \(T|J\) are “consistent.” Of particular interest for us is the case in which \(J\) consists of exactly four species. As in Introduction, assume that a “correct” tree \(T(J)\) has already been identified for the species of \(J\). Then, we hope that the phylogenetic tree \(T\) that we construct satisfies the criterion that \(T|J = T(J)\).

It is important to be able to measure the degree of inconsistency in a tree \(T\). We shall assume in this exposition that for each quartet \(J\), the tree \(T(J)\) is selected from among the various alternatives in figure 1 by minimizing the function \(w\). We may think of \(w(T(J))\) as the tree length of \(T(J)\), and we will so refer to it. The method, however, easily generalizes to the case in which \(w(T(J))\) is defined differently, as long as the objective is to make \(w\) as small as possible. If the user wanted to make use of maximum likelihood rather than parsimony, then the formulas work with \(w(T(J)) = -\log L(T(J))\), where \(L(T(J))\) is the likelihood of \(T(J)\).

Measuring the inconsistency of \(T\) with set \(S\) of leaves requires several steps: Suppose that \(J\) is a quartet (i.e., \(J \subseteq S\) and \(J\) contains exactly four elements). Suppose \(T(J)\) is the dogbone with leaves \(J\) with minimal value \(w(T(J))\). We define the excess \(e(T, J)\) of \(T\) at \(J\) as follows:

(a) If \(T|J \neq T(J)\), such that the correct tree \(T(J)\) is inconsistent with \(T\), define \(e(T, J) = w(T|J) - w(T(J))\).

(b) If \(T|J = T(J)\), such that the correct tree \(T(J)\) is consistent with \(T\), let \(U(T, J)\) denote the dogbone \(V\) with set \(J\) of leaves such that \(V \neq T(J)\) and such that \(w(V)\) is minimal among all such dogbones. Thus, \(U(T, J) \neq T(J)\), and for both of the dogbones \(V\) with set \(J\) of leaves for which \(V \neq T(J)\), we have \(w(U(T, J)) \leq w(V)\). We refer to \(U(T, J)\) as the “runner-up” dogbone. In this case, define \(e(T, J) = w(T|J) - w(U(T, J))\).

Define the “inconsistency” \(I(T)\) by \(I(T) = \max\{e(T, J) : J\) is a quartet from \(S\}\). Thus, \(I(T)\) is the maximum excess \(e(T, J)\), for which \(J\) ranges over all possible quartets.

If \(x\) is a particular taxon from \(S\), define the “local inconsistency” \(L(T, x)\) of \(T\) at \(x\) by \(L(T, x) = \max\{e(T, J) : J\) is a quartet from \(S\) and \(J\) contains \(x\}\). Thus, \(L(T, x)\) measures the maximum excess among all quartets containing \(x\). When necessary for clarity, we may refer to \(I(T)\) as the “global inconsistency” in order to distinguish it from the local inconsistency \(L(T, x)\) when localized to the taxon \(x\).

Note that in case (a) above, \(e(T, J) \geq 0\), since \(w(T(J))\) is minimal; in case (b) above, \(e(T, J) \leq 0\) for the same reason. A large positive \(I(T)\) indicates that \(T\) has a high degree of inconsistency and hence is unlikely to be correct. A small positive \(I(T)\) indicates that \(T\) exhibits only mild inconsistencies such as might be compatible with data errors or small amounts of convergence. A negative \(I(T)\) indicates complete consistency with all relevant quartets. See Willson (1998) for more details.

Another related measure for tree \(T\) is \(b(T)\), defined as the number of quartets \(J\) for which \(e(T, J) > 0\). Thus, \(b(T)\) counts the number of “bad” quartets \(J\) for which \(T|J\) is not a maximum parsimony tree. While \(b(T)\) is of interest, there are examples in which \(b(T)\) is small but \(I(T)\) is very large, undermining the usefulness of \(b(T)\) (see Willson 1998).

### Building a Tree by Adding Species One at a Time

Suppose that \(S\) is the set of all the species in question, and a tree \(W_i\) has been constructed with set \(S_i\) of species. We wish to insert into \(W_i\) a new species \(x\) from \(S - S_i\). We heuristically proceed to find the “best” placement. Thus, we assume that \(W_i\) is correct, and we see which new placement of species \(x\) is best. See figure 3 for an example of the possible placements of a new species.

For each possible placement \(i\) of \(x\) into the tree \(W_i\), we obtain a tentative tree \(W_i(x, i)\). Define \(g(x, i) = L(W_i(x, i), x)\). We refer to \(g(x, i)\) as the local inconsistency of the \(i\)th placement of \(x\). It tells the maximum excess (hence, the worst possible contradiction) involving species \(x\) if it is inserted into tree \(W_i\) with placement \(i\). A “large” positive value for \(g(x, i)\) indicates that placement \(i\) of species \(x\) is unlikely to be correct. A
“small” positive value indicates the possibility that \( W_i(x, i) \) is correct but that there are some errors in the data. (Typically, in a given real problem, however, we do not know which values are “large” and which values are “small.”) A negative value indicates that the placement is consistent with all the quartets involving \( x \).

From this point of view, the best possible placement of \( x \) is then the placement of \( x \) which minimizes \( g(x, i) \). By abuse of notation, we define \( L_i(x) \) as the minimum value of \( g(x, i) \) over all possible placements \( i \) of \( x \), and we call \( L_i(x) \) the “local inconsistency” for \( x \). Thus, \( L_i(x) = LI(W_i(x, i), x) \) for the best placement \( i \) of species \( x \). If there is a unique placement \( i \) for which \( g(x, i) = LI(x) \), define \( P(x) \) to be the \( i \) which minimizes \( g(x, i) \); we call \( P(x) \) the “best placement” of \( x \). The meanings of these quantities are as follows: If we accept \( P(x) \) as the correct placement, we obtain the tree \( W_i(x, P(x)) \). If \( L_i(x) < 0 \), then for every quartet \( J \) including \( x \), \( T(J) \) is consistent with \( W_i(x, P(x)) \). If \( L_i(x) \geq 0 \), then for any quartet \( J \) containing \( x \), the tree \( W_i(x, P(x)) \) is at most \( L_i(x) \) steps longer than \( T(J) \). For the worst quartet \( J \), the tree \( W_i(x, P(x)) \) is exactly \( L_i(x) \) steps longer than \( T(J) \). Thus, at worst, we contradict \( T(J) \) by \( L_i(x) \) steps. It is desirable that \( L_i(x) \) be small.

Occasionally, it may happen that there is more than one placement \( i \) for which \( g(x, i) = LI(x) \). In these situations, a tie-breaking rule is required. In our computer implementations, we have usually broken ties by preferring the placement which contradicts \( T(J) \) for the fewest quartets \( J \). Thus, for ties, we have computed the number \( b(x, i) \) of quartets \( J \subseteq S_k \cup \{x\} \) containing \( x \) such that \( e(W_i(x, i), J) > 0 \) (the number of “bad” quartets); among all \( i \) such that \( g(x, i) = LI(x) \), we have selected \( P(x) \) to equal the \( i \) for which \( b(x, i) \) is minimal. We therefore may proceed by inserting species \( x \) at the placement \( P(x) \) to obtain the tree \( W_{i+1} \).

As an example, I illustrate the process by placing species 5 into one of the positions of figure 3. Suppose that the data set yields the information given in table 1. Consider placement 5b. Then, \( e(W_5(5, 5b), \{2, 3, 4, 5\}) = 715 - 703 = 12 \), since placement 5b would induce \( W_5(5, 5b) \{2, 3, 4, 5\} = (5234) \), of tree length 715, while the shortest tree for that quartet is \((2345) \), of tree length 703. Since the placement did not induce the shortest tree, the excess is positive.

Next, \( e(W_5(5, 5b), \{1, 2, 4, 5\}) = 652 - 674 = -22 \). This is because placement 5b induces \( W_5(5, 5b) \{1, 2, 4, 5\} = (12) \), of tree length 652. Since this tree length is the smallest, we identify the runner-up tree length as 674, from \(((12)(45)) \), and compute a negative excess.

Similarly, we compute \( e(W_5(5, 5b), \{1, 3, 4, 5\}) = 706 - 698 = 8 \) and \( e(W_5(5, 5b), \{1, 2, 3, 5\}) = 613 - 636 = -23 \). The maximum excess obtained from placement 5b is then \( g(5, 5b) = 12 \). A similar computation can be done for each of the other placements. We find that \( g(5, 5e) = -8 \) and is smaller than any other value \( g(5, i) \). Hence, \( LI(5) = -8 \), and \( P(5) = 5e \). The best placement is 5e.

This procedure may be utilized recursively to build a tree for all the species. We start with some quartet \( J \) for which we have very high confidence in the correctness of \( T(J) \), and we let \( W_i = T(J), S_k = J \). Once \( W_k \) and \( S_k \) are known, if \( S_k \neq S \), then we select a species \( x \) in \( S - S_k \). We utilize the method described above to place \( x \) into \( W_k \) to obtain a tree \( W_{k+1} = W_k(x, P(x)) \) with species \( S_k+1 = S_k \cup \{x\} \). If \( S \) contains \( n \) species, then in this manner, we ultimately obtain a tree \( W_n \) containing all \( n \) taxa.

Unfortunately, it may happen that the order in which we choose to add species affects the final tree \( W_n \). We then have two approaches:

1. We may choose different orders in which we add the species and find the resulting tree in each case. Then we may combine the resulting trees by some method to obtain a consensus tree.
2. We may carefully select the order in which we add the species according to some criteria.

We give details about the first approach in Finding a Tree by Averaging the Trees from Many Replications, and we give details about the second approach in Building a Tree by Identifying the Strongest Signal.

### Finding a Tree by Averaging the Trees from Many Replications

The tree obtained by the method of the preceding section may depend on the order in which the species are added. Consequently, we choose an initial tree \( W_4 \) in which we have great confidence. Typically, this initial tree is of form \(((a b)(c d)) \), where \( a \) and \( b \) are very obviously close and \( c \) and \( d \) are very obviously close. For example, in Case Study: Is the Guinea Pig a Rodent? a = pygmy chimp, b = common chimp, c = harbor seal, and d = gray seal.

A systematic approach to finding an initial tree is to select \( W_4 = T(J) \) for some quartet \( J \) such that the tree \( T(J) \) has a much shorter tree length than any alternative dogbone for \( J \). For example, for each quartet \( J \), we may find the three possible resolved trees \( T_1(J), T_2(J), \) and \( T_3(J) \) for the species in \( J \). Reorder the trees so that \( w(T_1(J)) \leq w(T_2(J)) \leq w(T_3(J)) \). We then select as the initial \( J \) the quartet \( J \) maximizing the difference \( w(T_3(J)) - w(T_3(J)) \). This selection gives very strong evidence of the correctness of \( W_4 \).

We then generate a certain number \( m \) of random orders in which the remaining species may be added to \( W_4 \). Utilizing the method of the preceding section, we
thus obtain \( m \) trees \( T_i, i = 1, \ldots, m \). Some of these trees may coincide. We then find a way to “average” these trees to obtain some summary tree.

A particularly convenient method of summarizing the results of replications is to utilize the majority-rule consensus tree (Margush and McMorris 1981). Briefly, each edge in a labeled elementary tree \( T \) can be associated with a bipartition—i.e., a partition of the leaves of \( T \) into two nonempty disjoint sets. These sets consist of the leaves in each of the two components of the graph obtained if the edge of interest is removed from \( T \).

In more detail, each edge \( e \) in \( T \) is such that removing the edge \( e \) (but not the vertices at either end) disconnects \( T \) into two connected graphs \( T_1 \) and \( T_2 \). Each species in \( S \) lies in either \( T_1 \) or \( T_2 \). Let \( A \) denote the set of labels of vertices of \( T_1 \), and let \( B \) denote the set of labels of vertices of \( T_2 \). Then, \( \{A, B\} \) is a bipartition of \( S \); i.e., \( A \cup B = S \), \( A \cap B = \emptyset \), \( A \neq \emptyset \), \( B \neq \emptyset \). Note also that \( T_1 \) consists of all edges and vertices which lie on any path \( P_{ab} \) joining labeled vertices \( a \) and \( b \), where \( a \) and \( b \) are both in \( A \). Similarly, \( T_2 \) consists of all edges and vertices that lie on any path \( P_{cd} \) joining vertices \( c \) and \( d \), where \( c \) and \( d \) are both in \( B \). Informally, we think of \( A \) as the labels on one side of the edge \( e \) and \( B \) as the labels on the other side of \( e \).

The order of the sets \( A \) and \( B \) in a bipartition is irrelevant, so \( \{A, B\} = \{B, A\} \). For example, if \( S = \{1, 2, 3, 4, 5\} \), then \( A = \{1, 2, 4\}, B = \{3, 5\} \) form a bipartition \( \{A, B\} = \{B, A\} \).

Suppose that there are \( m \) trees \( T_i \). We count the number of times each bipartition is obtained in one of the trees \( T_i \). A given bipartition is a consensus bipartition if it occurs in more than half of the trees \( T_i \); i.e., if it occurs strictly more than \( m/2 \) times. Margush and McMorris (1981) prove that there exists a unique tree \( C \) (their consensus tree) whose edges correspond precisely to the consensus bipartitions.

The consensus tree is a useful summary of the trees \( T_i \) obtained by replication. Note that it need not be completely resolved. It is useful to label each edge with a number indicating the percentage of the replicates in which the edge occurred. Every edge attaching a single leaf, of necessity, occurs in 100% of the trees \( T_i \). By the definition of the consensus tree, each edge must occur in more than 50% of the trees \( T_i \). The percentages obtained measure the degree of confidence in each edge. These numbers are not to be confused with bootstrap figures.

Building a Tree by Identifying the Strongest Signal

We return to the situation of Building a Tree by Adding Species One at a Time. Suppose that \( S \) is the set of all the species in question, and a tree \( W_k \) has been constructed with set \( S \), of species. We wish to insert into \( W_k \) a new species from \( S - S_k \). We heuristically proceed to find the species and placement with the “strongest signal” for correctness. Thus, we assume that \( W_k \) is correct, and we see which new placement of some new species gives the “strongest signal” for correctness. The insertion of species \( x \) into the tree \( W_k \) at placement \( i \) leads to a tree \( W(x, i) \). In Building a Tree by Adding Species One at a Time, I described how to compute the local inconsistency \( g(x, i) \) at the \( i \)th placement of \( x \). Recall that best placement \( P(x) \) is the \( i \) which minimizes \( g(x, i); \) the corresponding local inconsistency is \( \text{LI}(x) = g(x, P(x)) \). Let \( r(x) = g(x, j) \) for the second-best placement \( J \) of \( x \). Thus, \( r(x) \) is the local inconsistency of the placement other than \( P(x) \) with the lowest local inconsistency; we call it the “runner-up inconsistency.” Any placement of \( x \) other than placement \( P(x) \) would force us to contradict some \( T(J) \) by at least \( r(x) \) steps.

The meaning of these quantities may be clarified by a hypothetical comparison (table 2) of the placements of three species \( x, y, \) and \( z \) into a tree \( W \).

<table>
<thead>
<tr>
<th>Placement ( i )</th>
<th>( g(x, i) )</th>
<th>( g(y, i) )</th>
<th>( g(z, i) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>75</td>
<td>18</td>
<td>45</td>
</tr>
<tr>
<td>2</td>
<td>42</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>38</td>
<td>10</td>
<td>21</td>
</tr>
<tr>
<td>5</td>
<td>48</td>
<td>7</td>
<td>36</td>
</tr>
<tr>
<td>6</td>
<td>90</td>
<td>14</td>
<td>78</td>
</tr>
</tbody>
</table>

* \( g(x, i) \) is the local inconsistency of inserting species \( x \) at placement \( i \). Low values are best.

From table 2, we see that placement 1 of species \( x \) leads to a local inconsistency of 75. Thus, placement 1 of species \( x \) yields a tree \( W_k(x, 1) \) such that for some quartet \( J \) the tree length of \( W_k(x, 1) \) is \( 75 \) steps longer than the tree length of \( T(J) \). Accepting placement 1 as correct would amount to saying that \( T(J) \) gives the wrong relationship among the species in \( J \) and that \( W_k(x, 1) \) is correct even though it is \( 75 \) steps longer than \( T(J) \). By contrast, placement 3 yields a tree \( W_k(x, 3) \) such that for the worst choice of \( J \), we have that \( W_k(x, 3) \) is only \( 8 \) steps longer than \( T(J) \). The best placement of species \( x \) is clearly \( i = 3 \), so \( P(x) = 3 \), and the local inconsistency for \( x \) is \( \text{LI}(x) = 8 \). Note that \( r(x) = 38 \), since 38 is the smallest \( g(x, i) \) greater than 8. If we agreed to accept placement 3 of species \( x \) as true, we would need to contradict some quartet’s tree length by 8. Any other placement would force us to contradict some quartet’s tree length by 38 or more.

A similar argument applies to the placement of species \( y \). Here, the best placement is the fifth one, so \( P(y) = 5 \) and \( \text{LI}(y) = 7 \). We may compute \( r(y) = 10 \). Thus, the best placement of \( y \) forces us to contradict some quartet’s tree length by 7, and any other placement forces us to contradict some quartet’s tree length by at least 10.

For species \( z \), there is a tie between placements 2 and 3 for the best placement. The data do not support any unique best placement. Hence, \( P(z) \) is not defined, and \( \text{LI}(z) = 12 \).

It is apparent that the third placement of \( x \) is much more strongly supported than the fifth placement of \( y \), even though \( \text{LI}(y) < \text{LI}(x) \). Indeed, it is hard to rule out placement 3 or 4 of species \( y \), while no placement of \( x \) other than placement 3 seems tenable. The fact that \( r(x) \)
and \( LI(x) \) are widely separated means that there is really little ambiguity in which placement of \( x \) is correct. By contrast, the fact that \( r(y) \) and \( LI(y) \) are so close means that there is considerable doubt about the correct placement of \( y \). There is, from the data, no way to choose the placement of \( z \), although placements 2 and 3 are preferred over any of the others.

In general, if there is a unique placement \( i \) for which \( g(x, i) = LI(x) \), define the strength \( s(x) \) of the placement to be \( s(x) = r(x) - LI(x) \). The strength \( s(x) \) correlates with our confidence in the correctness of the placement \( P(x) \).

In our example, we have \( s(x) = 38 - 8 = 30 \), while \( s(y) = 10 - 7 = 3 \) and \( s(z) = 12 - 5 = 7 \). Our high confidence in placement 3 of \( x \) compared to placement 5 of \( y \) is reflected in that fact that \( s(x) \) is much greater than \( s(y) \). The fact that there is insufficient information for a unique placement of \( z \) is reflected in the fact that \( s(z) = 0 \).

These considerations lead to a simple heuristic procedure: At each stage given a tree \( W_k \) with set \( S_k \) of species, for each species \( x \) in \( S - S_k \), we compute each \( g(x, i), P(x), r(x), \) and \( s(x) \). Select a species \( x \) for which \( s(x) \) is maximal; i.e., select the species \( x \) with the strongest placement. We form a new tree \( W_{k+1} \) by inserting \( x \) into \( W_k \) with placement \( P(x) \).

In the example, with species \( x, y, \) and \( z \) in contention for placement, the strengths are, respectively, 30, 3, and 0. Hence, we would choose to place \( x \) into the tree using placement 3, because the strength of its signal was greatest.

**Comparison with Other Methods on Artificial Data Sets**

In this section, I present some results that compare the effectiveness of these methods with that of some other methods on artificial data sets for which the correct tree is known. A tree \( T \) with 12 taxa was selected which was suggested by G. Hoelzer (personal communication) and related to Lyons-Weiler and Hoelzer (1997). For various values of a parameter \( M \), at each node of \( T \), sequences of 3,000 nt were generated such that the number of mutations (with replacement) between the sequences at either end of an edge was \( M \), except for two edges at opposite ends of the tree, for which the number of mutations was \( 10M \). The transition/transversion rate ratio \( \kappa \) was 4.0, indicating that, for example, the transition from A to G was four times as likely as the transversion from A to T. The two long edges at opposite ends of the tree ensured that long-branch attractions (Felsenstein 1978) would pose a problem in performing the reconstructions. For each quartet of four taxa corresponding to the leaves of the tree, the likelihoods were computed for each topology using the HKY method (Hasegawa, Kishino, and Yano 1985), and the topology was selected for which the likelihood was maximal. These computations were performed exactly as in the program PUZZLE (Strimmer and von Haeseler 1996). The tree was then reconstructed in two manners: (1) by PUZZLE with 1,000 puzzling steps, yielding a tree \( U \), and (2) by averaging the results of 100 replications as in *Finding a Tree by Averaging the Trees from Many Replications*, yielding a tree \( W \), for which we used the function \( w(T(J)) = -\log L(T(J)) \), with \( L \) denoting the likelihood. Each tree \( U \) reconstructed by PUZZLE was compared with the correct tree \( T \) by computing \( d_p \), the number of quartets \( Q \) on which the trees \( U \) and \( T \) differ (i.e., the number of sets \( Q \) consisting of four taxa such that \( U|Q \neq T|Q \)). Since there are 495 quartets in this case, it follows that, necessarily, \( d_p \geq 495 \). Similarly, each tree \( W \) reconstructed by our method of 100 replications was compared with \( T \) by computing \( d_p \), the number of quartets \( Q \) such that \( W \) and \( T \) differ on \( Q \). When \( d_p = 0 \) or \( d_r = 0 \), respectively, the reconstructed tree agreed with the tree \( T \) used to generate the data.

For \( M \leq 500 \), both methods reconstructed the correct tree \( T \) for all cases studied. Thus, both methods were very effective in this range. We study a sample of 65 cases in the range 500 \( \leq M \leq 900 \), for which the reconstruction of \( T \) becomes problematic; this difficulty is understandable since, for example, some edges have as many as 9,000 mutations among 3,000 sites, greatly obscuring the original relationships. For the 65 cases, the sample mean of \( d_p \) was 90.69 with a standard deviation of 132.76, while the sample mean of \( d_r \) was 26.08 with a standard deviation of 48.77. Assuming that these samples were from normal populations, we find that a 99% confidence interval for the difference in the population means corresponding to \( d_p - d_r \) is (19.35, 109.9). Hence, we have high confidence that the superior performance of our method was not due to chance. On average, the reconstruction by PUZZLE was wrong on 90.69/26.08 = 3.48 times as many quartets as was the reconstruction by our replication method.

Out of the 65 cases, both methods simultaneously reconstructed the correct tree 28 times, or 43% of the time; PUZZLE reconstructed a tree closer to the correct tree than did our method 16 times (25%), while our method reconstructed a tree closer to the correct tree than did PUZZLE 21 times (32%). It is noteworthy, however, that when the wrong tree was reconstructed by PUZZLE, it tended to be much further from the correct tree than in cases in which the wrong tree was reconstructed by our method. This may be seen in the distribution of the numbers \( d_p - d_r \), as shown in figure 4. When \( d_p - d_r > 0 \), so that \( U \) (constructed by PUZZLE) is further from \( T \) than is \( W \) (constructed by our method), then \( d_p - d_r \) tends to be large, showing that \( U \) is much worse than \( W \). When \( d_p - d_r < 0 \), so that \( W \) is further from \( T \) than is \( U \), then \( d_p - d_r \) is relatively close to zero, showing that \( U \) is only slightly better than \( W \).

The method of reconstruction described in the preceding section need not always find a tree containing all the taxa. For the cases indicated in figure 4, such failure happened frequently. Thus, the method of the preceding section is not as good for automatic reconstruction of a tree as is that of *Finding a Tree by Averaging the Trees from Many Replications*. I find results from the preceding section illuminating, however, in exploratory work in which I interact with the computer step by step, be-
cause it clearly identifies where the difficult placements occur and also identifies the likely rival placements.

Our choice of a tree with some very long branches yields data sets sufficiently problematic that maximum parsimony does not work well. Trees were reconstructed by PAUP*4.0 b1 (Swofford 1998) for a sample of data sets corresponding to 50 ≤ M ≤ 700 by using heuristic search with the maximum-parsimony criterion. In no cases did PAUP* reconstruct the correct tree; indeed, in every case studied, the number of quartets on which the correct tree was computed. The method of Finding a Tree by Averaging the Trees from Many Replications by averaging 100 replications was also used to reconstruct a tree W, and the number d_W of quartets for which W differed from the correct tree was computed. The reconstructions were correct trees when d_W = 0 or d_W = 0, respectively. For M ≤ 500, both methods always reconstructed the correct tree. The histogram tells the frequency of the numbers d_W = d_W for 65 data sets with the range 500 ≤ M ≤ 900. Columns have width 20. To clarify the diagram, the 28 cases in which d_W = d_W = 0 are omitted. When d_W > 0, the tree from PUZZLE is better than the tree constructed by the methods of this paper, since d_W is closer to zero than is d_W. When d_W > 0, the methods of this paper performed better than PUZZLE. In 16 cases, PUZZLE performed better than the methods of this paper (d_W < 0), but only slightly better, since d_W and d_W are close. In 11 cases, PUZZLE performed worse than the methods of this paper (d_W > 0), but the performance was much worse, since d_W and d_W were usually widely separated. On average, our reconstructions were closer to the correct tree.

Results Utilizing Parsimony for Each Quartet and Using Many Replications

We started with the tree ((1 2) (8 9)), in which the pygmy and common chimps are grouped together while the gray and harbor seals are also grouped together, because this tree had the shortest tree length among all 1,820 quartets. We then added species one at a time, in each case selecting the placement with lowest local inconsistency. For 100 replications, we followed the procedure described in Finding a Tree by Averaging the Trees from Many Replications with random orders of the added species. We obtained 100 trees T for i = 1, . . . , 100. Among the 100 replicates, there were 20 distinct trees. One tree occurred 24 times among the 100 replicates, and another tree occurred 21 times. No other tree occurred as many as 10 times.

We computed the majority-rule consensus tree for these 100 replicates by the method of Margush and McMorris (1981). The tree is drawn in figure 5. It agrees with the tree called F1 in Willson (1998). It is also the second most common tree among the 100 replicates, appearing 21 times. This tree T has inconsistency I(T) = 25 and b(T) = 148, where b(T) is the number of “bad” quartets J such that c(T, J) > 0.

The most frequently obtained tree, occurring 24 times among the replicates, is the tree called F1 in Willson (1998). Willson (1998) shows that F1 is minimally inconsistent (using the criterion of maximum parsimony for each quartet). It differs from figure 5 only in the placement of the horse closer to the (cow, whale) group than to the seal group. It has inconsistency I = 23 and b = 172.

Both of these trees place the guinea pig among the rodents, unlike the maximum-parsimony tree given in D’Erchia et al. (1996).
Results by Successively Adding Species with the Strongest Signal

We next utilized the method of **Building a Tree by Identifying the Strongest Signal** to build a tree in such a manner that at each stage we use the placement with the strongest signal. Again, we started with the quartet \(((1\ 2)\ (8\ 9))\) that groups the chimps together and the seals together. For each quartet \(J\), we selected \(T(J)\) by the criterion of maximum parsimony. Table 3 summarizes the results of applying the procedure to this data set. The resulting tree is given in figure 6. This tree does not place the guinea pig with the rodents.

The same tree is obtained if instead each \(T(J)\) is selected using maximum likelihood, as computed by PUZZLE, rather than maximum parsimony. It is the same tree obtained in figure 1b of D’Erchia et al. (1996) from genetic distances using the stationary Markov method with nucleotide supergenes of protein-coding genes (but not by maximum parsimony). It also agrees with the tree \(F_2\) obtained in Willson (1998) using a different automated procedure utilizing absolute inconsistency measures. It is interesting that the method of **Building a Tree by Identifying the Strongest Signal** yields a different result from that of **Finding a Tree by Averaging the Trees from Many Replications** for this data set.

### Table 3

Results of Applying the Methods of **Building a Tree by Identifying the Strongest Signal** to the Data Set Described in D’Erchia et al. (1996), Yielding the Tree Given in Figure 6

<table>
<thead>
<tr>
<th>Tree or Species (x)</th>
<th>(\text{LI}(x))</th>
<th>(s(x)^*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(((1\ 2)\ (8\ 9)))</td>
<td>-846</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>-212</td>
<td>217</td>
</tr>
<tr>
<td>5</td>
<td>-75</td>
<td>81</td>
</tr>
<tr>
<td>10</td>
<td>-72</td>
<td>81</td>
</tr>
<tr>
<td>4</td>
<td>-24</td>
<td>40</td>
</tr>
<tr>
<td>7</td>
<td>-21</td>
<td>30</td>
</tr>
<tr>
<td>6</td>
<td>-18</td>
<td>178</td>
</tr>
<tr>
<td>12</td>
<td>-3</td>
<td>14</td>
</tr>
<tr>
<td>11</td>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>13</td>
<td>25</td>
<td>7</td>
</tr>
<tr>
<td>16</td>
<td>30</td>
<td>5</td>
</tr>
</tbody>
</table>

* Species \(x\) is selected at each stage so as to maximize the signal strength \(s(x)\). The resulting local inconsistency is \(\text{LI}(x)\).

Conclusions and Extensions

This paper has described a heuristic method for building phylogenetic trees based on first determining a preferred tree \(T(J)\) for each collection \(J\) consisting of four species from the collection in question. The method requires that the choice of the preferred tree \(T(J)\) be made on the basis of minimizing a numerical function \(w\). Common choices of \(w(T)\) might be the tree length of \(T\), if one wishes to accept a criterion of maximum parsimony, or \(w(T) = -\log L(T)\), if, more generally, one wishes to accept a criterion of maximum likelihood,
where \( L(T) \) is the likelihood of \( T \). The method is versatile in that the procedures generalize readily to other choices of \( w(T) \).

Once the preferred trees \( T(J) \) are known, the next problem is to combine these trees in some manner to obtain a phylogenetic tree for all of the species. Inductively, we build a tree by adding one species at a time. Our methods use a numerical measure of the local inconsistency to find the best placement of each species. At each stage we may add a species \( x \) into a placement with the strongest signal strength as measured by \( s(x) \). Alternatively, we may choose a random order in which the species are added, add the species in this order each to the placement with lowest local inconsistency, and then summarize the results by giving a consensus tree for the various replicates we obtain. By either approach, trees \( T(J) \) for quartets in which we have low confidence because their choices were nearly tied are utilized only when trees in which we have higher confidence do not determine the placement.

The method presented in this paper differs from that of Willson (1998) because of the finer resolution in placing species. Willson (1998) places species so as to minimize the global inconsistency \( I(T) \) at each stage. At the later stages of a computer run, however, the global inconsistency \( I(T) \) does not yield the fine discrimination provided by the local inconsistency \( LI(x) \) and the signal strength \( s(x) \).

The results of the method presented in this paper may be compared with those of the program PUZZLE, described in Strimmer and von Haeseler (1996) and Strimmer, Goldman, and von Haeseler (1997), related to the quartet puzzling option in PAUP* (Swofford 1998). PUZZLE uses a quartet method in which maximum likelihood is utilized for each quartet. The trees for the various quartets are combined into a phylogenetic tree, however, in a manner different from ours, making no use of measures of local inconsistency and signal strength in order to choose the placements. The computer studies in Comparison with Other Methods on Artificial Data Sets show that both these methods of reconstructing a tree give the correct tree when the mutation rate is low, even when maximum parsimony reconstructs the wrong tree. When the mutation rates are higher, however, incorrect trees reconstructed by PUZZLE tend to be further from the correct tree than are incorrect trees reconstructed by our method.

Our method holds potential as an efficient means to build phylogenetic trees simultaneously for many taxa. Suppose, for example, that we have high confidence in a method for selecting the correct dogbone for a particular quartet (i.e., in a function \( w(T) \)). In that case, the placement procedure may be modified by carefully selecting a few quartets \( J \) which would uniquely determine the placement of a new species. In this manner, we could obtain phylogenetic information at a fast rate, and we could find a common tree for a large collection of taxa.

In cases in which a common alignment of many taxa is problematic, our method requires only a common alignment for a single quartet at a time. The errors in finding a common alignment of four species should be smaller than the errors in finding a common alignment of many more species. Optimized programs to align four species may then be utilized instead of general-purpose alignment programs.

At the website http://www.public.iastate.edu/~swillson/software.html, software may be found to accomplish many of the algorithms of this paper. The website contains source code written in C as well as sample data sets and instructions.

Acknowledgments

I wish to thank Gavin Naylor for well-considered criticisms concerning this subject. I thank L. A. Willson for suggestions about representing results graphically. I also thank the anonymous referee and the associate editor Michael Hendy for useful recommendations to improve the paper.

Literature Cited


Michael Hendy, reviewing editor

Accepted February 8, 1999