Chapter 6
Inferring Molecular Phylogeny

6.1 Introduction

The task of molecular phylogenetics is to convert information in sequences into an evolutionary tree for those sequences. A great (and ever increasing) number of methods have been described for doing this, which raises the inevitable question of how to come to grips with this plethora of possibilities. Two ways which seem useful to us are either to divide the methods by how they handle data, or to divide them by the approach taken when building trees. Both these divisions can help us appreciate the differences among the various tree building techniques. Given the rapid development of methods in this field, we cannot hope to cover completely every method that has been proposed, nor would this necessarily be helpful. Our goal is to cover the major methods, and to show how they interrelate.

6.1.1 Kinds of data: distances versus discrete characters

This division is based on how the data are treated; distance methods first convert

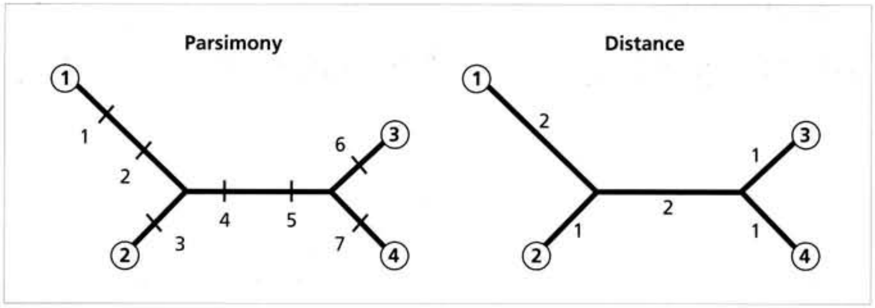


Fig. 6.1 A parsimony tree and a distance tree for the same sequence data. Note that both trees have the same topology and branch lengths, but that the parsimony tree identifies which site contributes to the length of each branch.

aligned sequences into a pairwise distance matrix, then input that matrix into a tree building method, whereas discrete methods consider each nucleotide site (or some function of each site) directly. As an example, consider the following sequences and corresponding (uncorrected) distance matrix:

sequences

sequences	sites 1	2	3	4	5	6	7
1	T	T	A	T	T	A	A
2	A	A	T	T	T	A	A
3	A	A	A	A	A	T	A
4	A	A	A	A	A	A	T

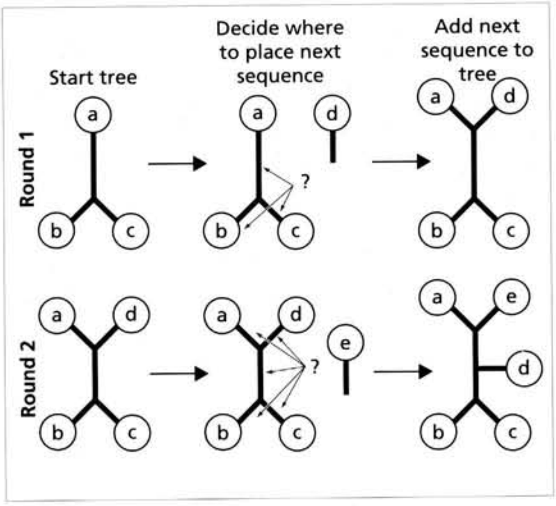
distances

sequences	**1**	**2**	**3**
2	3		
3	5	4	
4	5	4	2

The trees obtained by parsimony (a discrete method) and minimum evolution (a distance method) are identical in topology and branch lengths (Fig. 6.1). The parsimony analysis identifies seven substitutions and places them on the five branches of the tree. The distance tree apportions the observed distances between the sequences over the branches of the tree, and you can see that both methods arrive at the same estimates of the lengths of each branch. Under parsimony each of the seven sites requires one change, for a total of seven changes; if we sum the branch lengths on the distance tree we obtain the same value: $2 + 1 + 2 + 1 + 1 = 7$. Note, however, that the parsimony tree gives us the additional information of which site contributes to the length of each branch. Once we convert sequences into distances we lose this information. Furthermore, discrete methods allow us to infer the attributes of extinct ancestors, in this case extinct nucleotide sequences. These reconstructed ancestors can offer insights about molecular evolution (see Chapter 5).

6.1.2 Clustering methods versus search methods

Another way of dividing tree building methods is by the way they construct


Start tree
Decide where to place next sequence
Add next sequence to tree
Round 1
Round 2
a
b
c
d
e
?
Fig. 6.2 An example of how a clustering method builds a tree. The tree is constructed by starting with the tree for three sequences, then adding each remaining sequence in turn until finally all sequences have been added.

trees. **Cluster methods** follow a set of steps (an algorithm) and arrive at a tree. For example, if we have five sequences we might start with three of them (remember that there is only one possible unrooted tree for three sequences) and decide where to place the fourth sequence. Given the resulting tree for four sequences, we then decide where to add the fifth and last sequence to our tree (Fig. 6.2).

Clustering methods have the advantage of being easy to implement, resulting in very fast computer programs. Furthermore they almost always produce a single tree. This combination of speed and an apparently unambiguous answer is naturally very appealing, and accounts for much of the sustained popularity of clustering methods. However, they have some severe limitations as analytical tools. The result obtained from simple clustering algorithms often depends on the order in which we add the sequences to the growing tree. For example, had we started with sequences b, d and e rather than a, b and c in Fig. 6.2 we might have arrived at a different tree. But the biggest limitation is that cluster methods do not allow us to evaluate competing hypotheses, they merely produce a tree. It may be that two different trees could explain our data equally, or nearly equally, well. Unless we have some way to measure the fit between tree and data, we will not be aware of this.

Tree-building methods in the second class use **optimality criteria** to choose among the set of all possible trees (Fig. 6.3). This criterion is used to assign to each tree a 'score' or rank which is a function of the relationship between tree and data (examples include maximum parsimony and maximum likelihood). Optimality methods have the great advantage of requiring an explicit function that relates data and tree (for example, a model of how sequences evolve). These methods also allow us to evaluate the quality of any tree, hence we can compare how well competing hypotheses of evolutionary relationship fit the data. The Achilles' heel of optimality methods is that they are computationally

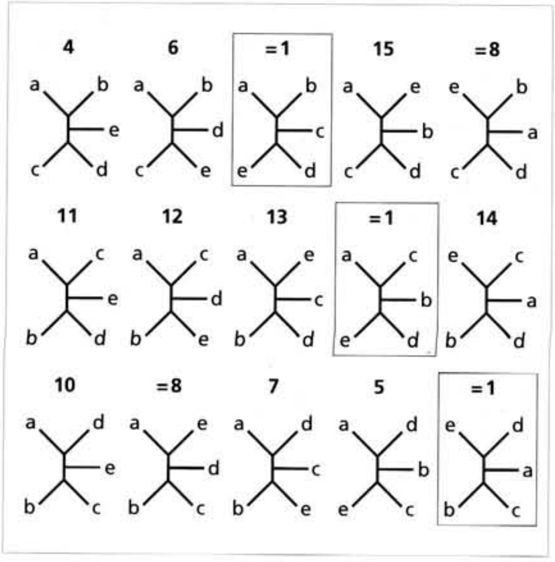


Fig. 6.3 An optimality method assigns to every possible tree a 'score' based on some measure of how the data relate to the tree. Based on this score each tree is ranked in order and typically the best tree (or trees) is used as the estimate of the phylogeny. In this example, there are three optimal trees (indicated by the ranking of first equal [=1]).

very expensive. An optimality method poses two problems that must be solved: firstly, for a given data set and a given tree, what is the value of the optimality criterion for that tree? For example, what is the minimum number of evolutionary events required to explain the observed data? Secondly, which of all the possible trees has the maximum value of this criterion? For example, which tree requires the fewest evolutionary events? The first problem is generally fairly straightforward; however, the second problem is rather more difficult as it belongs to a class of problems called NP-complete (see Box 6.1).

Box 6.1 NP-completeness and the problem of finding the best evolutionary tree

Despite the best efforts of mathematicians and computer scientists there is a set of problems called NP-complete (NP = non-deterministic polynomial) for which no efficient algorithms for their solution are known to exist. Members of this class of problems are all variants on the same problem, in that if one problem could be solved efficiently then all could be. The problem of finding the optimal evolutionary tree for a variety of criteria, including minimum evolution and maximum parsimony, is known to be NP-complete. In practical terms this means that for any reasonable number of sequences (e.g. more than 20) it is often impossible to guarantee that the optimal tree has in fact been found. Consequently, in many cases we must rely on heuristics (a euphemism for 'quick and dirty'). Trees found by such methods may turn out to be far from optimal, and the conclusions drawn from such trees may be somewhat suspect. A salutary example is the controversy over the geographic origins of human mitochondrial DNA in which different workers obtained quite different trees from different heuristic searches.

While for small numbers of sequences (e.g. no more than 20) it is often possible to find the optimal tree (or trees), in many cases this is not feasible, in which case we have to rely on heuristic methods. These are strategies designed to explore some subset of all the possible trees, in the hope that that subset will contain the optimal tree. A typical heuristic strategy is to start with a tree (which may be obtained using a simple clustering algorithm, or even chosen at random) and rearrange it, keeping any rearrangement that produces a better tree. Such algorithms are often called 'hill-climbing'. Imagine that you are in a valley and want to climb the highest hill, but you cannot see (some malicious text-book author has forced you to wear a blindfold). A reasonable strategy for climbing the highest hill would be to take a step, which may take you either a little bit downhill, a little bit uphill, or may keep you on a flat surface; if the step was downhill you are clearly not going up a peak, hence you retrace your step and try again; if the step is uphill you may be on the right track, and may wish to take another step in the same direction. If no step in any direction takes you up, but at least one takes you neither up nor down, then you are on a plateau. This plateau may be the top of the highest hill in the region, in which case you have succeeded; however, it may be merely a local high-point—being blindfolded you cannot tell.

Translating this analogy into the problem of tree searching, the 'hills' represent sets of locally optimal trees, and your steps are the tree rearrangements. Figure 6.4 shows two 'hills' or 'islands' of trees, and two possible starting trees, 1 and i. By rearranging the starting tree, and keeping only those trees that are better under our optimality criterion, we arrive at two different islands (a and b), one of which (b) is better than the other. The search that landed on island a would be unable to reach island b as to get there it would have to accept a suboptimal tree (tree 5) before reaching island b. If a set of possible trees contains more than one island then heuristic methods may land on a suboptimal island, and the optimal island goes undiscovered.

6.1.3 Subtree methods—assembling larger trees from smaller ones

Most algorithms for finding optimal phylogenetic trees use various techniques for rearranging trees to explore the phylogenetic 'landscape' comprising possible solutions (Fig. 6.4). If we can readily compute a score for each tree this approach is often reasonable. However, for some optimality criteria, such as maximum likelihood discussed below, computing a score for even a single tree is time consuming. Because the effectiveness of a heuristic search depends in part on how many trees we are willing to examine, this can hamper our search.

An alternative approach that has been used for both distance and discrete character methods is to divide the set of sequences into smaller sets and find the optimal tree for these subsets, which are then combined to form a larger tree. The smallest useful subset of an unrooted tree comprises four sequences (a **quartet**). For n sequences there are $\binom{n}{4} = n(n-1)(n-2)(n-3)/24$ possible

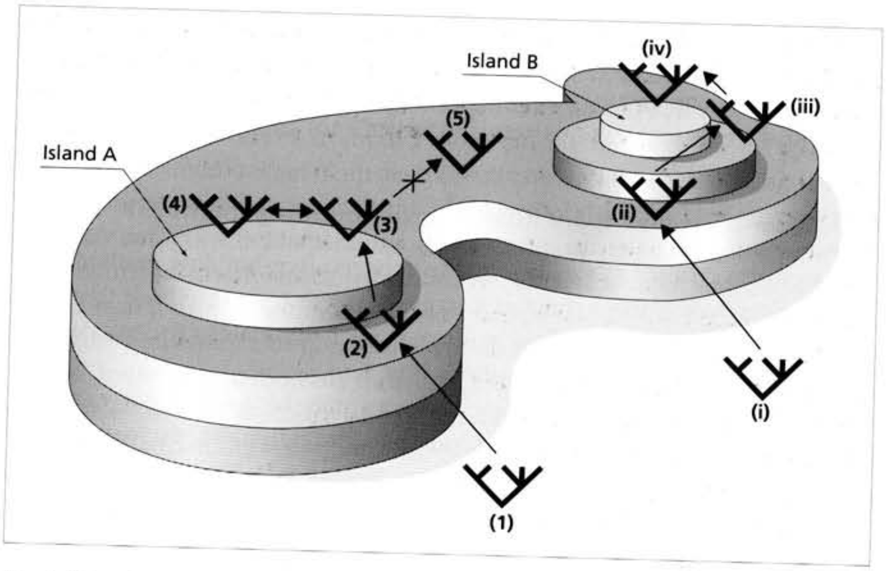


Fig. 6.4 Landscapes and the problem of islands of trees (locally optimal sets of trees). A hill climbing algorithm that started from tree 1 would succeed in finding trees 3 and 4 (which comprise island a), but would fail to discover that tree iv (island b) was even better, because to get to that tree it would have to cross a plateau of trees that were worse than trees 3 and 4. However, a search starting from tree i would succeed in finding the best tree. If the set of possible trees contains more than one island then heuristic methods may land on a suboptimal island, and the optimal island will not be discovered. After Maddison (1991).

quartets. Each quartet has three possible unrooted trees (see Table 2.1, p. 18) (Fig. 6.5). Each tree partitions the set of sequences {A, B, C, D} into two pairs: for example Q_1 corresponds to {A, B} and {C, D}. These two sets are each other's **neighbours**. Quartet-based tree building methods follow these two steps:

1 For each quartet identify which of the three possible trees is optimal for those four sequences.

2 Take all the four-sequence trees from step 1 and assemble them into a tree. Given perfect data the second step is trivial as there will be only one tree that all the subtrees will agree on. However, given homoplasy it may be that not all subtrees can be combined into one tree, in which case we want the tree that

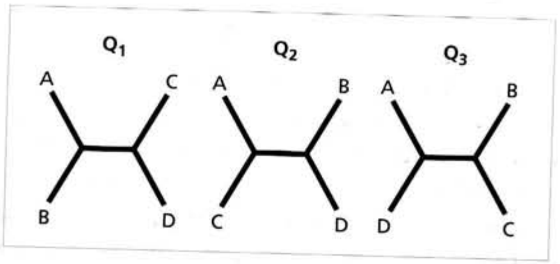


Fig. 6.5 The three possible trees for four taxa (a quartet).

accommodates the greatest number of the quartets. Finding this tree is, alas, also an NP complete problem (see Box 6.1) so it suffers from the same computational complexity that besets other methods. However, fast heuristic algorithms for assembling quartets into larger trees are available.

6.1.4 How do we compare different tree-building methods?

Based on the distinction we have made between tree-building methods that use distances versus those that use discrete characters, and methods that use a clustering algorithm versus those with an explicit optimality criterion, we can classify some commonly used methods (Fig. 6.6).

Note that methods from different classes may be related. For example, UPGMA can be considered a heuristic method for finding the best least squares ultrametric tree, and similarly neighbour joining is a heuristic method for estimating the minimum evolution tree.

Given the range of tree-building methods available, how can we decide which ones are better than others? David Penny and colleagues have suggested five desirable properties a tree-building method should have:

- **efficiency** (how fast is the method?);
- **power** (how much data does the method need to produce a reasonable result?);
- **consistency** (will it converge on the right answer given enough data?);
- **robustness** (will minor violations of the method's assumptions result in poor estimates of phylogeny?);
- **falsifiability** (will the method tell us when its assumptions are violated, i.e. that we should not be using the method at all).

Efficiency is effectively the time in which a computer program can find a tree using a given method. As we have seen above most, if not all, optimality

		Type of data	
		Distances	Nucleotide sites
Tree-building method	Clustering algorithm	UPGMA Neighbour joining	
	Optimality criterion	Minimum evolution	Maximum parsimony Maximum likelihood

Fig. 6.6 Some common phylogenetic methods classified by the method used to build the tree, and by the type of data used.

methods are NP
algorithms that
result, we have
optimality crite
the most parsim
of magnitude m
the same data.
time, heuristic s
than a search u
when the optim

The *power* o
before we can
might be theor
to be sequenced
is whether the
This desirable
we kept feedin

All tree-bu
the evolution
method retur
assumed a m
about evolutio
its underlying
know whethe
that is, the me
clock when t
that assumpt
and use anot

The ideal
method does
or more of th
is extremely
molecular cl
its chosen m
chapter we

6.2 Dista

Distance me
distance be
reconstruct
relationship
distance is a
are rarely, i

methods are NP-complete (see Box 6.1) and hence efficient tree searching algorithms that guarantee to find the best tree are unlikely to be found. As a result, we have to rely on heuristics for all but the smallest problems. Some optimality criteria can be evaluated more quickly than others; for example, the most parsimonious set of nucleotide substitutions can be calculated orders of magnitude more quickly than the likelihood of the same tree giving rise to the same data. One practical consequence of this is that in the same period of time, heuristic searches using parsimony can explore a much larger set of trees than a search using likelihood. Subtree methods (section 6.1.3) may be used when the optimality criterion is time consuming to compute.

The *power* of a method is a measure of how much data we need to collect before we can be reasonably sure of arriving at the correct result. A method might be theoretically very appealing, but if it requires huge numbers of sites to be sequenced then it may not be of much practical use. Another consideration is whether the method will converge on the true tree as we add more data. This desirable property is *consistency*; an inconsistent method would fail even if we kept feeding it more data.

All tree-building methods make (implicit or explicit) assumptions about the evolutionary process. Violation of these assumptions may result in a method returning a poor estimate of phylogeny, for example a method that assumed a molecular clock when there was none may be very misleading about evolutionary relationships. The sensitivity of a method to violations of its underlying model is a measure of its *robustness*. Ideally, we would like to know whether these violations are sufficient to rule out a particular model, that is, the method is *falsifiable*. A falsifiable method that assumed a molecular clock when there was none would allow us to test the clock assumption; if that assumption was found wanting then we would abandon that method and use another that did not make the clock assumption.

The ideal tree-building method would meet all five criteria, but such a method does not exist, nor is it likely to. All current methods emphasise one or more of these criteria at the expense of the remainder. For example, UPGMA is extremely fast (efficient) but is not robust to its implicit assumption of a molecular clock, whereas maximum likelihood is consistent (with respect to its chosen model of evolution) but computationally very intensive. In this chapter we will evaluate each method based on these five criteria.

6.2 Distance methods

Distance methods are based on the idea that if we knew the actual evolutionary distance between all members of a set of sequences, then we could easily reconstruct the evolutionary history of those sequences. This follows from the relationships between distances and trees outlined in Chapter 2: evolutionary distance is a tree metric and hence defines a tree. In practice, however, distances are rarely, if ever, exactly tree metrics, and hence one class of 'goodness of fit' methods seeks the metric tree that best accounts for the 'observed' distances (i.e. the pairwise distances calculated between the sequences). The second class of method seeks the tree whose sum of branch lengths is the minimum ('minimum evolution').

6.2.1 Goodness of fit measures

The goodness of fit F between observed distance d_{ij} and tree distances p_{ij} (see Chapter 2) for each pair of sequences i and j is given by

$$F_\alpha = \sum_{1 \le i < j \le n} \left| d_{ij} - p_{ij} \right|^\alpha \tag{6.1}$$

where α can take various values. If $\alpha = 1$ then the criterion is Farris's f statistic, if $\alpha = 2$ then F is the least-squares-fit criterion. As an example of the latter criterion, consider the distance matrix between hominoids shown in Table 6.1 and the corresponding additive tree (Fig. 6.7).

Table 6.1 Kimura 2-parameter distances between hominoid sequences (above diagonal) and tree distances obtained by least squares (below the diagonal) for the tree shown in Fig. 6.7. Tree distances larger than the observed distances are shown in bold, tree distances smaller than the observed are shown in italics.

	Human	Chimp	Gorilla	Orang-utan	Gibbon
Human	–	0.09190	0.1083	0.1790	0.2057
Chimp	0.0919	–	0.1134	0.1940	0.2168
Gorilla	*0.1068*	**0.1151**	–	0.1882	0.2170
Orang-utan	**0.1816**	*0.1898*	**0.1893**	–	0.2172
Gibbon	**0.2078**	*0.2160*	*0.2155*	0.2172	–

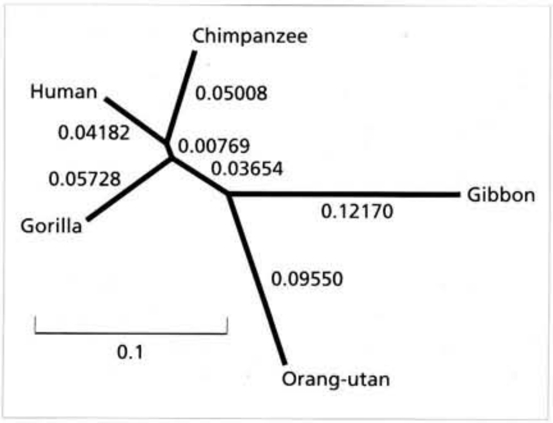


Fig. 6.7 Additive tree for hominoid mtDNA sequences showing branch lengths computed using least squares. The pairwise tree distances for this tree are given in the lower left triangle of Table 6.1.

The observed and tree distances are in close agreement, but note that some tree distances are larger than the observed, and some are less than the observed. While tree distances can be larger than observed distances due to homoplasy (i.e. multiple substitutions), the reverse is counter intuitive as tree distances less than those we observe between sequences imply that less evolutionary change took place than we actually observe. This contradiction (discussed further on p. 186) has led some workers to abandon the use of distance methods, or to use other methods of computing branch lengths from distances (see below).

In the example just given we were fitting an additive tree with $(2n-3)$ branches to $\binom{n}{2} = n(n-1)/2$ pairwise distances. However, measures of fit can also be applied to ultrametric trees, in which case there are $(n-1)$ independent branch lengths to be estimated. That there are fewer parameters to be estimated for an ultrametric tree, which has one more branch than an unrooted additive tree, may seem paradoxical, but this is due to the constraint that all terminal taxa are equidistant from the root. This constraint is equivalent to postulating a 'molecular clock' (a concept which is discussed in detail in Chapter 7). For example, in the ultrametric tree shown in Fig. 6.8 the branches e_1 and e_2 must be of the same length as they both represent the same interval of time (i.e. the time since sequences A and B diverged).

If the distances between sequences are ultrametric then both the ultrametric and additive trees would fit those distances equally well (indeed, the additive tree would be identical to the ultrametric tree). The greater the departure from the molecular clock, the more the data will depart from being ultrametric, and the greater the difference in fit between additive and ultrametric trees, the additive tree always being a better fit as it lacks the constraints imposed by the ultrametric tree. This property is employed by some tests of

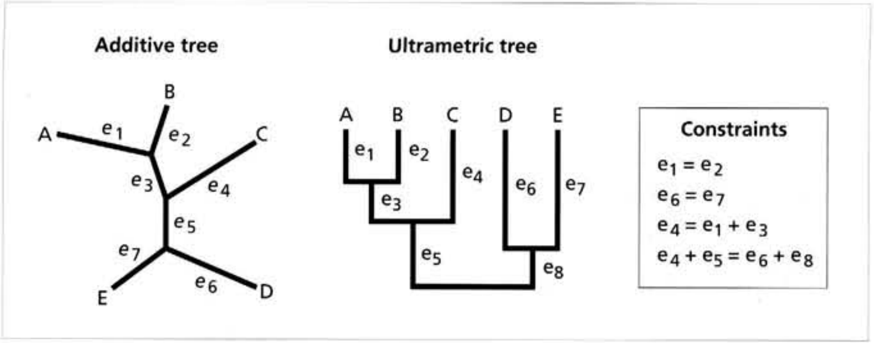


Fig. 6.8 Additive and ultrametric trees for the same sequences. Both trees specify the same cladistic relationships among the taxa, but whereas the additive tree has $(2n-3=7)$ independent branches the ultrametric tree has only $(n-1)=4$ because some branches are constrained to be equal to others, or to combinations of others. This constraint is equivalent to a molecular clock.

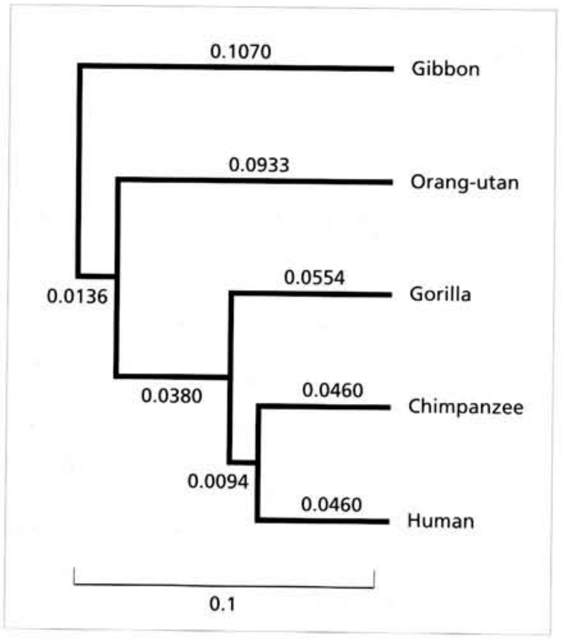


Fig. 6.9 Ultrametric tree for hominoid mtDNA with least squares branch lengths computed from the Kimura 2-parameter distances shown in the upper right triangle of Table 6.1. Compare with the additive tree in Fig. 6.7.

the molecular clock, such as the likelihood ratio test described later in this chapter (see p. 198).

Figure 6.9 shows an ultrametric tree for the same distances used to compute the additive tree in Fig. 6.7. The two trees specify the same cladistic relationships; however, the ultrametric tree also supplies a root (between the gibbon and the great apes including humans) and the branch lengths are slightly different. For instance, the additive tree apportions more evolutionary change to chimpanzee mtDNA than to human mtDNA, whereas in the ultrametric tree these two sequences are constrained to have exactly the same amount of evolutionary change since their divergence.

6.2.2 Minimum evolution

Given an unrooted metric tree for n sequences there are $(2n - 3)$ branches, each with length e_i. The sum of these branch lengths is the **length** L of the tree:

$$L = \sum_{i=1}^{2n-3} e_i \tag{6.2}$$

The minimum evolution tree (ME) is the tree which minimises L. This method is similar in spirit to parsimony, which we shall discuss below; however, the length in this case is computed from the pairwise distances between the sequences rather than from the fit of individual nucleotide sites to a tree. To use this method we need to be able to compute the length of any tree. However, these lengths are not always biologically valid (see p. 186).

Linear programming

Linear programming is a widely used mathematical technique to find the optimal solution to a problem, given a set of constraints (most spreadsheet programs can perform linear programming). When applied to finding the length of a tree, the two constraints that we must satisfy are that all the branch lengths are non-negative (i.e. $e_i \geq 0$ for all i), and that for any pair of sequences the tree distances are never less than the observed distances (i.e. $p_{i,j} \geq d_{i,j}$ for all i and j). Table 6.2 gives the observed pairwise distances for the hominoid DNA sequences, and the corresponding tree distances obtained for the tree shown in Fig. 6.10. The tree distances are in good agreement with the observed distances. Note that where the tree distances differ from the observed differences (e.g. between human and gibbon) the tree distances are always greater than the observed distances, satisfying the condition that the amount of evolutionary change inferred cannot be less than that which we have observed.

Table 6.2 Observed pairwise distances (p) between hominoid sequences (above diagonal) and tree distances computed by linear programming (below diagonal). Tree distances that differ from the observed are marked in bold. The corresponding tree is shown in Fig. 6.10.

	Human	Chimp	Gorilla	Orang-utan	Gibbon
Human	–	79	92	144	162
Chimp	79	–	95	154	169
Gorilla	92	**102**	–	150	169
Orang-utan	144	154	150	–	169
Gibbon	**163**	**173**	169	169	–

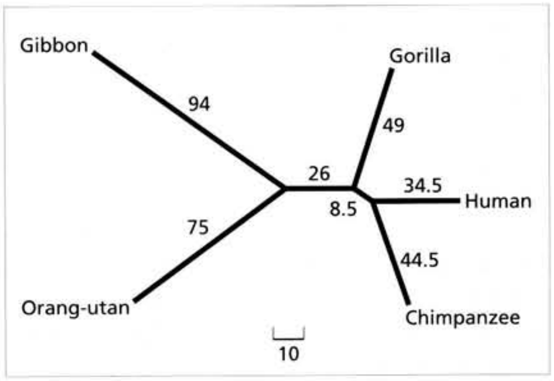


Fig. 6.10 Minimum evolution tree for the hominoid sequences with the branch lengths computed from the observed pairwise distances using linear programming. The total length of this tree is 331.5.

Least squares

More commonly, the branch lengths of the minimum evolution tree are estimated using least-squares methods. The branch lengths are estimated in the same way as for goodness of fit measures; however, rather than compare the fit of the observed distances the least squares branch lengths are added together to give the length of the tree.

6.2.3 Algorithms for finding distance trees

As with discrete characters, finding the optimal distance tree is computationally difficult. For small numbers of taxa exact methods can be used; for larger numbers we must rely on heuristics, of which there are a large number, some of which have largely been used only with distance data. We mention three here.

Neighbourliness

Neighbourliness makes use of the additive condition (see Chapter 2) where, given any four sequences, it is possible to label them A–D such that the distances between them obey this condition:

$$d(\text{A, B}) + d(\text{C, D}) \leq d(\text{A, C}) + d(\text{B, D}) = d(\text{A, D}) + d(\text{B, C}) \qquad (6.3)$$

Recalling from Chapter 2 that the additive condition defines a tree, this equation corresponds to the tree Q_1 shown in Fig. 6.5. On this tree A and B, and C and D are neighbours with respect to each other, hence the name of the method. Given data for n sequences that are perfectly additive (i.e. perfectly tree-like) we could use equation (6.3) to identify the additive subtrees for the data and from them construct the phylogeny for all n sequences. For actual data additivity may not hold, in which case we could seek the tree that has the greatest number of quartets for which equation (6.3) holds.

Neighbour joining

Neighbour joining (NJ) is a widely used method for tree building which combines computational speed with uniqueness of result—most implementations give a single tree. These two attributes (i.e. getting one tree, fast) have made it seem very appealing. Neighbour joining is a clustering method rather than an optimality method, and hence suffers from the limitation that it does not optimise a criterion of fit between tree and data. However, it is a good heuristic method for estimating the minimum evolution tree. One strategy for finding the ME tree is to first compute the NJ tree, then see if any local rearrangement of the NJ tree produces a shorter tree. Note that this strategy is not guaranteed to find the ME tree, for the reasons given in section 6.1.2. However, in practice the NJ tree is often the same or very similar to the ME tree.

Unweighted pair group method with arithmetic means (UPGMA)

UPGMA is one of the few tree-building methods that constructs an ultrametric tree (see Chapter 2). In an ultrametric tree all the tips are equidistant from the root of the tree, which is equivalent to assuming a molecular clock (see Chapter 7). Indeed, UPGMA is best viewed as a heuristic for finding the least squares ultrametric tree for a distance matrix.

Box 6.2 Are distance methods 'phenetic?'

A criticism often levelled at distance methods is that they are 'phenetic', and therefore inferior to other, 'phylogenetic', methods. Our view is that phenetics is a philosophy of systematics that eschews phylogenetic analysis as a search for the unknowable (because, with very few exceptions, no one has observed a phylogeny unfold over time). Hence phenetics favours summarising observed features of organisms using 'maximally informative' or 'predictive' classifications (which may or may not be actual phylogenies; if they are, this is merely a happy accident). Typically such classifications were constructed by computing a measure of similarity between organisms, then doing a cluster analysis on those similarities. Phenetics was rejected by most systematists by the late 1970s, largely because: (1) estimating evolutionary trees was thought to be a more worthwhile goal; (2) techniques (such as cladistics) existed which claimed to be able to estimate such trees; and (3) phenetics was unable to provide unambiguous reasons for choosing between a plethora of measures of similarity and clustering algorithms. However, if the goal is to reconstruct phylogeny then there are clear reasons for favouring one measure of similarity over another, and for favouring particular methods for converting distances into trees. This does not mean that all criticisms of distances raised by opponents of phenetics do not apply in this phylogenetic context, rather, we suggest that the criticism that a method is 'phenetic' is not, by itself, meaningful.

6.2.4 Objections to distance methods

In considering distance methods we should distinguish between methods for constructing the trees and methods for obtaining the distances. If the estimates of evolutionary distance are poor then the performance of a distance method may be adversely affected, which may not be a true reflection of the merits of the tree building method itself. Furthermore, many existing computer programs for finding distance trees lack the sophistication of equivalent programs for discrete data, making it difficult to determine if there is more than one optimal tree for example, or how many trees are nearly as good as the optimal tree. Again, this is a limitation of existing software, not distance methods as such. Hence in this section we consider objections to the use of distance data *per se*.

The major objections to distance methods are:

- summarising a set of sequences by a pairwise distance matrix loses information;
- branch lengths estimated by some distance methods may not be evolutionarily interpretable.

Distances lose information

If the original data are in the form of distances, such as those obtained from DNA hybridisation studies, then we have no other option but to use distances. However, if we have the sequences we have the option of analysing them directly or converting them into distances (Chapter 5). Doing the latter may lose information. For example, once converted to distances, we cannot trace the evolution of individual sites, or categories of sites on a tree, we have only an overall estimate of the relationship between tree and data. This can be clearly seen in Fig. 6.1 where the parsimony analysis of the original sequence data allows us to locate where in the tree each site changes, whereas the distance tree merely tells us how much change occurs along each branch.

Another way of expressing the loss of information inherent in converting sequences to distances is to compare the number of possible distance matrices with the number of possible data sets—the latter greatly exceed the former.

Uninterpretable branch lengths

It is possible to reconstruct branch lengths from distances that are mathematically valid but biologically difficult to interpret. For example, on p. 183 we used linear programming to fit observed distances between hominoid sequences to a tree. The length of that tree was 331.5 substitutions. This value raises two problems, the first being how to interpret 0.5 substitutions—a nucleotide can either be substituted or not; we cannot have half a substitution. One response to this difficulty is that branch lengths can represent two quite different quantities: the *expected* amount of evolutionary change and the *actual* or realised amount of change. For example, if we have a branch in a tree that corresponds to an interval of 1 Myr, and the rate of nucleotide substitution is 2.5% per Myr, then given a sequence of 100 bases in length we expect on average 2.5 substitutions to have occurred. Obviously, we cannot have half a substitution, so in reality some whole number of substitutions would have occurred, say 2 or 3 (or some other whole number). Under this interpretation, a branch length of 2.5 is entirely reasonable.

The second problem is that the figure of 331.5 substitutions is biologically not possible for this data set. As we shall see below when we discuss maximum parsimony, the minimum number of substitutions that must have occurred in the evolution of these particular sequences is 353. The tree length obtained by linear programming is internally consistent—but biologically impossible. In much the same way, when estimating branch lengths using least squares we can obtain tree distances that are less than the distances observed between the sequences (e.g. Table 6.1). Again, we cannot have less evolutionary change than we actually observe in the data.

6.3 Discrete methods

In contrast to distance methods, discrete methods operate directly on the sequences, or on functions derived from the sequences, rather than on pairwise distances. Hence they endeavour to avoid the loss of information that occurs when sequences are converted into distances. The two major discrete methods are **maximum parsimony** (MP) and **maximum likelihood** (ML). Maximum parsimony chooses the tree (or trees) that require the fewest evolutionary changes. Maximum likelihood chooses the tree (or trees) that of all trees is the one that is most likely to have produced the observed data. We also consider two methods that treat the data not as individual sites but as 'splits' (see Chapter 2): **spectral analysis** and **split decomposition**.

6.4 Maximum parsimony

The data for maximum parsimony comprise individual nucleotide sites. For each site the goal is to reconstruct the evolution of that site on a tree subject to the constraint of invoking the fewest possible evolutionary changes. Consider the following four sequences:

1 ATATT
2 ATCGT
3 GCAGT
4 GCCGT

Figure 6.11 shows the unrooted tree ((1,2),(3,4)) and two possible reconstructions of the evolution of the first site on that tree. In each reconstruction we have postulated which nucleotide the two internal nodes (ancestral sequences)

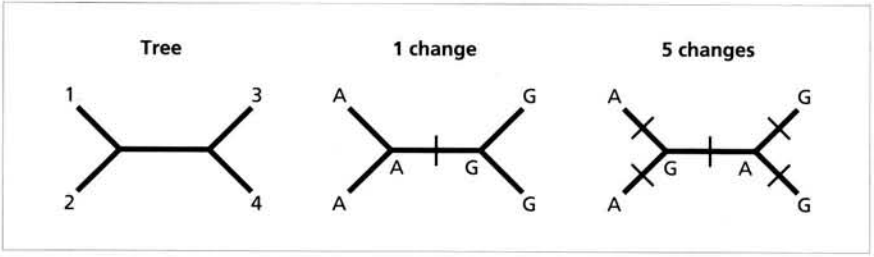


Fig. 6.11 An unrooted tree for four sequences 1–4 and two possible reconstructions of the evolution of the same site on that tree. One reconstruction requires one change, the other requires five. The former is more parsimonious.

possess. On branches where the nucleotide at each end differs we postulate a substitution. Note that our reconstruction does not specify where along that branch the change took place, merely that it did. Under the principle of parsimony the reconstruction that invokes a single substitution is preferred over the less parsimonious reconstruction that requires five substitutions. We can now find the most parsimonious reconstructions for each of the remaining sites (Fig. 6.12). Note that for the third site there are two equally parsimonious reconstructions, both requiring two steps.

The total number of evolutionary changes on a tree (often referred to as the tree's **length**) is simply the sum of the number of changes at each site. Hence, if we have k sites, each with a length of l, then the length L of the tree is given by

$$L = \sum_{i=1}^{k} l_i \tag{6.4}$$

Hence the tree in Fig. 6.11 has a length of 1 + 1 + 2 + 1 + 0 = 5 steps. The other two trees have lengths of six and seven steps (Table 6.3), so the tree ((1,2),(3,4)) is the most parsimonious tree.

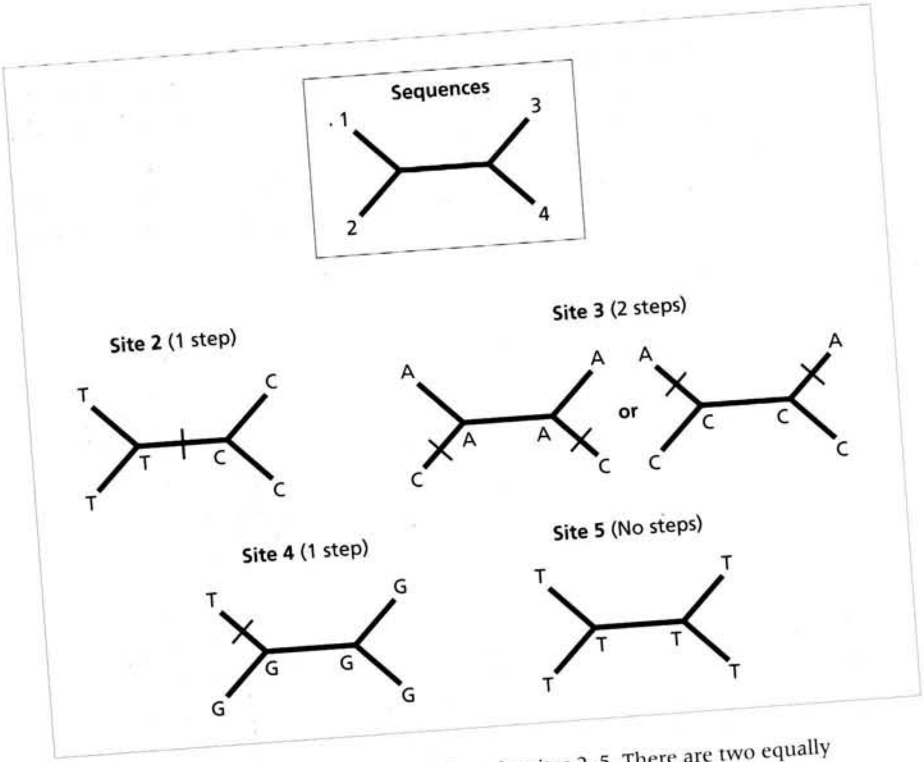


Fig. 6.12 Most parsimonious reconstructions for sites 2–5. There are two equally parsimonious reconstructions for site 3.

Table 6.3 Changes rec
for each site to fit the t
possible trees for four
sequences.

Notice that the nu
hence these sites c
sites are termed p
(each sequence h
only one sequenc

6.4.1 Generalis

When counting
the same 'cost',
transversion A -
matrix (Fig. 6.1
In the first n
has equal cost,
Note that the c
branch is the sa
The second
the cost of a tra

Fig. 6.13 Two
substitution m
corresponding
The value in
step matrix is
correspondin

Table 6.3 Changes required for each site to fit the three possible trees for four sequences.

Tree	Sites 1	2	3	4	5	Total
((1,2),(3,4))	1	1	2	1	0	5
((1,3),(2,4))	2	2	1	1	0	6
((1,4),(2,3))	2	2	2	1	0	7

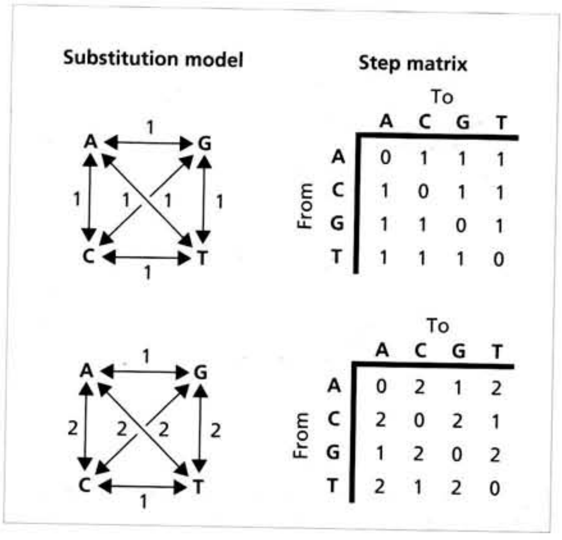
Notice that the number of steps at sites 4 and 5 are the same for all three trees, hence these sites do not discriminate between the three alternative trees. Such sites are termed **phylogenetically uninformative**; sites that are invariant (each sequence has the same nucleotide at the same position) or sites where only one sequence has a different nucleotide are examples of such sites.

6.4.1 Generalised parsimony

When counting changes in the above example, each substitution was accorded the same 'cost', so that the transition A → G counts as one step, as does the transversion A → C. Another way of representing this is by means of a step matrix (Fig. 6.13).

In the first model, which we have implicitly used above, each substitution has equal cost, so each cell in the corresponding step matrix has the entry '1'. Note that the cost of not changing is zero; if the nucleotide at each end of a branch is the same then we infer that no change has occurred.

The second model assigns a higher cost to transversions, in this case twice the cost of a transition. This is equivalent to saying the transversions are rarer



From \ To	A	C	G	T
A	0	1	1	1
C	1	0	1	1
G	1	1	0	1
T	1	1	1	0

From \ To	A	C	G	T
A	0	2	1	2
C	2	0	2	1
G	1	2	0	2
T	2	1	2	0

Fig. 6.13 Two different substitution models and the corresponding step matrices. The value in each cell in a step matrix is the 'cost' of the corresponding substitution.

than transitions, and therefore may be more reliable indicators of phylogeny. If we were to use this step matrix on the data discussed above, the number of steps for sites 1, 2 and 4 would not change as they all involve transitions. However, site 3 requires an A–C transversion. Trees 1 and 3 require this transversion to occur twice, hence for those sites the total cost for site 3 is 4. Tree 2 requires a single transversion with cost 2. Under this model tree 2 is now just as parsimonious as tree 1.

Generalised parsimony is very flexible in that we can specify a wide range of substitution models (see Chapter 5) using step matrices. However, it is not immediately obvious what cost to assign the different kinds of substitutions. We shall return to this point below.

6.4.2 Weighted parsimony

Not all sites may be equally phylogenetically useful. Sites that evolve very rapidly are likely to become quickly saturated so that the trace of history is overprinted and lost. Indeed, such sites may even be positively misleading. In contrast, sites that are conservative and evolve very slowly are less likely to suffer the effects of saturation. The relative value of different sites can be reflected by the **weight** w given to each site. Hence the length of a tree becomes

$$L = \sum_{i=1}^{k} w_i l_i \qquad (6.5)$$

The greater the phylogenetic value of the character, the greater the weight we might wish to assign it.

As with step matrices, weighting adds flexibility to parsimony analyses, but raises the question of exactly what weights to assign different sites.

6.4.3 Justification for parsimony

Among parsimony's advantages are that it is relatively straightforward to understand, it apparently makes few assumptions about the evolutionary process, it has been extensively studied mathematically, and some very powerful software implementations are available. However, the justification for choosing the most parsimonious tree as the best estimate of phylogeny is the subject of considerable controversy. Essentially two main arguments have been presented. The first is that parsimony is a methodological convention that compels us to maximise the amount of evolutionary similarity that we can explain as homologous similarity, that is, we want to maximise the similarity that we can attribute to common ancestry. Any character which does not fit a given tree requires us to postulate that the similarity between two sequences shown by that character arose independently in the two sequences—the similarity is due to homoplasy not homology. Hypotheses of homoplasy (such as convergence or parallel evolution) may be judged *ad hoc* in that they are attempts to explain why data do not fit a particular hypothesis. The most parsimonious tree minimises the number of *ad hoc* hypotheses required, and for that reason is preferred.

The second view is that parsimony is based on an implicit assumption about evolution, namely that evolutionary change is rare. Rarity of change implies that the tree that minimises change is likely to be the best estimate of the actual phylogeny. Under this view, parsimony may be viewed as an approximation to maximum likelihood methods (discussed below), and indeed it was in this context that parsimony methods were first proposed by Edwards and Cavalli-Sforza.

The debate between advocates of these two positions has been explored in great detail by Elliot Sober (1988), to which the reader is referred for more details. Of the two positions, the latter has the advantage that it is possible to explore the circumstances under which parsimony will fail to reconstruct the correct phylogeny, and to develop a framework in which parsimony can be compared with other methods.

6.4.4 Objections to parsimony

The principal objection to parsimony is that under some models of evolution it is not consistent (see section 6.1.4), that is, even if we add more data it is possible to obtain the wrong tree. The classic scenario where this might happen has been termed 'long branches attract'.

In the tree shown in Fig. 6.14, there are two unrelated sequences that are each separated from their ancestor by a long edge. In order for parsimony to recover the correct tree ((A, B),(C, D)) there must be more sites supporting

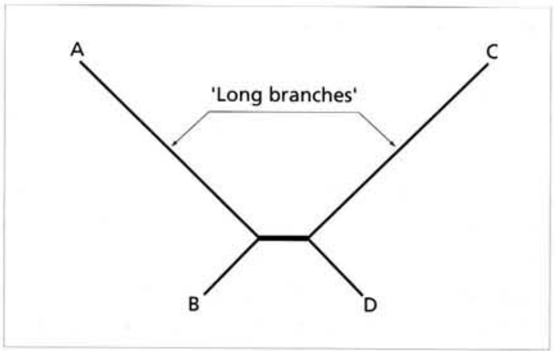


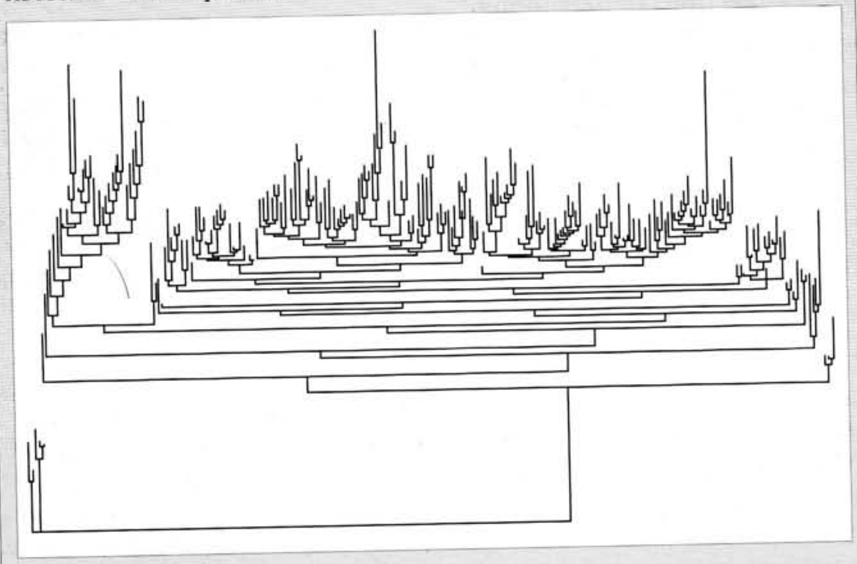
Fig. 6.14 The problem of 'long branches attracting'. The edges leading to sequences A and C are long relative to the other branches in the tree, reflecting the relatively greater number of substitutions that have occurred along those two edges. If the difference in numbers of substitutions is sufficiently great, there may be more apparent support for the split {{A, C},{B, D}} than for the split {{A, B},{C, D}}.

the split {{A, B},{C, D}} than either of the two other possible internal splits: {{A, C},{B, D}} and {{A, D},{B, C}}. If the internal edge is short relative to the long terminal edges then by chance alone A and C may acquire the same nucleotide independently. These convergences may outweigh the sites changing along the internal edge, and hence by the parsimony criterion the tree ((A, C), (B, D)) would be favoured. Under some circumstances, no matter how much data is added parsimony will obtain the wrong tree, hence it is inconsistent.

The problem of long branches attracting is most likely to occur when rates of evolution show considerable variation among sequences, or where the sequences being analysed are quite divergent. One strategy to reduce the effects of long edges is to add sequences that join onto those edges thus breaking them up (see Box 6.3).

Box 6.3 Big trees and long branches

The problem of 'long branch attraction' may be severe in the case of trees for four sequences. Paradoxically, it may be less of a problem for large phylogenies. The diagram below shows a tree for 228 angiosperm (flowering plant) 18S ribosomal DNA sequences.

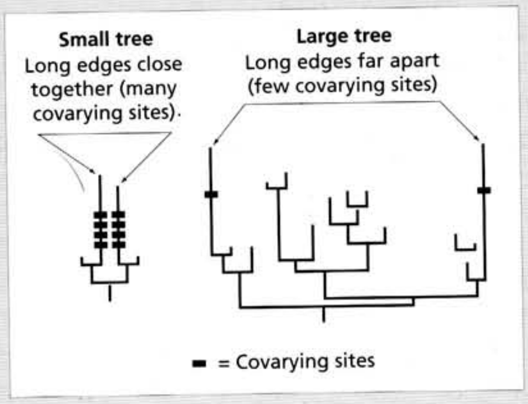


This tree contains several prominent long edges. To test our ability to recover this tree Hillis (1996) generated artificial data sets by simulating the evolution of DNA sequences of various lengths on this tree, then analysed those data sets using a range of tree-building methods (this approach of simulating data on a tree is called 'parametric bootstrapping' and is discussed in section 6.8.3). Parsimony and neighbour joining both did surprisingly well, requiring only 5000 nucleotides to recover the original tree. The crucial problem posed by

continued

Box 6.3 *continued*

'long branches' is not so much the *length* of the branches, but that the *same* substitutions have occurred along the two branches, fooling tree-building methods into joining them together. The probability of such covarying homoplasies is less if the branches are widely separated. Intuitively, if the long branches are close together then their ancestral states probably closely resembled each other, whereas branches far apart in a tree likely had very different ancestral states. It is more likely that similar substitutions will occur in two lineages that were similar to start with than in two lineages that were very different. Hence, in small trees (such as the four-taxon trees much used in simulations—see section 6.7.3), the rapid rate of evolution in the two lineages that were initially quite similar is more likely to yield the same substitutions in the two edges than in large trees.


Small tree
Long edges close together (many covarying sites).
Large tree
Long edges far apart (few covarying sites)
▬ = Covarying sites

6.5 Maximum likelihood

Given competing explanations for a particular outcome, which explanation should we choose? The principle of **likelihood**, which we encountered in Chapter 5, suggests that the explanation that makes the observed outcome the most likely (i.e. the most probable) occurrence is one to be preferred. Put more formally, if given some data D, and a hypothesis H, the likelihood of that data is given by

$$L_D = \Pr(D \mid H) \tag{6.6}$$

which is the probability of obtaining D given H. In the context of molecular phylogenetics D is the set of sequences being compared, and H is a phylogenetic tree, hence we want to find the likelihood of obtaining the observed sequences given a particular tree. The tree that makes our data the most probable evolutionary outcome is the **maximum likelihood estimate** of the phylogeny.