The Phylogenetic Handbook

A Practical Approach to DNA and Protein Phylogeny

Edited by
Philippe Lemey, Marco Salemi, and Anne-Mieke Vandamme

Contents

6

Phylogenetic inference using maximum likelihood methods

Heiko A. Schmidt and Arndt von Haeseler

THEORY

6.1 Introduction

The concept of ***likelihood*** refers to situations that typically arise in natural sciences in which given some data **D**, a decision must be made about an adequate explanation of the data. Thus, a specific model and a hypothesis are formulated in which the model as such is generally not in question. In the phylogenetic framework, one part of the model is that sequences actually evolve according to a tree. The possible hypotheses include the different tree structures, the branch lengths, the parameters of the ***model of sequence evolution***, and so on. By assigning values to these elements, it is possible to compute the probability of the data under these parameters and to make statements about their plausibility. If the hypothesis varies, the result is that some hypotheses produce the data with higher probability than others. Coin-tossing is a standard example. After flipping a coin $n = 100$ times, $h = 21$ heads and $t = 79$ tails were observed. Thus, $\mathbf{D} = (21, 79)$ constitutes a sufficient summary of the data. The model then states that with some probability, $\theta \in [0, 1]$ heads appear when the coin is flipped. Moreover, it is assumed that the outcome of each coin toss is independent of the others, that θ does not change during the experiment, and that the experiment has only two outcomes (head or tail). The model is now fully specified. Because both heads and tails were obtained, θ must be larger than zero and smaller than 1. Moreover, any probability textbook explains that the probability to observe exactly $H = h$ heads in n tosses can be calculated according to the binomial distribution:

$$\Pr[H = h] = \binom{n}{h} \theta^h (1 - \theta)^{n-h} \tag{6.1}$$

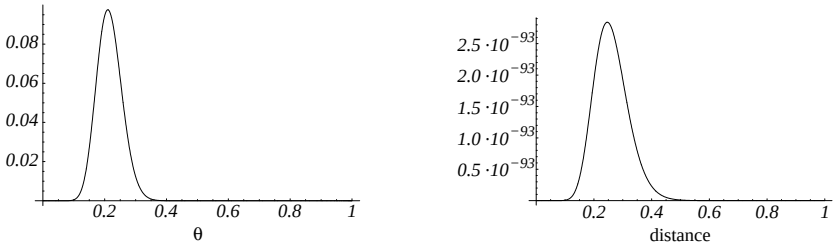


Fig. 6.1. Left: Likelihood function of a coin-tossing experiment showing 21 heads and 79 tails. Right: Likelihood function of the Jukes-Cantor model of sequence evolution for a sequence with length 100 and 21 observed differences.

Equation 6.1 can be read in two ways. First, if θ is known, then the probability of $h = 0, \ldots, n$ heads' in n tosses can be computed. Second, Equation 6.1 can be seen as a function of θ, where n and h are given; this defines the so-called ***likelihood function***

$$L(\theta) = \Pr[H = h] = \binom{n}{h} \theta^h (1-\theta)^{n-h} \tag{6.2}$$

From Figure 6.1, which illustrates the likelihood function for the coin-tossing example, it can be seen that some hypotheses (i.e., choices of θ) generate the observed data with a higher probability than others. In particular, Equation 6.2 becomes maximal if $\theta = \frac{21}{100}$. This value also can be computed analytically. For ease of computation, first compute the logarithm of the likelihood function, which results in sums rather than products:

$$\log[L(\theta)] = \log \binom{n}{h} + h \log \theta + (n-h) \log(1-\theta) \tag{6.3}$$

The problem is now to find the value of θ ($0 < \theta < 1$) maximizing the function. From elementary calculus, it is known that relative extrema of a function, $f(x)$, occur at critical points of f, i.e., values x_0 for which either $f'(x_0) = 0$ or $f'(x_0)$ is undefined. Differentiation of Equation 6.3 with respect to θ yields:

$$L'(\theta) = \frac{\partial \log[L(\theta)]}{\partial \theta} = \frac{h}{\theta} - \frac{n-h}{1-\theta} \tag{6.4}$$

This derivative is equal to zero if $\theta_0 = \frac{h}{n}$, positive, i.e. $L'(\theta) > 0$, for $0 < \theta < \theta_0$, and negative for $\theta_0 < \theta < 1$, so that $\log[L(\theta)]$ attains its maximum at $\theta_0 = \frac{h}{n}$. We say $\hat{\theta} = \frac{h}{n}$ is the ***maximum-likelihood estimate (MLE)*** of the probability of observing a head

in a single coin toss (the hat ˆ notation indicates an estimate rather than the unknown value of θ). In other words, when the value of θ is selected that maximizes Equation 6.3, the observed data are produced with the highest likelihood, which is precisely the ***maximum-likelihood (ML) principle***. However, the resulting likelihoods are usually small (e.g., $L(21/100) \approx 0.0975$); conversely, the likelihoods of competing hypotheses can be compared by computing the odds ratio. Note that the hypothesis that the coin is fair ($\theta = \frac{1}{2}$) results in a likelihood of $L(\frac{1}{2}) \approx 1.61 \cdot 10^{-9}$; thus, the MLE of $\hat{\theta} = 0.21$ is $6 \cdot 10^7$ times more likely to produce the data than $\theta = 0.5$! This comparison of odds ratios leads to the statistical test procedure discussed more or less detailed in Chapters 8, 10-12, and 14.

In evolution, point mutations are considered chance events, just like tossing a coin. Therefore, at least in principle, the probability of finding a mutation along one branch in a **phylogenetic tree** can be calculated by using the same maximum-likelihood framework discussed previously. The main idea behind phylogeny inference with maximum-likelihood is to determine the tree topology, branch lengths, and parameters of the evolutionary model (e.g., transition/transversion ratio, base frequencies, rate variation among sites) (see Chapter 4) that maximize the probability of observing the sequences at hand. In other words, the likelihood function is the conditional probability of the data (i.e., sequences) given a hypothesis (i.e., a model of substitution with a set of parameters θ and the tree τ, including branch lengths):

$$L(\tau, \theta) = \Pr(\text{Data}|\tau, \theta) = \Pr(\text{aligned sequences}|\text{tree, model of evolution}) \quad (6.5)$$

The MLEs of τ and θ (named $\hat{\tau}$ and $\hat{\theta}$) are those making the likelihood function as large as possible:

$$\hat{\tau}, \hat{\theta} = \underset{\tau,\theta}{\operatorname{argmax}}\, L(\tau, \theta) \quad (6.6)$$

Before proceeding to the next section, some cautionary notes are necessary. First, the likelihood function must not be confused with a probability. It is defined in terms of a probability, but it is the probability of the observed event, not of the unknown parameters. The parameters have no probability because they do not depend on chance. Second, the probability of getting the observed data has nothing to do with the probability that the underlying model is correct. For example, if the model states that the sequences evolve according to a tree, although they have recombined, then the final result will still be a single tree that gives rise

to the maximum-likelihood value (see also Chapter 15). The probability of the data given the MLE of the parameters does not provide any hints that the model assumptions are in fact true. One can only compare the maximum-likelihood values with other likelihoods for model parameters that are elements of the model. To determine whether the hypothesis of tree-like evolution is reasonable, the types of relationship allowed among sequences must be enlarged; this is discussed in Chapter 21.

6.2 The formal framework

Before entering the general discussion about maximum-likelihood tree reconstruction, the simplest example (i.e., reconstructing a maximum-likelihood tree for two sequences) is considered. A tree with two **taxa** has only one branch connecting the two sequences; the sole purpose of the exercise is reconstructing the branch length that produces the data with maximal probability.

6.2.1 The simple case: Maximum-likelihood tree for two sequences

In what follows, it is assumed that the sequences are evolving according to the Jukes and Cantor model (see Chapter 4). Each position evolves independently from the remaining sites and with the same **evolutionary rate**. The alignment has length l for the two sequences $S_i = \left(s_i^1, \ldots, s_i^l\right)$, $(i = 1, 2)$, where s_i^j is the nucleotide, the amino acid, or any other letter from a finite alphabet at sequence position j in sequence i. The likelihood function is, then, according to Equation 4.31:

$$L(d) = \prod_{j=1}^{l} \pi_{s_1^j} P_{s_1^j s_2^j}\left(-\frac{4d}{3}\right) \tag{6.7}$$

where d, the number of substitutions per site, is the parameter of interest and $P_{xy}(t)$ is the probability of observing nucleotide y if nucleotide x was originally present, and $\pi_{s_1^j}$ is the probability of character s_1^j in the equilibrium distribution. From Equations 4.12a and 4.12b, the following is obtained:

$$P_{xy}\left(-\frac{4}{3}d\right) = \begin{cases} \frac{1}{4}\left(1 + 3\exp\left[-\frac{4}{3}d\right]\right) \equiv \tilde{P}_{xx}(d), & \text{if } x = y \\ \frac{1}{4}\left(1 - \exp\left[-\frac{4}{3}d\right]\right) \equiv \tilde{P}_{xy}(d), & \text{if } x \neq y \end{cases} \tag{6.8}$$

To infer d, the relevant statistic is the number of identical pairs of nucleotides (l_0) and the number of different pairs (l_1), where $l_0 + l_1 = l$.

```
L20571   ...AAAGTAATGAAGAAGAACAACAGGAAGTCATGGAGCTTATACATA...
AF10138  ...ATGGAGAAGAAGAAG---------AGACTCTGGCTAAGTTATTGT...
X52154   ...ATGGAGAAGAAGAAG---------AGAGACTGGAACAGCTTATCC...
U09127   ...ATGGGGATAGAGAGGAATTATCCTTGCTGGTGGACATGGGGGATT...
U27426   ...AGGGGGATACAGATGAATTGGCAACACTTGTGGAAATGGGGAACT...
U27445   ...AAGGGGATACGGACGAATTGGCAACACTTCTGGAGATGGGGAACT...
U067158  ...AGGGGGACACTGAGGAATTATCAACAATGGTGGATATGGGGCGTC...
U09126   ...GAGGGGATACAGAGGAATTGGAAACAATGGTGGATATGGGGCATC...
U27399   ...AGGGAGATGAGGAGGAATTGTCAGCATTTGTGGGGATGGGGCACC...
U43386   ...AGGGAGATGCAGAGGAATTATCAGCATTTATGGAAATGGGGCATC...
L02317   ...AAGGAGATCAGGAAGAATTATCAGCACTTGTGGAGATGGGGCACC...
AF025763 ...AAGGGGATCAGGAAGAATTGTCAGCACTTGTGGAGATGGGGCATG...
U08443   ...AAGGAGATGAGGAAGCATTGTCAGCACTTATGGAGAGGGGGCACC...
AF042106 ...AAGGGGATCAGGAAGAATTATCGGCACTTGTGGACATGGGGCACC...
```

Fig. 6.2. Part of the mtDNA sequence alignment used as a relevant example throughout the book.

Therefore, the alignment is summarized as $\mathbf{D} = (l_0, l_1)$ and the score is computed as:

$$\log[L(d)] = C + l_0 \log\left[\tilde{P}_{xx}(d)\right] + l_1 \log\left[\tilde{P}_{xy}(d)\right] \tag{6.9}$$

which is maximal if

$$d = -\frac{3}{4} \log\left[1 - \frac{4}{3} \cdot \frac{l_1}{l_1 + l_0}\right]. \tag{6.10}$$

This result is not influenced by the constant C, which only changes the height of the maximum but not its 'location'.

Please note that, the MLE of the number of substitutions per site equals the method-of-moments estimate (see Equation 4.15a). Therefore, the maximum-likelihood tree relating the sequences S_1 and S_2 is a straight line of length d, with the sequences as endpoints.

This example was analytically solvable because it is the simplest model of sequence evolution and, more importantly, because only two sequences – which can only be related by *one* tree – were considered. The following sections set up the formal framework to study more sequences.

6.2.2 The complex case

When the data set consists of $n > 2$ aligned sequences, rather than computing the probability $P_{xy}(t)$ of observing two nucleotides x and y at a given site in two sequences, the probability of finding a certain column or pattern of nucleotides in the data set is computed. Let D_j denote the nucleotide pattern at site $j \in \{1, \ldots, l\}$ in the alignment (Figure 6.2). The unknown probability obviously depends on the model of sequence

evolution, M, and the tree, τ relating the n sequences with the number of substitutions along each branch of the tree (i.e., the branch lengths). In theory, each site could be assigned its own model of sequence evolution according to the general time reversible model (see Chapter 4) and its own set of branch lengths. Then, however, the goal to reconstruct a tree from an alignment becomes almost computationally intractable and, hence, several simplifications are needed. First, it is assumed that each site s in the alignment evolves according to the same model M; for example, the Tamura-Nei (TN) model (see Equations 4.32a, b, c) (i.e., γ, κ, and π are assumed the same for each site in the alignment). The assumption also implies that all sites evolve at the same rate μ (see Equation 4.24). To overcome this simplification, the rate at a site is modified by a rate-specific factor, $\rho_j > 0$. Thus, the ingredients for the probability of a certain site pattern are available, and

$$\Pr\left[D_j|\tau, M, \rho_j\right], \quad j = 1, \ldots, l \tag{6.11}$$

specifies the probability to observe pattern D_j. If it is also assumed that each sequence site evolves independently (i.e., according to τ and M, with a site specific rate ρ_j), then the probability of observing the alignment (data) $\mathbf{D} = (D_1, \ldots, D_l)$ equals the product of the probabilities at each site, as follows:

$$L\left(\tau, M, \rho|\mathbf{D}\right) \equiv \Pr\left[\mathbf{D}|\tau, M, \rho\right] = \prod_{j=1}^{l} \Pr\left[D_j|\tau, M, \rho_j\right] \tag{6.12}$$

When the data are fixed, Equation 6.12 is again a likelihood function (like Equations 6.2 and 6.5), which allows for the two ways of looking at it (see the previous section). First, for a fixed choice of τ, M, and the site rate vector ρ, the probability to observe the alignment $\mathbf{D}$ can be computed with Equation 6.11. Second, for a given alignment $\mathbf{D}$, Equation 6.12 can be used to find the **MLEs**.

In what follows, the two issues are treated separately. However, to simplify the matter, it is assumed that the site-specific rate factor ρ_j is drawn from a **Γ-distribution** with expectation 1 and variance $\frac{1}{\alpha}$ [38, 40], where α defines the shape of the distribution (see also Section 4.6.1).

6.3 Computing the probability of an alignment for a fixed tree

Consider the tree τ with its branch lengths (i.e., number of substitutions per site), the model of sequence evolution M with its parameters (e.g.,

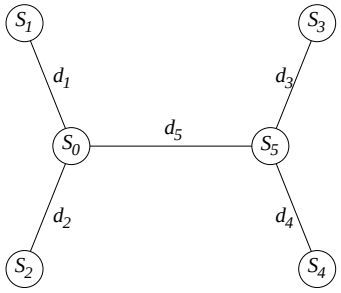


Fig. 6.3. Four-sequence tree, with branch lengths d_1, d_2, d_3, and d_4 leading to sequences S_1, S_2, S_3, and S_4 and branch length d_5 connecting the 'ancestral' sequences S_0 and S_5.

transition/transversion ratio, stationary base composition), and the site-specific rate factor $\rho_j = 1$ for each site j. The goal is to compute the probability of observing one of the 4^n possible patterns in an alignment of n sequences. The tree displayed in Figure 6.3 illustrates the principle for four sequences ($n = 4$). Because the model M is a submodel of the GTR class – that is, a time-reversible model (see Chapter 4) – we can assign any point as a root to the tree for the computation of its likelihood (Pulley Principle, cf. [7]). Here, we will assume that evolution started from sequence S_0 and then proceeded along the branches of tree τ with branch lengths d_1, d_2, d_3, d_4, and d_5. To compute $\Pr[D_j, \tau, M, 1]$ for a specific site j, where $D_j = (s_1^j, s_2^j, s_3^j, s_4^j)$ are the nucleotides observed, it is necessary to know the ancestral states s_0^j and s_5^j. The conditional probability of the data given the ancestral states then will be as follows:

$$\Pr[D_j, \tau, M, \mathbf{1} |\, s_0^j, s_5^j] = P_{s_0^j s_1^j}(d_1) \cdot P_{s_0^j s_2^j}(d_2) \cdot P_{s_0^j s_5^j}(d_5) \cdot P_{s_5^j s_3^j}(d_3) \cdot P_{s_5^j s_4^j}(d_4) \quad (6.13)$$

The computation follows immediately from the considerations in Chapter 4. However, in almost any realistic situation, the ancestral sequences are not available. Therefore, one sums over all possible combinations of ancestral states of nucleotides gaining a so-called maximum average likelihood [32]. As discussed in Section 4.4, nucleotide substitution models assume **stationarity**; that is, the relative frequencies of A, C, G, and T (π_A, π_C, π_G, π_T) are at equilibrium. Thus, the probability for nucleotide s_0^j will equal its stationary frequency $\pi(s_0^j)$, from which it follows that

$$\Pr[D_j, \tau, M, \mathbf{1}] = \sum_{s_0^j} \sum_{s_5^j} \pi(s_0^j) P_{s_0^j s_5^j}(d_1) \cdot P_{s_0^j s_2^j}(d_2) \cdot P_{s_0^j s_5^j}(d_5) \cdot P_{s_5^j s_3^j}(d_3) \cdot P_{s_5^j s_4^j}(d_4) \quad (6.14)$$

Although this equation looks like one needs to compute exponentially many summands, the sum can be efficiently assessed by evaluating the likelihoods moving from the end nodes of the tree to the root [7]. In each step, starting from the leaves of the tree, the computations for two nodes are joined and replaced by the joint value at the ancestral node (see Section 6.3.1 for details). This process bears some similarity to the computation of the minimal number of substitutions on a given tree in the ***maximum parsimony*** framework [10] (see Chapter 8). However, contrary to maximum parsimony, the distance (i.e., number of substitutions) between the two nodes is considered. Under the maximum parsimony framework, if two sequences share the same nucleotide, then the most recent common ancestor also carries this nucleotide (see Chapter 8). In the maximum-likelihood framework, this nucleotide is shared by the ancestor only with a certain probability, which gets smaller if the sequences are only very remotely related.

6.3.1 Felsenstein's pruning algorithm

Equation 6.14 shows how to compute the likelihood of a tree for a given position in a sequence alignment. To generalize this equation for more than four sequences, it is necessary to sum all the possible assignments of nucleotides at the $n-2$ inner nodes of the tree. Unfortunately, this straightforward computation is not feasible, but the amount of computation can be reduced considerably by noticing the following recursive relationship in a tree. Let $D_j = (s_1^j, s_2^j, s_3^j, \ldots, s_n^j)$ be a pattern at a site j, with tree τ and a model M fixed. Nucleotides at inner nodes of the tree are abbreviated as x_i with $i = n+1, \ldots, 2n-2$. For an inner node i with offspring o_1 and o_2, the vector $\mathbf{L}_j^i = \big(L_j^i(A), L_j^i(C), L_j^i(G), L_j^i(T)\big)$ is defined recursively as

$$L_j^i(s) = \left[\sum_{x\in\{A,C,G,T\}} P_{sx}(d_{o_1}) L_j^{o_1}(x)\right] \cdot \left[\sum_{x\in\{A,C,G,T\}} P_{sx}(d_{o_2}) L_j^{o_2}(x)\right] \quad s \in \{A,C,G,T\} \tag{6.15}$$

and for the leaves

$$L_j^i(s) = \begin{cases} 1, & \text{if } s = s_i^j \\ 0, & \text{otherwise} \end{cases} \tag{6.16}$$

where d_{o_1} and d_{o_2} are the number of substitutions connecting node i and its descendants in the tree (cf. Fig. 6.4). Without loss of generality, it is assumed that the node $2n-2$ has three offspring: o_1, o_2, and

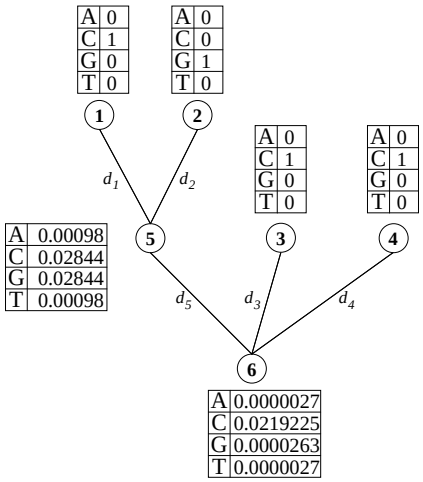


Fig. 6.4. Likelihood computation on a three-taxa tree for a alignment site pattern $D_j = (C, G, C, C)$ with all branch lengths $d_1, \ldots d_5$ set to 0.1. According to Eq. 6.8 the probability to observe no mutation after $d = 0.1$ is 0.9058 and for a specific nucleotide pair 0.0314. The values $L_j^i(s)$ at the leaves are computed with Eq. 6.16, those at the internal nodes with Eq. 6.15. For example, to obtain $L_j^5(C)$ at node five, Eq. 6.15 reduces to $P_{CC}(d_1) \cdot P_{CG}(d_2) = 0.9058 \cdot 0.0314$. The position likelihood according to Eq. 6.17 is 0.0054886, the according log-likelihood is -5.2051.

o_3, respectively. For this node, Equation 6.15 is modified accordingly. This equation allows an efficient computation of the likelihood for each alignment position (cf. Fig. 6.4) by realizing that

$$\Pr\left[D_j, \tau, M, 1\right] = \sum_{s \in \{A,C,G,T\}} \pi_s L_j^{2n-2}(s) \tag{6.17}$$

Equation 6.17 then can be used to compute the likelihood of the full alignment with the aid of Equation 6.12. In practice, the calculation of products is avoided, moving instead to log-likelihoods; that is, Equation 6.12 becomes

$$\log\left[L\left(\tau, M, 1\right)\right] = \log\left[\prod_{j=1}^{l} \Pr\left[D_j, \tau, M, 1\right]\right] = \sum_{j=1}^{l} \log\left[\Pr\left[D_j, \tau, M, 1\right]\right] \tag{6.18}$$

6.4 Finding a maximum-likelihood tree

Equations 6.15 through 6.18 show how to compute the probability of an alignment, if everything were known. In practice, however, branch lengths of the tree are unknown. Branch lengths are computed numerically by maximizing Equation 6.18; that is, by finding those branch lengths for tree r maximizing the log-likelihood function, which is accomplished by applying numerical routines like Newton-Raphson or Brent's method [23]. Such a computation is usually time-consuming and typically the result depends on the numerical method.

Nevertheless, maximizing the likelihood for a single tree is not the biggest challenge in phylogenetic reconstruction; the daunting task is to actually find the tree among all possible tree structures that maximizes the global likelihood. Unfortunately, for any method that has an explicit optimality criterion (e.g., maximum parsimony, distance methods, and maximum-likelihood), no efficient algorithms are known that guarantee the localization of the best tree(s) in the huge space of all possible tree topologies. The naïve approach to simply compute the maximum-likelihood value for each tree topology is prohibited by the huge number of tree structures, even for moderately sized data sets. The number of (unrooted) tree topologies increases tremendously with the number taxa (n) can be computed for n sequences according to

$$t_n = \frac{(2n5)!}{2^{n-3}(n-3)!} = \prod_{i=1}^{n}(2i-5). \tag{6.19}$$

When computing the maximum-likelihood tree, the model parameters and branch lengths have to be computed for each tree, and then the tree that yields the highest likelihood is selected. Because of the numerous tree topologies, testing all possible trees is impossible, and it is also computationally not feasible to estimate the model parameters for each tree. Thus, various heuristics are used to suggest reasonable trees, including stepwise addition (e.g., used in Felsenstein's PHYLIP package: program DNAML, [9]) and ***star decomposition*** (MOLPHY, [2]) as well as the ***neighbor-joining*** (NJ) algorithm [26]. These methods are discussed in Chapters 8, 10, and 5, respectively. However, to make this chapter self-consistent we briefly summarize the various heuristics. In our parlance, we are looking for the tree with the highest likelihood. However the tree rearrangement operations themselves are independent of the objective function.

6.4.1 Early heuristics

Stepwise addition was probably among the first heuristics to search for a maximum-likelihood of a tree. The algorithm proceeds as follows: The procedure starts from the unrooted tree topology for three taxa randomly selected from the list of n taxa. Then one reconstructs the corresponding maximum likelihood tree. To extend this tree we randomly pick one of the remaining $n-3$ taxa. This taxon is then inserted into each branch of the best tree. The branch, where the insertion leads to the highest likelihood, will be called insertion branch. Thus, we have a local decision criterion that selects the tree with the highest likelihood from a list of $2k-3$ trees, if k taxa are already in the sub-tree. The resulting tree will then be used to repeat the procedure. After $n-3$ steps a maximum-likelihood tree is obtained, that is at least locally optimal. That means given the insertion order of the taxa and given the local decision criterion no better tree is possible.

However, we have only computed the maximum-likelihood for $\sum_{i=3}^{n}(2i-5) = (n-2)^2$ trees. Thus, it is possible that another insertion order of the taxa will provide trees with a higher likelihood. To reduce the risk of getting stuck in such local optima tree-rearrangement operations acting on the full tree were suggested.

6.4.2 Full-tree rearrangement

Full-tree rearrangement operations change the structure of a given tree with n leaves. They employ the following principle. From a starting tree a number of trees (the neighborhood of the starting tree) are generated according to specified rules. For each resulting tree, the maximum-likelihood value is computed. The tree with the highest likelihood is then used to repeat the procedure. The rearrangement typically stops if no better tree is found. This tree is then said to be a locally optimal tree. The chance of actually having determined the globally optimal tree, however, depends on the data and the size of neighborhood.

Three full-tree rearrangement operations are currently popular: ***Nearest neighbor interchange (NNI)***, ***sub-tree pruning and regrafting (SPR)*** and ***tree-bisection and reconnection (TBR)***, confer Figure 6.5 and for details Chapter 8. Depending on the operation the size of the neighborhood grows linearly (NNI), quadratically (SPR), or cubically (TBR) with the number of taxa in the full tree.

Different approaches are applied to reduce the increase of compu-

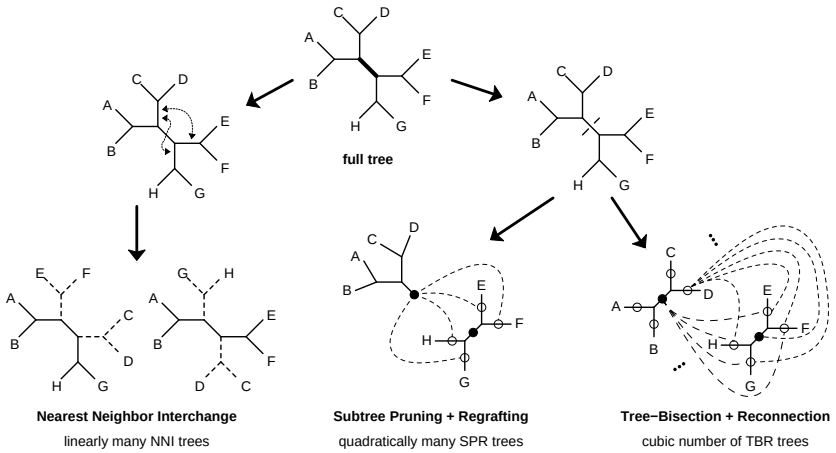


Fig. 6.5. The three basic tree rearrangement operations (NNI, SPR, and TBR) on the thick branch in the full tree. In SPR and TBR all pairs of 'circled' branches among the two subtrees will be connected (dashed lines), except the two filled circles to each other, since this yields the full tree again.

tation time of the SPR or TBR, while still take advantage of their extended neighborhood. We will briefly describe some programs and search schemes. Some of these packages have implemented different variants and extension of the insertion or rearrangement operations. We will not explain them in full detail, but rather refer to the corresponding publications.

6.4.3 DNAML and fastDNAml

The DNAML program (PHYLIP package, [9]) and its descendant, fastDNAml [22, 33], search by stepwise addition. Although not turned on by default, the programs allow to apply SPR rearrangements after all sequences have been added to the tree.

Moreover, fastDNAml provides tools to do full tree rearrangements after each insertion step. The user may choose either NNI or SPR, and can also restrict the SPR neighborhood by setting a maximal number of branches to be crossed between pruning and inserting point of the subtree.

6.4.4 PHYML and PHYML-SPR

PHYML [14] reduces the running time by a mixed strategy. They use a fast distance based method, BioNJ [12], to quickly compute a full initial tree. Then they apply **fastNNI** operations to optimize that tree. During fastNNI all possible NNI trees are evaluated (optimizing only the branch crossed by the NNI) and ranked according to their ML value. Those NNIs which increase the ML value most, but do not interfere with each other, are simultaneously applied to the current tree. Simultaneously applying different NNIs saves time and makes it possible to walk quickly though tree space. On the new current tree fastNNI is repeated until no ML improvement is possible.

Due to their limited range of topological changes NNIs are prone to get stuck in local optima. Hence, a new SPR-based version, **PHYML-SPR** [15], has been devised taking advantage of the larger neighborhood induced by SPR. To compensate for the increased computing time, PHYML-SPR evaluates the SPR neighborhood of the current tree by fast measures like distance-based approaches to determine a ranked list of most promising SPR tree candidates (cf. [15] for more details). Their likelihood is then assessed by only optimizing the branch lengths on the path from the pruning to the insertion point. If a better tree is found it is used as new current tree.

A fixed number of best candidate trees according to their likelihood are then optimized by adjusting all branch lengths. If now a tree has a higher likelihood than the current one, this tree is replaces the old one.

PHYML-SPR allows to alternate SPR and fastNNI-based iterations. Iteration continues until no better tree is found.

6.4.5 IQPNNI

IQPNNI [39] uses BioNJ [12] to compute the starting tree and fastNNI for likelihood optimization. IQPNNI, however, applies a different strategy to reduce the risk of of getting stuck in local optima. When the current tree cannot be improved, IQPNNI randomly removes taxa from the current tree and re-inserts them using a fast quartet-based approach. The new tree is again optimized with fastNNI. If the new tree is better, then it becomes the new starting tree, otherwise the original current tree is kept.

This procedure is either repeated for a user-specified number of itera-

tions or IQPNNI applies a built-in stopping rule, that uses a statistical criterion, to abandon further search [39].

Furthermore, IQPNNI provides ML tree reconstruction for various codon models like [13] or [41]. Refer to Chapter 14 for details on such complex models.

6.4.6 RAxML

The RAxML program [30] builds the starting tree based on Maximum Parsimony (Chapter 8) and optimizes with a variant of SPR called **lazy subtree rearrangement** (LSR, [31]). LSR combines two tricks to reduce the computational demand of SPR operations. First, it assigns a maximal distance between pruning and insertion point for the SPR operations to restrict the size of the neighborhood. The maximal SPR distance (< 25 branches) is determined at the start of the program. Second, LSR optimizes only the branch that originates at the pruning point and the three newly created at the insertion point. The LSRs are repeated many times always using the currently best tree. For the 20 best trees, found during the LSR, the final ML-value is re-optimized by adjusting all branch lengths. The LSR and re-optimization is repeated until no better tree is found.

6.4.7 Simulated Annealing

Simulated annealing [16] is an attempt to find the maximum of complex functions (possibly with multiple peaks), where standard (hill climbing) approaches may get trapped in local optima. One starts with an initial tree, then samples the tree-space by accepting with a reasonable probability a tree with a lower likelihood (down-hill move). Trees with higher likelihood (up-hill moves) are always accepted. This is conceptually related to Markov Chain Monte Carlo (see Chapters 7 and 8). However, as the process continues the down-hill probability is decreased. This decrease is modeled by a so-called cooling schedule. The term 'annealing' is borrowed from crystal formation. Initially (high temperature) there is a lot of movement (almost every tree is accepted), then as the temperature is lowered the movements get smaller and smaller. If the decrease in temperature is modeled adequately then the process will eventually find the ML tree. However, to model the decrease in temperature is not trivial.

First introduced in a parsimony context [19, 5], simulated annealing

to reconstruct ML trees is applied by SSA [27] and RAxML-SA [29]. Furthermore, [11] use simulated annealing to construct alignments and trees simultaneously.

6.4.8 Genetic algorithms

Genetic algorithms (GA) are an alternative search technique to solve complex optimization problems. They borrow the nomenclature and the optimization decision from evolutionary biology. In fact, GA are a special category of evolutionary algorithms [3].

The basic ingredients of GA are a population of individuals (in our case a collection of trees) a fitness function (maximum likelihood function according to Equation 6.17) that determines the offspring number. According to the principles of evolution a tree can mutate (change in branch lengths, NNI, SPR, TBR operations), even trees can exchange sub-trees (recombination). For the mutated tree the fitness function is computed. The individuals of the next generation are then randomly selected from the mutant trees and the current non-mutated trees according to their fitness (selection step). Typically one also keeps track of the fittest individual (the tree with the best likelihood). After several generation evolution stops and the best tree is output.

After having been introduced to phylogenetics in the mid-nineties (e.g., [20]), GARLI [42], METAPIGA [17], and GAML [18] are examples for applications of GA in phylogenetic inference.

6.5 Branch support

As should be clear by now, none of the above methods guarantee to detect the optimal tree. Hence, biologists usually apply a plethora of methods, and if those reconstruct similar trees one tends to have mre confidence in the result.

Typically tree reconstruction methods are searching for the best tree, leaving the user with a single tree and ML value but without any estimate of the reliability of its subtrees.

Several measures are used to assess the certainty of a tree or its branches. The ML values from competing hypotheses can be used in a likelihood ratio test (LRT, see Chapter 10, 11, and 14) or other tests (Chapter 12).

The support of branches are often assessed by employing statistical principals. The most widely used approach to assess branch support

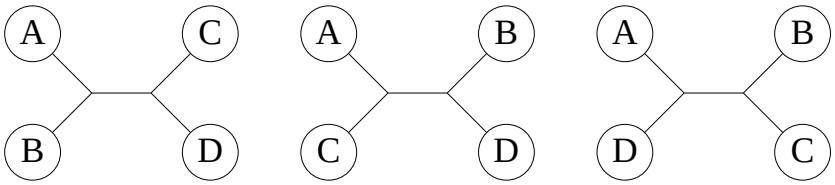


Fig. 6.6. The three different informative tree topologies for the quartet $q = (A, B, C, D)$
.

seems to be ***bootstrapping*** [6, 8], where pseudo-samples are created by randomly drawing with replacement l columns from the original l-column alignment, i.e., a column from the data alignment can occur more than once or not at all in a pseudo-sample. From each pseudo-samples a tree is reconstructed and a consensus tree is constructed, incorporating those branches that occur in the majority of the reconstructed trees. These percentages are used as indicator for the reliability of branches. See Chapter 5 for details on branch support analysis using the bootstrap.

Very similar to the bootstrap is jackknifing, where only a certain percentage of the columns are drawn without replacement [24].

Finally, the trees sampled in a Bayesian MCMC analysis are usually summarized in a consensus tree ([25], see also next chapter).

Another method to measure branch support is the quartet puzzling method implemented in the TREE-PUZZLE software, that will be explained in the following section. Although TREE-PUZZLE is nowadays not faster than most of the above mentioned ML methods, it is usually faster than running at least 100 bootstraps with an ML method and certainly faster than a Bayesian MCMC analysis.

6.6 The quartet puzzling algorithm

Quartet puzzling [35] utilizes quartets, i.e., groups of four sequences. Quartets are the smallest set of taxa for which more than one tree topology exists. The three different quartet tree topologies are shown in Figure 6.6. Quartet-based methods use the advantage that the quartet trees can be quickly evaluated with maximum-likelihood. However, there exist $\binom{n}{4} = \frac{n!}{4!(n-4)!}$ possible quartets in a set of n taxa.

Quartet puzzling is performed in 4 steps:

Parameter estimation: First TREE-PUZZLE estimates the parameters for the evolutionary model. To this end

(i) The pairwise distance matrix D is estimated for all pairs of sequences in the input alignment and a Neighbor Joining tree is constructed from D.

(ii) Then, maximum-likelihood branch lengths are computed for the NJ topology and parameters of the sequence evolution are estimated.

(iii) Based on these estimates, a new D and NJ tree are computed and Step (ii) is repeated.

Steps (ii) and (iii) are repeated until the estimates of the model parameters are stable.

ML step: To produce the set of tree topologies, the likelihoods of all $3 \times \binom{n}{4}$ quartet tree topologies are evaluated. Then, for each quartet and each topology the corresponding highest likelihood is stored. The algorithm takes into account that two topologies may have similar likelihoods (partly resolved quartet) or even no topology (unresolved quartet) gains sufficient support [34].

Puzzling step: Based on the set of supported quartet topologies, trees are constructed by adding taxa in random order. Each taxon is inserted into that branch least contradicted by the set of relevant quartet trees.

This step is repeated many times with different input orders, producing a large set of intermediate trees.

Consensus step: The set of intermediate trees is subsequently summarized by a majority rule consensus tree, the so-called quartet puzzling tree, where the percent occurrences for each branch are considered **puzzle support values**.

6.7 Likelihood-mapping analysis

The chapter so far has discussed the problem of reconstructing a phylogenetic tree and assessing the reliability of its branches. A maximum-likelihood approach may also be used to study the amount of **evolutionary information** contained in a data set. The analysis is based on the maximum-likelihood values for the three possible four taxa trees. If L_1, L_2, and L_3 are the likelihoods of trees T_I, T_2, and T_3, then one computes the ***posterior probabilities*** of each tree T_i as $p_i = \frac{L_i}{L_l+L_2+L_3}$. Since the p_i terms sum to 1, the probabilities p_1, p_2, and p_3 can be reported

simultaneously as a point P lying inside an equilateral triangle, each corner of the triangle representing one of the three possible tree topologies (Figure 6.7A). If P is close to one corner – for example, the corner T_1 – the tree T_1 receives the highest support. In a maximum-likelihood analysis, the tree T_i, which satisfies $p_i = \max\{p_l, p_2, p_3\}$, is selected as the MLE. However, this decision is questionable if P is close to the center of the triangle. In that case, the three likelihoods are of similar magnitude; in such situations, a more realistic representation of the data is a star-like tree rather than an artificially **strictly bifurcating tree** (see Section 1.6).

Therefore, the ***likelihood-mapping method*** [36] partitions the area of the equilateral triangle into seven regions (Figure 6.7B). The three trapezoids at the corners represent the areas supporting strictly bifurcating trees (i.e., Areas 1, 2, and 3 in Figure 6.7B). The three rectangles on the sides represent regions where the decision between two trees is not obvious (i.e., Areas 4, 5, and 6 in Figure 6.7B for trees 1 and 2, 2 and 3, and 3 and 1). The center of the triangle represents sets of P points where all three trees are equally supported (i.e., Area 7 in Figure 6.7B). Given a set of n aligned sequences, the likelihood-mapping analysis works as follows. The three likelihoods for the three tree topologies of each possible quartet (or of a random sample of the quartets) are reported as a dot in an equilateral triangle like the one in Figure 6.7A. The distribution of points in the seven areas of the triangle (see Figure 6.7B) gives an impression of the tree-likeness of the data. Note that because the method evaluates quartets computed from n sequences, which one of the three topologies is supported by any corner of the triangle is not relevant. Only the percentage of points belonging to the areas 1, 2, and 3 is relevant to get an impression about the amount of tree-likeness in the data. To summarize the three corners (Areas $1+2+3$; see Figure 6.7B) represent fully resolved tree topologies; Area 7 represents star-like phylogenies (Figure 6.7B); the three Areas $4+5+6$ (see Figure 6.7B) represent network-like phylogeny, where the data support conflicting tree topologies (see also Chapter 21).

From a biological standpoint, a likelihood mapping analysis showing more than $20-30\%$ of points in the star-like or network-like area suggests that the data are not reliable for phylogenetic inference. The reasons why an alignment may not be suitable for tree reconstruction is multiple, e.g., noisy data, alignment errors, recombination, etc. In the latter case, methods that explore and display conflicting trees, such as **bootscanning** (see Chapter 16), ***split decomposition*** or **NeighborNet** (see

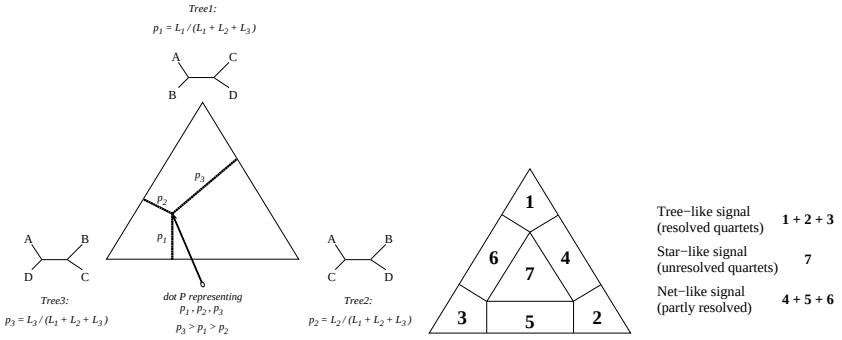


Fig. 6.7. Likelihood mapping. (A) The three posterior probabilities p_1, p_2, and p_3 for the three possible unrooted trees of four taxa are reported as a point (P) inside an equilateral triangle, where each corner represents a specific tree topology with likelihood L_1, L_2, and L_3, respectively. (B) Seven main areas in the triangle supporting different evolutionary information.

Chapters 21 for network analysis) may give additional information. A more detailed study on quartet mapping is given in [21].

PRACTICE

6.8 Software packages

A number software packages are available to compute maximum-likelihood trees from DNA or amino-acid sequences. A detailed list can be found at Joe Felsenstein's web site, `http://evolution.genetics.washington.edu/PHYLIP/software.html`.

Because program packages are emerging at a rapid pace, the reader is advised to visit this web site for updates.

6.9 An illustrative example of a ML tree reconstruction

In what follows, the `hivALN.phy` file (available at `http://www.thephylogenetichandbook.org`) will be analyzed with the latest version 3.2 of IQPNNI to infer a maximum likelihood tree and TREE-PUZZLE 5.3 [28] to compute support values for the branches of the ML tree.

6.9.1 Reconstructing an ML tree with IQPNNI

Place the `hivALN.phy` file in the same folder as the IQPNNI executable and start `iqpnni`. The following text appears:

```
WELCOME TO IQPNNI 3.2 (sequential version)

Please enter a file name for the sequence data:
```

Type the filename `hivALN.phy` **and press enter.**

```
GENERAL OPTIONS
 o                  Display as outgroup? L20571
 n                 Number of iterations? 200
 s                        Stopping rule? No, stop after 200 iterations

IQP OPTIONS
 p   Probability of deleting a sequence? 0.3
 k               Number representatives? 4

SUBSTITUTION PROCESS
 d          Type of sequence input data? Nucleotides
 m                Model of substitution? HKY85 (Hasegawa et al. 1985)
 t           Ts/Tv ratio (0.5 for JC69)? Estimate from data
 f                     Base frequencies? Estimate from data

RATE HETEROGENEITY
 r          Model of rate heterogeneity? Uniform rate

quit [q], confirm [y], or change [menu] settings:
```

Each option can be selected to change the setting by typing the corresponding letter. For example, if the user types `m` then each keystroke will change the model of sequence evolution. After a number of strokes the default `HKY85` reappears. The letter `n` allows to change the number of iterations. By default the number of iterations is set to twice the number of sequences. The user is advised to set the limit as high as possible or to use the stopping rule.

For example, a typical run to infer a tree based on DNA sequences would start with the following setting:

```
GENERAL OPTIONS
 o                  Display as outgroup? L20571
 n                 Number of iterations? 200
 s                        Stopping rule? No, stop after 200 iterations
```

```
IQP OPTIONS
 p   Probability of deleting a sequence? 0.3
 k               Number representatives? 4

SUBSTITUTION PROCESS
 d          Type of sequence input data? Nucleotides
 m                Model of substitution? HKY85 (Hasegawa et al. 1985)
 t           Ts/Tv ratio (0.5 for JC69)? Estimate from data
 f                     Base frequencies? Estimate from data

RATE HETEROGENEITY
 r          Model of rate heterogeneity? Gamma distributed rates
 i       Proportion of invariable sites? No
 a   Gamma distribution parameter alpha? Estimate from data
 c      Number of Gamma rate categories? 4

quit [q], confirm [y], or change [menu] settings:
```

Entering `y` starts the program. As model of sequence evolution HKY is selected (see Sections 4.6 and 4.9). Rate heterogeneity is modeled with a Γ-distribution. The shape parameter α of the Γ-distribution is estimated with the aid of four discrete categories (see Section 4.6.1). An appropriate model can also be selcted based on one of the approaches discussed in Chapter 10.

IQPNNI also infers trees from amino acid sequence alignments and it is possible to compute trees from coding DNA sequences using different codon models.

During the optimization messages like

```
(1) Optimizing gamma shape parameter ...
Gamma distribution shape = 0.57224
Optimizing transition/transversion ratio ...
Transition/transversion ratio: 1.7074
LogL = -17408.85673

(2) Optimizing gamma shape parameter ...
Gamma distribution shape = 0.54188
Optimizing transition/transversion ratio ...
Transition/transversion ratio: 1.7951
LogL = -17396.4208
```

will appear.

After optimizing the model parameters the IQPNNI continues with the tree search as described. Whenever a better tree is found IQPNNI outputs

```
Doing Nearest Neighbour Interchange... 1 s
We have constructed the initial tree !!!
The currently best log likelihood = -17395.94491
29 Iterations / time elapsed = 0h:0m:8s  (will finish in 0h:0m:47s)
GOOD NEWS: BETTER TREE FOUND: THE CURRENTLY BEST LOG LIKELIHOOD = -17392.7264
76 Iterations / time elapsed = 0h:0m:19s  (will finish in 0h:0m:31s)
120 Iterations / time elapsed = 0h:0m:30s  (will finish in 0h:0m:20s)
179 Iterations / time elapsed = 0h:0m:41s  (will finish in 0h:0m:4s)
200 Iterations / time elapsed = 0h:0m:45s
```

Looking at the output above highlights, that IQPNNI is could well escape the local optimum it was stuck for the first 28 iterations. Strategies only based on BioNJ and fastNNI like the original PHYML strategy, which is roughly equivalent to the reconstruction of the initial tree, would have finished on that local optimum.

Then IQPNNI optimizes the model parameters as described above and re-estimates the likelihood of the final tree:

```
Optimizing the final tree topology as well as branch lengths...
Final best log likelihood: -17392.72496
Constructing the majority rule consensus tree...
Estimating site rates by empirical Bayesian...
The results were written to following files:
   1. hivALN.phy.iqpnni
   2. hivALN.phy.iqpnni.treefile
   3. hivALN.phy.iqpnni.treels
   4. hivALN.phy.iqpnni.rate
Total Runtime: 0h:0m:46s
Finished!!!
```

The outfile of IQPNNI, a text file here called `hivALN.phy.iqpnni`, summarizes the results of the phylogenetic analyses. `hivALN.phy.iqpnni.treefile` contains the ML tree in NEWICK format. The tree can be displayed with the TreeView program. The reconstructed tree is shown in Figure 6.8a.

Valuable information can be obtained from the stopping rule even if the analysis is not finished by the stopping rule: If more than three times

a better tree has been found during the iterations IQPNNI estimates how many additional iterations are necessary to be sure according to a 5% confidence level that the current search will not produce a tree with a better likelihood. If the rule suggests more iterations then executed, one is advised to re-run the analysis. IQPNNI also offers the option to continue the current analysis (see the IQPNNI manual for details).

Repeating the example will produce slightly different output, that means, the best tree might be found in different iterations. This is due the stochastic component in the the IQPNNI procedure (cf. Section 6.4.5) when randomly removing and then re-inserting taxa into the tree.

6.9.2 Getting a tree with branch support values using quartet puzzling

Most tree reconstruction methods output only a tree topology but do not provide information about the significance of the branching pattern.

Here, we will analyze the above data with TREE-PUZZLE to get support values for the IQPNNI tree.

To do this, place the `hivALN.phy` file (available at `http://www.thephylogenetichandbook.org`) in the same folder as the TREE-PUZZLE binary and run the executable. The following text appears:

```
WELCOME TO TREE-PUZZLE 5.3.

Please enter a file name for the sequence data:
```

Type the filename `hivALN.phy` and press enter.

```
Input data set (hivALN.phy) contains 14 sequences of length 2352
   1. L20571
   2. AF103818
   3. X52154
   4. U09127
   5. U27426
   6. U27445
   7. AF067158
   8. U09126
   9. U27399
  10. U43386
  11. L02317
```

```
  12. AF025763
  13. U08443
  14. AF042106
(consists very likely of nucleotides)

GENERAL OPTIONS
 b                      Type of analysis?  Tree reconstruction
 k                 Tree search procedure?  Quartet puzzling
 v          Quartet evaluation criterion?  Approximate maximum likelihood (ML)
 u              List unresolved quartets?  No
 n              Number of puzzling steps?  1000
 j              List puzzling step trees?  No
 9   List puzzling trees/splits (NEXUS)?  No
 o                   Display as outgroup?  L20571 (1)
 z     Compute clocklike branch lengths?  No
 e                   Parameter estimates?  Approximate (faster)
 x             Parameter estimation uses?  Neighbor-joining tree
SUBSTITUTION PROCESS
 d           Type of sequence input data?  Auto: Nucleotides
 h              Codon positions selected?  Use all positions
 m                 Model of substitution?  HKY (Hasegawa et al. 1985)
 t    Transition/transversion parameter?  Estimate from data set
 f                Nucleotide frequencies?  Estimate from data set
RATE HETEROGENEITY
 w           Model of rate heterogeneity?  Uniform rate

Quit [q], confirm [y], or change [menu] settings:
```

Each option can be selected and changed by typing the corresponding letter. For example, if the user types `b` repeatedly, then the option will change from `Tree reconstruction` to `Likelihood mapping` to `Tree reconstruction` again. The letter `k` cycles from `Quartet puzzling` to `Evaluate user defined trees` to `Consensus of user defined trees` to `Pairwise distances only (no tree)`. A typical run to infer a tree based on DNA sequences starts with the following setting:

```
GENERAL OPTIONS
 b                      Type of analysis?  Tree reconstruction
 k                 Tree search procedure?  Quartet puzzling
 v          Quartet evaluation criterion?  Approximate maximum likelihood (ML)
 u              List unresolved quartets?  No
```

```
 n             Number of puzzling steps?  10000
 j             List puzzling step trees?  No
 9   List puzzling trees/splits (NEXUS)?  No
 o                  Display as outgroup?  L20571 (1)
 z     Compute clocklike branch lengths?  No
 e                  Parameter estimates?  Approximate (faster)
 x            Parameter estimation uses?  Neighbor-joining tree
SUBSTITUTION PROCESS
 d          Type of sequence input data?  Auto: Nucleotides
 h             Codon positions selected?  Use all positions
 m                Model of substitution?  HKY (Hasegawa et al. 1985)
 t    Transition/transversion parameter?  Estimate from data set
 f               Nucleotide frequencies?  Estimate from data set
RATE HETEROGENEITY
 w          Model of rate heterogeneity?  Gamma distributed rates
 a   Gamma distribution parameter alpha?  Estimate from data set
 c      Number of Gamma rate categories?  8

Quit [q], confirm [y], or change [menu] settings:
```

By entering `y`, TREE-PUZZLE computes a quartet puzzling tree based on 10,000 intermediate trees, using approximate likelihoods to estimate the quartet trees. The model parameter estimates are also approximated and are based on an NJ tree computed at the beginning of the optimization routine. In the example, the HKY model is selected (see Sections 4.6 and 4.9). Again Γ-distributed rate heterogeneity over sites is modeled, where the shape parameter α is estimated with the aid of eight discrete categories (see Section 4.6.1). If these settings are confirmed (type '`y`'), the following output will appear on the screen:

```
Optimizing missing substitution process parameters
Optimizing missing rate heterogeneity parameters
Optimizing missing substitution process parameters
Optimizing missing rate heterogeneity parameters
Optimizing missing substitution process parameters
Optimizing missing rate heterogeneity parameters
Writing parameters to file hivALN.phy.puzzle
Writing pairwise distances to file hivALN.phy.dist
Computing quartet maximum likelihood trees
Computing quartet puzzling trees
Computing maximum likelihood branch lengths (without clock)
```

```
All results written to disk:
       Puzzle report file:         hivALN.phy.puzzle
       Likelihood distances:       hivALN.phy.dist
       Phylip tree file:           hivALN.phy.tree

The parameter estimation took 11.00 seconds (= 0.18 minutes = 0.00 hours)
The ML step took              8.00 seconds (= 0.13 minutes = 0.00 hours)
The puzzling step took        2.00 seconds (= 0.03 minutes = 0.00 hours)
The computation took 23.00 seconds (= 0.38 minutes = 0.01 hours)
     including input 272.00 seconds (= 4.53 minutes = 0.08 hours)
```

The puzzle report file `hivALN.phy.puzzle`, the most important file, summarizes all results of the phylogenetic analyses. Because the content of the report file is self-explanatory, it is not discussed here. `hivALN.phy.dist` contains the matrix of pairwise distances based on the model parameters. `hivALN.phy.tree` contains the quartet puzzling tree (Figure 6.8b) in NEWICK notation that can be displayed with the TreeView program (see Chapter 5 and Figure 5.5).

TREE-PUZZLE allows to output of all different tree topologies computed during puzzling steps (option `j`). In the HIV example, 685 different intermediate trees were found, in which the most frequent tree occurred at about 6.6%. Therefore, the quartet puzzling algorithm can be used to generate a collection of plausible candidate trees, and this collection can subsequently be employed to search for the most likely tree. It is known that the consensus tree does not necessarily coincide with the maximum-likelihood tree, especially when the consensus is not fully resolved [4]. Thus, to get the maximum-likelihood tree option `j` should be changed to unique topologies, which will be output in the `.ptorder` file. A typical line of this file looks like the following:

```
[1. 657 6.57 14 68510000](L20571,((AF10138,X52154),
  (U09127,(((U27426,U27445), (U067158,U09126)),
  ((U27399,U43386),(((L02317,AF042106),AF025763),
  U08443))))));
```

The first column is a simple numbering scheme, in which each tree is numbered according to its frequency (i.e., second column and third column). Column four (14) gives the first time among $10,000$ (column 6) puzzling steps, when the tree was found. Column five shows how many different trees were found (685).

Similar to bootstrap analysis (cf. Chapter 5), the resulting intermediate trees and, hence, the resulting support values might differ slightly due to the randomization of the insertion order in the puzzling step (cf. Section 6.6). Consequently, also the number of unique intermediate tree topologies and their percentages will also vary slightly. Please note, the intermediate tree found most often does not necessarily coincides with the maximum likelihood tree, but often the ML tree is among the intermediate trees.

To compute the maximum-likelihood tree among all intermediate trees, run TREE-PUZZLE again with the following settings:

```
GENERAL OPTIONS
b                    Type of analysis? Tree reconstruction
k               Tree search procedure? Evaluate user defined trees
z   Compute clocklike branch lengths? No
e                 Parameter estimates? Approximate (faster)
x          Parameter estimation uses? Neighbor-joining tree

SUBSTITUTION PROCESS
d        Type of sequence input data? Auto: Nucleotides
h           Codon positions selected? Use all positions
m              Model of substitution? HKY (Hasegawa et al., 1985)
t  Transition/transversion parameter? Estimate from data set
f             Nucleotide frequencies? Estimate from data set

RATE HETEROGENEITY
w        Model of rate heterogeneity? Gamma distributed rates
a Gamma distribution parameter alpha? Estimate from data set
c    Number of Gamma rate categories? 8

Quit [q], confirm [y], or change [menu] settings: y
```

Now TREE-PUZZLE will compute the maximum-likelihood values for all intermediate trees using the model parameter estimates from the iterative procedure. The computation takes time, but it provides more insight about the data and likelihood surface of the trees. The resulting `.puzzle` file shows the results of the ML analysis for the different topologies at the end of the file. In addition, the likelihoods of the trees are compared by various test (cf. Chapter 8 for details).

If one enables option `9`, TREE-PUZZLE outputs all splits and trees

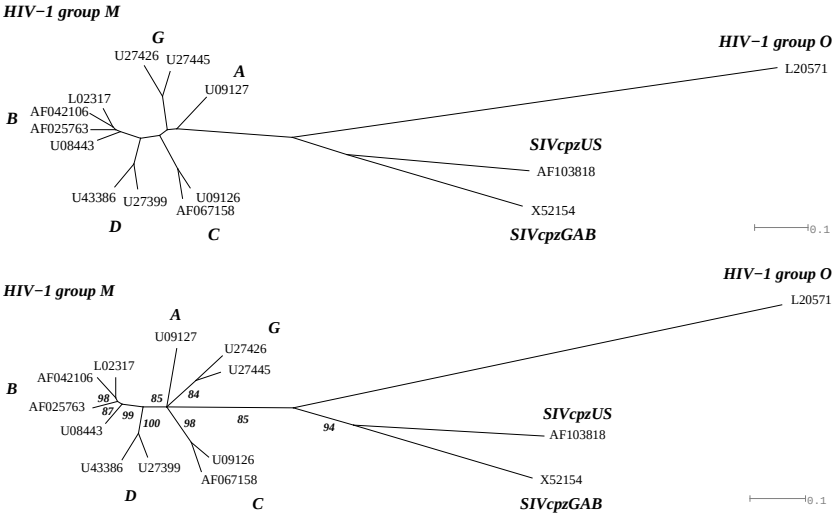


Fig. 6.8. ML-tree reconstructed with IQPNNI (above) and the quartet puzzling consensus tree for the HIV/SIV dataset. The major groups of HIV-1 and the Group M subtypes are indicated in bold, the puzzle support values are given at the branches.

in NEXUS format to the file `hivALN.phy.nex` which can be analyzed with the SplitsTree software (see Chapter 21).

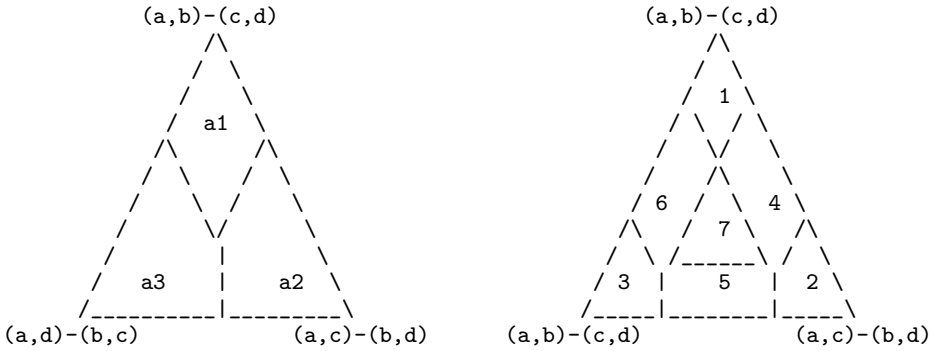
6.9.3 Likelihood-mapping analysis of the HIV data set

Option `b` in the `GENERAL OPTIONS` menu of TREE-PUZZLE switches from `Tree reconstruction` to `Likelihood mapping`. When the number of quartets for the data is below 10,000, the program computes the posterior probabilities of the quartet trees (see Section 6.7) for all quartets; otherwise, only the posterior probabilities for $10,000$ random quartets are computed. In the latter case, the user can decide how many quartets to evaluate by selecting from the `GENERAL OPTIONS` menu option `n`, number of quartets, and typing the number. Typing `0` forces TREE-PUZZLE to analyze all the possible quartets; however, this is time-consuming for large data sets. A random selection serves the same purpose.

Table 6.1 shows part of the outfile from the likelihood-mapping analysis of the $1,001$ possible quartets for the HIV data set. 92.72% of all quartets are tree-like, i.e., they are located close to the corners of the

Table 6.1. *Likelihood-mapping analysis with TREE-PUZZLE*
TREE-PUZZLE writes results of the likelihood mapping analysis at the end of the report file `hivALN.phy.puzzle` in the `LIKELIHOOD MAPPING STATISTICS` section. The results are given for the whole dataset but also for each sequence to help identifying outliers. For example, for the HIV dataset (alignment: `hivALN.phy`; substitution model: HKY with Γ-distributed rate heterogeneity, parameters estimated via maximum likelihood):

```
LIKELIHOOD MAPPING STATISTICS
```




```
[...]

Quartet resolution per sequence:

     name       #quartets    resolved         partly           unresolved
-------------------------------------------------------------------------------
  1     L20571       286    261 ( 91.26)     12 (  4.20)     13 (  4.55)
  2   AF103818       286    271 ( 94.76)     14 (  4.90)      1 (  0.35)
  3     X52154       286    260 ( 90.91)     13 (  4.55)     13 (  4.55)
  4     U09127       286    255 ( 89.16)     17 (  5.94)     14 (  4.90)
  5     U27426       286    260 ( 90.91)     15 (  5.24)     11 (  3.85)
  6     U27445       286    273 ( 95.45)     11 (  3.85)      2 (  0.70)
  7   AF067158       286    261 ( 91.26)     18 (  6.29)      7 (  2.45)
  8     U09126       286    263 ( 91.96)     13 (  4.55)     10 (  3.50)
  9     U27399       286    270 ( 94.41)      9 (  3.15)      7 (  2.45)
 10     U43386       286    264 ( 92.31)      8 (  2.80)     14 (  4.90)
 11     L02317       286    268 ( 93.71)     12 (  4.20)      6 (  2.10)
 12   AF025763       286    268 ( 93.71)     13 (  4.55)      5 (  1.75)
 13     U08443       286    267 ( 93.36)     12 (  4.20)      7 (  2.45)
 14   AF042106       286    271 ( 94.76)      9 (  3.15)      6 (  2.10)
-------------------------------------------------------------------------------
                    1001    928 ( 92.71)     44 (  4.40)     29 (  2.90)


Overall quartet resolution:

Number of resolved        quartets (regions 1+2+3):    928 (=  92.71%)
Number of partly resolved quartets (regions 4+5+6):     44 (=   4.40%)
Number of unresolved      quartets (region 7):          29 (=   2.90%)
```

TREE-PUZZLE also outputs a drawing of the likelihood-mapping triangles in encapsulated Postscript format (`hivALN.phy.eps`) that can be printed or imported in a graphics application to be edited.

triangle. Only 4.4% of all quartets lie in the rectangles and 2.9% in the central triangle: they represent the unresolved part of the data. Because most of the quartets are tree-like and only a fraction of about 7% do not support a unique phylogeny, an overall phylogenetic tree with a good resolution is expected. Nevertheless, the percentages of up to 4.9% unresolved quartets still implies a conciderable amount of noisy or conficting signal in the dataset.

Please note that, if the likelihood mapping is repeated on a ramdom subset, the resulting percentages may naturally differ slightly.

6.10 Conclusions

The HIV quartet puzzling consensus tree calculated in the previous section is by and large resolved and all resolved branches coincide with branches in the IQPNNI ML tree (Figure 6.8a). In contrast, quartet puzzling tree contains a **polytomy** (see Section 1.6) joining Subtypes A, C, G, and the B/D clade (Figure 6.8). This is not surprising in light of the noise in the data revealed by likelihood mapping. Such a lack of resolution may suggest a **star-like radiation** at the origin of the HIV-1 Group M subtypes, but it could also imply the presence of **inter-subtype recombination** in HIV-1. The latter speculation from the first edition of this book [37] was recently substanciated by the findings of [1] that subtype G might be a recombinant instead of a pure subtype as previously thought.

Such issues are analyzed further in Chapters 15, 16, and 21, which discuss general methods for detecting and investigating recombination and conflicting phylogenetic signals in molecular data.

Bibliography

Abecasis, A. B., Lemey, P., Vidal, N., de Oliveira, T., Peeters, M., Camacho, R., Shapiro, B., Rambaut, A. and Vandamme, A.-M. (2005) Recombination is confounding the early evolutionary history of hiv-1: subtype g is a circulating recombinant form. *J. Virol.*, **81**, in press.

Adachi, J. and Hasegawa, M. (1996) *MOLPHY Version 2.3 – Programs for Molecular Phylogenetics Based on Maximum Likelihood*, volume 28 of *Computer Science Monographs*. Institute of Statistical Mathematics, Minato-ku, Tokyo.

Bäck, T. and Schwefel, H.-P. (1993) An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, **1**, 1–23.

Cao, Y., Adachi, J. and Hasegawa, M. (1998) Comment on the quartet puzzling method for finding maximum-likelihood. *Mol. Biol. Evol.*, **15**, 87–89.

Dress, A. and Krüger, M. (1987) Parsimonious phylogenetic trees in metric spaces and simulated annealing. *Adv. Appl. Math.*, **8**, 8–37.

Efron, B. (1979) Bootstrap methods: Another look at the jackknife. *Ann. Stat.*, **7**, 1–26.

Felsenstein, J. (1981) Evolutionary trees from DNA sequences: a maximum likelihood approach. *J. Mol. Evol.*, **17**, 368–376.

Felsenstein, J. (1985) Confidence limits on phylogenies: An approach using the bootstrap. *Evolution*, **39**, 783–791.

Felsenstein, J. (1993) *PHYLIP manual, version 3.5*. Department of Genetics, University of Washington, Seattle.

Fitch, W. M. (1971) Toward defining the course of evolution: Minimum change for a specific tree topology. *Syst. Zool.*, **20**, 406–416.

Fleissner, R., Metzler, D. and von Haeseler, A. (2005) Simultaneous statistical multiple alignment and phylogeny reconstruction. *Syst. Biol.*, **54**, 548–561.

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

Goldman, N. and Yang, Z. (1994) A codon–based model of nucleotide substitution for protein–coding DNA sequences. *Mol. Biol. Evol.*, **11**, 725–736.

Guindon, S. and Gascuel, O. (2003) A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Syst. Biol.*, **52**, 696–704.

Hordijk, W. and Gascuel, O. (2006) Improving the efficiency of SPR moves in phylogenetic tree search methods based on maximum likelihood. *Bioinformatics*, **21**, 4338–4347.

Kirkpatrick, S., Gelatt Jr., C. D. and Vecchi, M. P. (1983) Optimisation using simulated annealing. *Science*, **220**, 671–680.

Lemmon, A. R. and Milinkovitch, M. C. (2002) The metapopulation genetic algorithm: An efficient solution for the problem of large phylogeny estimation. *Proc. Natl. Acad. Sci. USA*, **99**, 10516–10521.

Lewis, P. O. (1998) A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data. *Mol. Biol. Evol.*, **15**, 277–283.

Lundy, M. (1985) Applications of the annealing algorithm to combinatorial problems in statistics. *Biometrika*, **72**, 191–198.

Matsuda, H. (1995) Construction of phylogenetic trees from amino acid sequences using a genetic algorithm. In *Proceedings of the Genome Informatics Workshop VI*, pp. 19–28, Universal Academy Press.

Nieselt-Struwe, K. and von Haeseler, A. (2001) Quartet-mapping, a generalization of the likelihood-mapping procedure. *Mol. Biol. Evol.*, **18**, 1204–1219.

Olsen, G. J., Matsuda, H., Hagstrom, R. and Overbeek, R. (1994) fastDNAml: A tool for construction of phylogenetic trees of DNA sequences using maximum likelihood. *Comput. Appl. Biosci.*, **10**, 41–48.

Press, W. H., Flannery, B. P., Teukolsky, S. A. and Vetterling, W. T. (1992) *Numerical Recipes in C: The Art of Scientific Computing*. 2nd edition, Cambridge University Press, Cambridge.

Quenouille, M. H. (1956) Notes on bias in estimation. *Biometrika*, **43**, 353–336.

Ronquist, F. and Huelsenbeck, J. P. (2003) MrBayes 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics*, **19**, 1572–1574.

Saitou, N. and Nei, M. (1987) The neighbor–joining method: A new method

for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, **4**, 406–425.

Salter, L. A. and Pearl, D. K. (2001) Stochastic search strategy for estimation of maximum likelihood phylogenetic trees. *Syst. Biol.*, **50**, 7–17.

Schmidt, H. A., Strimmer, K., Vingron, M. and von Haeseler, A. (2002) TREE-PUZZLE: Maximum likelihood phylogenetic analysis using quartets and parallel computing. *Bioinformatics*, **18**, 502–504.

Stamatakis, A. (2005) An efficient program for phylogenetic inference using simulated annealing. In *Online Proceedings of the 4th IEEE International Workshop on High Performance Computational Biology (HICOMB 2005)*, p. 8, Denver.

Stamatakis, A. (2006) RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics*, **22**, 2688–2690.

Stamatakis, A. P., Ludwig, T. and Meier, H. (2005) RAxML-III: a fast program for maximum likelihood-based inference of large phylogenetic trees. *Bioinformatics*, **21**, 456–463.

Steel, M. and Penny, D. (2000) Parsimony, likelihood, and the role of models in molecular phylogenetics. *Mol. Biol. Evol.*, **17**, 839–850.

Stewart, C. A., Hart, D., Berry, D. K., Olsen, G. J., Wernert, E. A. and Fischer, W. (2001) Parallel implementation and performance of fastDNAml - a program for maximum likelihood phylogenetic inference. In *Proceedings of the International Conference on High Performance Computing and Communications - SC2001*, pp. 191–201.

Strimmer, K., Goldman, N. and von Haeseler, A. (1997) Bayesian probabilities and quartet puzzling. *Mol. Biol. Evol.*, **14**, 210–213.

Strimmer, K. and von Haeseler, A. (1996) Quartet puzzling: A quartet maximum–likelihood method for reconstructing tree topologies. *Mol. Biol. Evol.*, **13**, 964–969.

Strimmer, K. and von Haeseler, A. (1997) Likelihood–mapping: A simple method to visualize phylogenetic content of a sequence alignment. *Proc. Natl. Acad. Sci. USA*, **94**, 6815–6819.

Strimmer, K. and von Haeseler, A. (2003) Phylogeny inference based on maximum likelihood methods with tree-puzzle. In Salemi, M. and Vandamme, A.-M. (eds.), *The Phylogentic Handbook*, pp. 137–159, Cambridge University Press, Cambridge, UK.

Uzzel, T. and Corbin, K. W. (1971) Fitting discrete probability distributions to evolutionary event. *Science*, **172**, 1089–1096.

Vinh, L. S. and von Haeseler, A. (2004) IQPNNI: Moving fast through tree space and stopping in time. *Mol. Biol. Evol.*, **21**, 1565–1571.

Wakeley, J. (1993) Substitution rate variation among sites in hypervariable region 1 of human mitochondrial DNA. *J. Mol. Evol.*, **37**, 613–623.

Yang, Z. and Nielsen, R. (1998) Synonymous and nonsynonymous rate variation in nuclear genes of mammals. *J. Mol. Evol.*, **46**, 409–418.

Zwickl, D. J. (2006) *Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion*. Ph.D. thesis, University of Texas, Austin, USA.