Recursions for statistical multiple alignment

Jotun Hein*, Jens Ledet Jensen†‡, and Christian N. S. Pedersen§

*Department of Statistics, Oxford University, South Parks Road, Oxford OX1 3SY, United Kingdom; and †Department of Theoretical Statistics, Institute of Mathematics, and §Department of Computer Science, University of Aarhus, Ny Munkegade, DK-8000 Aarhus C, Denmark

Communicated by Joseph Felsenstein, University of Washington, Seattle, WA, September 29, 2003 (received for review April 28, 2003)

Algorithms are presented that allow the calculation of the probability of a set of sequences related by a binary tree that have evolved according to the Thorne–Kishino–Felsenstein model for a fixed set of parameters. The algorithms are based on a Markov chain generating sequences and their alignment at nodes in a tree. Depending on whether the complete realization of this Markov chain is decomposed into the first transition and the rest of the realization or the last transition and the first part of the realization, two kinds of recursions are obtained that are computationally similar but probabilistically different. The running time of the algorithms is $O(\prod_{i=1}^{d} L_i)$, where L_i is the length of the ith observed sequences and d is the number of sequences. An alternative recursion is also formulated that uses only a Markov chain involving the inner nodes of a tree.

backward recursion | emission probability | forward recursion | hidden Markov chain | states

Proteins and DNA sequences evolve predominantly by substitutions, insertions, and deletions of single letters or strings of these elements, where a letter is either a nucleotide or an amino acid. During the last two decades, the analysis of the substitution process has improved considerably and has been based increasingly on stochastic models. The process of insertions and deletions has not received the same attention and is presently being analyzed by optimization techniques, for instance maximizing a similarity score as first used by Needleman and Wunsch (1).

Thorne, Kishino, and Felsenstein (2) proposed a well defined time-reversible Markov model for insertions and deletions [denoted more briefly as the Thorne, Kishiro, and Felsenstein (TKF) model] that allowed a proper statistical analysis for two sequences. Such an analysis can be used to provide maximum likelihood sequence alignments for pairs of sequences or to estimate the evolutionary distance between two sequences. Recently, an algorithm was presented by Steel and Hein (3) that allows statistical alignment of sequences related by a star-shaped tree, a tree with one inner node. Hein (4) formulated an algorithm that calculates the probability of observing a set of sequences related by a given tree in time $O((\prod_i L_i)^2)$, where L_i is the length of the ith sequence. This is also the time required by Steel and Hein's algorithm (3). Holmes and Bruno (5) used the algorithm by Hein (4) to design a Gibbs sampler that has the potential of analyzing a higher number of sequences than the exact algorithms. The present work accelerates, extends, and formalizes the algorithm in ref. 4. In particular, the time requirement for the algorithm presented here is reduced to $O(\prod_i L_i)$.

The TKF model is formulated in terms of links and associated letters. To each link is associated a letter that undergoes changes, independently of other letters, according to a reversible substitution process (identical to the site-substitution process, where insertions and deletions are not allowed). A link and its associated letter are deleted after an exponentially distributed waiting time with mean $1/\mu$. While a link is present, it gives rise to new links at the rate λ. A new link is placed immediately to the right of the link from which it originated, and the associated letter is chosen from the stationary distribution of the substitution process. At the left of the sequence is a so-called immortal link that never dies and gives rise to new links at the rate λ, preventing the process from becoming extinct.

For the TKF model on a tree, the defining parameters are the death rate μ and the birth rate λ, as described above, together with a time parameter τ for each edge of the tree. The time parameter τ defines how long the process runs along a given edge. When the process splits into two subprocesses at an inner node, the two subprocesses are independent.

The main probabilistic aspects of the TKF model are given by Eqs. **2–4** and **8** below. The structure of probabilities **3** and **4** allows us to write the joint probability of observed sequences at the leaves of a tree together with the alignment and the unobserved sequences at inner nodes of the tree as a Markov chain along the sequences observed, until the process reaches an absorbing state. The process of observed sequences therefore becomes a hidden Markov chain. Having obtained this identification, we can use traditional methods for obtaining a recursion for the calculation of the probability of the observed sequences. In particular, we state two recursions, one corresponding to splitting the process according to the first state of the Markov chain, and the other corresponding to splitting the process according to the last state of the Markov chain. In *Approach 1*, a state of the hidden Markov chain describes an element in the alignment for the whole tree, which gives a recursion with time complexity $O(\prod_i L_i)$ when implemented using dynamic programming. In *Approach 2*, we take a state of the hidden Markov chain to be an element in the alignment of the tree consisting of inner nodes only. This gives a recursion with time complexity $O((\prod_i L_i)^2)$; however, this can be reduced to $O(\prod_i L_i)$, and actually we obtain a recursion with slightly fewer terms than that considered in *Approach 1*. We start in *Preliminaries* by defining the states of our hidden Markov chain and finding the transition probabilities of the Markov chain. This section introduces necessary notation to allow for a precise mathematical formulation.

Preliminaries

Notation. We consider a tree with d' inner nodes and d leaves. The inner nodes are numbered from 1 to d', with 1 being the root and where the ancestor $a(i)$ of i is to be found in $\{1, 2, \ldots, i-1\}$. The leaves are numbered from $d'+1$ to $d'+d$, with the descendants of inner node i being numbered before the descendants of inner node j for $j > i$. For a tree with two inner nodes and four leaves, the numbering can be seen in Fig. 1.

The evolutionary time distance from the ancestor $a(z)$ of a node z to the node z is $\tau(z)$. The observed sequences are S_j for $j = d'+1, \ldots, d'+d$, where S_j is the observed sequence at the leaf j. The length of S_j is L_j, and the ath entry of S_j is denoted $S_j(a)$. We write $S_j(a:b)$ for entries from a to b with a and b included. We let S denote the collection of sequences, and for two d-dimensional vectors u, v indexed by $j = d'+1, \ldots, d'+d$ and with integer entries $S(u:v)$ denote the collection of subsequences $S_j(u_j:v_j)$. To compare two d-dimensional vectors u, v, the notation

$$u > v \text{ if } u_j > v_j \ \forall_j, \quad \text{and} \quad u \overset{w}{>} v \text{ if } u_j > v_j \text{ for some } j$$

is used, with similar definitions for other relations. To shorten the formulae, we write for two vectors, K, l, with $l \geq 0$, $S[K, l] = S((K - l + 1):K)$. Finally, L is the vector with entries L_j.

Abbreviation: TKF model, Thorne–Kishino–Felsenstein model.

‡To whom correspondence should be addressed. E-mail: jlj@imf.au.dk.

www.pnas.org/cgi/doi/10.1073/pnas.2036252100

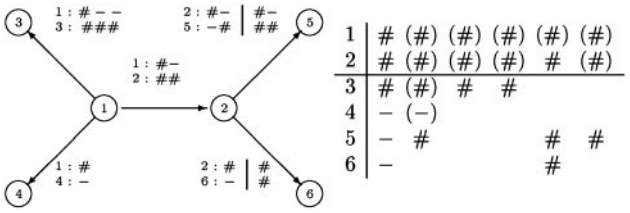

Fig. 1. A tree with four leaves, where the link a root 1 survives at inner node 2 and produces a new link at leaf 2.

The notation $1(E)$ is used for the function that is 1 when the expression E is true and 0 otherwise. We use the symbol # for a link, and when following the fate of a link along the tree, we write # at node i if the link is present, and we write – at node i if the link died along the edge from $a(i)$ to i.

Markov Structure of the TKF Model for Two Sequences. In this subsection, the TKF model from time zero to time τ is considered. We rewrite the probabilities for the deletion of a link and for the number of new links that appear before time τ in such a way that we recognize a Markov structure along the sequences with states

$$\binom{\#}{\#}, \binom{\#}{-}, \binom{-}{\#}, \quad [1]$$

corresponding to survival of the link, deletion of the link, and insertion of a new link.

Let $V = 1$ if the link survives, and let $V = 0$ if the link dies. Because the death rate is μ, we have

$$P(V = 1) = \exp(-\mu\tau) \quad \text{and} \quad P(V = 0) = 1 - \exp(-\mu\tau). \quad [2]$$

Let N be the number of new links after time τ. From Thorne, Kishino, and Felsenstein (2), we have

$$P(N = k|V = 1) = (1 - \lambda\beta)(\lambda\beta)^k, \quad k \geq 0, \quad [3]$$

$$P(N = k|V = 0) = \begin{cases} 1 - \kappa & k = 0 \\ \kappa(1 - \lambda\beta)(\lambda\beta)^{k-1} & k \geq 1, \end{cases} \quad [4]$$

where

$$\beta = \frac{1 - \exp((\lambda - \mu)\tau)}{\mu - \lambda\exp((\lambda - \mu)\tau)} \quad \text{and} \quad \kappa = 1 - \frac{\mu\beta}{1 - \exp(-\mu\tau)}.$$

From these formulae, we find

$$P(N \geq k|V = 1) = (\lambda\beta)^k, \quad k \geq 0,$$

$$P(N \geq k|V = 0) = \kappa(\lambda\beta)^{k-1}, \quad k \geq 1,$$

and this implies

$$P(N \geq k + 1|N \geq k, V = 1) = \lambda\beta, \quad k \geq 0, \quad [5]$$

$$P(N \geq k + 1|N \geq k, V = 0) = \begin{cases} \kappa & k = 0 \\ \lambda\beta, & k \geq 1 \end{cases}. \quad [6]$$

The important point for establishing a Markov structure along the sequences is that Eqs. **5** and **6** are equal and independent of k for $k \geq 1$. The independence of k gives the Markov structure for the number of new links, that is, a new link is added with probability $\lambda\beta$, and we stop adding new links with probability $1 - \lambda\beta$. We can thus generate V and N by a Markov chain with transition probabilities

	$\binom{\#}{\#}$	$\binom{\#}{-}$	$\binom{-}{\#}$	stop
start	$e^{-\mu\tau}$	$1 - e^{-\mu\tau}$	0	0
$\binom{\#}{\#}$	0	0	$\lambda\beta$	$1 - \lambda\beta$
$\binom{\#}{-}$	0	0	κ	$1 - \kappa$
$\binom{-}{\#}$	0	0	$\lambda\beta$	$1 - \lambda\beta$

[7]

To interpret the whole alignment as a Markov chain, we note that the number K of links at stationarity has the following distribution. (see Thorne, Kishino, and Felsenstein, ref. 2),

$$P(K = k) = \gamma^k(1 - \gamma), \quad k \geq 0, \quad \gamma = \frac{\lambda}{\mu}. \quad [8]$$

Again this corresponds to a Markov chain where we add a link with probability γ, and we stop adding more links with probability $1 - \gamma$. Having reached the stop state in system **7**, we thus add a new link at time zero with probability γ and start a new round of the Markov chain in Eq. **7**. We can combine this into a Markov chain on the states in Eq. **1** together with an End state as follows:

	$\binom{\#}{\#}$	$\binom{\#}{-}$	$\binom{-}{\#}$	End
$\binom{\#}{\#}$	$(1 - \lambda\beta)\gamma e^{-\mu\tau}$	$(1 - \lambda\beta)\gamma(1 - e^{-\mu\tau})$	$\lambda\beta$	$(1 - \lambda\beta)(1 - \gamma)$
$\binom{\#}{-}$	$(1 - \kappa)\gamma e^{-\mu\tau}$	$(1 - \kappa)\gamma(1 - e^{-\mu\tau})$	κ	$(1 - \kappa)(1 - \gamma)$
$\binom{-}{\#}$	$(1 - \lambda\beta)\gamma e^{-\mu\tau}$	$(1 - \lambda\beta)\gamma(1 - e^{-\mu\tau})$	$\lambda\beta$	$(1 - \lambda\beta)(1 - \gamma)$

[9]

As an example, the $\binom{\#}{\#}$, $\binom{\#}{\#}$ entry corresponds to going to the stop in Eq. **7** from $\binom{\#}{\#}$, adding a link at time zero with probability γ, and going to $\binom{\#}{\#}$ from start in Eq. **7**.

When considering the TKF model on a tree, we will need the terms in Eq. **7** for each edge of the tree. Because we number the edges by the node at the end of the edge, we introduce for each node $j > 1$ the terms

$$b(\#, \#; j) = \lambda\beta(j), \quad b(\#, -; j) = 1 - b(\#, \#; j),$$

$$b(-, -; j) = \frac{\mu\beta(j)}{1 - \exp(-\mu\tau(j))},$$

$$b(-, \#; j) = 1 - b(-, -; j),$$

$$s(\#; j) = \exp(-\mu\tau(j)), \quad s(-; j) = 1 - s(\#; j),$$

where $\beta(j) = \{1 - \exp((\lambda - \mu)\tau(j))\}/\{\mu - \lambda\exp((\lambda - \mu)\tau(j))\}$.

States. Because the probability of an alignment on the tree is the product of the probabilities of the pairwise alignments along the edges, we can use the hidden Markov structure presented in *Markov Structure of the TKF Model for Two Sequences* for pairwise alignment to obtain the hidden Markov structure for alignment on the tree. To obtain this, we need to choose states for the hidden Markov model on the tree that allow us to identify the states of the hidden Markov model for the pairwise alignments along the edges.

When translating a set of pairwise alignments between the nodes $(a(j), j)$, $2 \leq j \leq (d + d')$ into a sequence of states for the multiple alignment, we will use the convention that if a birth at node i and a birth at node $j > i$ both are the result of a birth at node $z < i$, then the birth at node j will appear before the birth at node i in the sequence.

Table 1. States of the Markov chain for the tree in Fig. 1

	Column number										
Node	1	2	3	4	5	6	7	8	9	10	11
1	#	(#)	(#)	(#)	(#)	#	(#)	(#)	(#)	(#)	(#)
2	#	(#)	(#)	(#)	(#)	-	(-)	(-)	#	(#)	(#)
3	a_3	(a_3)	(a_3)	(a_3)	#	a_3	(a_3)	#			
4	a_4	(a_4)	(a_4)	#		a_4	#				
5	a_5	(a_5)	#						a_5	(a_5)	#
6	a_6	#							a_6	#	
	16	8	4	2	1	4	2	1	4	2	1

Any of the variables a_i can be either # or −. The last row gives the number of states of the indicated form.

A state represents two things, a new event in part of the tree and a "history" in the complementary part of the tree. The two together give information on which new events are possible in the next state. A state ξ consists of some subsets of nodes together with a value $\xi(z) \in \{\#, -\}$ for the nodes z in these subsets. The new event attached to a state ξ is a birth of a link at some node $t(\xi)$ and, if $t(\xi)$ is an inner node, the survival ($\xi(z) = \#$) or nonsurvival ($\xi(z) = -$) along the tree down from $t(\xi)$. We let $T(\xi)$, respectively $L(\xi)$, be the set of inner nodes $z > t(\xi)$, respectively leaves, where we have survival or where the link died on the edge leading to the node. The history corresponds to a birth at node 1 and the survival ($\xi(z) = \#$) or nonsurvival ($\xi(z) = -$) along inner nodes $z < t(\xi)$, with the property that the link survived at the ancestor $a(t(\xi))$. We let $H(\xi)$ be the set of inner nodes $z < t(\xi)$ where the link survived or died on the edge leading to the node. Furthermore, if $t(\xi)$ is a leaf, the history contains an inner node $h(\xi) < a(t(\xi))$, $h(\xi) \in H(\xi)$ with $\xi(h(\xi)) = \#$ and a set $HL(\xi)$ of leaves $z > t(\xi)$ being descendant of $h(\xi)$ and for which the link at $h(\xi)$ survived or died on the edge leading to the leaf. For a state where $t(\xi)$ is an inner node, the next state can have a birth of a new link in any of the nodes in $H(\xi) \cup \{t(\xi)\} \cup T(\xi) \cup L(\xi)$, and for a state with $t(\xi)$ a leaf, a new link can be born at the nodes in $H(\xi) \cup HL(\xi) \cup \{t(\xi)\}$. Note that the history is defined in such a way as to respect our convention of the ordering of the births.

To exemplify the definitions above, let us consider the tree in Fig. 1 with two inner nodes. We represent the states as six-dimensional columns with values # or − in $\{t(\xi)\} \cup T(\xi) \cup L(\xi)$, with values (#) or (−) in $H(\xi) \cup HL(\xi)$, and with no value in the remaining nodes. All 45 possible states are listed in Table 1. Column 1 of Table 1 gives the 16 states corresponding to a birth at node 1 that survived at inner node 2. That the birth is at node 1 leaves no room for a history. Column 4 gives the two states corresponding to a birth at leaf 4 with $H(\xi) = \{1, 2\}$, $h(\xi) = 1$, and $HL(\xi) = \{3\}$. There are no values at leaves 5 and 6 due to our convention of the ordering of the births. Column 9 gives the four states corresponding to a birth at inner node 2. Here there are no values at leaves 3 and 4 due to our convention. In Fig. 1, the translation between the set of pairwise alignments and the states of the multiple alignment is illustrated. Fig. 1 displays the situation where there is one link only at node 1. This link survives at inner node 2 and produces a new link at node 2. The original link does not survive at leaves 4–6, but produces a new link at leaf 5. The original link survives at leaf 3 and produces two new links at this leaf. The new link at inner node 2 survives in both leaves 5 and 6 and produces a new link at leaf 5. The set of states in the multiple alignment is shown in Fig. 1 *Right*. The first state is the birth of the link at node 1 together with the survival and nonsurvival of this link. State 2 is the birth at node 5 coming from the original link at node 1. States 3 and 4 are the two births at node 3. State 5 is the birth of a new link at node 2 together with the survival at nodes 5 and 6. Note that there are no values in this state at nodes 3 and 4 due to the convention that a birth at inner node 2 implies that all births at nodes 3 and 4 have been handled. Finally, the last state is the birth at node 5 originating from the new link at node 2.

As mentioned in the beginning of this subsection, we get a Markov chain because we can identify all the pairwise alignments along the edges from the states we use. To illustrate this, let us write down the probability of the alignments in Fig. 1 as follows:

$$\begin{array}{llll} s(\#;2) & & b(\#,\#;2) & b(\#,-;2) \\ s(\#;3) & b(\#,\#;3)b(\#,\#;3)b(\#,-;3) & & \\ s(-;4) & b(-,-;4) & & \\ s(-;5)\, b(-,\#;5) & b(\#,-) & s(\#,5) & b(\#,\#;5)b(\#,-;5) \\ s(-;6)\, b(-,-;6) & & s(\#,6) & b(\#,-;6) \end{array} . \qquad [10]$$

Here each row represents the probability of the alignment along one of the edges. The terms in a row have been spread out to align terms vertically, as explained below. The terms $s(\#;j)$ and $s(-;j)$ are the two entries in the first row of Eq. **7** corresponding to survival and nonsurvival of a link along the edge leading to node j, $b(\#,\#;j)$, $b(\#,-;j)$ are the two entries from the second and fourth row of Eq. **7**, and $b(-,\#;j)$ and $b(-,-;j)$ are the two entries from the third row of Eq. **7**. The first row of Eq. **10** gives the probability of the alignment between nodes 1 and 2, which is given through the survival of the link together with the probability of a birth of a new link and the probability of no further links. The product of the terms in each column in Eq. **10** represents a transition probability in the chain with 45 states, except for the first column that has to be combined with the probability related to leaving a previous state, and the last column that has to be combined with the probability related to the next link at root 1. As can be seen, each column is a function of the corresponding consecutive set of states in Fig. 1.

When stating the transition probabilities, it is convenient to have the following notation. For a state ξ, we let $\gamma = \gamma(r, \xi)$ be the state we enter when having a birth at the leaf r. If $t(\xi)$ is an inner node $r \in L(\xi)$, $H(\gamma) = H(\xi) \cup \{t(\xi)\} \cup T(\xi)$, $h(\gamma) = t(\xi)$, $HL(\gamma) = \{z \in L(\xi) | z < r\}$, and $\gamma(z) = \xi(z)$ for nodes in these sets. If $t(\xi)$ is a leaf $r \in HL(\xi) \cup \{t(\xi)\}$, $H(\gamma) = H(\xi)$, $h(\gamma) = h(\xi)$, $HL(\gamma) = \{z \in HL(\xi) | z < r\}$, and $\gamma(z) = \xi(z)$ for nodes in these sets. The set of all states is denoted Ξ, and the subset of states ξ with $t(\xi)$ an inner node is denoted Ξ_1.

Transition Probabilities. A transition probability $p(x,y)$ of going from state x to state y can be written formally as $p(x, y) = \text{stop} \cdot \text{new} \cdot \text{survival}$, where stop gives the probability of no new links at certain nodes, new gives the probability of a new link at a particular node, and survival gives the probability of the fate of the new link. We thus find for a state $\xi \in \Xi_1$

$$p(\xi, \gamma(r, \xi)) = \prod_{j \in L(\xi), j > r} b(\xi(j), -, j) b(\xi(r), \#, r). \qquad [11]$$

For a state ξ with $t(\xi)$ a leaf, we get with $s = t(\xi)$

$$p(\xi, \gamma(r, \xi)) = b(\#, -, s)^{1(s>r)} \prod_{j \in HL(\xi),\, r<j<s} b(\xi(j), -, j) \times b(\xi(r), \#, r)^{1(s>r)}\, b(\#, \#, r)^{1(s=r)}. \qquad [12]$$

For two states $\xi, \eta \in \Xi_1$, a transition from ξ to η is possible only if η corresponds to a new link at one of the inner nodes from which ξ allows the introduction of new links. This can be formulated formally as

$$t(\eta) \in H(\xi) \cup \{t(\xi)\} \cup T(\xi), \quad \eta(j) = \xi(j), \quad j \in H(\eta),$$

$$H(\eta) = \{j \in H(\xi) \cup \{t(\xi)\} \cup T(\xi) | j < t(\eta)\}.$$

In this case, the transition probability is

$$p(\xi, \eta) = \prod_{j\in L(\xi)} b(\xi(j), -, j) \prod_{j\in T(\xi)\cup H(\xi),\, j>t(\eta)} b(\xi(j), -, j) \times b(\xi(t(\eta)), \#; t(\eta))^{1(t(\eta)>1)} (\lambda/\mu)^{1(t(\eta)=1)} \times \prod_{j\in T(\eta)\cup L(\eta)} s(\eta(j), j). \quad [13]$$

When $t(\xi)$ is a leaf and $t(\eta)$ is an inner node, the transition from ξ to η requires

$$t(\eta) \in H(\xi), \quad \eta(j) = \xi(j), \quad j \in H(\eta),$$
$$H(\eta) = \{j \in H(\xi) | j < t(\eta)\},$$

and the transition probability $p(\xi, \eta)$ is

$$b(\#, -, t(\xi)) \prod_{j\in HL(\xi)} b(\xi(j), -, j) \prod_{j\in H(\xi),\, j>t(\eta)} b(\xi(j), -, j) \times b(\xi(t(\eta)), \#; t(\eta))^{1(t(\eta)>1)} (\lambda/\mu)^{1(t(\eta)=1)} \prod_{j\in T(\eta)\cup L(\eta)} s(\eta(j), j). \quad [14]$$

For a transition to the end state, only the first line of Eqs. **13** and **14** should be used, multiplied by $(1 - \lambda/\mu)$, and with $t(\eta) = 1$. Finally, the transition probabilities from the immortal state I can be calculated as if I corresponds to the state ξ_0 with a new link at node 1 that survives in all of the tree.

Algorithms

In this section, we present two algorithms for computing the probability of the observed sequences S_j, for $j = d' + 1, \ldots, d' + d$, being related by the given evolutionary tree. Both algorithms are based on the hidden Markov chain described in the previous section but differ in their choice of states. In the first algorithm, the states describe the alignment for both inner nodes and leaves. The running time is $O(\prod_{j=d'+1}^{d'+d} L_j) = O(L_{\max}^d)$, where $L_{\max}$ is the maximum length of the observed sequences. In the second algorithm, the states describe the alignment for inner nodes only. The running time is now $O(L_{\max}^{2d})$, but the algorithm can be rewritten to obtain an $O(L_{\max}^d)$ running time as in the first approach. The principle for deriving the algorithms is classical and very well known: we consider what happens in either the first or last step of the Markov chain.

Approach 1: Inner Nodes and Leaves. ***Notation.*** We consider a Markov process $x_0, x_1, \ldots, x_N$ that starts in the initial state I and stops at a random time $N + 1$ in the end state E. Thus $x_0 = I, x_i \in \Xi$, for $i = 1, \ldots, N$, and $x_{N+1} = E$. The transition probability going from x to y is $p(x, y)$ as described in *Transition Probabilities*. A state $\xi \in \Xi_1$, corresponding to the birth of a link at an inner node, emits a letter in those observed sequences S_z for which $z \in L_1(\xi) = \{u \in L(\xi) | \xi(u) = \#\}$. A state $\xi \notin \Xi_1$ emits a letter in the sequence $S_{t(\xi)}$ only. For any state $x \in \Xi$, we let

$$l(x) = (l_{d'+1}(x), l_{d'+2}(x), \ldots, l_{d'+d}(x)) \quad [15]$$

be a vector indexed by the numbering of the observed sequences and consisting of ones in those coordinates for which x emits a letter and zeroes in the other coordinates:

$$l_j(x) = \begin{cases} 1 & \text{if } x \text{ emits a letter in sequence } S_j \\ 0 & \text{otherwise.} \end{cases} \quad [16]$$

For the state x_i in the hidden Markov chain, we use the shorthand notation l^i for the vector $l(x_i)$. The lengths L^i of the sequences emitted by the first i states $x_1, \ldots, x_i$ can then be written as $L^i = \sum_{r=1}^{i} l^r$. With this notation, the state x_i emits the letters $S(L^{i-1} + 1 : L^{i-1} + l^i) = S[L^i, l^i]$, where $S_j(L_j^{i-1} + 1 : L_j^{i-1} + l_j^i)$ is the empty set if $l_j^i = 0$. The probability that a state x emits the vector of letters s (with the possibility that some of the coordinates of s are equal to the empty set) is $p_e(s|x)$.

Backward recursion. For an arbitrary vector $K \geq 0$ and state $x_0 \in \Xi$, we define $F(K|x_0) = P(S(K + 1 : L)|x_0)$, that is, the probability that the sequences $S((K + 1) : L)$ are produced by a set of states $x_1, x_2, \ldots$ given that the Markov chain starts in the state x_0. Clearly, $P(S(1 : L)|x_0 = I) = F(0|I)$. Summing over the states of the Markov chain $F(K|x_0)$ is given by

$$\sum_{n=0}^{\infty} \sum_{x_1,\ldots,x_n\in\Xi: K+L^n=L} p(x_n, E)(S[K + L^i, l^i]|x_i). \quad [17]$$

When $K <^w L$ and $K \leq L$, the recursion for $F(K|x_0)$, with $\tilde{L}^i = \sum_{r=2}^{n} l(x_i)$, is

$$\begin{aligned} F(K|x_0) &= \sum_{n=1}^{\infty} \sum_{x_1\in\Xi} p(x_0, x_1) p_e(S[K + l^1, l^1]|x_1) \\ &\quad \times \sum_{\substack{x_2,\cdots,x_n\in\Xi: \\ (K+l(x_1))+\tilde{L}^n=L}} p(x_n, E) \\ &\quad \times \prod_{i=1}^{n} p(x_{i-1}, x_i) p_e(S[K + l^1 + \tilde{L}^i, l^i]|x_i) \\ &= \sum_{z\in\Xi} p(x_0, z) p_e(S[K + l(z), l(z)]|z) F(K + l(z)|z). \end{aligned} \quad [18]$$

When $K = L$, the recursion is

$$F(L|x_0) = p(x_0, E) + \sum_{z\in\Xi: l(z)=0} p(x_0, z) F(L|z). \quad [19]$$

Recursion **18** states that the probability of the sequences $S((K + 1) : L)$ produced by states $x_1, x_2, \ldots$, given that we start in state x_0, is a sum over the possible states of x_1. Each term in the sum is the product of the transition probability of going from x_0 to x_1, the emission probability for those letters emitted by x_1, and the probability of the remaining sequences $S((K + l(x_1) + 1) : L)$ given that we start in x_1. If in recursion **18** we replace the summation by the max operation, we obtain a recursion for finding the alignment with the highest probability. This is known as the Viterbi algorithm in the hidden Markov model literature.

Hein, Jensen, and Pedersen (6) also derive a forward recursion by separating out the contribution from x_n instead of x_0. Computationally, there is no difference between the forward and backward recursions. However, the latter has an interpretation as a probability, thereby making it easier to understand.

Emission probabilities. For a full description of the TKF model, we need a model for the substitution process. We let $f_{b|a}^{\tau}$ be the probability for the substitution of a letter a by b over a time period τ. The stationary probabilities for this transition matrix are denoted by π.

When a state corresponds to a birth of a new link in one of the leaves only, that is, $t(\xi)$ is a leaf, the emitted vector s has a letter at the node $t(\xi)$ only, and the emission probability is simply the stationary probability $\pi(s(t(\xi)))$. For a state $\xi \in \Xi_1$ corresponding to a birth of a new link at inner node $t(\xi)$, the emitted vector s has letters at those nodes $z \in L(\xi)$ for which $\xi(z) = \#$. With $a(j)$, the ancestor of a node j, and with

$$T_1(\xi) = \{z \in T(\xi) | \xi(z) = \#\}, \quad L_1(\xi) = \{z \in L(\xi) | \xi(z) = \#\}, \quad [20]$$

we can write the emission probability as

Table 2. Example of alignment

True	Maximal	CLUSTAL W
TTATAT-G-ACTTG-CC-GG	T-TATA-TGACTTGCCGG	TTATAT--GAC-TTGC-CGG
CT-TCG-G-ACGTG-GC-TC	C-T-TC-GGACGTGGCTC	CT-T-C-GGACGTGGCTC--
CCAAAC-GGACGTTAC--GC	CCAAAC-GGACGTTACGC	CCAAAC-GGACGTTACGC
-AAAACAG-GACTTAT-A-C	A-AAACAGGAC-TTATAC	-AAAACAGGAC-TTATAC--

Four sequences and their true alignment that generated the sequences, the maximal alignment obtained from the Viterbi algorithm, and the alignment obtained from the CLUSTAL W program (www.ebi.ac.uk/clustalw).

$$p_e(s|\xi) = \sum_{\upsilon_{t(\xi)}} \pi(\upsilon_{t(\xi)}) \sum_{\upsilon_z, z\in T_1(\xi)} \prod_{z\in T_1(\xi)} f^{\pi(z)}_{\upsilon_z|\upsilon_{a(z)}} \prod_{z\in L_1(\xi)} f^{\pi(z)}_{s_z|\upsilon_{a(z)}}. \quad [21]$$

This formula simply says that the probability of the emitted letters $s_z, z \in L_1(\xi)$ is the sum of the joint probability of the ancestral and emitted letters over the possible values of the ancestral letters.

Implementation and analysis. Let us briefly discuss how to implement the recursion given by Eqs. **18** and **19**. There is a complication, in that there will always be terms on the right-hand side of the equations for which $K + l(z) = K$ or $l(z) = 0$. The states $\xi \in \Xi_1$ for which $l(\xi) = 0$ are characterized by having $\xi(z) = -$ for all $z \in L(\xi)$, that is, the new link does not survive at any of the leaves. Let us denote this class of states by C. Imagine that for some K, the term $F(\tilde{K}|x)$ has been calculated for all $\tilde{K} >^w K, \tilde{K} \geq K$ and all $x \in \Xi$. For each $x \in C$, recursion **18** gives

$$F(K|x) = \sum_{z\in C} p(x, z)F(K|z) + \omega(x), \quad [22]$$

with $\omega(x)$ known. Let Q be the matrix with entries $p(z_1, z_2), z_1, z_2 \in C$. Then, because the entries are nonnegative and the sum along a row is < 1, the matrix $I_C - Q$, where I_C is the identity matrix, is invertible, and the set of linear equations **22** has a unique solution. Having solved this system of equations, we can next calculate $F(K|x)$ for $x \notin C$ directly from Eq. **18**, or from Eq. **19** when $K = L$. Also in the case of the Viterbi algorithm for finding the alignment with the highest probability, we must, for a given K in the recursion, first solve for the states in C. The boundary conditions for the recursion are $F(K|x) = 0$ when $K >^w L$.

To run the algorithm, we need to calculate $F(K|x)$ for any $K \leq L$ and for any $x \in \Xi$. The number of steps needed is therefore of the order $N \prod_{i=1}^{d} L_i$, where N is the number of elements in the set Ξ.

For illustration purposes, we have implemented recursion **18** as well as the Viterbi algorithm and an algorithm for simulating alignments conditional on the observed sequences for the case of four observed sequences. No attempt to optimize the program has been made, and the program therefore runs only on short sequences. As an example, we use a set of simulated sequences kindly supplied by Yun Song (Oxford University, Oxford). The parameters used in the simulation are $\lambda = 0.05$, $\mu = 0.052$, and the Jukes–Cantor model for substitution where the rate of leaving a state is 0.3. All edges of the tree have lengths 1. We use the same parameters when finding the maximal alignment. The true alignment that generated the sequences and the maximal alignment can be seen in Table 2. We have also included an alignment obtained from the CLUSTAL W program by Higgins, Thompson, and Gibson (7). The total probability of the observed sequences is 7.62×10^{-41}, as obtained from recursion **18**, and the probability of the maximal alignment is 2.04×10^{-43}, contributing only 0.27% to the total probability.

The maximal alignment and the CLUSTAL W alignment agree on aligning GAC in the middle. We have run 500 simulations of the conditional alignment given the observed sequences, and in 78% of the cases, we find that GAC is aligned. CLUSTAL W aligns the last C of the four sequences, and this is not seen in the maximal alignment. In the 500 simulations from the conditional alignments, we never encountered a case where the last C was aligned. Generally, the possibility of simulating alignments from the conditional distribution given the observed sequences allows us to make statements on the reliability of features seen in an alignment.

Approach 2: Inner Nodes Only. ***Notation.*** In *Approach 1*, a state described a column of the alignment for all of the inner nodes and leaves, and a state emitted at most one letter in each of the observed sequences. In this section, we will instead let the states describe the inner nodes only, which in turn necessitates the emission of arbitrary long subsequences among the observed sequences. This implies an extra sum in the recursion, thus seemingly making the recursion more complicated. However, we can rewrite the recursion, ending up with a recursion of the same complexity as before and with fewer terms than in *Approach 1*.

More precisely, a state ξ is a birth of a new link at an inner node and is characterized by the node $t(\xi)$ at which the link is born; the set $T(\xi)$ of inner nodes describes the fate of the link, and the set $H(\xi)$ gives the history of the new link. As before, $L(\xi)$ is the set of leaves descending from $t(\xi)$ or from the nodes in $T_1(\xi)$ (see Eq. **20**). As in Eq. **15**, $l(x)$ denotes the lengths of the emitted subsequences. Contrary to Eq. **16**, $l(x)$ is no longer determined by the state x; the state determines only at which nodes it is possible to emit letters:

$$l_j(x) = \begin{cases} \geq 0 & j \in L(x) \\ 0 & \text{otherwise.} \end{cases} \quad [23]$$

We again use $l^i = l(x_i)$, furthermore take $l^0 \geq 0$ to be the length of the subsequence emitted by the immortal link, and define $L^i = \sum_{r=0}^{i} l^r$ to be the lengths of the sequences emitted by the first i states. The emission probability $q_e(K, l|x)$ is now both the probability of emitting subsequences of length $l_j, j = d' + 1, \ldots, d' + d$ and the probability that the emitted letters are $S_j((K_j - l_j + 1) : K_j)$. To state this probability, we define

$$L_1(\xi, l) = \{z \in L(\xi)|l_z > 0\}.$$

In the formula below, u denotes the subset of the leaves $L_1(\xi, l)$ at which the link survives from the ancestral inner nodes. To shorten the formula, we define $q(\#; z) = s(-; z)b(-, \#; z)$, $q(-; z) = s(-; z)b(-, -; z)$, $A_z^{m,l} = s(\#; z)\pi(S_z(m_z + 2 : m_z + l_z))$, and $B_z^{m,l} = s(-; z)b(-, \#; z)\pi(S_z(m_z + 1 : m_z + l_z))$, where $\pi(S_j(a : b)) = \prod_{i=a}^{b} \pi(S_j(i))$. Then the probability that the state ξ emits the subsequences $S(m + 1 : m + l)$ is $q_e(m + l, l|\xi)$, given by

$$\prod_{z\in L(\xi)\setminus L_1(\xi,l)} q(-; z) \prod_{z\in L_1(\xi,l)} b(\#, \#; z)^{l_z-1} b(\#, -; z) \times \sum_{u\subseteq L_1(\xi,l)} f(m, u, \xi) \prod_{z\in u} A_z^{m,l} \prod_{z\in L_1(\xi,l)\setminus u} B_z^{m,l}. \quad [24]$$

The function $f(m, u, \xi)$ is the emission probability for the first letter at those leaves where we have survival of the link and is given by $p_e(s|\xi)$ from Eq. **21**, with $L_1(\xi)$ replaced by u and s_z replaced by $S_z(m_z + 1)$. Furthermore,

$$q_e(l, l|I) = \prod_{d'<z\leq d'+d} b(\#, \#; z)^{l_z} b(\#, -; z)\pi(S_z(1 : l_z)).$$

Backward recursion. The backward recursion is obtained by defining

$$F(K|x_0) = \sum_{n=0}^{\infty} \sum_{x_1,\ldots,x_n\in\Xi} \sum_{l^0,l^1,\ldots,l^n:K+L^n=L} q_e(K + l^0, l^0|x_0) \times \left(\prod_{i=1}^{n} p(x_{i-1}, x_i) q_e(K + L^i, l^i|x_i) \right) p(x_n, E), \quad [25]$$

which equals $P(S(K+1:L)|x_0)$. Separating out the sum over the first state x_1, we find

$$F(K|x_0) = q_e(L, L-K|x_0)p(x_0, E) + \sum_{\xi\in\Xi}\sum_{l(\xi)} p(x_0, \xi)q_e(K+l(\xi), l(\xi)|\xi)F(K+l(\xi)|\xi) \quad [26]$$

for $K <^w L$ and $K \le L$, and when $K = L$, the recursion is

$$F(L|x_0) = p(x_0, E) + \sum_{\xi\in\Xi} p(x_0, \xi)P(l(\xi)=0|\xi)F(L|\xi). \quad [27]$$

A forward recursion can be derived in the same way. The details are described by Hein, Jensen, and Pedersen (6).

Reduction of complexity. For the recursions described in the previous subsection, we need to calculate $F(K|x)$ for any value of $K \le L$. This takes of the order $O(L_{\max}^d)$ steps. Each step here, however, involves the sum over l (see Eq. **26**) and therefore requires of the order $O(L_{\max}^d)$ steps. The time complexity of the algorithms is thus of the order $O(L_{\max}^{2d})$. The algorithms are therefore inferior to those given in *Approach 1*. It turns out, though, that we can rewrite the algorithms in such a way that the resulting time complexity becomes $O(L_{\max}^d)$ and where the constant factor hidden by the O notation is slightly smaller than for the algorithms of *Approach 1*. We start by inserting Eq. **24** into recursion **26**.

$$F(K|x) = q_e(L, L-K|x)p(x, E) + \sum_{\xi\in\Xi} p(x, \xi) \times \sum_{\substack{l_z\ge 0,\ z\in L(\xi);\\ l_z=0, z\notin L(\xi)}} \prod_{z\in L(\xi)\setminus L_1(\xi,l)} q(-;z) \prod_{z\in L_1(\xi,l)} b(\#,\#;z)^{l_z-1} \times \sum_{u\subseteq L_1(\xi,l)} f(K,u,\xi)\prod_{z\in u} A_z^{K,l} \prod_{z\in L_1(\xi,l)\setminus u} B_z^{K,l} F(K+l|\xi), \quad [28]$$

where, as before, u is the subset of the leaves $L_1(\xi, l)$ at which we have survival from the ancestral inner node. If we let w be the subset of the leaves $L(\xi)\setminus u$ at which there is no survival, but the number of new links is positive, and we introduce the subset v of $u \cup w$ at which $l_z \ge 2$, this gives

$$F(K|x) = q_e(L, L-K|x)p(x, E) + \sum_{\xi\in\Xi} p(x, \xi) \times \sum_{u\subseteq L(\xi)} f(K,u,\xi) \sum_{w\subseteq L(\xi)\setminus u} \prod_{z\in L(\xi)\setminus(u\cup w)} q(-;z) \times \sum_{v\subseteq(u\cup w)} \prod_{z\in u} s(\#;z) \prod_{z\in w} q(\#;z) \times \pi(S_z(K_z+1)) \prod_{z\in v} b(\#,\#;z) G(K+1(u\cup w)|\xi, v), \quad [29]$$

where $1(u \cup w)$ is 1 when $z \in (u \cup w)$ and 0 when $z \notin (u \cup w)$. The function G is defined as

$$G(M|\xi, v) = \sum_{m_z\ge 1,\ z\in v;\ m_z=0, z\notin v} F(M+m|\xi) \times \prod_{z\in v} b(\#,\#;z)^{m_z-1} \pi(S_z(M_z+1:M_z+m_z)) \quad [30]$$

for a nonempty subset v of $L(\xi)$, and $G(M|\xi, \emptyset) = F(M|\xi)$.

We can obtain a recursion for G by splitting the sum in Eq. **30** into $\sum_{\tilde v\subseteq v}\sum_{m_z\ge 2, z\in\tilde v; m_z=1, z\in(v\setminus\tilde v); m_z=0, z\notin v}$, where $\tilde v$ can be the empty subset. This gives

$$G(M|\xi, v) = \prod_{z\in v}\pi(S_z(M_z+1)) \sum_{\tilde v\subseteq v}\prod_{z\in\tilde v} b(\#,\#;z) \times \sum_{m_z\ge 2, z\in\tilde v; m_z=1, z\in(v\setminus\tilde v); m_z=0, z\notin v} G(M+1(v)|\xi, \tilde v). \quad [31]$$

Combining Eqs. **29** and **31**, we have established a recursion involving the functions $F(K|\xi)$ and $G(K|\xi, v)$. For the tree in Fig. 1, the recursions of *Algorithms* involves 45 terms, whereas the number of terms for the recursions in this section is 24.

Discussion

This work presents algorithms that have the same complexity as the traditional nonstatistical multiple alignment algorithm in Sankoff (8). The statistical alignment approach to sequence analysis differs relative to the optimization approach in focusing on obtaining the probability of the sequences under the given model, rather than obtaining an alignment. Among molecular biologists, however, it is popular to consider the actual alignment, and the one chosen is typically the alignment that contributes the most to the probability of the observed sequences. The latter can be calculated by simple modifications of the central recursions of this work, where a summation operator is substituted by a maximization operator. Several additional problems have to be solved to make the algorithm of this paper useful in real data analysis. Besides actually implementing the algorithm, it needs to be coupled to a numerical optimization method to find maximum likelihood estimates of the unspecified parameters, such as branch lengths, substitution parameters, and insertion and deletion rates. This method can then be used to analyze up to, say, four sequence of realistic lengths (hundreds of base pairs/amino acids). Elementary computational tricks can extend this to six or seven sequences; beyond this, radically different methods will have to be applied. Jensen and Hein (9) suggested a simulation technique where the basic step is the simulation of the alignment of a three-star tree. The Gibbs sampler proposed by Holmes and Bruno (5) is based on samplings that require pairwise alignments only. This is a faster operation, whereas the Gibbs sampler proposed by Jensen and Hein (9) achieves a more efficient mixing per move. From the perspective of a biologist, the underlying model for this paper can be criticized. First, the assumption that all insertions/deletions are only one nucleotide/amino acid long does not conform to the biological reality and should be relaxed. Second, the assumption that all positions in a sequence evolve according to the same rates is also unrealistic. Formulating models and ways to calculate the relevant probabilities in such models is a major challenge to the field if a statistical approach to alignment is to be of widespread use.

1. Needleman, S. B. & Wunsch, C. D. (1970) *J. Mol. Biol*. **48,** 443–453.
2. Thorne, J. L., Kishino, H. & Felsenstein, J. (1991) *J. Mol. Evol*. **33,** 114–124.
3. Steel, M. & Hein, J. (2001) *Appl. Math. Lett*. **14,** 679–684.
4. Hein, J. (2001) *Proc. Pac. Symp. Biocomp*. 179–190.
5. Holmes, I. & Bruno, W. J. (2001) *Bioinformatics* **17,** 803–820.
6. Hein, J., Jensen, J. L. & Pedersen, C. N. S. (2002) *Recursions for Multiple Statistical Alignment* (Dept. of Theoretical Statistics, University of Aarhus, Aarhus, Denmark), Technical Report no. 425.
7. Higgins, D. G., Thompson, J. D. & Gibson, T. J. (1994) *Nucleic Acids Res*. **22,** 4673–4680.
8. Sankoff, D. (1975) *SIAM J. Appl. Math*. **78,** 35–42.
9. Jensen, J. L. & Hein. J. (2002) *Gibbs Sampler for Statistical Multiple Alignment* (Dept. of Theoretical Statistics, University of Aarhus, Aarhus, Denmark), Technical Report no. 429.