Statistica Sinica **15**(2005), 889-907

GIBBS SAMPLER FOR STATISTICAL MULTIPLE ALIGNMENT

Jens Ledet Jensen and Jotun Hein

University of Aarhus and Oxford University

Abstract: For a set of sequences, related by a binary tree, that have evolved according to the Thorne-Kishino-Felsenstein model, a Gibbs sampler is presented for simulating the ancestral sequences and their alignments. The updating step consists in updating the ancestral sequence and its three alignments within a 3-star tree. We compare the Gibbs sampler with the algorithm suggested recently by Holmes and Bruno.

Key words and phrases: 3-star tree, EM-algorithm, evolutionary model, hidden Markov model, Markov chain Monte Carlo.

1. Introduction

In this paper we describe a Markov Chain Monte Carlo method for sampling multiple sequence alignments within the Thorne, Kishino and Felsenstein (1991) model (*TKF-model*) on a binary tree. We use a Gibbs sampler where in each step an ancestral sequence and its alignments with the three neighbours is simulated conditional on the three neighbouring sequences. While developing the method described in this paper the article by Holmes and Bruno (2001) appeared. In that paper a Gibbs sampler is given that is computationally simpler than the one described in this paper. In each step Holmes and Bruno (2001) either update the alignment between two sequences given the two sequences, or update the ancestral sequence given its alignments with the three neighbours while allowing the insertion of new letters in the ancestral sequence that are not aligned to any of the letters in the three neighbours (see Appendix A below). We compare our method with that of Holmes and Bruno (2001) in terms of mixing properties and efficiency.

The aim of the present method is to use a simple stochastic model to obtain a distribution of alignments and a likelihood function. The traditional optimization-based methods rely on a score function and a heuristic algorithm. These methods have been improved over decades, but are limited in ignoring the stochastic nature of the evolutionary process. In Section 5 we consider a data example and compare the output from the CLUSTAL W program (www.ebi.ac.uk/clustalw) with the information that can be obtained from the method in this paper.

A forerunner to the present statistical alignment is Zhu, Liu and Lawrence (1998), where a posterior distribution on the set of alignments of input sequences is produced. Their method gives a uniform prior distribution on alignments with k (possibly longer than one nucleotide/amino acid) insertion-deletions, and then uses a PAM score matrix to give a likelihood of each alignment within this set. Their method is restricted to a pair of sequences, uses an arbitrary (non-evolutionary) prior on alignments, and finally includes an evolutionary modelling for the substitution process only, not for the insertion-deletion process.

1.1. TKF-model

In the TKF-model (Thorne, Kishino and Felsenstein (1991)), each letter in a sequence develops independently of the other letters according to a birth and death process with birth rate λ and death rate $\mu > \lambda$. When a new letter is born it is inserted to the right of the letter giving birth. The new letter is chosen according to a distribution π. At the very left end of the sequence is a birth process with rate λ (immigration) so that the sequence will not eventually die out. While a letter is alive it is subject to a Markovian substitution process with stationary probabilities given by π, and with the transition probability of a change from w_1 to w_2 within a time span τ given by $f(w_2|w_1;\tau)$. The stationary distribution of a sequence S of length L is given by

$$P(S) = (1-\gamma)\gamma^L \prod_{i=1}^{L} \pi(S[i]), \quad \gamma = \frac{\lambda}{\mu}, \tag{1}$$

where $S[i]$ is the ith element in the sequence.

If a sequence S_1 evolves into S_2 during a time span τ we can summarise the evolution in terms of the alignment of some of the letters in S_1 with some of the letters in S_2 (survival of these letters), in terms of deletions (deaths) of some of the letters, in terms of insertions (births), and finally in terms of substitutions for the aligned letters. The TKF-model for this summary information can be reformulated as a hidden Markov model (Durbin, Eddy, Krogh and Mitchison (1998) and Hein (2001)). The three basic states for the underlying Markov chain are match (M), deletion (D), and insertion (I), where M is the alignment of a letter in sequence one with a letter in sequence two, D is a letter in sequence one that did not survive, and I is the birth of a letter in sequence two. Apart from the three states above there is also an *end state* ($\mathcal{E}$) in order to model the random lengths of the sequences. We use the symbol # to denote the presence of a letter (nucleotide or amino acid) and the symbol $-$ to denote the absence of a letter. To give the transition probabilities in the Markov chain, we define for notational convenience

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

and introduce the terms

$$\begin{aligned} & b(\#,\#) = \gamma\beta, \quad b(\#,-) = 1 - b(\#,\#), \\ & b(-,\#) = 1 - \frac{\beta}{1-\exp(-\mu\tau)}, \quad b(-,-) = 1 - b(-,\#), \\ & s(\#) = \exp(-\mu\tau), \quad s(-) = 1 - s(\#), \end{aligned} \tag{2}$$

where $b(\cdot,\#)$ has the interpretation of the probability of a birth, $b(\cdot,-)$ is the probability of no birth, and $s(\#)$ is the probability of survival. The transition probabilities are then

	M	D	I	$\mathcal{E}$
M	$b(\#,-)\gamma s(\#)$	$b(\#,-)\gamma s(-)$	$b(\#,\#)$	$b(\#,-)(1-\gamma)$
D	$b(-,-)\gamma s(\#)$	$b(-,-)\gamma s(-)$	$b(-,\#)$	$b(-,-)(1-\gamma)$
I	$b(\#,-)\gamma s(\#)$	$b(\#,-)\gamma s(-)$	$b(\#,\#)$	$b(\#,-)(1-\gamma)$

(3)

The state M emits a letter in both sequences and the distribution of the two letters (w_1, w_2) is $\pi(w_1)f(w_2|w_1;\tau)$. The deletion state D emits a letter in sequence one only, and the insertion state I emits a letter in seqeunce two only, in both cases the distribution of the letter is $\pi(\cdot)$.

The immigration part of the model is incorporated by saying that the Markov chain starts in the state $\mathcal{I}$ that has the same transition probabilities as the match state M, and the initial state $\mathcal{I}$ does not emit any letters (also called the immortal state).

In Hein, Jensen and Pedersen (2003) a detailed description is given of how to formulate the TKF-model on a binary tree as a hidden Markov model. We use this below for the special case of a 3-star tree.

1.2. Notation and Gibbs idea

We have η observed sequences $S_1, \ldots, S_\eta$, one for each leaf of a binary tree with $\nu = \eta - 2$ interior nodes. The unobserved sequences at the inner nodes are denoted $T_{\eta+1}, \ldots, T_{\eta+\nu}$. The root of the tree is taken as the interior node numbered $\eta + \nu$. Any interior node $\eta + 1 \leq i < \eta + \nu$ has an ancestor $a(i)$ among the interior nodes $i+1, \ldots, \eta+\nu$ and two descendants $d_1(i)$ and $d_2(i)$ among the interior nodes $\eta+1, \ldots, \eta+i-1$ and the leaves. For the root the ancestor $a(\eta+\nu)$ is replaced by a descendant; for a leaf j the ancestor $a(j)$ is among the interior nodes. An example of a tree with 4 observed sequences is given in Figure 1, and an example with 7 observed sequences is given in Figure 2.

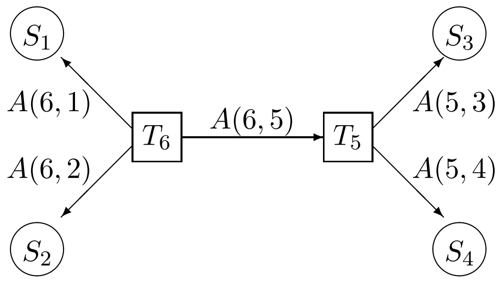

Figure 1. A tree with four observed sequences.

The branch from the node $a(j)$ to the node j is numbered j so that the set of branches is $j = 1, \ldots, \eta + \nu - 1$. Branch number j has a length τ_j and an alignment $A(a(j), j)$ consisting of a sequence with terms M, D, and I.

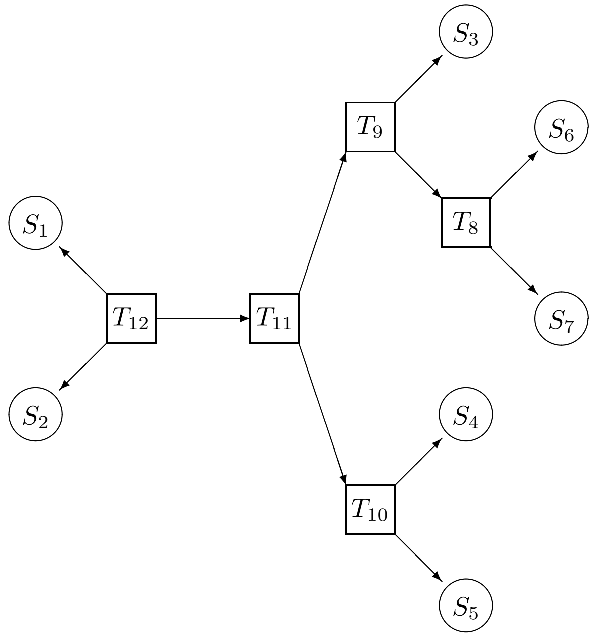

Figure 2. A tree with seven observed sequences.

The TKF-model gives the joint probability of all the sequences $(T_{\eta+1}, \ldots, T_{\eta+\nu})$, $(S_1, \ldots, S_\eta)$, and all the alignments $A(a(j), j)$. In this paper we consider simulation of $T_{\eta+1}, \ldots, T_{\eta+\nu}$ and the alignments $A(a(j), j)$, conditionally on the value of the observed sequences $(S_1, \ldots, S_\eta)$. Each step in the simulation consists in simulating a 3-star tree conditionally on the sequences at the three leaves. Thus for each $r = \eta+1, \ldots, \eta+\nu$, we consider the 3-star with interior node r and leaves $a(r), d_1(r), d_2(r)$, and simulate a new value of the sequence T_r and the alignments

$A(r, a(r))$, $A(r, d_1(r))$ and $A(r, d_2(r))$ from the conditional distribution given the sequences at the three leaves. This conditional distribution is given in (11) below. For the tree in Figure 1 we simulate the 3-star tree with node 5 being the interior node and next the 3-star tree with node 6 being the interior node.

To get initial values of the interior sequences $T_{\eta+1}, \ldots, T_{\eta+\nu}$ we use an algorithm for simulating a 2-star tree equivalent to the one described below for a 3-star tree. In particular we have a formula equivalent to (11) for the sequential simulation of an interior sequence. For $r = \eta + 1, \ldots, \eta + \nu$, we simulate T_r given the sequences at $d_1(r)$ and $d_2(r)$.

2. 3-star Tree

2.1. States and transition probabilities

We consider a 3-star tree where we let T be the sequence at the interior node and let S_1, S_2, S_3 be the sequences at the three leaves. The evolutionary time distances along the branches are τ_1, τ_2, τ_3. In this section we describe the TKF-model for the 3-star tree as a hidden Markov chain. There are two sets of states. The first set of states correspond to having a letter in the ancestral sequence T, and recording the set J of leaves at which the letter survives. Thus $j \in J$ means that the letter survided at this leaf, and $j \notin J$ means that the letter did not survive at this leaf. We denote such a state by $M(J)$, where $J = \emptyset$ is also a possibility. The second set of states correspond to births (insertions) at a subset J of the three leaves, and we denote such a state by $I(J)$, where $J \neq \emptyset$. As in the case with two sequences there is also an end state $\mathcal{E}$. From a state $M(J)$ we can go to any other state. From a state $I(J_1)$ we can go to any other state $M(J_2)$, but only to states $I(J_2)$ with $J_2 \subseteq J_1$. The set of 15 states of the form $M(J)$ or $I(J)$ is denoted Ξ in the following.

To state the transition probabilities we define $\beta(j)$, $b(\#, \#; j)$, $b(\#, -; j)$, $b(-, \#; j)$, $b(-, -; j)$, $s(\#; j)$, and $s(-; j)$ as in (2), with τ replaced by τ_j, $j = 1, 2, 3$. Furthermore, for a state x of the form $M(J)$ or $I(J)$ we define for $j = 1, 2, 3$, $x_j = \#$ if $j \in J$ and $x_j = -$ if $j \notin J$. The transition probability $p(x, y)$ of going from the state x to the state y is then

	$y = M(J_2)$	$y = I(J_2)$	$y = \mathcal{E}$
$x = M(J_1)$	$B(\#, \#)\gamma \left(\prod_{j=1}^{3} s(y_j; j)\right)$	$B(\#, -)$	$B(\#, \#)(1 - \gamma)$
$x = I(J_1)$	$B(-, \#)\gamma \left(\prod_{j=1}^{3} s(y_j; j)\right)$	$B(-, -)$	$B(-, \#)(1 - \gamma)$

(4)

with

$$\begin{aligned} B(\#,\#) &= \textstyle\prod_{j=1}^{3} b(x_j,-;j), \quad B(\#,-) = \prod_{j=1}^{3} b(x_j,y_j;j),\\ B(-,\#) &= \textstyle\prod_{\{j\in J_1\}} b(\#,-;j), \quad B(-,-) = \prod_{\{j\in J_1\}} b(\#,y_j;j). \end{aligned}$$

The initial state $\mathcal{I}$ corresponds to $M(\{1,2,3\})$ and does not emit any letters.

A state $M(J)$ emits a letter w_0 at the interior node and emits letters w_j at the leafs $j \in J$. A state $I(J)$ emits letters w_j at the leafs $j \in J$ only. The emission probabilities are

$$\begin{aligned} p_e^0(w|M(J)) &= \pi(w_0)\textstyle\prod_{\{j\in J\}} f(w_j|w_0;\tau_j),\\ p_e^0(w|I(J)) &= \textstyle\prod_{\{j\in J\}} \pi(w_j). \end{aligned} \tag{5}$$

We will also be using the marginal probability of (w_1,w_2,w_3) given the state, obtained by summing over w_0 in the previous expression,

$$\begin{aligned} p_e(w|M(J)) &= \textstyle\sum_{w_0} \pi(w_0)\prod_{\{j\in J\}} f(w_j|w_0;\tau_j),\\ p_e(w|I(J)) &= \textstyle\prod_{\{j\in J\}} \pi(w_j). \end{aligned} \tag{6}$$

2.2. Simulating a 3-star tree

We denote the length of the sequence S_j by L_j, $j = 1,2,3$. A subsequence starting in a and ending in b is denoted $S_j[a:b]$. If $a > b$ we interpret $S_j[a:b]$ as the empty set. For column vectors u and v with integer entries we let $S[u:v]$ denote the three subsequences $S_j[u_j:v_j]$, $j = 1,2,3$. For a state $x = M(J)$ we define $t(x) = 1$ and a 3-dimensional vector $l(x)$ with 1 at coordinates $j \in J$ and 0 at the remaining coordinates. For a state $x = I(J)$ we define $t(x) = 0$ and $l(x)$ as above.

The multiple alignment for a 3-star tree is given through the states $x^0,\ldots,x^N$, where $x^0 = \mathcal{I}$ is the initial state that does not emit any letters, $x^i \in \Xi$, $i = 1,\ldots,N$, and x^{N+1} is the end state. Here N is random. Let

$$L^i = l(x^1) + \cdots + l(x^i), \quad t^i = t(x^1) + \cdots + t(x^i).$$

Thus L^i is the part of the sequences in S explained by the first i states of the alignment. We can write the joint probability of the sequences and the alignment as

$$\begin{aligned} &P(N=n, x^1,\ldots,x^n,T,S)\\ &\quad = p(x^n,\mathcal{E})\prod_{i=1}^{n} p(x^{i-1},x^i)p_e^0\left(T[t^{i-1}+1:t^i], S[L^{i-1}+1:L^i]\,\middle|x^i\right), \end{aligned} \tag{7}$$

where n and $x^1, \dots, x^n$ are such that $L^n = l(x^1) + \cdots + l(x^n) = L$, with L being the vector of lengths of the sequences. Note, that if we sum this expression over the possible letters of the ancestral sequence T the term p_e^0 is replaced by the term $p_e(S[L^{i-1}+1 : L^i]|x^i)$.

To obtain the marginal probability of a part of the alignment we introduce the function $F(K|x^0)$, where K is a column vector of integers and x^0 is any state among Ξ, given by

$$\begin{aligned} &F(K|x^0) \\ &= \sum_{n=0}^{\infty} \sum_{x^1,\dots,x^n \in \Xi : K+L^n = L} p(x^n, \mathcal{E}) \prod_{i=1}^{n} p(x^{i-1}, x^i) p_e(S[K+L^{i-1}+1 : K+L^i]|x^i), \end{aligned} \tag{8}$$

where the inner sum is zero if there is no $x^1, \dots, x^n$ with $K + L^n = L$. In particular, $F(K|x) = 0$ if there exists j with $K_j > L_j$. This function gives the marginal probability of the sequences $S[K+1 : L]$ given that the initial state is x^0. In particular the marginal probability of the sequences at the three leaves is

$$P(S) = F(0|\mathcal{I}). \tag{9}$$

From (7) and (8) we find the following marginal probability

$$\begin{aligned} &P\left(N \geq k, x^1, \dots, x^k, T[1 : t^k], S\right) \\ &\quad = \Big\{ \prod_{i=1}^{k} p(x^{i-1}, x^i) p_e^0 \left(T[t^{i-1}+1 : t^i], S[L^{i-1}+1 : L^i] \,\middle|\, x^i\right) \Big\} F(L^k|x^k), \end{aligned}$$

and using (9) we get

$$\begin{aligned} &P(N \geq k, x^1, \dots, x^k, T[1 : t^k]|S) \\ &\quad = \Big\{ \prod_{i=1}^{k} p(x^{i-1}, x^i) p_e^0 \left(T[t^{i-1}+1 : t^i], S[L^{i-1}+1 : L^i] \,\middle|\, x^i\right) \Big\} \frac{F(L^k|x^k)}{F(0|I)}. \end{aligned} \tag{10}$$

Dividing (10) by the same expression with k replaced by $k-1$ we obtain

$$\begin{aligned} &P\left(N \geq k, x^k, T[t^{k-1}+1 : t^k] \,\middle|\, S, N \geq k-1, x^1, \dots, x^{k-1}, T[1 : t^{k-1}]\right) \\ &\quad = p(x^{k-1}, x^k) p_e^0 \left(T[t^{k-1}+1 : t^k], S[L^{k-1}+1 : L^k] \,\middle|\, x^k\right) \frac{F(L^k|x^k)}{F(L^{k-1}|x^{k-1})}. \end{aligned} \tag{11}$$

From (11) we can sequentially simulate $(x^1, T[1 : t^1]), (x^2, T[t^1+1 : t^2]), \dots$ if $F(K|x)$ is known for any x and any $K \leq L$.

In order to calculate $F(K|x)$ we make a recursion from (8). We do this by separating the sum into the sum over x^1 and the sum over the remaining variables. For $K_j \leq L_j$, $j = 1, 2, 3$, and $K_j < L_j$ for at least one j, we get

$$F(K|x) = \sum_{x \in \Xi} p(x,z) p_e \left(S[K+1 : K+l(z)] \,|x \right) F(K+l(z)|z), \tag{12}$$

and for $K = L$ we find

$$F(L|x) = p(x,\mathcal{E}) + p(x,D_0) F(L|D_0), \quad D_0 = M(\emptyset). \tag{13}$$

The recursion (12) is solved in the following way. If $F(\tilde{K}|x)$ has been found for all x and all $\tilde{K}$ with $\tilde{K} \geq K$ and $\tilde{K}_j > K_j$ for at least one j, we first find $F(K|D_0)$ from

$$F(K|D_0)(1-p(D_0,D_0)) = \sum_{z \in \Xi, z \neq D_0} p(D_0,z) p_e \left(S[K+1 : K+l(z)] \,|x \right) F(K+l(z)|z),$$

and next find $F(K|x)$, $x \neq D_0$, from (12). The start of the recursion is given by

$$\begin{aligned} F(L|D_0) &= \frac{p(D_0,\mathcal{E})}{1-p(D_0,D_0)}, \\ F(L|x) &= p(x,\mathcal{E}) + p(x,D_0) F(L|D_0), \quad x \neq D_0. \end{aligned} \tag{14}$$

Note that when we have simulated the alignment $x^1, \ldots, x^N$ for the 3-star with interior node r and leaves $a(r)$, $d_1(r)$ and $d_2(r)$ we can immediately read off the alignments $A(r,a(r))$, $A(r,d_1(r))$ and $A(r,d_2(r))$.

3. Complexity and mixing

3.1. Mixing

We first consider the problem of simulating the ancestor and the three alignments of a 3-star tree. Details of the simulation experiments are given in Appendix B. For a 3-star tree our algorithm is designed to simulate directly from the conditional distribution given the three observed sequences. When simulating according to the method of Holmes and Bruno (2001) (see Appendix A below for a description), we find that the mixing is not fast. In the upper part of Figure 3 is a plot of the first 200 autocorrelations on a logarithmic scale based on 100,000 simulated values of the number of deletions that are not followed by an insertion in branch 2 (one value corresponds to one round of updating the three alignments and updating the ancestral sequence). The lower curve in the plot is for three sequences of lengths around 75, and the upper curve is for three sequences of lengths around 150. We see that, apart from an initial phase, there seems to be an exponential decrease of the autocorrelations. We have estimated the slope using the correlations for lags 50 to 150. Based on the first 50 values and the exponential decrease, we have also estimated the sum of the autocorrelations, $\sum_k r_k$. When calculating the variance of the average $\bar{x} = \sum_{i=1}^n x_i$ based on n simulated values, we have that $n\text{Var}(\bar{x}) \approx 1 + 2\sum_k r_k$ in the limit $n \to \infty$. If for example $1 + 2\sum_k r_k = 100$, the interpretation is that we need to simulate $100n$ observations in order to have the same precision as compared with the situation of n independent values when n is large.

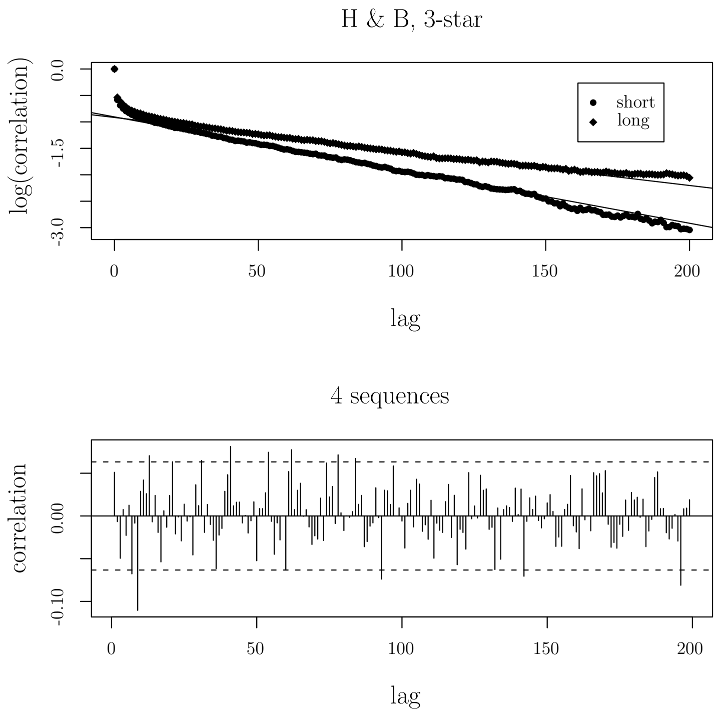

Figure 3. Autocorrelations for the number of deletions along branch 2. In the upper plot is the logarithm of the autocorrelations using the method of Holmes and Bruno (2001) for a 3-star three with sequences of lengths around 75 (short) and 150 (long), respectively. In the lower plot is the autocorrelations for four sequences and using the method of this paper.

In Table 1 we have given the result for the algorithm of Holmes and Bruno (2001) for a 3-star tree as well as for the tree in Figure 1 with four observed sequences and for the tree in Figure 2 with seven observed sequences. The actual numbers here of course depend on the choice of parameter values (see Appendix B). However, the underlying dependency that influences most the properties of the algorithm is the one given in (3). So for a wide range of parameters we expect the same qualitative features as those reported in this section. The exception is when the time parameter τ is either very small or very large.

For the algorithm reported in this paper there is by construction no correlation for the 3-star tree and in the other cases there is very little correlation. An example is given in the lower part of Figure 3 where we consider a tree with four observed sequences (see Figure 1). The autocorrelations are based on 1,000 simulated values, where each value corresponds to an updating of each of the two 3-star trees embedded in the tree in Figure 1.

Table 1. Correlations for the Holmes and Bruno (2001) algorithm.

Length	3-star, 75	3-star, 150	4-seq, 75	4-seq, 150	7-seq, 75
Slope on log scale	-0.0101	-0.0064	-0.0061	-0.0061	-0.0055
$1+2\sum_k r_k$	82	127	130	140	176

3.2. Complexity

Without any refinements our algorithm has complexity L^3, where L is a typical length of a sequence, due to the calculation of $F(K|x)$ via the recursion (12). This can be reduced since $F(K|x)$ will be practically zero outside a band around a 'typical alignment'. The latter can for example be taken as the maximum likelihood alignment or the best alignment within some other scoring system. In the runs reported in this paper there are only few insertions and deletions and we simply take a fixed band around a line in the three dimensional space. Similarly, the algorithm of Holmes and Bruno (2001) has complexity L^2 and this can be reduced by using a band only. For the runs with sequences of length approximately 75 we have used a band of width 20, and for the runs with sequences of length approximately 150 we have used a band of width 30. With these band widths we were not able to detect any difference in the marginal probability of the sequences (9) calculated without and with the use of a band. The band width will generally depend on the separation of the sequences and theoretical calculations can be made on the probability of large excursions. The use of corner cutting or a band limited matrix is a standard technique in dynamical programming and was used for instance by Ukkonen (1985). Hein et al. (2000) used this to achieve a major acceleration for pairwise statistical alignment, and we refer to this paper for further discussion of the technique. Using a band of width w, a rough calculation gives the following complexity measures

$$\begin{aligned}\text{Holmes and Bruno}: & \; k_2 \times 3 \times L \times w + k_3 \times L,\\ \text{Ours}: & \; k_3 \times 15 \times L \times w \times w,\end{aligned}$$

where k_2 is the number of branches, and k_3 is the number of interior nodes. The numbers 3 and 15 are the number of states for aligning two sequences and for aligning a 3-star tree, respectively. These complexity measures seem to be in good agreement with the actual CPU times reported in Table 2. The CPU times reported in Table 2 are based on 1,000 updatings for the Holmes and Bruno algorithm and on 100 updatings for our algorithm. These CPU times are very stable as to the number of updatings used.

Table 2. CPU running times in seconds for 100 rounds of updating of the alignment. The bottom row gives the efficiency of the algorithm in this paper as compared to the algorithm in Holmes and Bruno (2001), using values from Table 1.

	3-star,75 $w = 20$	3-star,150 $w = 30$	4-seq,75 $w = 20$	4-seq,150 $w = 30$	7-seq,75 $w = 20$
H & B	4.99	15.04	9.94	27.77	18.77
Ours	158.5	775.4	330.3	1524.0	830.2
Ratio	32	52	33	55	44
$(1 + 2\sum_k r_k)$/Ratio	2.6	2.5	3.9	2.6	4.0

From the bottom row of Table 2 we see that in all the runs our algorithm is more efficient for estimating mean values. Furthermore, if we can design a version of the algorithm that uses a fixed band, irrespective of the lengths of the sequences, then the efficiency of our algorithm as compared to that of Holmes and Bruno (2001) will increase with the lengths of the sequences.

4. Maximum Likelihood Estimation

4.1. Full likelihood for sequences and the alignments

For two sequences S_1 and S_2 with an alignment $A = \{z^1, \ldots, z^n\}$, where z^i is one of the states M, D, or I, the probability of the sequence S_2 and the alignment A given the sequence S_1 is

$$P(S_2, A|S_1; \tau) = \tilde{p}(z^n, \mathcal{E}; \tau) \prod_{i=1}^{n} \tilde{p}(z^{i-1}, z^i; \tau) \tilde{p}_e^c \left(S_2[L_2^{i-1}+1 : L_2^i] \,\middle|\, S_1[L_1^{i-1}+1 : L_1^i], z^i; \tau \right). \quad (15)$$

Here L_1 and L_2 are the lengths of the sequences, $\tilde{p}(\cdot, \cdot; \tau)$ is the transition probability from (3), and $\tilde{p}_e^c$ is a conditional emission probability

$$\tilde{p}_e^c(w_2|w_1, z; \tau) = \begin{cases} f(w_2|w_1; \tau) & z = M, \\ 1 & z = D, \\ \pi(w_2) & z = I. \end{cases} \quad (16)$$

We can next state the full likelihood $L_f(\theta)$ for the multiple alignment on the tree, that is, the joint probability of $S_1,\ldots,S_\eta$, $T_{\eta+1},\ldots,T_{\eta+\nu}$, and all the pairwise alignments $A(a(j),j)$, as a function of the parameters θ defining the model. Using the notation $S_{\eta+j}=T_{\eta+j}$ we find

$$L_f(\theta)=P(T_{\eta+\nu})\prod_{j=1}^{\eta+\nu-1}P\left(S_j,A(a(j),j)\left|T_{a(j)};\tau_j\right.\right), \tag{17}$$

where $P(T_{\eta+\nu})$ is calculated as in (1).

The marginal likelihood $L_m(\theta)$ based on the observed sequences S_r, $r=1,\ldots,\eta$, is obtained by summing $L_f(\theta)$ over the ancestral sequences and their alignments. The ratio $L_m(\theta_2)/L_m(\theta_1)$ can be calculated as a mean value

$$\frac{L_m(\theta_2)}{L_m(\theta_1)}=E_{\theta_1}\left(\frac{L_f(\theta_2)}{L_f(\theta_1)}\left|S_r,r=1,\ldots,\eta\right.\right). \tag{18}$$

Thus we can use the Gibbs sampler from Section 3 to generate samples from the conditional distribution given S_r, $r=1,\ldots,\eta$, and thereby approximate (18). However, unless θ_2 is close to θ_1, the ratio $L_f(\theta_2)/L_f(\theta_1)$ will have a variance growing exponentially in the length of the sequences. Instead we use the EM-algorithm described next.

4.2. Simulated EM-algorithm

The parameters of the model are the stationary frequencies $\pi(\cdot)$, the birth rate λ, the death rate μ, the parameters of the substitution probabilities (in our case ψ taking care of the difference between transversions and transitions), and the times τ_i along the branches of the tree. It is quite natural to think of the information in the data as consisting of two parts, one part is what can be learned from the sequences concerning the marginal distribution of a sequence, and the other part is what can be learned about the evoulutionary process when considering the changes between two sequences. The marginal distribution contains information on π and $\gamma=\lambda/\mu$, whereas the changes between the sequences mainly provides information on the times τ_i and the transversion/transition ratio ψ.

Using this line of thinking we first estimate the stationary probabilities π from the empirical frequencies in the observed sequences $S_1,\ldots,S_\eta$. Also we estimate $\gamma=\lambda/\mu$ from the average length of the observed sequences

$$\begin{aligned}\hat\pi(a)&=\sum_{j=1}^{\eta}\sum_{i=1}^{L_j}1(S_j[i]=a)/\sum_{j=1}^{\eta}L_j,\\ \hat\gamma&=\frac{\bar L}{1+\bar L},\quad \bar L=\frac{1}{\eta}\sum_{i=1}^{\eta}L_i.\end{aligned} \tag{19}$$

When π and γ have been fixed the full likelihood (17) can, apart form a data dependent term, be written as

$$\begin{aligned} L_f(\theta) = \prod_{j=1}^{\eta+\nu-1} & \Big\{ b(\#,\#;j)^{N(\#,\#;j)} b(\#,-;j)^{N(\#,-;j)} \\ & \times b(-,\#;j)^{N(-,\#;j)} b(-,-;j)^{N(-,-;j)} s(\#;j)^{N(\#;j)} s(-;j)^{N(-;j)} \\ & \times \prod_{w_1,w_2} f(w_2|w_1;j)^{K(w_1,w_2;j)} \Big\}, \end{aligned} \tag{20}$$

where $\theta = (\mu, \psi, \tau_j : j = 1, \ldots, \eta + \nu - 1)$, with ψ the parameters in the substitution matrix. Here $N(\#,\#;j)$ counts the number of times we have the term $b(\#,\#;j)$ in the transition probabilities (see (3)) in the alignment $A(a(j),j)$. All the other counts $N(\cdot)$ are defined similarly, and $K(w_1,w_2;j)$ is the number of substitutions of w_1 by w_2 along the branch j. To use the EM-algorithm we must simulate the mean values of all the count statistics in the conditional distribution given the observed sequences and under the parameter value θ_1, say. A new value θ_2 is then found by maximising (20) with the counts replaced by their mean values.

We use an iterative procedure to maximize (20). We first find, with $\phi = (\mu, \psi)$,

$$\hat{\tau}_j(\phi), \quad j = 1, \ldots, \eta + \nu - 1,$$

for a fixed value of ϕ, and next find a new value of ϕ by maximising

$$L_f(\tilde{\phi}, \hat{\tau}(\phi) : j = 1, \ldots, \eta + \nu - 1)$$

with respect to $\tilde{\phi}$. The reason for this procedure is that finding $\hat{\tau}_j(\phi)$ is a one dimensional search problem, since L_f factorizes for a fixed value of ϕ.

We have tried the above method on a 3-star tree. For a 3-star tree we can compare with the estimates obtained by maximising the likelihood function directly. We have considered a situation with ψ fixed and thus we maximize with respect to $(\mu, \tau_1, \tau_2, \tau_3)$. In Table 3 are the results from 30 steps of the simulated EM-algorithm, where in each step we simulate the ancestral and its alignments 1,000 times. Included are also the averages $\bar{l}_f$ of the full log likelihood function, where $l_f = \log(L_f)$ with L_f given in (17), as well as the marginal log likelihood l_m based on the three observed sequences. The last column in Table 3 shows the steady increase of the likelihood function during the EM-steps, and that we have come close to the maximal value after 30 steps. The parameter estimates have not yet come close to the maximum likelihood estimates, but this shows that the likelihood function is very flat in a large region of the parameter space. This is not surprising since the expected number of substitutions and deletions are each around 6 only.

Table 3. EM-algorithm for a 3-star tree.

	μ	τ_1	τ_2	τ_3	$\bar{l}_f$	l_m
Start	0.100	0.80	0.80	0.80	19.41	-1.95
Iteration 5	0.085	1.15	0.87	1.21	-2.65	-0.43
Iteration 10	0.079	1.27	0.83	1.39	-6.58	-0.25
Iteration 15	0.078	1.32	0.76	1.48	-4.44	-0.17
Iteration 20	0.077	1.35	0.68	1.53	-1.82	-0.12
Iteration 25	0.077	1.38	0.63	1.57	-0.87	-0.09
Iteration 30	0.078	1.39	0.61	1.59	0.00	-0.08
MLE	0.106	1.55	0.28	1.62		0.00

For the tree in Figure 1 we also made 30 steps in the simulated EM-algorithm. The results can be seen in Table 4.

Table 4. EM-algorithm for the tree in Figure 1.

	μ	τ_1	τ_2	τ_3	τ_4	τ_5	$\bar{l}_f$
Start	0.100	0.80	0.80	0.80	0.80	0.80	-1.68
Iteration 5	0.101	0.94	0.79	0.71	0.72	0.75	4.83
Iteration 10	0.102	1.01	0.76	0.68	0.69	0.71	7.40
Iteration 15	0.107	1.09	0.72	0.70	0.67	0.67	3.58
Iteration 20	0.110	1.11	0.66	0.69	0.66	0.67	3.14
Iteration 25	0.113	1.14	0.64	0.70	0.66	0.67	1.34
Iteration 30	0.115	1.15	0.62	0.69	0.65	0.66	0.00

5. An Example

In this section we consider ten sequences of 5S RNA. These ten 5S RNA are chosen very widely from the domain of life and their most recent common ancestor probably existed more than 3 billion years ago. The total branch length on the phylogeny relating these sequnces corresponds to observing evolution for more than 10 billion years. Historically, 5S RNAs have been among the first molecules to be used for phylogenies of the kingdoms of life since they are short (about hundred nucleotides) and present in very diverse organisms. In the late 80s such sequences were at the limit that multiple optimisation alignments could handle. Today the phylogenetic relationship of plants, animals, etc., are done by comparing a large set of molecules that has representatives in the relevant organisms. The evolutionary tree relating the ten sequences are shown in Figure 4. Due to the shortness of the ten molecules, some aspects of the tree are not well determined. Thus, the edge between nodes 12 and 13 is so short that it is not statistically different from zero.

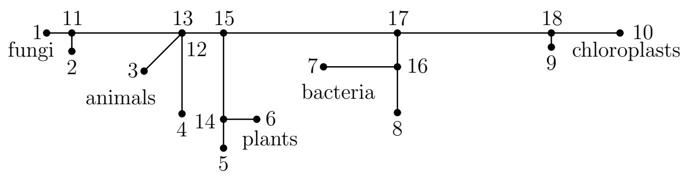

Figure 4. The evolutionary tree for ten 5 S RNA sequences. The lengths of the edges are proportional to the estimated time parameters τ_i. The ten sequences are 1: *Auricularia auricula-judae*, 2: *Auricularia edulis*, 3: *C. elegans*, 4: *Gallus gallus*, 5: *Equisitum arvense*, 6: *Cycad revoluta*, 7: *Bacillus brevis*, 8: *Bacillus firmus*, 9: *Jungermannia subulata*,and 10: *Dryopteris acuminata*.

We ran the simulated EM-algorithm as described above with γ and π estimated directly from the observed sequences and with the transversion/transition rate fixed at $\psi = 0.4$ (based on the results in Hein (1990)). We made ten steps in the EM-algorithm where in each step the mean values were based on 100 updatings of the complete tree. In Table 5 are the maximum of the log likelihood of the alignment from the 100 updatings as well as the average of the 100 updatings. As can be seen after ten iterations these values are stabilized. The estimated time parameters τ_i have been used as the lengths of the edges in Figure 4.

Table 5. EM-algorithm for the tree in Figure 4. In each step of the algorithm 100 updatings of the complete tree are used. The table gives the maximum and the average of the 100 values of the log likelihood of the alignments, with the value 1,386 added.

run	1	2	3	4	5	6	7	8	9	10
max	57.6	125.0	137.0	164.3	150.4	149.3	161.7	152.0	150.1	151.0
average	0.4	69.5	97.8	114.6	116.1	116.8	120.1	118.5	114.4	119.3

To illustrate how our method can be used we consider the alignment of the ten sequences using the CLUSTAL W program (www.ebi.ac.uk/clustalw, default settings being used). The alignment obtained by this method has 19 columns that are conserved (the columns marked by an asterisk in Table 6). We want to evaluate the reliability of the statement that a column is conserved. We can run our algortihms to obtain samples from the posterior distribution of alignments given the ten observed sequences and thereby estimate the probability of a particular column being conserved. Using the estimated parameter values from the EM-algorithm we made 1,000 updates and calculated the fraction for which a particular column was conserved. Many of the columns marked by an asterisk are highly conserved (10 of these columns had a posterior probability greater than 95%). Except for a few cases the probability of the column being conserved is bigger than 80%. In one case, though, the probability is quite low: column 16 out of the 19 columns marked by an asterisk has a posterior probability of being conserved of around 16%. This kind of information cannot be obtained from a alignment programme as CLUSTAL W and requires a probabilistic modelling of the evolution and ways of calculating complicated posterior distributions.

Table 6. Alignment of ten 5 S RNA sequences. Columns marked by an asterisk are conserved.

```
 1:  A----TCCACGGCCATAGGACTCTGAAAGCACTGCATCCCGT-CCGATCTGCAAAGTTAACCAGAG
 2:  A----TCCACGGCCATAGGACTGTGAAAGCACCGCATCCCGT-CTGATCTGCGCAGTTAAACACAG
 3:  G----CTTACGACCATATCACGTTGAATGCACGCCATCCCGT-CCGATCTGGCAAGTTAAGCAACG
 4:  G----CCTACGGCCATCCCACCCTGGTAACGCCCGATCTCGT-CTGATCTCGGAAGCTAAGCAGGG
 5:  GT---GGTGCGGTCATACCAGCGCTAATGCACCGGATCCCAT-CAGAACTCCGCAGTTAAGCGCGC
 6:  G----GGTGCGATCATACCAGCGTTAATGCACCGGATCCCAT-CAGAACTCCGCAGTTAAGCGCGC
 7:  T----CTGGTGATGATGGCGGAGGGGACACACCCGTTCCCATACCGAACACGGCCGTTAAGCCCTC
 8:  T----CTGGTGGCGATAGCGAGAAGGTCACACCCGTTCCCATACCGAACACGGAAGTTAAGCTTCT
 9:  T---TCTGGTGTCTCAGGCGTGGAGGAACCACACCAATCCATCCCGAACTTGGTGGTGAAACTCTA
10:  T-ATTCTGGTGTCCCAGGCGTAGAGGAACCACACCGATCCATCTCGAACTTGGTGGTGAAACTCTG
11:  A----TCCACGGCCATAGGACTCTGAAAGCACCGCATCCCGT-CCGATCTGCGAAGTTAAACACAG
12:  G----CCTACGACCATACCACCCTGAAAGCACCCCATCCCGT-CCGATCTCGGAAGTTAAGCAGGG
13:  G----CCTACGACCATACCACCCTGAAAGCACCCCATCCCGT-CCGATCTCGGAAGTTAAGCAGGG
14:  G----GGTGCGATCATACCAGCGTTAATGCACCGGATCCCAT-CAGAACTCCGCAGTTAAGCGCGC
15:  G----CGTGCGACCATACCAGCGTGAAAGCACCCGATCCCAT-CCGAACTCGGAAGTTAAGCGCGC
16:  T----CTGGTGACGATGGCGGGGAGGTCACACCCGTTCCCATACCGAACACGGAAGTTAAGCTCGT
17:  T----CTGGTGACGATGGCGGGGAGGTCACACCCGATCCCATACCGAACTCGGAAGTTAAACTCGT
18:  T---TCTGGTGTCTCAGGCGTGGAGGAACCACACCAATCCATCCCGAACTTGGTGGTGAAACTCTA
               *                  * *       * *   ** *      *  ** *

 1:  TACCGCC-CAGTTAG-TACCACGGTGGGGGACCACGCGGGAA-TCCTGGGTGCTG-T-GGT-T---
 2:  TGCCGCC-TAGTTAG-TACCATGGTGGGGGACCACATGGGAA-TCCTGGGTGCTG-T-GGT-T---
 3:  TTGAGTC-CAGTTAG-TACTTGGATCGGAGACGGCCTGGGAA-TCCTGGATGTTG-TAAGC-T---
 4:  TCGGGCC-TGGTTAG-TACTTGGATGGGAGACCTCCTGGGAA-TACCGGGTGCTG-TAGGCTT---
 5:  TTGGGCCAGAA-CAG-TACTGGGATGGGTGACCTCCCGGGAAGTCCTG-GTGCCG-CACCC-C---
 6:  TTGGGTTGGAG-TAG-TACTAGGATGGGTGACCTCCTGGGAAGTCCTA-ATATTG-CACCC-TT--
 7:  CAGCGCC-AA--TGG-TACTTGCTCCGCAGGGA-GCCGGGAG-AGTAGGACGTCGCCAGGC-----
 8:  CAGCGCC-GA--TGG-TAGTT-AGGGGCTGTCC-CCTGTGAG-AGTAGGACGCTGCCAGGC-----
 9:  TTGCGGT-GA--CGA-TACTGTAGGGGAAGCCC-GATGGAAA-AATAGCTCGACGCCAGGA---T-
10:  CCGCGGT-AA--CCAATACTCGGGGGG-GGCCC-TGCGGAAA-AATAGCTCGATGCCAGGA---TA
11:  TACCGCC-TAGTTAG-TACCATGGTGGGGGACCACATGGGAA-TCCTGGGTGCTG-T-GGT-T---
12:  TTGCGCC-GAGTTAG-TACTTGGATGGGAGACCACATGGGAA-TCCTGGGTGCTG-TAGGC-T---
13:  TTGCGCC-GAGTTAG-TACTTGGATGGGAGACCACATGGGAA-TCCTGGGTGCTG-TAGGC-T---
14:  TTGGGCCAGAG-TAG-TACTGGGATGGGTGACCTCCTGGGAAGTCCTG-GTGCTG-CACCC-T---
15:  TTGCGCC-GAG-TAG-TACTTGGATGGGAGACCACCTGGGAA-TCCTGGGTGCTG-CAGGC-T---
16:  CAGCGCC-GA--TGG-TACTTGAGCGGCAGGCC-CCTGGGAG-AGTAGGACGCTGCCAGGC-----
17:  CAGCGCC-GA--TGG-TACTTGAGTGGCAGGCC-CCTGGGAA-AATAGGGCGCTGCCAGGC-----
18:  CTGCGGT-GA--CGA-TACTGTAGGGGAAGCCC-TATGGAAA-AATAGCTCGATGCCAGGA---T-
         *           **        *          *  *             *
```

The maximal value of the likelihood of the alignment obtained during the 1,000 simulated alignments was $-1,219.6$ ($= 166.4 - 1,386$, compare Table 5). The corresponding alignment is given in Table 6. The columns marked by an asterisk are the conserved columns and include the one mentioned above having a low probability of being conserved.

6. Discussion

We have shown that it is feasible to simulate multiple alignments within the TKF-model using a Gibbs sampler where in each step a 3-star tree is updated. The Gibbs sampler seems to be mixing very rapidly. Contrary to this the algorithm suggested in Holmes and Bruno (2001) seems to have long mixing times. For the runs reported here the algorithm of Holmes and Bruno (2001) is less efficient than the algorithm suggested in this paper. Via the EM-algorithm we can obtain maximum likelihood estimates of the parameters of the model.

The Gibbs sampler suggested in this paper is not restricted to the exact form of the TKF-model. A more general hidden Markov model simply implies a different state space and different transition probabilities for the 3-star tree in Section 2. In particular one may wish to include the possibility of going to the immortal state instead of the *end* state. This will introduce an extra parameter in the model so that $\gamma = \lambda/\mu$ is no longer determined by the lengths of the sequences.

The algorithm produces samples from the conditional distribution of alignments and ancestral sequences given the observed sequences. One can therefore estimate the probabilities of different evolutionary events. The algorithm does not point to one particular alignment although, in a long run of the algorithm, one can of course choose the alignment with the highest value of the full likelihood function.

Appendix A. Holmes and Bruno algorithm

In Holmes and Bruno (2001) two different updating steps are used. The first one is the ordinary simulation of an alignment of two sequences S_1 and S_2. This means that a set of states $x^1, \ldots, x^N$ of the form M, D, or I is simulated. Here the kth state x^k is simulated from the distribution

$$P\left(x^k \,\middle|\, S, x^1, \ldots, x^{k-1}\right) = \tilde{p}(x^{k-1}, x^k)\tilde{p}_e\left(S[L^{k-1}+1 : L^k] \,\middle|\, x^k\right) \frac{\tilde{F}(L^k|x^k)}{\tilde{F}(L^{k-1}|x^{k-1})}, \tag{21}$$

where $\tilde{p}(\cdot,\cdot)$ is the transition probability from (3), $\tilde{p}_e$ is the emission probability

$$\tilde{p}_e\left(\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} | x\right) = \begin{cases} \pi(w_1)f(w_2|w_1;\tau) & x = M, \\ \pi(w_1) & x = D, \\ \pi(w_2) & x = I, \end{cases}$$

and $\tilde{F}$ satisfies for $K \neq L$ the recursion

$$F(K|x) = \sum_{z, K+l(z) \leq L} \tilde{F}(K + l(z)|z)\tilde{p}(x,z) \\ \times \tilde{p}_e\left(S_2[K_2+1 : K_2+l_2(z)]\,|S_1[K_1+1 : K_1+l_1(z)], z\right),$$

where z is one of the states M, D, or I. The recursion is started by $\tilde{F}(L|x) = \tilde{p}(x, \mathcal{E})$.

The second updating step in Holmes and Bruno (2001) can be explained as follows. We consider a 3-star tree where we have an alignment along each of the three branches. These three alignments are translated into an alignment $\bar{x}^1, \ldots, \bar{x}^{\bar{N}}$ with states $\bar{x}^i$ of the form $M(J)$ or $I(J)$. We first remove all those $\bar{x}^i$ that equal D_0 from (13). This gives us the reduced set of states $x^1, \ldots, x^N$. Then we insert new states equal to D_0 in between x^{i-1} and x^i, the length m_i of the inserted states has distribution

$$P(m_i = k) = \begin{cases} \dfrac{p(x^{i-1}, D_0)p(D_0, x^i)}{p(x^{i-1}, x^i) + p(x^{i-1}, D_0)p(D_0, x^i)} & k = 0, \\ \dfrac{p(x^{i-1}, x^i)(1 - p(D_0, D_0))}{p(x^{i-1}, x^i) + p(x^{i-1}, D_0)p(D_0, x^i)} p(D_0, D_0)^{k-1} & k > 0. \end{cases}$$

Finally we update, for $i = 1, \ldots, N$, the ancestral letter at the interior node corresponding to the state x^i. This means that if $x_0^i = -$, there is no letter to update and if $x_0^i = \#$, we choose a letter w_0 according to the distribution

$$p(w_0) \propto \pi(w_0) \prod_{\{j \geq 1 : x_j^i = \#\}} f(S_j[L_j^{i-1} + 1]|w_0; \tau_j).$$

Appendix B. Details of simulation experiments

In the simulations we model sequences of nucleotides so that the possible letters are A, G, C, T. The substitution process is described by the rate matrix

$$q(w_1, w_2) = \frac{\pi(w_2)\psi^{1_{tv}(w_1, w_2)}}{c},$$

where 1_{tv} is one if substituting w_2 for w_1 is a transversion and 1_{tv} is zero otherwise. The scaling constant c was taken to be $\pi(G) + \psi(\pi(C) + \pi(T))$, so that there is a unit rate for a change of the letter A.

We considered the tree in Figure 1 with four observed sequences. We generated observed sequences from the TKF-model using the parameter settings $\mu = 0.1$, $\lambda = 0.099$, $\psi = 0.2$, $\pi = (0.2, 0.3, 0.2, 0.3)$ and $\tau_1 = \tau_2 = \tau_3 = \tau_4 = \tau_5 = 0.8$. These values were chosen so as to be realistic in many situations. The stationary probabilities deviates to a moderate degree from the uniform distribution. The death rate μ and the times τ_i are such that roughly 8% of the nucleotides will be deleted. This is slightly bigger than the value found in Hein et al. (2000) when studying α globin and β globin. The transition/transversion bias varies and is especially pronounced in mitochondria. In mammalian nuclear genes a bias of 4 in favour of transitions is not unrealistic (Li (1997)).

For the investigation of the mixing properties we generated observed sequences of lengths $(75, 74, 79, 79)$ and $(156, 147, 145, 146)$, respectively. When simulating the ancestral sequences and their alignments we took γ and π as given in (19), used the true values of μ, ψ, and took the branch lengths to be $\tau = 0.8$. As the initial value for the ancestral sequences we simply took $T_5 = S_1$ and $T_6 = S_4$. When simulating from the tree in Figure 2, similar choices were made.

References

Durbin, R., Eddy, S., Krogh, A. and Mitchison, G. (1998). *Biological Sequence Analysis: Probabilistic Models of Protein and Nucleic Acids.* Cambridge University Press, Cambridge, UK.

Hein, J. (1990). A unified approach to phylogenies and alignments. *Methods in Enzymology* **183**, 625-644.

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

Hein, J., Jensen, J. L. and Pedersen, C. N. S. (2003). Recursions for statistical multiple alignment. *Proc. Natl. Acad. Sci. USA* **100**, 14960-14965.

Hein, J., Wiuf, C. Knudsen, B., Møller, M. B. and Wibling, G. (2000). Statistical alignment: computational properties, homology testing and goodness-of-fit. *J. Mol. Bio.* **302**, 265-279.

Holmes, I. and Bruno, W. J. (2001). Evolutionary HMMs: a Bayesian approach to multiple alignment. *Bioinformatics* **17**, 803-820.

Li, W.-L. (1997). *Molecular Evolution.* Sinauer, Sunderland Massachusetts.

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.

Ukkonen, E. (1985). Algorithms for approximate string matching. *Infor. and Control* **64**, 100-118.

Zhu, J., Liu, J. S., and Lawrence, C. E. (1998). Bayesian adaptive sequence alignment algorithms. *Bionformatics* **14**, 25-39.

Department of Theoretical Statistics, Institute of Mathematics, University of Aarhus, Ny Munkegade, DK-8000 Aarhus C, Denmark.

E-mail: jlj@imf.au.dk

Department of Statistics, Oxford University, The Peter Medawar Building for Pathogen Research, South Parks Road, Oxford OX1 3SY, England.

E-mail: hein@stats.ox.ac.uk

(Received September 2002; accepted January 2005)