Syst. Biol. 68(2):252–266, 2019

DOI:10.1093/sysbio/syy059
Advance Access publication September 15, 2018

A Simulation-Based Approach to Statistical Alignment

ELI LEVY KARIN[1], HAIM ASHKENAZY[1], JOTUN HEIN[1,2], AND TAL PUPKO[1,*]

[1]*School of Molecular Cell Biology & Biotechnology, George S. Wise Faculty of Life Sciences, Tel-Aviv University, Tel-Aviv 69978, Israel; and*
[2]*Department of Statistics, University of Oxford, Oxford, UK*
[*]*Correspondence to be sent to: School of Molecular Cell Biology & Biotechnology, George S. Wise Faculty of Life Sciences, Tel-Aviv University, Tel-Aviv 69978, Israel; E-mail: talp@post.tau.ac.il*

Received 04 February 2018; reviews returned 10 September 2018; accepted 10 September 2018
Associate Editor: David Bryant

Abstract.—Classic alignment algorithms utilize scoring functions which maximize similarity or minimize edit distances. These scoring functions account for both insertion–deletion (indel) and substitution events. In contrast, alignments based on stochastic models aim to explicitly describe the evolutionary dynamics of sequences by inferring relevant probabilistic parameters from input sequences. Despite advances in stochastic modeling during the last two decades, scoring-based methods are still dominant, partially due to slow running times of probabilistic approaches. Alignment inference using stochastic models involves estimating the probability of events, such as the insertion or deletion of a specific number of characters. In this work, we present SimBa-SAl, a simulation-based approach to statistical alignment inference, which relies on an explicit continuous time Markov model for both indels and substitutions. SimBa-SAl has several advantages. First, using simulations, it decouples the estimation of event probabilities from the inference stage, which allows the introduction of accelerations to the alignment inference procedure. Second, it is general and can accommodate various stochastic models of indel formation. Finally, it allows computing the maximum-likelihood alignment, the probability of a given pair of sequences integrated over all possible alignments, and sampling alternative alignments according to their probability. We first show that SimBa-SAl allows accurate estimation of parameters of the long-indel model previously developed by Miklós et al. (2004). We next show that SimBa-SAl is more accurate than previously developed pairwise alignment algorithms, when analyzing simulated as well as empirical data sets. Finally, we study the goodness-of-fit of the long-indel and TKF91 models. We show that although the long-indel model fits the data sets better than TKF91, there is still room for improvement concerning the realistic modeling of evolutionary sequence dynamics. [Long-indel model; pairwise alignment; sequence simulations; SimBa-SAl; statistical alignment.]

Through the course of evolution, a sequence changes with respect to its ancestor by substitution as well as insertion–deletion (indel) events, yielding a modified, possibly longer, or shorter sequence. An alignment of a sequence to its ancestor represents the homology relationships at the position level. Alignment inference is a challenging task that has been studied for almost five decades (e.g., Needleman and Wunsch 1970; Vingron and Waterman 1994; Chatzou et al. 2016). Alignments are typically prerequisites for at least three major classes of investigation: inferring phylogenies, detecting homologous regions, which point at common origin, and possibly function, and analyzing the nature of molecular evolution in terms of rates and selection (Yang 2014; Graur 2016). Due to their great importance for these evolutionary analyses, constructing alignments has been dubbed "The Holy Grail of Sequence Analysis" (Gusfield 1997) and it continues to pose algorithmic challenges to this day (Kemena and Notredame 2009; Iantorno et al. 2014).

Following the pioneering work of Needleman and Wunsch (1970) who aligned two sequences by optimizing a similarity score, distance minimizing alternatives were presented by a series of authors (e.g., Sankoff 1972; Sellers 1974; Wagner and Fischer 1974). Much effort was invested in choosing optimal parameters for these algorithms, weighting indels of various sizes (Waterman et al. 1976; Gotoh 1982) and developing faster and memory efficient techniques (Hirschberg 1975; Ukkonen 1985; Myers 1986).

Bishop and Thompson (1986) were the first to treat alignment inference within stochastic evolutionary process theory. However, later models in the field heavily rely on the TKF91 model (Thorne et al. 1991). In that seminal work, only indels of a single nucleotide or amino acid were allowed. Thorne et al. (1992) extended their model to allow for longer indels of a geometric length distribution. However, overlapping indels were not allowed, which is biologically unrealistic. Miklós et al. (2004) extended statistical alignment to longer indels. Although their work formed a robust foundation for statistical alignment, very few studies applied this model to analyze sequence data. One reason for the paucity in follow-up studies is that the long-indel model is conceptually more challenging compared with TKF91 or score-based non-stochastic alignment methods. Furthermore, the naïve implementation of pairwise alignment (PWA) inference under this model is quartic in the unaligned sequence lengths, which is very slow. Together, these challenges resulted in a lack of available software for statistical PWA inference using genuine probabilistic approaches.

Despite these difficulties there are strong arguments for the advantage of using probabilistic approaches to align sequences. One such advantage is the ability to estimate parameters concerning indel rates and sizes,

which can be highly informative regarding how indel dynamics change between genes and organisms (Ophir and Graur 1997; Hamilton et al. 2003; Lunter 2007; Chen et al. 2009; Levy Karin et al. 2017). Furthermore, unbiased sampling of alternative alignment solutions, rather than relying on a single point estimate, is possible when employing a statistical approach. Moreover, a statistical framework allows for model selection techniques to determine which indel model best fits specific data. Finally, stochastic models of indels are required for a realistic simulation of sequences (e.g., Stoye et al. 1998; Cartwright 2005; Fletcher and Yang 2009; Sipos et al. 2011).

Hidden Markov Models (HMMs) have been used for the analysis of biological sequences for more than 20 years (e.g., Baldi et al. 1994; Krogh et al. 1994). These first HMMs were clearly stochastic but did not correspond to an evolutionary model, that is, they were a stochastic representation of the sequence family members around an equilibrium state and they ignored any underlying tree structure and ancestor-descendant relationships. However, most continuous time indel Markov models can be expressed as HMMs. For example, the TKF91 and TKF92 models were formulated as HMMs by Hein (2001) and Holmes and Bruno (2001) and the long-indel model has an HMM representation as well (Miklós et al. 2004). HMMs have since been widely improved and are used in many advanced alignment algorithms, including BaliPhy (Redelings et al. 2005) and Clustal Omega (Sievers et al. 2011). Nevertheless, the translation of continuous time Markov models to HMMs is often not immediate (Holmes 2017) and the major difference between the two arises in the analysis of multiple sequences, as discussed below.

The first key step toward statistical multiple sequence alignment (MSA) was taken by Steel and Hein (2001). They presented the application of the TKF91 model to a star tree with several branches. The ancestral sequence of such a star tree can be generated by an HMM with three states: "start", "base" (nucleotide or amino acid), and "stop". Most subsequent MSA algorithms are a generalization of this model (e.g., Lunter et al. 2003). One key difference between describing an MSA using a set of HMMs versus a continuous time Markov process is that only in the latter, the model parameters are shared among all tree branches while in HMM-based methods, a different pair HMM is assumed for each branch of the phylogenetic tree (e.g., Holmes 2003). Thus, constructing the appropriate HMM for many sequences implies a great increase in the size of the state space.

Herein, we introduce a novel computational approach for statistical alignment. We developed a versatile simulation-based procedure to calculate the probability of the basic building blocks of the long-indel statistical alignment. Our proposed procedure serves as an alternative to the original trajectory-based computation to estimate these probabilities. Furthermore, we accelerate the fundamental inference algorithm to greatly reduce running times. We demonstrate the applicability of the method using simulations as well as analyses of the HOMSTRAD 2017 benchmark database. Finally, we examine attributes concerning model fitting by extracting summary statistics from input PWAs and comparing them to summary statistics extracted from corresponding sets of simulated PWAs.

Materials and Methods

Generation of test data sets under the long-indel model

We define a "skeleton alignment" to be an alignment over a binary alphabet: each entry is either "-" (gap) or "#" (non-gap). In the long-indel model, the skeleton alignment is dictated by three parameters: r, μ, and t (described in detail in the section The Long-indel Model). We used the SimBa-SAl simulator to produce 540 PWA skeletons: 20 skeletons from each of 27 parameter combinations: $r = \{0, 0.3, 0.6\}$, $\mu = \{0.04, 0.07, 0.1\}$, and $t = \{0.25, 0.5, 1\}$. The ancestral sequence in each such skeleton had exactly 200 non-gap characters. We then simulated a substitution process over each such skeleton using the JTT replacement matrix (Jones et al. 1992).

Generation of test data sets under the INDELible model

To test the robustness of PWA inference with the long-indel model to model misspecification, PWAs were also simulated using INDELible (Fletcher and Yang 2009). The INDELible model relies on three parameters: IR, A (described in detail in the section Robustness to Model Misspecification), and a branch length t. INDELible was used to simulate PWAs under 27 parameter combinations: $A = \{1.1, 1.3, 1.7\}$, $IR = \{0.01, 0.02, 0.05\}$, and $t = \{0.25, 0.5, 1\}$. Altogether, 540 (20 $\times$ 27) true PWAs were simulated. In each simulation, the ancestral sequence length was set to 200 and the substitution process followed the JTT replacement matrix (Jones et al. 1992) along a branch of length t. The maximal indel length was restricted to 50. Columns composed of only gap characters were removed.

Column score computation

Each inferred PWA was compared with its corresponding true PWA in order to evaluate its accuracy. The accuracy measure is the column score, the computation of which is fully described in the supplementary material of Satija et al. (2009).

Gotoh and MAFFT pairwise inference procedures

We computed PWAs based on the Gotoh (1982) algorithm using the NEEDLE program, which is part of EMBOSS package version 6.3.0 (Rice et al. 2000). MAFFT PWAs were computed using version 7.123b and employing the FFT-NS-2 strategy (Katoh and Standley 2013). For both inference methods default parameter

values were used. Both programs use the BLOSUM62 substitution matrix (Henikoff and Henikoff 1992).

SimBa-SAl implementation and source code

The programs (SimBa-SAl simulator and SimBa-SAl inference) are implemented in C++. The source code as well as its documentation are available from https://github.com/elileka. To avoid underflows, computations are carried out in log-space. Using the Gumbel-max method allows sampling from a vector of log-probabilities according to the probabilities. Summing probabilities is possible by implementing the log-sum-exp trick, in which the largest log probability is subtracted from all other log probabilities and the exponent of each difference is computed before summation.

Model and Algorithm

The long-indel model

The long-indel model is a full probabilistic evolutionary Markov model that includes substitutions, insertions and deletions (Miklós et al. 2004). It is a time-reversible, context-independent model that allows indels of arbitrarily long-sequence segments. The formation of indels under this model is controlled by three parameters: μ, r, and γ. The identity of the characters inserted or deleted in an indel event is independent of the event itself. Events are described to occur rightward of the starting position. The parameter μ is the total rate a given position is deleted by events of any length that started in that position or to its left. The following geometric function is proposed to describe μ_k, the rate of deletions of size k ($k \geq 1$) that start at a given position:

$$\mu_k = \mu(1-r)^2 r^{k-1} \tag{1}$$

This formulation assures the total rate a position is deleted by an event of any size is equal to μ. The r parameter controls the tendency of events to be short (r close to 0) or long (r close to 1).

The rate of insertions of size k that start to the right of a given position is λ_k. The time reversibility of the model implies a detailed balance (Miklós et al. 2004), which in turn, implies the following concerning the equilibrium probability of a sequence of length n:

$$v(n) = \gamma^n(1-\gamma), \tag{2a}$$

and that the following equation holds for every k (and specifically for $k=1$):

$$\gamma^k = \frac{\lambda_k}{\mu_k}. \tag{2b}$$

Thus,

$$\lambda_k = \gamma^k \times \mu_k. \tag{3}$$

The total rate of deletion events, d, that start at a given position is

$$d = \sum_{k=1}^{\infty} \mu(1-r)^2 r^{k-1} = \mu(1-r). \tag{4}$$

Similarly, i, the total rate of insertion events that start at a given position, is

$$i = \sum_{k=1}^{\infty} \mu_k \gamma^k = \sum_{k=1}^{\infty} \gamma\mu(1-r)^2(\gamma r)^{k-1} = \frac{\mu\gamma(1-r)^2}{1-\gamma r}. \tag{5}$$

Thus, the total rate of events of any kind that start at a given position is $q = d + i$.

In this study, we limited the maximal indel size to be 50, that is, events longer than 50 are considered to be of length 50.

Simulating under the long-indel model

We implemented a pairwise sequence simulator under the long-indel model (SimBa-SAl simulator). In this work, we used it for two purposes: estimating indel event probabilities (see below) and benchmarking PWA inference procedures. Following Miklós et al. (2004), we embed an ancestral sequence A in an infinite sequence. In practice, since we limit the indel size to 50 characters, we can embed the sequence within a finite sequence, whose length we denote N. We simulate the evolution of A and its flanking regions along a branch of length t. The need to simulate with flanking regions stems from the fact that indels can start to the left of A and continue beyond the right border of A, exposing the left end positions to uneven indel rates (see Miklós et al. 2004). The simulation takes advantage of independence between the substitution and indel processes as assumed in the long-indel model. Thus, we focus solely on the indel process and produce an "alignment skeleton," in which each character is either "-" (gap) or "#" (non-gap). In order to allow for the reconstruction of the true alignment at the end of the simulation, each character is assigned a unique ID (as described by Ezawa 2016). Homologous characters share the same ID. If a character is deleted from the evolving sequence, its ID is removed with it and if a character is inserted, it is accompanied with a newly generated ID. The simulation is initiated with an ancestral sequence of length N (including the flanking regions of the embedded ancestral sequence). As the sequence evolves, its length changes. Let M be the length of the evolving sequence at a given time point and q as defined after equation 5. A waiting time for the next event is drawn from an exponential distribution with parameter $M \times q$. If the waiting time exceeds the branch length t, the simulation ends. The type (insertion/deletion) and size of the event are sampled according to the relative indel event rates. The leftmost starting position of the event is sampled uniformly across all positions. If the event is a deletion of size k, the starting position and $(k-1)$ positions to its right are removed. If

the event exceeds the evolving sequence, it is truncated. Similarly, if the event is an insertion, k characters are inserted right to the starting position.

At the end of the simulation, the descendant sequence is compared with the ancestral sequence. Shared IDs represent homologous characters. IDs that are not found in the descendant indicate these positions were deleted, whereas IDs greater-than or equal-to N indicate characters that were inserted in the descendant. The PWA is then trimmed to remove flanking regions of length 200 characters from each side, which might have been exposed to uneven indel rates.

Division of a PWA into segments

Miklós et al. (2004) described the division of a PWA between an ancestral sequence A and a descendant sequence D into conditionally independent alignment segments, as detailed below. The PWA probability computed in this study is conditioned on the ancestral sequence A. The computation of this probability is demonstrated with the following simple example, where 'A' and 'D' with a subscript denote amino acids or nucleotide bases:

A_0	A_1	A_2	–	A_3	A_4	A_5	A_6
-	D_0	–	D_1	D_2	–	D_3	-

We first note that the evolution of the sequence can be described using four events: $\delta-$ death, $\sigma-$ survival, $\rho-$ replacement, and $\iota-$ insertion. In the above example, A_0 dies (denoted $\delta(A_0)$). Similarly, A_1 survives ($\sigma(A_1)$) and is substituted by D_0 ($\rho(A_1,D_0)$) and D_1 is inserted ($\iota(D_1)$). Using this notation, the PWA probability is

$$P(PWA|A)=P(\delta(A_0),\sigma(A_1),\rho(A_1,D_0),\delta(A_2),\iota(D_1),\sigma(A_3),\rho(A_3,D_2),\delta(A_4),\sigma(A_5),\rho(A_5,D_3),\delta(A_6)).$$

The events up to and including a match (homologous position) on the right divide the list of events into segments. For example, the second segment C_2 consists of the events:

$$C_2=\delta(A_2),\iota(D_1),\sigma(A_3),\rho(A_3,D_2).$$

The probability of the alignment can be rephrased using conditional probability as:

$$P(PWA|A)=P(C_1,C_2,C_3,C_4)=P(C_1)\times P(C_2|C_1)\times P(C_3|C_1,C_2)\times P(C_4|C_1,C_2,C_3).$$

Since indel events do not cross match positions, given the segment to its immediate left, each segment is independent of any other preceding segments:

$$P(PWA|A)=P(C_1)\times P(C_2|C_1)\times P(C_3|C_2)\times P(C_4|C_3).$$

Furthermore, the dependence of each segment on the previous one reduces to the dependence on a match position on the left (denoted as M):

$$P(PWA|A)=P(C_1)\times P(C_2|M)\times P(C_3|M)\times P(C_4|M).$$

Division of a segment into independent components

Under the model, the indel process is independent of the substitution process, which allows factorizing each segment. For example, the second segment can be factorized as:

$$P(C_2|M)=P(\delta(A_2),\iota(D_1),\sigma(A_3),\rho(A_3,D_2)|M)=P(\delta(A_2),\iota(D_1),\sigma(A_3)|M)\times P(\rho(A_3,D_2)|M).$$

Of note, $P(\delta(A_2))$ and $P(\sigma(A_3))$ do not depend on the identities of A_2 and A_3 since the entire alignment computation is conditioned on the ancestral sequence A and the indel process is assumed to be context-independent. Hence, these probabilities can be written as $P(\delta(\#_1))$ and $P(\sigma(\#))$, where # denotes any character and the subscript indicates that the deleted sequence is of length 1. Further, $P(\iota(D_1))$ is equal to the insertion probability of a substring of one character in length (independent of the character identity) times the probability that the identity of the inserted character is D_1. This probability can be represented as $P(\iota(\#_1))\times P(D_1)$. And thus:

$$P(C_2|M)=P(\delta(\#_1),\iota(\#_1),\sigma(\#)|M)\times P(D_1)\times P(\rho(A_3,D_2)).$$

By this factorization, each segment is decomposed into two components; one that is character-independent and one that is character-dependent, which does not depend on M. The character-independent component is termed a chop as defined by Miklós et al. (2004).

The L, N, R, and B chops

There are four types of chops depending on their position with respect to adjacent segments. The first segment from the left (C_1 in the example above) harbors the L chop. This chop is not conditioned on a match to the left and includes the first $\sigma(\#)$ factor. We follow the notation of Miklós et al. (2004) and denote $P(L_{i,j})$ as the probability of an L chop in which i ancestral characters were deleted and j descendant characters were inserted. In the above example, the segment C_1 is associated with $L_{1,0}$. An N chop is a chop that is conditioned on a match position M to its left and includes a $\sigma(\#)$ factor, that is, an N chop always ends with a match position. An N chop will always be to the right of an L or an N chop. As in the case of an L chop, $P(N_{i,j})$ is the probability of an N chop in which i ancestral characters were deleted and j descendant characters were inserted. In the above example, the chop associated with the second segment has a probability $P(N_{1,1})=P(\delta(\#_1),\iota(\#_1),\sigma(\#)|M)$. An R chop is the last chop to the right. It is also conditioned on a match position M to its left. However, it does not include a $\sigma(\#)$ factor. The probability $P(R_{i,j})$ is defined similarly to $P(L_{i,j})$ and $P(N_{i,j})$. Of note, if the alignment ends on a match, the last chop is $R_{0,0}$, that is, conditioned on the last match, neither ancestral characters were deleted nor descendant characters were inserted. Finally, in theory there can be cases in which the alignment

does not include even a single match position. In such a case there is only a single chop in the alignment and its probability is denoted $P(B_{i,j})$, where i is the length of the ancestral sequence and j is the length of the descendant sequence.

Thus, the PWA probability of the above example and using the above notation is:

$$\begin{aligned} P(PWA|A) = & P(L_{1,0}) \times P(\rho(A_1, D_0)) \\ & \times P(N_{1,1}) \times P(D_1) \times P(\rho(A_3, D_2)) \\ & \times P(N_{1,0}) \times P(\rho(A_5, D_3)) \\ & \times P(R_{1,0}). \end{aligned}$$

Character-dependent probabilities

Any of the well-established substitution stochastic models, such as JC, HKY85, and GTR for nucleotides and JTT, LG, and WAG for amino acids can be used to compute the character-dependent components of each segment (see Arenas 2015 for a review of these models).

A simulation-based approach to estimate chop probabilities

Miklós et al. (2004) developed an elaborate trajectory-based method for estimating chop probabilities. To compute a specific chop probability, this trajectory-based method enumerates over all possible paths of sequence states. Each of these states is a result of a specific indel event at a specific time. Due to the infinite number of possible trajectories, boundaries over the number of events need to be set. Let Q be the maximal number of events, then the time complexity of computing each chop probability is $O(Q^2)$. Moreover, the trajectory computation depends on the indel model specification. In this work, we propose an alternative procedure for estimating chop probabilities, which is simulation-based. Our method consists of simulating a long PWA under the model and then using this alignment to estimate and tabulate the probabilities of the various chops (see below and Fig. 1). Importantly, our method decouples the estimation of chop probabilities from alignment inference computations (e.g., computing the probability of a given PWA and finding the maximum-likelihood PWA).

Tabulating N, L, R, and B chop probabilities

For a given long-indel parameter combination and branch length t, the SimBa-SAl simulator is used to generate a long-true PWA (the length of the ancestral sequence including the flanking regions is 1,000,000). To estimate the probability of each $N_{i,j}$ chop, the alignment is scanned from left to right. As detailed above, a segment associated with an N chop ends on a match, conditioned on a match to its left. The probability of the chop is

$$P(N_{i,j}) = \frac{|N_{i,j}|}{\sum_{i,j}|N_{i,j}|}$$

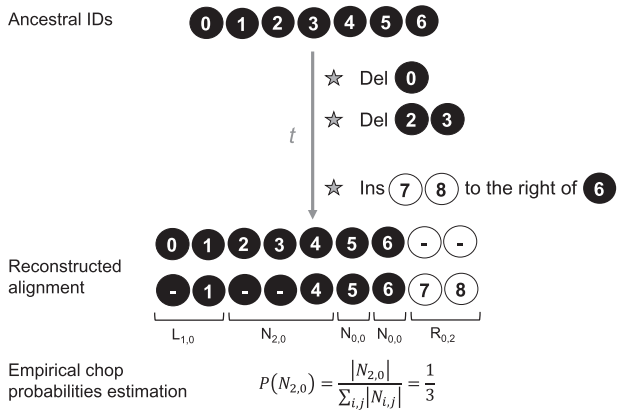

FIGURE 1. A simulation-based estimation of chop probabilities. A simulation to estimate chop probabilities starts with an ancestral sequence of length N (here, $N=7$; throughout this study, $N=$ 1,000,000). The simulation focuses solely on indel events; thus, the actual character identities are irrelevant. Indel events occur along a branch of length t in accordance with indel event rates (see Main Text). At the end of the simulation, the true PWA is reconstructed based on character IDs. Chop probabilities are estimated from this alignment.

where $|N_{i,j}|$ denotes the total number of observed chops of type $N_{i,j}$.

The L, R, and B chops occur at one or both alignment ends. Therefore, their empirical estimation requires several simulated alignments of various lengths. Instead of repeated simulations, the same long-simulated PWA used for the estimation of the N chops is divided into segments of various sizes. The sizes of the segments reflect the stationary length distribution (see Eq. 2a). Chop probabilities are computed similarly to the probability of an N chop.

Consistency checks

Appendix B of the long-indel paper (Miklós et al. 2004) presents equations (30–34), which hold in case chop probabilities are consistent. We use these equations to evaluate the empirical estimation procedure under the long-indel model. Of note, the sums of B chops in equation (31) and of R chops in equation (32) in Miklós et al. (2004) are missing a factor of $(1-\gamma)$. We thus added this factor to the consistency checks performed by our program. Moreover, despite not being explicitly stated in their paper, equation (33) should hold only in the case $t=1$.

Generation of chop tables

We used the SimBa-SAl simulator to compute chop tables under 55,480 parameter combinations of r, μ, and t. In all simulations, we fixed $\gamma=0.99$ (see section Parameter Optimization below). The parameter values were: $r=\{0, 0.05, \ldots, 0.9\}, \mu=\{0.01, 0.02, \ldots, 0.2\}$, and $t=\{0.05, 0.06, \ldots, 1.5\}$. These ranges represent reasonable values of each of the model parameters. The set of r parameter values covers the entire plausible range, the

TABLE 1. Inference procedures and dynamic programming matrices

Goal	Required matrices
Computing the ML-PWA conditioned on A	S
Sampling a random PWA according to its probability	Z
Computing the probability of D conditioned on A (summing over all possible PWAs)	P
Computing the probability that a specific chop-associated segment is included in a PWA of A and D	X, P

A = ancestral sequence; D = descendant sequence.

set of μ values spans cases with very few deletions (0.01) to a very high deletion to substitution rate ratio (0.2) and the t values cover a range starting with highly conserved sequences up to highly diverged ones. The computed chop tables under each of these parameter combinations are available from the authors.

Inference using precomputed chop tables

There are several inference procedures concerning an ancestral sequence A and a descendant sequence D. All these inference procedures rely on similar dynamic programming algorithms to compute various matrices (Table 1). Below we modify the dynamic programming procedure described by equations (14) and (15) in Miklós et al. (2004) to compute the maximum-likelihood pairwise alignment (ML-PWA) rather than the probability of D conditioned on A. The computation of the ML-PWA relies on two stages. In the first stage, a matrix S is computed, where S^i_j is the probability of the partial ML-PWA of the first $(i+1)$ ancestral characters and the first $(j+1)$ descendant characters, such that character A_i is matched with character D_j ($i=0,\ldots,|\mathrm{A}|-1$ and $j=0,\ldots,|\mathrm{D}|-1$):

$$S^i_j = max\left\{\begin{array}{l} P(L_{i,j})\times P_t(A_i\rightarrow D_j)\times\prod_{k=0}^{j-1} q(D_k), \\ max_{n,m}\left\{S^{i-n-1}_{j-m-1}\times P(N_{n,m})\times P_t(A_i\rightarrow D_j)\times\prod_{k=j-m}^{j-1} q(D_k)\right\}\end{array}\right\} \quad (6)$$

where $P_t(A_i\rightarrow D_j)$ and $q(D_k)$ are computed according to the transition and stationary probabilities of the substitution process, respectively. The limits of n and m are $0\le n\le(i-1), 0\le m\le(j-1)$.

In the next stage, the complete ML-PWA is recovered by using the S matrix. Since each element S^i_j refers to the partial alignment whose last match is between characters A_i and D_j, the rest of the alignment is a segment associated with an R chop of the remaining characters in A and in D:

$$P(ML-PWA) = max\left\{\begin{array}{l} P(B_{|A|,|D|})\times\prod_{k=0}^{|D|-1} q(D_k), \\ max_{n,m}\left\{S^{|A|-n-1}_{|D|-m-1}\times P(R_{n,m})\times\prod_{k=|D|-m}^{|D|-1} q(D_k)\right\}\end{array}\right\} \quad (7)$$

where $0\le n\le(|A|-1)$ and $0\le m\le(|D|-1)$.

In equations (6) and (7), in case the number of inserted descendant characters is equal to 0 (i.e., when $j=0$ or $m=0$), the product associated with their insertion is equal to 1.

As detailed in Table 1, additional dynamic programming computations that can be carried out by SimBa-SAl rely on several matrices: S, Z, P, and X. Sampling a PWA is possible by replacing the "max" operator in equations (6) and (7) above with sampling chop-associated segments according to their relative probabilities. This modification of equation (6) results in the Z matrix. The P matrix can be thought of as the "forward components" matrix. Its computation is similar to that of the S matrix, with the "max" operator replaced with summation. Element P^i_j is the sum of probabilities of all partial alignments of the first $(i+1)$ ancestral characters and $(j+1)$ descendant characters such that character A_i is matched with character D_j. Using P and replacing the 'max' operator with summation in equation (7) allow computing the probability of D conditioned on A (i.e., considering all possible alignments). This modification of equations (6) and (7) is used in the parameter optimization procedure (see details below). The X matrix can be thought of as the "backward components" matrix. Element X^i_j is the sum of probabilities of all partial alignments of ancestral characters $(i+1)$ to the end and the descendant characters $(j+1)$ to the end, conditioned on a match between A_i and D_j.

Time complexity

The time complexity of computing any of the matrices above using the original method is $O(n^4\times Q^2)$, where each of the unaligned sequences is of length $O(n)$ and $O(Q^2)$ is the time complexity of the chop probability computation (see above). Using pre-computed chop probabilities, the complexity is $O(n^4\times c(V))$, where V is the size of the chop probabilities table and $c(V)$ is the cost of a look up operation (under current implementation, $c(V)=\log(V)$). Even this reduced time complexity becomes limiting because of the $O(n^4)$ component. We thus introduced additional accelerations.

Accelerations

In order to reduce the running times of the dynamic programming procedure, we have implemented a chop-based acceleration. This acceleration modifies equation (6) when computing element S_j^i (similarly for matrices Z, P, and X) to consider only previous S_{j-m-1}^{i-n-1} values for which the corresponding $N_{n,m}$ is found in the precomputed chop table. Thus, the number of previous S values depends on the size of the N chop table and not on the value of i and j (i.e., it does not depend on the length of the unaligned sequences). This modification results in time complexity of $O(n^2 \times V)$. Of note, excluding computations for $N_{n,m}$ chops not found in the table implicitly assigns 0 as the probability of any chop not observed in the simulation procedure.

Parameter optimization

The above procedure describes the computation of an ML-PWA for a specific set of model parameters. However, the long-indel model parameters: r, μ, γ, and the branch length t that best fit a given data set are unknown. As different values of these parameters may result in different ML-PWAs, these parameters need to be inferred. We fixed the value of the γ parameter to be 0.99. This decision is based on the equilibrium length distribution (see Eq. 2a). As this length is geometrically distributed with parameter γ, the expected sequence length n is $E(v(n)) = \frac{\gamma}{1-\gamma}$. Thus, under a reasonable assumption that $E(v(n)) > 100$, we obtain that the value of the γ parameter should be greater than 0.99. Since γ is between 0 and 1, we fixed it to be 0.99.

Throughout this work, parameter combinations were restricted to a set of 55,480 options (see details above). This discretization allows using precomputed chop tables, thus decoupling the inference procedure from the one-time task of computing chop probabilities, which accelerates the inference procedure. The score of each parameter combination is computed based on the P matrix, that is, the probability of D conditioned on A, marginalized over all possible PWAs. Our goal is to find the parameter combination (r, μ, and t) that maximizes this likelihood. We propose two optimization schemes. The first scheme is a brute-force ML paradigm, which considers all parameter combinations, returning the parameter combination that yielded the highest likelihood. The number of dynamic programming computations in this scheme is equal to the number of considered parameter combinations and it is thus exhaustive. In the alternative optimization scheme, we rely on a hill climbing search procedure. The initial parameter combination for the search is chosen by a quick scan of the parameter space (see Supplementary Material available on Dryad at https://doi.org/10.5061/dryad.p069231). Next, the likelihood of the initial parameter combination is computed. Each parameter combination has "immediate neighbors," which share exactly two out of three of its parameter values and have a minimal distance from its third parameter. For example, one of the neighbors of the combination $(r = 0.05, \mu = 0.01, t = 0.34)$ is $(r = 0.05, \mu = 0.02, t = 0.34)$. The likelihood of each of the immediate neighbors of the initial parameter combination is computed. If one of the neighbors has higher likelihood, it replaces the initial parameter combination and its immediate neighbors are examined. The replacement of a combination by a better neighbor continues until no improvement in likelihood is achieved.

RESULTS

Reconstruction of simulation parameters

Using the SimBa-SAl simulator and a JTT substitution process (Jones et al. 1992), we generated a set of 540 true PWAs under 27 parameter combinations of the long-indel model (see Materials and Methods). We applied the SimBa-SAl inference procedure with the chop-based acceleration and the initial-PWA optimization scheme to the 540 unaligned simulated sequence pairs. We first measured the ability of the SimBa-SAl inference procedure to identify the parameters that generated the sequences. The inference error of each of the three parameters was computed as $(\hat{p} - p)^2$, where p is the true parameter value and $\hat{p}$ is the value inferred by SimBa-SAl. Over all 540 cases, the average inference errors were 0.0011 ± 0.0009, 0.0057 ± 0.0048, and 0.026 ± 0.0281 for the μ, t, and r parameters, respectively. We next aimed to examine how parameter accuracy depends on the parameters used for simulating the PWAs (Fig. 2, Supplementary Fig. S1, available on Dryad). We found that the estimation error of all three parameters was considerably higher when the r value was high (0.6) rather than low or medium (0 or 0.3). For example, the average estimation error of μ in data sets with a low or medium r value is 0.0008, whereas it is 0.0018 for data sets with a high r value (for full details see Supplementary Table S1, available on Dryad). A high r value compared with a low r value combined with the same t and μ parameters means fewer indel events. Thus, data sets with a high r value contain less indels, making it harder to infer the parameters behind them.

Comparison to Gotoh and MAFFT pairwise procedures

We next set to evaluate the accuracy of the ML-PWAs inferred with SimBa-SAl. To this end, we measured the alignment accuracy by computing the column score of the ML-PWA with respect to its corresponding true PWA (see Materials and Methods). We compared these scores to the measured scores for PWAs computed by two commonly used PWA methods: the Gotoh algorithm as implemented in the package EMBOSS (Rice et al. 2000), and MAFFT PWA procedure (Katoh and Standley 2013). Gotoh and MAFFT were run with

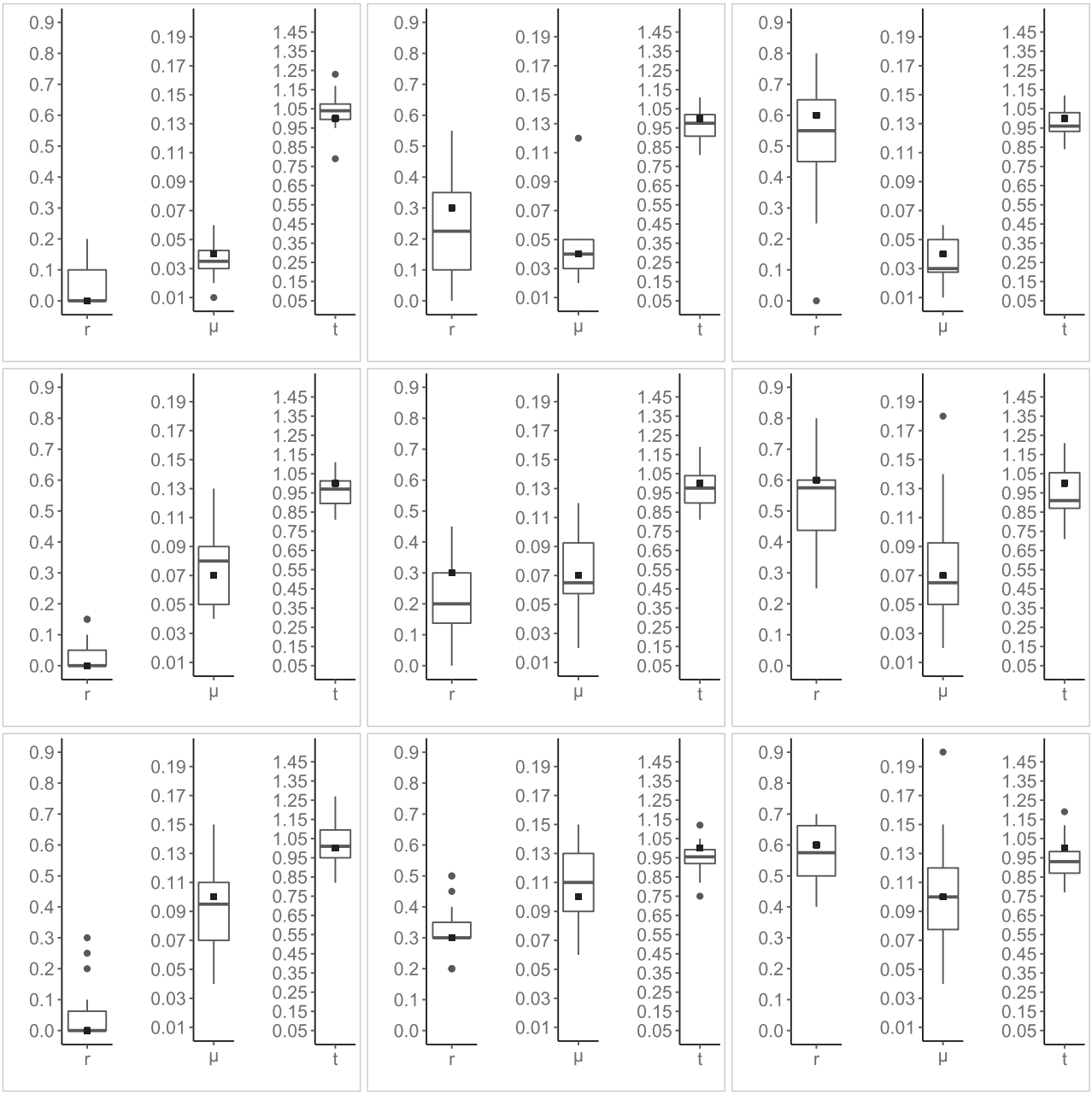

FIGURE 2. Parameter reconstruction. Five hundred and forty simulated PWAs were generated by considering 27 parameter combinations of the long-indel model. The unaligned sequences of each PWA were given to the SimBa-SAl inference procedure and the parameters inferred for each dataset were compared with the true ones (full square). In each plot, the three y axes reflect the possible range of parameter values. The figure depicts the accuracy of parameter estimation for nine combinations in which $t = 1$. All combinations are presented in Supplementary Fig. S1, available on Dryad.

default parameters (see Materials and Methods). Over all parameter combinations, the average column scores were 0.907, 0.908, and 0.929 for Gotoh, MAFFT, and SimBa-SAl, respectively. The difference between Gotoh and MAFFT was not statistically significant (Wilcoxon paired test $P > 0.23$), whereas the difference between SimBa-SAl and MAFFT was (Wilcoxon paired test $P < 2.2e^{-16}$, Fig. 3). We next tested whether the increased accuracy is limited to a specific set of model parameters. Our analysis clearly shows that this is not the case: the increased accuracy of the SimBa-SAl inference procedure was observed across most sets of individual parameter combinations (Supplementary Fig. S2, available on Dryad).

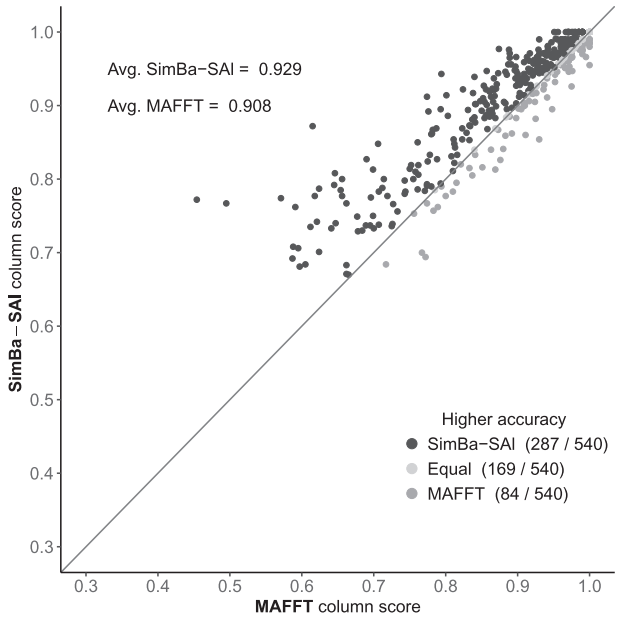

FIGURE 3. Inference accuracy of long-indel model PWAs. Five hundred and forty simulated PWAs were generated by considering 27 parameter combinations of the long-indel model. The unaligned sequences of each PWA were given as input to MAFFT and SimBa-SAl. The accuracy of SimBa-SAl is higher than that of MAFFT. "Avg." indicates the average column score.

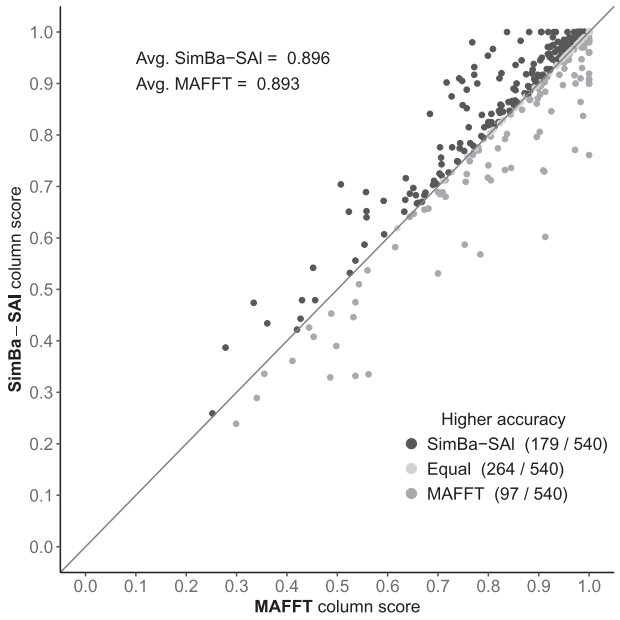

FIGURE 4. Inference accuracy of INDELible PWAs. Five hundred and forty PWAs were generated by considering 27 parameter combinations of the INDELible model. The unaligned sequences of each PWA were given as input to MAFFT and SimBa-SAl. The accuracy of SimBa-SAl is higher than that of MAFFT.

Robustness to model misspecification

Next, we measured the effect of model misspecification on the SimBa-SAl inference procedure. To this end, we used INDELible (Fletcher and Yang 2009) to generate PWAs under an alternative indel formation model. The INDELible model is controlled by two parameters: *IR* and *A*. The *IR* parameter is the rate of indel events relative to a substitution rate of 1. The *A* parameter is the shape parameter of a power law distribution from which the size of events is drawn. Of note, the *IR* parameter is not equivalent to the μ parameter of the long-indel model. While the μ parameter of the long-indel model refers to the total deletion rate per position by an event of any size, the *IR* parameter reflects the rate of indel events, which start at a given position. Moreover, the INDELible model is a misspecification of the long-indel model with respect to the following model choices: 1) the function of indel lengths; 2) as opposed to the long-indel model, the rate of insertions is assumed to be equal the rate of deletions, which also means the INDELible model is non-reversible. We generated 20 true PWAs from each of 27 INDELible parameter combinations and a JTT substitution process (see Materials and Methods). These true PWAs were used to test the performance and accuracy of Gotoh, MAFFT and SimBa-SAl. Over all parameter combinations, the average column scores were 0.888, 0.893, and 0.896 for Gotoh, MAFFT, and SimBa-SAl, respectively. The difference between SimBa-SAl and MAFFT was statistically significant (Wilcoxon paired test $P < 0.0074$, Fig. 4). We further repeated these analyses for each set of parameters separately and the same trend was observed (Supplementary Fig. S3, available on Dryad). These results suggest that the long-indel model is robust to moderate model misspecifications. Finally, as can be seen in the lower average column scores, the INDELible test data sets pose a slightly harder challenge for the alignment inference methods than the long-indel test data sets.

Alignment accuracy measured on the HOMSTRAD 2017 database

We downloaded 630 structurally-derived protein PWAs from the HOMSTRAD 2017 database (accessed: 28 December 2017; Mizuguchi et al. 1998; Stebbings and Mizuguchi 2004). For each of the 630 PWAs we first measured the longest stretch of consecutive gap characters. We filtered out 460 PWAs in which the longest stretch was found at one of the alignment ends, as these cases are suspected for non-homogenous rate of indel events along the sequences. We then ran Gotoh, MAFFT, and SimBa-SAl on the unaligned sequences of the 170 retained data sets. With average column scores of 0.742, 0.753, and 0.75 for Gotoh, MAFFT, and SimBa-SAl, respectively, the differences in the measured accuracies of the three methods were not statistically significant (Wilcoxon paired test $P > 0.25$ for all three comparisons). We next reanalyzed the HOMSTRAD 2017 data sets using SimBa-SAl, this time sampling 100 PWAs according to their probability. In 33 of the 170 cases, the true PWA was found in the sample and in eight out of these 33 the ML-PWA was not the true PWA. Of note, the column scores computed for the HOMSTRAD 2017

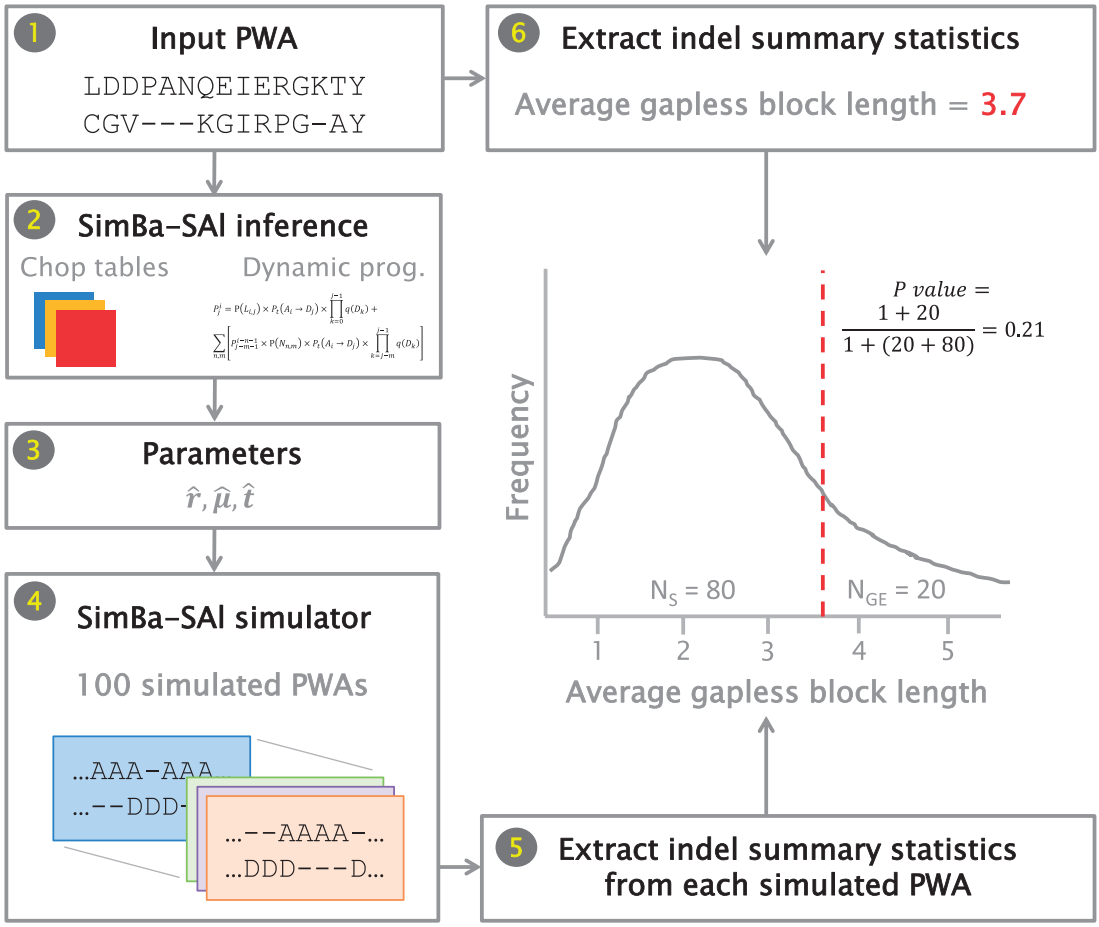

FIGURE 5. Model fitting testing scheme. Model parameters are inferred from an input PWA by SimBa-SAI (Stages 1, 2, and 3) and 100 simulated PWAs are generated under those parameters (Stage 4). The same summary statistic is extracted from the simulated PWAs (Stage 5) and from the input PWA (Stage 6). The number of simulations in which the summary statistic is greater than or equal to the value computed based on the input PWA (N_{GE}) and the number of simulations in which it is smaller (N_S) are used to compute an empirical P value for the input PWA.

database are substantially lower than those obtained when analyzing simulated data sets, suggesting that the indel dynamics characterizing HOMSTRAD 2017 alignments are more complex than the indel dynamics generated by both INDELible and the long-indel model.

Model fitting

To investigate the reason for the lower scores computed for the HOMSTRAD 2017 database, we aimed to better characterize how well the long-indel model reflects indel dynamics in nature. In case the model captures the essence of the dynamics, it is expected that summary statistics concerning indel formation extracted from an input biological PWA and from a set of corresponding PWAs simulated under the model parameters that best fit this biological PWA should be similar. Here, we focused on the following three summary statistics: 1) the average length of a block of positions without any gaps ("average gapless block length"), 2) the longest gapless block, and 3) the ratio of gap to non-gap characters in the alignment. Our test of model fitting is designed to assign an empirical P value to each examined input PWA with respect to each of the three summary statistics by comparing it to 100 corresponding PWAs simulated under the model. The stages of the test are depicted in Figure 5.

We first focused on the long-indel test PWAs. These data sets were simulated under the long-indel model and thus, the model should fit them. Specifically, when considering a summary statistic extracted from each input PWA, we expect its value to come from the same distribution of values as its 100 corresponding simulated PWAs. In other words, we expect the distribution of empirical P values to be uniform between 0 and 1. Focusing on the long-indel test data sets allowed us to examine the impact of the SimBa-SAI parameter inference procedure. We first measured the fit of the model when the corresponding simulated PWAs were generated using the same parameters behind each of the input PWAs (skipping stage 2 in Fig. 5). As expected, we found that the distribution of the 540 empirical P values was uniform between 0 and 1 for all three summary

statistics (Fig. 6A). We next examined the distributions of empirical P values when the computation relied on simulated PWAs based on the long-indel model parameters inferred by SimBa-SAl. Here, we found slight deviations from the expected uniform distribution (Fig. 6B).

We next examined the fit of the TKF91 and the long-indel models to the 170 data sets of the HOMSTRAD 2017 database. Of note, TKF91 is a private case of the long-indel model in which $r=0$. We thus measured the fit of this model by running the SimBa-SAl inference procedure on each input of the HOMSTRAD 2017 database with chop tables in which $r=0$ (2,920 tables out of all 55,480 tables of the long-indel model). For the long-indel model, we found strong deviations from the uniform distribution for all three summary statistics (Fig. 6C). Specifically, 32% and 21% of the empirical P values associated with the average gapless block length and longest gapless block, were smaller than 0.05, respectively. In addition, 41% of the empirical P values associated with the ratio of gap to non-gap characters were greater than 0.94. Together, these deviations suggest that the long-indel model tends to over-align the HOMSTRAD PWAs. For the TKF91 model, the deviations were even stronger (Fig. 6D). Specifically, 63% and 55% of the empirical P values associated with the average gapless block length and longest gapless block, were smaller than 0.05, respectively, and 53% of the empirical P values associated with the ratio of gap to non-gap characters were greater than 0.94. This suggests that not only the TKF91 model over-aligns the HOMSTRAD PWAs, the insertion or deletion of a single character at a time makes it deviate even more from the indel dynamics observed in real biological data sets. Put together, these results concerning the long-indel and TKF91 models suggest there is still much room for improvement in the modeling of indel dynamics in biological sequence data (see Discussion).

The unrealistic indel patterns of the TKF91 model are demonstrated in the analysis of the HOMSTRAD GAD (family: GatB and aspartyl tRNA synthetases) domain. Figure 7 depicts the HOMSTRAD true GAD PWA as well as the ML-PWA estimates based on either the long indel or the TKF91 models. Clearly, in this example, the ML-PWA under the long-indel model is closer to the HOMSTRAD PWA than the ML-PWA under the TKF91 model. Relative to the real GAD PWA, the TKF91 ML-PWA is characterized by an overabundance of very short indels, which result in a shorter average gapless block, a shorter maximal gapless block and a higher ratio of gap to non-gap characters. In contrast, the ML-PWA under the long-indel model, although different from the true PWA, is characterized by indel summary statistics that resemble those of the true PWA.

Running times

The SimBa-SAl parameter optimization procedure used throughout this work saves costly dynamic programming computations. The average number of dynamic programming procedures performed in the hill climbing search was 19.4 ± 14.9 on the long-indel test data sets, 19.8 ± 14.4 on the INDELible test data sets, and 46.7 ± 42 on the HOMSTRAD 2017 database. On average, a single dynamic programming procedure, which uses the chop-acceleration method, took 4.9 ± 4.1 s on the long-indel test data sets, 19.9 ± 18.4 s on the INDELible test data sets, and 48.6 ± 107.3 s on the HOMSTRAD 2017 database. All computations were carried out on a computer cluster where each node has 2 2.50GHz Intel(R) Xeon(R) CPUs with 64GB shared RAM. Each of the two CPUs has 20 cores.

Discussion

In this work we present SimBa-SAl, a simulation-based method for statistical alignment. Analysis with SimBa-SAl is performed in two stages. First, chop probabilities under a specific model and its parameters are estimated using a simulation. The tabulated results of this one-time computation are used in all subsequent analyses. Second, using the precomputed chop probabilities, PWA estimates are inferred using dynamic programming procedures as listed in Table 1. Dividing the analysis into these two stages relies on the independence of the indel process from the substitution process as well as on the decomposition of an alignment to independent chop-associated segments. Beyond these assumptions, the SimBa-SAl approach offers great flexibility in the choice of a generative model for indel formation utilized in the first stage. In this study, we chose to focus on the long-indel model. One of the major limitations of inference with this model is the long running times under a naïve implementation, as also noted by Miklós et al. (2004). We proposed two accelerations to the algorithm in addition to decoupling chop probability computations from the inference procedure. The first acceleration is based on excluding N chops from consideration in the dynamic programming procedure (Eq. 6). In such an acceleration, a chop should be excluded from computation if its probability is small enough. However, defining a criterion for "small enough" is not straight-forward. The SimBa-SAl approach offers a natural interpretation of this criterion by excluding chops that were not observed in simulations. The second acceleration saves dynamic programming procedures with respect to parameter choice. We show the utility in a hill climbing optimization procedure to select a parameter combination among a large set of precomputed chop tables. Of note, both these accelerations are applied in the second stage of the SimBa-SAl method (the inference procedure), indifferent to the model that generated the chop tables.

Our analyses of the performance of the SimBa-SAl method focused on accuracies measured with respect to the ML-PWA. This allowed us to compare SimBa-SAl to Gotoh and MAFFT, which produce point estimates of

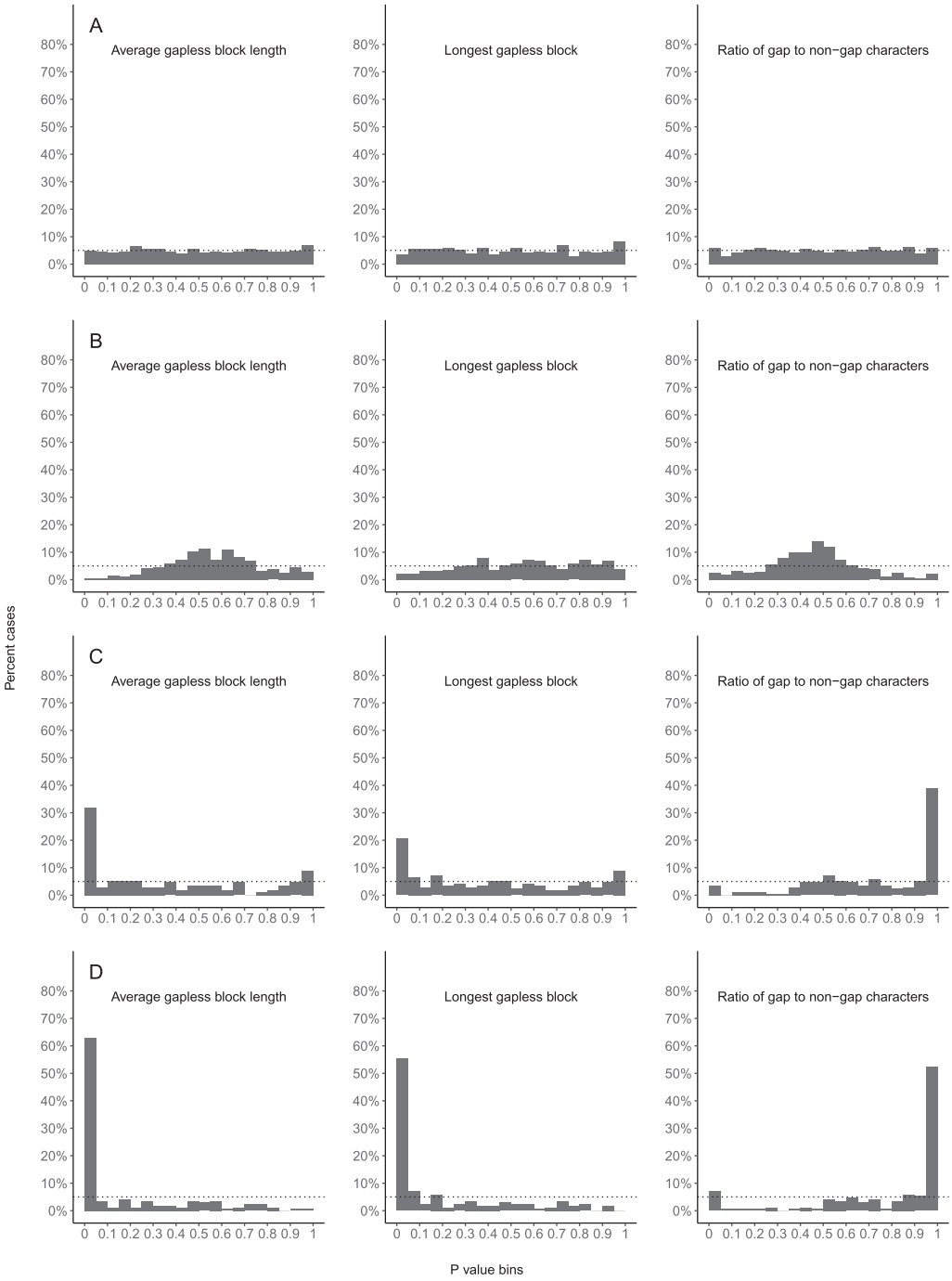

FIGURE 6. The fit of the long-indel and TKF91 models. The empirical *P* values computed for a collection of datasets with respect to each of the three summary statistics were divided into 20 equally-sized bins between 0 and 1. The dashed line indicates the expected 5% of *P* values in each bin. When the input PWAs are from the long-indel model and each corresponding set of PWAs is simulated under the same parameters as the input, the empirical *P* values follow a uniform distribution between 0 and 1 (A). When the input PWAs are from the long-indel model and each corresponding set of PWAs is simulated under the parameters inferred by SimBa-SAl, the empirical *P* values exhibit slight deviations from a uniform distribution between 0 and 1 (B). When the input PWAs are from the HOMSTRAD 2017 database and each corresponding set of PWAs is simulated under the parameters inferred by SimBa-SAl under the long-indel model, the empirical *P* values exhibit strong deviations from a uniform distribution between 0 and 1 (C). When the input PWAs are from the HOMSTRAD 2017 database and each corresponding set of PWAs is simulated under the parameters inferred by SimBa-SAl using only the TKF91 model, the empirical *P* values exhibit the strongest deviations from a uniform distribution between 0 and 1 (D).

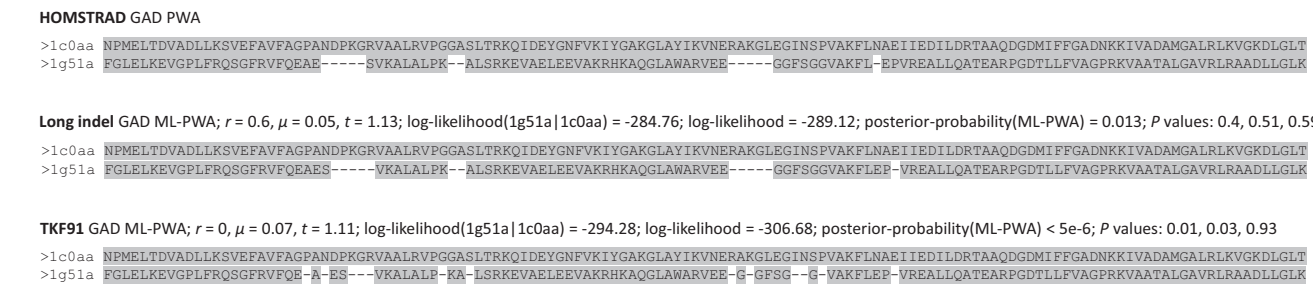

FIGURE 7. The fit of the long-indel and TKF91 models to the GAD domain. The HOMSTRAD true PWA, SimBa-SAl long-indel ML-PWA, and SimBa-SAl TKF91 ML-PWA are presented one above the other. For clarity of indel patterns, non-gaps are presented with a gray background. Three summary statistics were examined: 1) the average gapless block length, 2) the longest gapless block, and 3) the ratio of gap to non-gap characters in the alignment. The estimated parameter values as well as the empirical *P* values for each of the three summary statistics are presented above the SimBa-SAl PWAs. The log-likelihood of each alignment as well as the log of the total probability of the descendant conditioned on the ancestor were computed as detailed in Table 1 and in the section "Inference using pre-computed chop tables."

the alignment. We found that SimBa-SAl with the long-indel model offered greater-than or equal-to accuracy on simulated data sets as well as on the HOMSTRAD 2017 database.

Unlike Gotoh and MAFFT, SimBa-SAl has the ability to account for alignment uncertainty. Several software that account for alignment uncertainty were previously developed: Baliphy (Redelings et al. 2005), StatAlign (Novák et al. 2008), and FSA (Bradley et al. 2009). Baliphy assumes a phylogenetic tree of the analyzed sequences, where each branch is associated with its own HMM. In this formulation, indel formation is not based on an explicit stochastic indel process. StatAlign is based on TKF92 and thus poses its limitations (see Introduction). FSA is based on pair HMMs to estimate posterior probabilities for matches and indels between each pair of sequences. An annealing step follows to combine these probabilities to form an MSA. Thus, although these programs explicitly account for alignment uncertainty, they do not analyze substitutions and indels in a unified manner as a continuous time stochastic process along a phylogenetic tree. One way in which SimBa-SAl can be used to account for uncertainty is to consider a sample of PWAs, drawn based on their probabilities, rather than a single point estimate. When analyzing the HOMSTRAD 2017 database, we showed that in certain cases, such a sample contains the true PWA even when the true PWA is not the ML-PWA. The importance of sampling becomes even clearer when examining the posterior probability of any single point estimate. In the GAD example presented in Figure 7, we show that the ML-PWAs (computed either under the long-indel or the TKF91 model) have low posterior probabilities (<0.013 in both cases), which demonstrates the need to account for alignment uncertainty. Furthermore, a sample of PWAs could then be used for various applications, such as constructing better protein structure models and homology testing.

When we examined model fitting, we found that although the long-indel model fits the HOMSTRAD 2017 database better than the TKF91 model, there is still room for improvement when it comes to constructing realistic models of indel formation. The modularity of SimBa-SAl should allow the integration of various indel formation models, which relax some of the assumptions of the long-indel model. The first of these assumptions is time reversibility of the indel process. This assumption, which is often the result of mathematical convenience considerations, is not necessarily realistic with respect to the evolutionary process of biological sequences. Specifically, this assumption implies the same equilibrium length distribution for all analyzed sequences. Moreover, it implies a detailed balance in which deletions are the mirror image of insertions, that is, the rate of inserting k characters times the stationary probability of length n is equal to the rate of deleting k characters times the stationary probability of length $(n+k)$. Rivas and Eddy (2008) have proposed a non-reversible Markov model over an alphabet where a gap is an additional character. However, their model does not allow for long indels. Using the framework developed in this work, it is possible to extend the long-indel model to a non-reversible version by assuming independent parameters for the insertion and deletion rates. A second assumption concerns the description of indel lengths using a geometric function, which might not be realistic (Chang and Benner 2004; Ezawa 2016). Because SimBa-SAl uses simulations to evaluate chop probabilities, and because given the chop probabilities, the computation is independent of the indel length distribution, it is relatively straight-forward to explore different indel length functions. Another strong assumption of the long-indel model is that indels occur uniformly along the sequence. One possible extension can include simulations and inferences under indel-rich and indel-poor zones. Finally, the current indel model assumes that indels are context-independent, which is clearly false (Benner et al. 1993). However, relaxing this assumption is more challenging as it would require introducing dependencies between the substitution and indel processes, which would complicate the decomposition of a PWA into chop-associated segments. This would, in turn, result in high computation times for statistical alignments.

SimBa-SAl is designed to compute pairwise statistical alignments. Extending it to analyze multiple sequences is a laudable, albeit challenging, goal. It will require the definition of a chop with respect to multiple sequences or alternatively, the recursive construction of pairwise chops with respect to their mutual ancestor (as developed by Steel and Hein 2001 for the TKF91 model). Accelerations, such as the chop-based acceleration will be of great importance in order to maintain reasonable running times.

Supplementary Material

Data available from the Dryad Digital Repository: http://dx.doi.org/10.5061/dryad.p069231.

Funding

This study was supported by an Israel Science Foundation grant 802/16 to T.P. While conducting this research, J.H. was a visiting scholar of the Edmond J. Safra Center for Bioinformatics at Tel-Aviv University. E.L.K. is a recipient of a FEBS long-term fellowship and an EMBO non-stipendiary long-term fellowship.

Acknowledgments

The authors thank the Associate Editor and Jeff Thorne for their helpful comments and suggestions.

References

Arenas M. 2015. Trends in substitution models of molecular evolution. Front. Genet. 6:319.
Baldi P., Chauvin Y., Hunkapiller T., McClure M. 1994. Hidden Markov models of biological primary sequence information. Proc. Natl. Acad. Sci. U. S. A. 91:1059–1063.
Benner S.A., Cohen M.A., Gonnet G.H. 1993. Empirical and structural models for insertions and deletions in the divergent evolution of proteins. J. Mol. Biol. 229:1065–1082.
Bishop M.J., Thompson E.A. 1986. Maximum likelihood alignment of DNA sequences. J. Mol. Biol. 190:159–165.
Bradley R.K., Roberts A., Smoot M., Juvekar S., Do J., Dewey C., Holmes I., Pachter L. 2009. Fast statistical alignment. PLoS Comput. Biol. 5(5): e1000392.
Cartwright R.A. 2005. DNA assembly with gaps (Dawg): simulating sequence evolution. Bioinformatics 21:iii31–iii38.
Chang M.S.S., Benner S.A. 2004. Empirical analysis of protein insertions and deletions determining parameters for the correct placement of gaps in protein sequence alignments. J. Mol. Biol. 341:617–631.
Chatzou M., Magis C., Chang J.-M., Kemena C., Bussotti G., Erb I., Notredame C. 2016. Multiple sequence alignment modeling: methods and applications. Brief. Bioinform. 17:1009–1023.
Chen J.-Q., Wu Y., Yang H., Bergelson J., Kreitman M., Tian D. 2009. Variation in the ratio of nucleotide substitution and indel rates across genomes in mammals and bacteria. Mol. Biol. Evol. 26:1523–1531.
Ezawa K. 2016. General continuous-time Markov model of sequence evolution via insertions/deletions: are alignment probabilities factorable? BMC Bioinformatics. 17:304.
Fletcher W., Yang Z. 2009. INDELible: a flexible simulator of biological sequence evolution. Mol. Biol. Evol. 26:1879–1888.
Gotoh O. 1982. An improved algorithm for matching biological sequences. J. Mol. Biol. 162:705–708.
Graur D. 2016. Molecular and genome evolution. Oxford: Oxford University Press.
Gusfield D. 1997. Multiple string comparison—the holy grail. In: Algorithms on strings, trees, and sequences. Cambridge: Cambridge University Press. p. 332–369.
Hamilton M.B., Braverman J.M., Soria-Hernanz D.F. 2003. Patterns and relative rates of nucleotide and insertion/deletion evolution at six chloroplast intergenic regions in new world species of the Lecythidaceae. Mol. Biol. Evol. 20:1710–1721.
Hein J. 2001. An algorithm for statistical alignment of sequences related by a binary tree. Pac. Symp. Biocomput. 2001:179–190.
Henikoff S., Henikoff J.G. 1992. Amino acid substitution matrices from protein blocks. Proc. Natl. Acad. Sci. U. S. A. 89:10915–10919.
Hirschberg D.S. 1975. A linear space algorithm for computing maximal common subsequences. Commun. ACM. 18:341–343.
Holmes I. 2003. Using guide trees to construct multiple-sequence evolutionary HMMs. Bioinformatics. 19 Suppl 1:i147–i157.
Holmes I., Bruno W.J. 2001. Evolutionary HMMs: a Bayesian approach to multiple alignment. Bioinformatics. 17:803–820.
Holmes I.H. 2017. Solving the master equation for Indels. BMC Bioinformatics. 18:255.
Iantorno S., Gori K., Goldman N., Gil M., Dessimoz C. 2014. Who watches the watchmen? An appraisal of benchmarks for multiple sequence alignment. Methods Mol Biol. 2014;1079:59–73.
Jones D.T., Taylor W.R., Thornton J.M. 1992. The rapid generation of mutation data matrices from protein sequences. Bioinformatics. 8:275–282.
Katoh K., Standley D.M. 2013. MAFFT multiple sequence alignment software version 7: improvements in performance and usability. Mol. Biol. Evol. 30:772–780.
Kemena C., Notredame C. 2009. Upcoming challenges for multiple sequence alignment methods in the high-throughput era. Bioinformatics. 25:2455–2465.
Krogh A., Brown M., Mian I.S., Sjolander K., Haussler D. 1994. Hidden Markov models in computational biology: applications to protein modeling. J. Mol. Biol. 235:1501–1531.
Levy Karin E., Shkedy D., Ashkenazy H., Cartwright R.A., Pupko T. 2017. Inferring rates and length-distributions of indels using approximate Bayesian computation. Genome Biol. Evol. 9:1280–1294.
Lunter G. 2007. Probabilistic whole-genome alignments reveal high indel rates in the human and mouse genomes. Bioinformatics 23:i289–i296.
Lunter G.A., Miklós I., Song Y.S., Hein J. 2003. An efficient algorithm for statistical multiple alignment on arbitrary phylogenetic trees. J. Comput. Biol. 10:869–889.
Miklós I., Lunter G.A., Holmes I. 2004. A "Long Indel" model for evolutionary sequence alignment. Mol. Biol. Evol. 21:529–540.
Mizuguchi K., Deane C.M., Blundell T.L., Overington J.P. 1998. HOMSTRAD: A database of protein structure alignments for homologous families. Protein Sci. 7:2469–2471.
Myers E.W. 1986. An O(ND) difference algorithm and its variations. Algorithmica. 1:251–266.
Needleman S.B., Wunsch C.D. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol. 48:443–453.
Novák Á., Miklós I., Lyngsø R., Hein J. 2008. StatAlign: an extendable software package for joint Bayesian estimation of alignments and evolutionary trees. Bioinformatics. 24:2403–2404.
Ophir R., Graur D. 1997. Patterns and rates of indel evolution in processed pseudogenes from humans and murids. Gene 205:191–202.
Redelings B.D., Suchard M.A., Lewis P. 2005. Joint Bayesian estimation of alignment and phylogeny. Syst. Biol. 54:401–418.
Rice P., Longden I., Bleasby A. 2000. EMBOSS: the european molecular biology open software suite. Trends Genet. 16:276–277.
Rivas E., Eddy S.R. 2008. Probabilistic phylogenetic inference with insertions and deletions. PLoS Comput Biol 4(9): e1000172.
Sankoff D. 1972. Matching sequences under deletion-insertion constraints. Proc. Natl. Acad. Sci. U. S. A. 69:4–6.
Satija R., Novák Á., Miklós I., Lyngsø R., Hein J. 2009. BigFoot: Bayesian alignment and phylogenetic footprinting with MCMC. BMC Evol. Biol. 9:217.

Sellers P.H. 1974. An algorithm for the distance between two finite sequences. J. Comb. Theory, Ser. A 16:253–258.
Sievers F., Wilm A., Dineen D., Gibson T.J., Karplus K., Li W., Lopez R., McWilliam H., Remmert M., Söding J., Thompson J.D., Higgins D.G. 2011. Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. Mol. Syst. Biol. 7:539.
Sipos B., Massingham T., Jordan G.E., Goldman N. (2011) PhyloSim - Monte Carlo simulation of sequence evolution in the R statistical computing environment - BMC Bioinformatics 12:104.
Stebbings L.A., Mizuguchi K. 2004. HOMSTRAD: recent developments of the homologous protein structure alignment database. Nucleic Acids Res. 32:203D–207.
Steel M., Hein J. 2001. Applying the Thorne-Kishino-Felsenstein model to sequence evolution on a star-shaped tree. Appl. Math. Lett. 14:679–684.
Stoye J., Evers D., Meyer F. 1998. Rose: generating sequence families. Bioinformatics 14:157–163.
Thorne J.L., Kishino H., Felsenstein J. 1991. An evolutionary model for maximum likelihood alignment of DNA sequences. J. Mol. Evol. 33:114–124.
Thorne J.L., Kishino H., Felsenstein J. 1992. Inching toward reality: an improved likelihood model of sequence evolution. J. Mol. Evol. 34:3–16.
Ukkonen E. 1985. Algorithms for approximate string matching. Inf. Control. 64:100–118.
Vingron M., Waterman M.S. 1994. Sequence alignment and penalty choice. Review of concepts, case studies and implications. J. Mol. Biol. 235:1–12.
Wagner R.A., Fischer M.J. 1974. The string-to-string correction problem. J. ACM. 21:168–173.
Waterman M.S., Smith T.F., Beyer W.A. 1976. Some biological sequence metrics. Adv. Math. (N. Y). 20:367–387.
Yang Z. 1996. Phylogenetic analysis using parsimony and likelihood methods. J. Mol. Evol. 42:294–307.
Yang Z. 2014. Molecular evolution: a statistical approach. Oxford: Oxford University Press.