Recursions for Statistical Multiple Alignment

Jotun Hein

Department of Statistics, Oxford University,

The Peter Medawar Building for Pathogen Research,

South Parks Road, Oxford OX1 3SY, England

Jens Ledet Jensen*

Department of Theoretical Statistics, Institute of Mathematics,

Ny Munkegade, DK-8000 Aarhus C, Denmark.

Phone: +45 89423526 Fax: +45 86131769 Email: jlj@imf.au.dk

Christian N.S. Pedersen

Department of Computer Science, Ny Munkegade,

DK-8000 Aarhus C, Denmark

Keywords: backward recursion, emission probability, forward recursion, hidden markov chain, states, transition probability

Abstract

Algorithms are presented that allows 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 recursions are based on a Markov chain generating sequences and their alignment at nodes in a tree.

Dependent 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 are $O(\prod_{i=1}^{d} L_i)$, where L_i is the length of the i'th observed sequence and d is the number of sequences - leaves at the binary tree. An alternative recursion is also formulated that only uses a Markov chain involving the inner nodes of a tree.

1 Introduction

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 increasingly been based on stochastic models. The process of insertions and deletions have not received the same attention and is presently being analysed by optimization techniques for instance maximizing a similarity score as first used by Needleman and Wunch (1970).

In 1991 Thorne, Kishino and Felsenstein proposed a well defined time reversible Markov model for insertions and deletions (denoted more briefly as the *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 (2000) that allowed statistical alignment of sequences related by a star shaped tree - a tree with one inner node. Hein (2001) formulated an algorithm that calculate the probability of observing a set of sequences related by a given tree in $O((\prod_i L_i)^2)$ time, where L_i is the length of the i'th sequence. This is also the time required by the algorithm in Steel and Hein (2000). The present article accelerates, extends and formalizes the algorithm in Hein (2001). In particular the time requirement for the algorithm presented here is $O(\prod_i L_i)$.

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 is 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 very left of the sequence is a socalled immortal link that never dies and gives rise to new links at the rate λ. This prevents the process form 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 for how long the process runs along a given edge. A scaling of the times are needed in that multiplying all the τ's by a constant c and dividing μ and λ by c will give the same process. 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 as follows. At equilibrium the probability that the sequence has l links is $(1-\frac{\lambda}{\mu})(\frac{\lambda}{\mu})^l$. Given that the sequence has length l the equilibrium probability of a specific sequence of letters is $\prod_{i=1}^{l}\pi(S(i))$, where $S(i)$ is the i'th letter in the sequence and $\pi(\cdot)$ is the equilibrium distribution of the substitution process. The probability that a link survives over a time span τ and during that time gives rise to k new links is given by (2) and (3) below, and the same probability when the link does not survive is given in (2) and (4). For the immortal link the probability of k new links is given by (3). Thus the probability of two aligned sequences can be written in terms of the stationary probability of sequence 1 together with the product of the appropiate probabilities for the fate of each link and associated letter in sequence 1 as seen in sequence 2. The latter involves (2), (3), and (4), the stationary probabilities for the letters associated to the new links, and the substitution probability in the case of survival of a link from sequence 1 to sequence 2.

The structure of the probabilities (3) and (4) allow 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 observed until the process reaches an absorbing state. The process of observed sequences therefore become 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 one corresponding to splitting the process according to the last state of the Markov chain. In section 3.1 a state of the hidden Markov chain describes an element in the alignment for the whole tree, and this gives a recursion of complexity $O(\prod_i L_i)$. In section 3.2 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 of 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 the one considered in section 3.1.

We start in section 2 by defining the states of our hidden Markov chain and finding the transition probabilities of the Markov chain. This section introduces necessary notation in order to allow for a precise mathematical formulation.

2 Preliminaries

2.1 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 found in $\{1, 2, \ldots, i-1\}$. The leaves are numbered from $d'+1$ to $d'+d$ with the descendants of the inner node i being numbered before the descendants of the inner node j for $j > i$. Thus for a tree with two inner nodes and four leaves the numbering can be seen in Figure 3 and for a tree with four inner nodes and 6 leaves the numbering can be seen in Figure 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 a'th entry of S_j is denoted $S_j(a)$. We write $S_j(a:b)$ for the 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 $j = d' +$

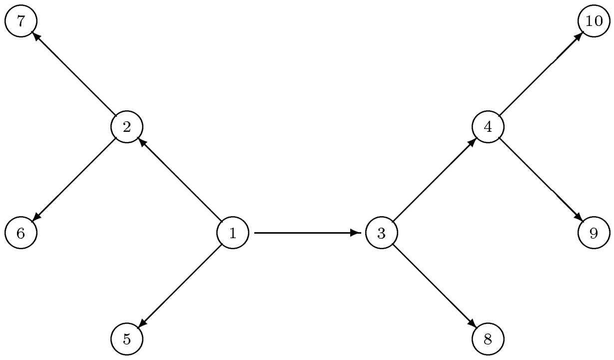


Figure 1: A rooted tree with ten nodes

$1, \ldots, d' + d$ 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 and 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.

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.

2.2 Markov structure of *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 appears before time τ in such a way that we recognize a Markov structure on the states

$$\begin{pmatrix} \# \\ \# \end{pmatrix}, \quad \begin{pmatrix} \# \\ - \end{pmatrix}, \quad \begin{pmatrix} - \\ \# \end{pmatrix}, \tag{1}$$

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

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

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

Let N be the number of new links after time τ. From Thorne, Kishiono, and Felsenstein (1991) we have

$$P(N=k|V=1) \;=\; (1-\lambda\beta)(\lambda\beta)^k, \quad k \geq 0, \tag{3}$$

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

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

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

and this implies

$$P(N \geq k+1|N \geq k, V=1) \;=\; \lambda\beta, \quad k \geq 0, \tag{5}$$

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

The important point for establishing a Markov structure is that (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

$$\begin{array}{c|cccc} & \binom{\#}{\#} & \binom{\#}{-} & \binom{-}{\#} & \text{stop} \\ \hline \text{start} & \mathrm{e}^{-\mu\tau} & 1-\mathrm{e}^{-\mu\tau} & 0 & 0 \\ \binom{\#}{\#} & 0 & 0 & \lambda\beta & 1-\lambda\beta \\ \binom{\#}{-} & 0 & 0 & \kappa & 1-\kappa \\ \binom{-}{\#} & 0 & 0 & \lambda\beta & 1-\lambda\beta \end{array} \tag{7}$$

To interpret the whole alignment as a Markov chain we note that the number K of links at stationarity has distribution (Thorne, Kishiono, and Felsenstein 1991)

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

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 the system (7) we thus add a new link at time zero with probability γ and start a new round of the Markov chain in (7). We can combine this into a Markov chain on the states in (1) together with an *End* state as follows:

$$\begin{array}{c|cccc} & \binom{\#}{\#} & \binom{\#}{-} & \binom{-}{\#} & \text{End} \\ \hline \binom{\#}{\#} & (1-\lambda\beta)\gamma \mathrm{e}^{-\mu\tau} & (1-\lambda\beta)\gamma(1-\mathrm{e}^{-\mu\tau}) & \lambda\beta & (1-\lambda\beta)(1-\gamma) \\ \binom{\#}{-} & (1-\kappa)\gamma \mathrm{e}^{-\mu\tau} & (1-\kappa)\gamma(1-\mathrm{e}^{-\mu\tau}) & \kappa & (1-\kappa)(1-\gamma) \\ \binom{-}{\#} & (1-\lambda\beta)\gamma \mathrm{e}^{-\mu\tau} & (1-\lambda\beta)\gamma(1-\mathrm{e}^{-\mu\tau}) & \lambda\beta & (1-\lambda\beta)(1-\gamma) \end{array} \tag{8}$$

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

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

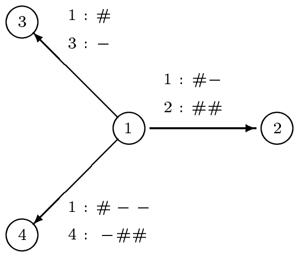
$$\begin{aligned} &b(\#,\#;j) = \lambda\beta(j), \quad b(\#,-;j) = 1 - b(\#,\#;j), \\ &b(-,\#;j) = 1 - \frac{\mu\beta(j)}{1-\exp(-\mu\tau(j))}, \quad b(-,-;j) = 1 - b(-,\#;j), \\ &s(\#;j) = \exp(-\mu\tau(j)), \quad s(-;j) = 1 - s(\#;j), \end{aligned}$$

where

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

2.3 States

Since 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 subsection 2.2 for pairwise alignment to obtain the hidden Markov structure for alignment



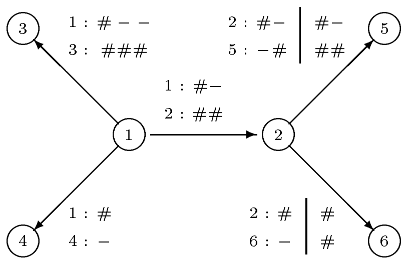
1	#	(#)	(#)	(#)
2	#	(#)	(#)	#
3	–	(–)	(–)	
4	–	#	#	

Figure 2: A simple 3-star tree, where we consider the fate of one mortal link.

on the tree. In order to obtain this we need to choose states for the hidden Markov model on the tree that allows us to identify the states of the hidden Markov model for the pairwise alignments along the edges. To understand how we do this it is best first to look at some examples.

The simplest non-trivial example is a 3-star tree. We consider the fate of one mortal link. As an example consider the situation in figure 2, where the mortal link survives along the edge from root 1 to leaf 2 and produces a new link at leaf 2, but does not survive along the edges from root 1 to leaves 3 and 4. We represent this situation by the states shown on the right in figure 2. The important thing is that from these states we can reconstruct the three pairwise alignments. The rule for writing the states is that we first write the survival and non-survival along the three edges in the first state, then we write the new links at leaf 4, next the new links at leaf 3, and finally the new links at leaf 2. When writing the states for these new links, we remember the situation for those edges left for consideration (this is represented by putting parenthesis around the symbols). We think of this as a “history”. For the 3-star tree we get a total of $2^3 + 2^2 + 2 + 1 = 15$ states.

Next, let us consider the situation in figure 3, where we have a tree with four leaves. Here we have a situation where the link at the root 1 survives at the inner node 2 and produces a new link at node 2. The original link does not survive at any of the leaves 4,5, and 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 leaf 5 and leaf 6 and produces a new link at leaf 5. As before the first state represents the survival or non-survival along the three of the link at the root. We next run through the



1	#	(#)	(#)	(#)	(#)	(#)
2	#	(#)	(#)	(#)	#	(#)
3	#	(#)	#	#		
4	–	(–)				
5	–	#			#	#
6	–				#	

Figure 3: A tree with four leaves, where the link at root 1 survives at inner node 2 and produces a new link at leaf 2.

leaves in reverse order to represent the new links at the leaves. There are no new links at leaf 6 and so the first one to represented is the new link at leaf 5. As before we represent the "history" by symbols in parenthesis. There are no new links at leaf 4, but two new links at leaf 3. Having finished the new links at the leaves we turn to the possible new links at the inner node 2. For each new link at node 2 we first represent the survival or non-survival at the the two leaves 5 and 6 and next the new links at leaf 6 followed by the new links at leaf 5. The number of states for this tree becomes

$$(2^4 + 2^3 + 2^2 + 2^1 + 2^0) + (2^2 + 2^1 + 2^0) + (2^2 + 2^1 + 2^0) = 45,$$

where the first set of numbers represents the case where the link at the root survives at node 2, the second set represents the case where the link does not survive at node 2 (in which case there is nothing at leaves 5 and 6), and the last set represents a new link at node 2 (in which case there is nothing at leaves 3 and 4). All the 45 states are listed as the columns in Table 1. In this table a_i is either # or $-$. The last row gives the number of states of the indicated form, that is, 2^k with k the number of a's in the column. The first 16 states (column 1) each represents a new link at the root 1 that survives at inner node 2. Columns 2 to 5 represent the birth of a new link at one of the leaves originating from the state in column 1. Columns 6 to 8 represent the cases related to a new link at the root 1 that does not survive at inner node 2, and columns 9 to 11 represent the cases related to the birth of a new link at inner node 2. The history, represented by symbols in parenthesis is used to show what transitions are possible from the state, that is, at what

node	column number										
	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

Table 1: States of the Markov chain for the tree in Figure 3. Any of the variables a_i can be either # of $-$. The last row gives the number of states of the indicated form.

positions can a new link be born.

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 that we use. To illustrate this let us write down the probability of the alignments in Figure 3

$$\begin{array}{rccccccc}
s(\#;2) & \times & \times & \times & b(\#,\#;2) & \times & b(\#,-;2) \\
\times s(\#;3) & \times & b(\#,\#;3) & b(\#,\#;3) & b(\#,-;3) & & \\
\times s(-;4) & \times & b(-,-;4) & & & & \\
\times s(-;5) & b(-,\#;5) & b(\#,-) & \times & s(\#,5) & b(\#,\#;5) & b(\#,-;5) \\
s(-;6) & b(-,-;6) & \times & \times & s(\#;6) & b(\#,-;6) &
\end{array} \tag{9}$$

Here each row represents the probability of the alignment along one of the edges. The terms $s(\#;j)$ and $s(-;j)$ are the two entries in the first row of (7) corresponding to survival and non-survival of a link along the edge leading to node j, $b(\#,\#;j)$ and $b(\#,-;j)$ are the two entries from the second and fourth row of (7), and $b(-,\#;j)$ and $b(-,-;j)$ are the two entries from the third row of (7). The first row of (9) gives the probability of the alignment between node 1 and 2, and 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 (9) represents a transistion 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 the root 1. As can be seen each column is a function of the corresponding consequtive set of states in figure 3.

From the examples above we recognize two classes of states. One class corresponds to a new link at an inner node j, together with the fate of this link along the tree, and together with a history consisting of a link at the root and the fate of this link at inner nodes $i < j$ and with the property that the link is present at $a(j)$. The fate of the new link along the tree constitutes a subtree $\tilde{T}$. From a state as described above further new links can be born from nodes of the subtree $\tilde{T}$ or from nodes that are included in the history. Note that there are no leaves in the history because of the convention that new links at the leaves are treated before new links at inner nodes. We denote this class of states by Ξ_1.

The second class of states corresponds to a new link at a leaf k together with a history. The history is defined through a $j \leq a(k)$. It consist of a link at the root and the fate of this link at inner nodes $i < j$ and with the property that the link is present at $a(j)$. Furthermore, the history contains the fate of a link starting at j and containing the leaf k, and where leaves $l > k$ are not included. From this state further new links can be born from the node k or form nodes in the history. We denote the second class of states by Ξ_2.

For the tree in Figure 1 the states $\xi \in \Xi_1$, corresponding to the births of new links at inner nodes, are illustrated in Table 2. The total number of states in $\Xi = \Xi_1 \cup \Xi_2$ is 271 for this tree.

We next introduce some notation for the different states. For a state ξ from the first class we let $t(\xi)$ be the node at which the new link is introduced, and let $T(\xi)$ be the set of inner nodes $j > t(\xi)$ where the link survived or where the link died along the edge to j. Thus $\{t(\xi), T(\xi)\}$ is the subtree describing the fate of the new link among inner nodes. Next we let $L(\xi)$ be the set of leaves at which the new link survived or where the link died along the edge leading to the leaf. Finally, we let $H(\xi)$ be the set of inner nodes $i < t(\xi)$ describing the history of ξ. Formally, ξ is identified by the sets

$$t(\xi),\ T(\xi),\ L(\xi),\ H(\xi),$$

node	column number												
	1	2	3	4	5	6	7	8	9	10	11	12	13
1	#	#	#	#	#	#	(#)	(#)	(#)	(#)	(#)	(#)	(#)
2	−	#	#	#	−	−	(#)	(−)	(#)	(−)	(#)	(−)	#
3	−	−	#	#	#	#	(#)	(#)	#	#	#	#	−
4			−	#	−	#	#	#	−	−	#	#	
5	a_5	a_5	a_5	a_5	a_5	a_5							
6		a_6	a_6	a_6									a_6
7		a_7	a_7	a_7									a_7
8			a_8	a_8	a_8	a_8			a_8	a_8	a_8	a_8	
9				a_9		a_9	a_9	a_9			a_9	a_9	
10				a_{10}		a_{10}	a_{10}	a_{10}			a_{10}	a_{10}	
	2	8	16	64	4	16	4	4	2	2	8	8	4

Table 2: States of the Markov chain corresponding to the birth of a new link at an inner node for the tree in Figure 1. Any of the variables a_i can be either # of $-$. The last row gives the number of states of the indicated form.

and by the value of ξ at the nodes in these sets. A state from the second class is most easily described as a modification of a state from the first class. Thus if $r \in L(\xi)$ we let $\gamma(r, \xi)$ be the state from the second class where we have a new link at the leaf r. The history of this state is then given by the value of ξ at

$$H(\gamma(r, \xi)) = H(\xi) \cup T(\xi) \cup \{j | j \in L(\xi), j < r\}.$$

As an example if we consider the state in the nine'th column of Table 2 we have $t(\xi) = 3$, $T(\xi) = \{3, 4\}$, $L(\xi) = \{8\}$, and $H(\xi) = \{1, 2\}$.

2.4 Transition probabilities

A transition probability $p(x, y)$ of going from the state x to the state y can formally be written 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 $\xi \in \Xi_1$

$$p(\xi, \gamma(r, \xi)) = \left\{ \prod_{j \in L(\xi), j > r} b(\xi(j), -, j) \right\} b(\xi(r), \#, r), \tag{10}$$

and for $s \geq r$

$$\begin{aligned} &P(\gamma(s, \xi), \gamma(r, \xi)) \\ &= \left\{ b(\#, -, s)^{1(s>r)} \prod_{j \in L(\xi), r<j<s} b(\xi(j), -, j) \right\} b(\xi(r), \#, r)^{1(s>r)} b(\#, \#, r)^{1(s=r)}. \end{aligned} \tag{11}$$

For two states $\xi, \eta \in \Xi_1$ a transition from ξ to η is only possible 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

$$\begin{aligned} &t(\eta) \in H(\xi) \cup T(\xi), \\ &H(\eta) = \{ j \in H(\xi) \cup T(\xi) | j < t(\eta) \}, \\ &\eta(j) = \xi(j), \ j \in H(\eta). \end{aligned}$$

In this case the transition probability is

$$\begin{aligned} P(\xi, \eta) = &\left\{ \prod_{j \in L(\xi)} b(\xi(j), -, j) \prod_{j \in T(\xi) \cup H(\xi), j > t(\eta)} b(\xi(j), -, j) \right\} \\ &\times b(\xi(t(\eta)), \#; t(\eta))^{1(t(\eta)>1)} (\lambda/\mu)^{1(t(\eta)=1)} \left\{ \prod_{j \in T(\eta) \cup L(\eta)} s(\eta(j), j) \right\}, \end{aligned} \tag{12}$$

and

$$\begin{aligned} P(\gamma(r, \xi), \eta) = &\left\{ b(\#, -, r) \prod_{j \in L(\xi), j<r} b(\xi(j), -, j) \prod_{j \in T(\xi) \cup H(\xi), j > t(\eta)} b(\xi(j), -, j) \right\} \\ &\times b(\xi(t(\eta)), \#; t(\eta))^{1(t(\eta)>1)} (\lambda/\mu)^{1(t(\eta)=1)} \left\{ \prod_{j \in T(\eta) \cup L(\eta)} s(\eta(j), j) \right\}. \end{aligned} \tag{13}$$

For a transition to the end state the first line of (12) and (13) should be used only, 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 $\xi_0 \in \Xi_1$ with a new link at the root that survives in all of the tree.

3 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 the 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 wellknown: we consider what happens in either the first step or the last step of the Markov chain.

3.1 Approach 1: Annotation of inner nodes and leaves

3.1.1 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 subsection 2.4. 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 $z \in L_1(\xi) = \{u \in L(\xi) | \xi(u) = \#\}$. For a state on the form $\gamma(r, \xi)$ the state emits a letter in the sequence S_r only. For any state $x \in \Xi$ we let

$$l(x) = (l_{d'+1}(x), l_{d'+2}(x), \ldots, l_{d'+d}(x)) \tag{14}$$

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:

$$l_j(x) = \begin{cases} 1 & \text{if } x \text{ emits a letter in sequence } S_j \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

For the state x_i in the hidden Markov chain we use the short hand 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 $p_e(s|x)$.

3.1.2 Backward recursion

Summing over the states of the Markov chain we have

$$P(S(1:L)|x_0 = I) = \sum_{n=0}^{\infty} \sum_{x_1,\ldots,x_n \in \Xi : L^n = L} p(x_n, E) \prod_{i=1}^{n} p(x_{i-1}, x_i) p_e(S[L^i, l^i]|x_i). \tag{16}$$

We will obtain a recursion by separating out the contribution from the first term x_1. Define for an arbitrary vector $K \geq 0$ and state $x_0 \in \Xi$

$$\begin{aligned} F(K|x_0) &= \sum_{n=0}^{\infty} \sum_{x_1,\ldots,x_n \in \Xi : K+L^n = L} p(x_n, E) \prod_{i=1}^{n} p(x_{i-1}, x_i) p_e(S[K+L^i, l^i]|x_i) \\ &= P(S(K+1:L)|x_0), \end{aligned}$$

that is, the probability that the sequences $S((K+1):L)$ are produced by the states $x_1, x_2, \ldots$ given that the Markov chain starts in the state x_0. The inner sum is zero if there are no $x_1, \ldots, x_n$ with $K + L^n = L$. Clearly, $P(S(1:L)|x_0 = I) = F(0|I)$. When

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

$$\begin{aligned} F(K|x_0) &= \sum_{n=1}^{\infty} \sum_{x_1 \in \Xi} p(x_0, x_1) p_e(S[K + l^1, l^1]|x_1) \sum_{x_2,\ldots,x_n \in \Xi : (K + l(x_1)) + \tilde{L}^n = L} p(x_n, E) \\ &\quad \times \prod_{i=2}^{n} p(x_{i-1}, x_i) p_e(S[K + l^1 + \tilde{L}^i, l^i]|x_i) \\ &= \sum_{z \in \Xi} p(x_0, z) p_e(S[K + l(z), l(z)]|z) F(K + l(z)|z). \end{aligned} \tag{17}$$

When $K = L$ the recursion is

$$F(L|x_0) = p(x_0, E) + \sum_{z \in \Xi : l(z) = 0} p(x_0, z) F(L|z). \tag{18}$$

The recursion (17) states that the probability of the sequences $S((K+1) : L)$ produced by the states $x_1, x_2, \ldots$ given that we start in the 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 the recursion (17) 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.

3.1.3 Forward recursion

In this subsection we obtain a recursion by separating out the contribution from x_n in (16). Define for an arbitrary vector $K \geq 0$ and state $x \in \Xi$

$$H(K, x) = \sum_{n=0}^{\infty} \sum_{x_1,\ldots,x_n \in \Xi : L^n = K} \left(\prod_{i=1}^{n} p(x_{i-1}, x_i) p_e(S[L^i, l^i]|x_i) \right) p(x_n, x), \tag{19}$$

where $x_0 = I$ and the inner sum is zero if there are no $x_1, \ldots, x_n$ with $L^n = K$. Defining $H(L, E)$ as above with x replaced by the end state E we have $P(S(1 : L)|x_0 = I) = H(L, E)$. We will use the convention $H(K, x) = 0$ if some of the coordinates of K are less than zero.

Splitting the sum in (19) we obtain for $K \overset{w}{>} 0$ and $K \geq 0$ the recursion

$$H(K, x) = \sum_{z \in \Xi} H(K - l(z), z) p_e(S[K, l(z)]|z) p(z, x). \tag{20}$$

When $K = 0$ the recursion becomes

$$H(0, x) = p(x_0, x) + \sum_{z \in \Xi : l(z)=0} H(0, z)p(z, x). \tag{21}$$

Computationally there is no difference between the use of $H(K, x)$ or $F(K|x_0)$. However, the latter has an interpretation as a probability thereby making it easier to understand the recursion.

3.1.4 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, it is of the form $\gamma(r, \xi)$, the emitted vector s has a letter at the node r only, and the emission probability is simply the stationary probability $\pi(s(r))$. 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

$$T_1(\xi) = \{z \in T(\xi)|\xi(z) = \#\}, \quad L_1(\xi) = \{z \in L(\xi)|\xi(z) = \#\}, \tag{22}$$

we can write the emission probability as

$$p(s|\xi) \;=\; \sum_{v_{t(\xi)}} \pi(v_{t(\xi)}) \sum_{v_z, z \in T_1(\xi)} \prod_{z \in T_1(\xi)} f(v_z|v_{a(z)};\tau(z)) \prod_{z \in L_1(\xi)} f(s_z|v_{a(z)};\tau(z)). \tag{23}$$

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.

3.1.5 Implementation and analysis

Let us briefly discuss how to implement the recursion given by (17) and (18). 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 $\tilde{K} \overset{w}{>} K$, $\tilde{K} \geq K$, and all $x \in \Xi$. For each $x \in C$ the recursion (17) gives

$$F(K|x) = \sum_{z \in C} p(x,z) F(K|z) + \omega(x) \tag{24}$$

with $\omega(x)$ known. Let Q be the matrix with entries $p(z_1, z_2)$, $z_1, z_2 \in C$. Then since the entries are nonnegative and the sum along a row is less than 1 the matrix $I_C - Q$, where I_C is the identity matrix, is invertible, and the set of linear equations (24) has a unique solution. Having solved this system of equations we can next calculate $F(K|x)$ for $x \notin C$ directly from (17), or from (18) 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 $K \overset{w}{>} L$.

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 illustrtion purposes we have implemented the recursion (**??**) as well as the Viterbi algortihm and an algortihm for simulating alignments conditioned on the observed sequences for the case of four observed sequences. No attempt to optimize the program has been made and the program therefore only runs on short sequences. As an example we have taken the first part of the sequences in Example 1 in Bucka-Larsen, Caprani and Hein (1999). The data and the alignment with the highest probability are given in Table 3. The alignment has been obtained on choosing the parameter values $\mu =$, $\lambda =$, $\theta =$(need explanation), and $\tau(j) = 1$ for all j.

The alignment in Table 3 agrees with the one shown in Bucka-Larsen, Caprani and Hein (1999) (based on aligning 8 sequences) for the two first columns, but not for the third column. It is interesting that when simulating the alignment 500 times we do in no cases see the third column of Bucka-Larsen, Caprani and Hein (1999).

C	$-$	C	A	A	G	T	G	C	T	$-$	A	G	$-$	A	T	C	G	$-$	T
C	$-$	T	G	C	G	$-$	G	A	T	$-$	A	C	$-$	A	T	A	A	$-$	C
C	A	T	G	A	G	A	T	T	G	$-$	A	C	T	A	T	T	A	C	C
C	A	C	T	C	T	A	G	T	C	C	A	C	A	A	A	C	G	A	C

Table 3: Four sequences and their maximal alignment.

3.2 Approach 2: Annotation of inner nodes only

3.2.1 Notation

In section 3.1 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 less terms than in section 3.1.

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 describing the fate of the link, and the set $H(\xi)$ giving the history of the new link. For the tree in Figure 1 the states are now given in the upper part of Table 2. As before $L(\xi)$ is the set of leaves descending from $t(\xi)$ or from the nodes in $T_1(\xi)$ (see (22)). As in (14) $l(x)$ denotes the lenghts of the emitted subsequences. Contrary to (15) $l(x)$ is no longer determined by the state x, the state only determines at which nodes it is possible to emit letters:

$$l_j(x) = \begin{cases} \geq 0 & j \in L(x) \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

We again use $l^i = l(x_i)$ and 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 $q_e(S(u:v)|x)$ is now both the probability of emitting subsequences of length $v_j - u_j$, $j = d'+1, \ldots, d'+d$, and the probability that the emitted letters are $S_j(u_j : v_j)$. To state this probability we define the set

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

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. Then the probability that the state ξ emits the subsequences $S(m+1 : m+l)$ is

$$\begin{aligned}
&q_e(S(m+1 : m+l)|\xi) \qquad (26)\\
&= \left(\prod_{z \in L(\xi)\setminus L_1(\xi,l)} s(-;z)b(-,-;z)\right)\left(\prod_{z \in L_1(\xi,l)} b(\#,\#;z)^{l_z-1}b(\#,-;z)\right)\\
&\quad \times \sum_{u \subseteq L_1(\xi,l)} f(m,u,\xi)\left(\prod_{z\in u} s(\#;z)\pi(S_z(m_z+2 : m_z+l_z))\right)\\
&\quad \times \left(\prod_{z \in L_1(\xi,l)\setminus u} s(-;z)b(-,\#;z)\pi(S_z(m_z+1 : m_z+l_z))\right)
\end{aligned}$$

where $\pi(S_j(a:b)) = \prod_{i=a}^{b} \pi(S_j(i))$. The function $f(m,u,\xi)$ is the emission probablity for the first letter at those leaves where we have survival of the link and is given by $p_e(s|\xi)$ from (23) with $L_1(\xi)$ replaced by u and with s_z replaced by $S_z(m_z+1)$. Furthermore,

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

3.2.2 Backward recursion

Instead of (16) we write

$$\begin{aligned}
&P(S(1:L)|x_0 = I) \qquad (27)\\
&= \sum_{n=0}^{\infty} \sum_{x_1,\ldots,x_n \in \Xi} \sum_{l^0,l^1,\ldots,l^n : L^n = L} q_e(S(1:l^0)|I)\left(\prod_{i=1}^{n} p(x_{i-1},x_i)q_e(S[L^i,l^i]|x_i)\right) p(x_n,E).
\end{aligned}$$

The backward recursion is obtained by defining

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

and separating out the sum over the first state x_1 we find

$$\begin{aligned}
&F(K|x_0) \qquad (28)\\
&= q_e(S(K+1:L)|x_0)p(x_0,E) + \sum_{\xi\in\Xi}\sum_{l(\xi)} p(x_0,\xi) q_e(S[K+l(\xi),l(\xi)]|\xi) F(K+l(\xi)|\xi)
\end{aligned}$$

for $K \overset{w}{<} L$ and $K \leq L$, and when $K = L$ the recursion is

$$F(L|x_0) = p(x_0,E) + \sum_{\xi\in\Xi} p(x_0,\xi) P(l(\xi)=0|\xi) F(L|\xi). \qquad (29)$$

The *forward* recursion is derived in exactly the same way, now with $H(k,x)$ form (19) defined as

$$\begin{aligned}
&H(K,x) \qquad (30)\\
&= \sum_{n=0}^{\infty} \sum_{x_1,\ldots,x_n\in\Xi} \sum_{l^0,l^1,\ldots,l^n:L^n=K} q_e(S(1:l^0)|I) \left(\prod_{i=1}^{n} p(x_{i-1},x_i) q_e(S[L^i,l^i]|x_i)\right) p(x_n,x).
\end{aligned}$$

The forward recursion was presented in Hein (2001), where $H(K,x)$ was erroneously referred to as a probability.

3.2.3 Reduction of complexity

For the recursions described in the previous subsection we need to calculate either $H(K,x)$ or $F(K|x)$ for any value of $K \leq L$. This takes of the order $O(L_{\max}^d)$ steps. Each step here, however, involves the sum over l, see (28), and therefore requires of the order $O(L_{\max}^d)$ steps. The complexity of the algorithms are therefore of the order $O(L_{\max}^{2d})$ steps. The algorithms are therefore inferior to the algorithms given in Section 3.1. It turns out, though, that we can rewrite the algorithms in such a way that the resulting complexity is $O(L_{\max}^d)$ and that the constant factor is actually slightly smaller here than for the algorithms of Section 3.1.

We start by inserting (26) into the recursion (28). To shorten the formula we define $q(\#; z) = s(-; z)b(-, \#; z)$ and $q(-; z) = s(-; z)b(-, -; z)$. We then find

$$\begin{aligned}
&F(K|x) \qquad\qquad (31)\\
&= q_e(S(K+1:L)|x)p(x,E)\\
&\quad + \sum_{\xi\in\Xi} p(x,\xi) \sum_{l_z\geq 0, z\in L(\xi);\, l_z=0, z\notin L(\xi)} \left(\prod_{z\in L(\xi)\setminus L_1(\xi,l)} q(-;z)\right)\left(\prod_{z\in L_1(\xi,l)} b(\#,\#;z)^{l_z-1}\right)\\
&\quad \times \sum_{u\subseteq L_1(\xi,l)} f(K,u,\xi)\left(\prod_{z\in u} s(\#;z)\pi(S_z(K_z+2:K_z+l_z))\right)\\
&\quad \times \left(\prod_{z\in L_1(\xi,l)\setminus u} q(\#;z)\pi(S_z(K_z+1:K_z+l_z))\right) F(K+l|\xi)
\end{aligned}$$

where, as before, u is the subset of the leaves $L_1(\xi, l)$ at which we have survival from the ancestral inner node. Next, we let w be the subset of the leaves $L(\xi) \setminus u$ at which there is not survival, but the number of new links is positive. Then (31) becomes

$$\begin{aligned}
&F(K|x) \qquad\qquad (32)\\
&= q_e(S(K+1:L)|x)p(x,E)\\
&\quad + \sum_{\xi\in\Xi} p(x,\xi) \sum_{u\subseteq L(\xi)} f(K,u,\xi) \sum_{w\subseteq L(\xi)\setminus u} \left(\prod_{z\in L(\xi)\setminus(u\cup w)} q(-;z)\right)\\
&\quad \times \sum_{l_z\geq 1, z\in(u\cup w);\, l_z=0, z\notin(u\cup w)} \left(\prod_{z\in u\cup w} b(\#,\#;z)^{l_z-1}\right)\left(\prod_{z\in u} s(\#;z)\pi(S_z(K_z+2:K_z+l_z))\right)\\
&\quad \times \left(\prod_{z\in w} q(\#;z)\pi(S_z(K_z+1:K_z+l_z))\right) F(K+l|\xi)
\end{aligned}$$

Finally, we introduce the subset v of $u \cup w$ at which $l_z \geq 2$. This gives

$$
\begin{aligned}
F(K|x) &= q_e(S(K+1:L)|x)p(x,E) \\
&\quad + \sum_{\xi\in\Xi} p(x,\xi) \sum_{u\subseteq L(\xi)} f(K,u,\xi) \sum_{w\subseteq L(\xi)\setminus u} \left(\prod_{z\in L(\xi)\setminus(u\cup w)} q(-;z) \right) \\
&\quad \times \sum_{v\subseteq(u\cup w)} \left(\prod_{z\in u} s(\#;z) \right) \left(\prod_{z\in w} q(\#;z)\pi(S_z(K_z+1)) \right) \\
&\quad \times \sum_{l_z\geq 2, z\in v;\, l_z=1, z\in(u\cup w)\setminus v;\, l_z=0, z\notin(u\cup w)} F(K+l|\xi) \\
&\quad \times \left(\prod_{z\in v} b(\#,\#;z)^{l_z-1} \pi(S_z(K_z+2:K_z+l_z)) \right) \\
&= q_e(S(K+1:L|x)p(x,E) \\
&\quad + \sum_{\xi\in\Xi} p(x,\xi) \sum_{u\subseteq L(\xi)} f(K,u,\xi) \sum_{w\subseteq L(\xi)\setminus u} \left(\prod_{z\in L(\xi)\setminus(u\cup w)} q(-;z) \right) \\
&\quad \times \sum_{v\subseteq(u\cup w)} \left(\prod_{z\in u} s(\#;z) \right) \left(\prod_{z\in w} q(\#;z)\pi(S_z(K_z+1)) \right) \\
&\quad \times \left(\prod_{z\in v} b(\#,\#;z) \right) \sum_{\tilde{l}_z\geq 1, z\in v;\, \tilde{l}_z=0, z\notin v} F(K+1(u\cup w)+\tilde{l}|\xi) \\
&\quad \times \left(\prod_{z\in v} b(\#,\#;z)^{\tilde{l}_z-1} \pi(S_z((K+1(u\cup w))_z+1:(K+1(u\cup w))_z+\tilde{l}_z)) \right),
\end{aligned} \tag{33}
$$

where

$$
1(u\cup w)_z = \begin{cases} 1 & z\in(u\cup w) \\ 0 & z\notin(u\cup w) \end{cases}
$$

Let us denote the last sum in (33) by G, that is,

$$
G(M|\xi,v) = \sum_{m_z\geq 1, z\in v;\, m_z=0, z\notin v} F(M+m|\xi) \prod_{z\in v} b(\#,\#;z)^{m_z-1} \pi(S_z(M_z+1:M_z+m_z)). \tag{34}
$$

for a nonempty subset v of $L(\xi)$, and

$$
G(M|\xi,\emptyset) = F(M|\xi).
$$

Thus (33) becomes

$$
\begin{aligned}
F(K|x) &= q_e(S(K+1:L)|x)p(x,E) &(35)\\
&+\sum_{\xi\in\Xi} p(x,\xi) \sum_{u\subseteq L(\xi)} f(K,u,\xi) \sum_{w\subseteq L(\xi)\setminus u}\left(\prod_{z\in L(\xi)\setminus(u\cup w)} q(-;z)\right)\\
&\times \sum_{v\subseteq(u\cup w)}\left(\prod_{z\in u} s(\#;z)\right)\left(\prod_{z\in w} q(\#;z)\pi(S_z(K_z+1))\right)\\
&\times\left(\prod_{z\in v} b(\#,\#;z)\right) G(K+1(u\cup w)|\xi,v).
\end{aligned}
$$

We can obtain a recursion for G by splitting the sum in (34) into

$$\sum_{\tilde{v}\subseteq v}\ \sum_{m_z\geq 2, z\in\tilde{v};\, m_z=1, z\in(v\setminus\tilde{v});\, m_z=0, z\notin v}$$

where $\tilde{v}$ can be the empty subset. This gives

$$
\begin{aligned}
&G(K|\xi,v) &(36)\\
&= \prod_{z\in v}\pi(S_z(M_z+1))\sum_{\tilde{v}\subseteq v}\prod_{z\in\tilde{v}} b(\#,\#;z) \sum_{m_z\geq 2, z\in\tilde{v};\, m_z=1, z\in(v\setminus\tilde{v});\, m_z=0, z\notin v} G(M+1(v)|\xi,\tilde{v}).
\end{aligned}
$$

Combining (35) and (36) we have established a recursion involving the functions $F(K|\xi)$ and $G(K|\xi,v)$. For the tree in Figure 1 the recursions of section 3 involves 271 terms whereas the number of terms in this section is 142. For the tree in Figure 3 the numbers are 45 and 24, respectively.

3.2.4 Implementation and analysis

For the recursion in (36) with $v\neq\emptyset$ there is no problem with self reference. For the recursion in (35) the self reference problem is handled as in subsection 3.1.5. Imagine that $F(\tilde{K}|x)$ and $G(\tilde{K}|x,v)$ have been found for all $\tilde{K}\overset{w}{>}K$, $\tilde{K}\geq K$, for all x and for all v. Then (35) takes the form

$$F(K|x)=\sum_{\xi\in\Xi} p(x,\xi)\left(\prod_{z\in L(\xi)} q(-;z)\right)F(K|\xi)+\omega(x) \tag{37}$$

where $\omega(x)$ is known. Defining Q to be the matrix with entries

$$p(x_1,x_2)\left(\prod_{z\in A_{x_2}} q(-;z)\right)$$

for $x_1, x_2 \in \Xi$ we have that $I_\Xi - Q$ is invertible since the entries of Q are nonnegative and the sum along a row is less than 1. Thus the set of linear equations (37) has a unique solution.

The boundary conditions for the recursions are $F(K|x) = 0$ and $G(K|\xi, v) = 0$ for $K \overset{w}{>} L$.

4 Discussion

This paper presents an algorithm that has the same complexity as the traditional nonstatistical multiple alignment algorithm (Sankoff, 1975). 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, not in obtaining an alignment. Among molecular biologists it is, however, 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 paper, 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 analyse up to, say, four sequence of realistic lengths (hundreds of base pairs/amino acids). Elementary computational tricks can extend this to six-seven sequences, and beyond this radically different methods will have to be applied. In Jensen and Hein (2002) a simulation technique is suggested where the basic step is the simulation of the alignment of a 3-star tree.

From the perspective of a biologist the underlying model for this paper can be criticized. Firstly, the assumption that all insertions-deletions are only one nucleotide/amino acid long does not conform to the biological reality and should be relaxed. Secondly, 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.

References

Hein, J. (2001). An algorithm for statistical alignment of sequences related by a binary tree. In Altman, R.B., Dunker, A.K., Hunter, L., Lauderdale, K. and Klein, T.E. (eds). Pacific Symposium on Biocomputing. World Scientific. Singapore. pp. 179-190.

Needleman, S.B. and Wunsch, C.D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* **48**, 443-53.

Sankoff, D. (1975). Minimal mutation trees of sequences. *SIAM J. Appl. Math.* **78**, 35-42.

Steel, M. and Hein, J. (2000). Applying the Thorne-Kishino-Felsenstein model to sequence evolution of a star tree. *Appl. Math. Lett.* **14**, 679-84.

Thorne, J.L., Kishino, H. and Felsenstein, J. (1991). An evolutionary model for maximum likelihood alignment of DNA sequences. *J. Mol. Evol.* **33**, 114-124.