Efficient and Robust Global Amino Acid Sequence Alignment with Uncertain Evolutionary Distance

Matthias C. M. Troffaes
Universiteit Gent, SYSTeMS Research Group
Technologiepark Zwijnaarde 914, 9052 Zwijnaarde, Belgium
Matthias.Troffaes@UGent.be

Abstract

When aligning genetic sequences, we have to rely on estimates of evolutionary distance between sequences and their closest common ancestor. In practice, many alignments are performed on short sequences, and unfortunately, for such sequences it is well-known that estimation of evolutionary distance is subject to serious errors. Without additional information about the sequences, it is hardly possible to improve existing estimators. This paper addresses how imprecise probability theory allows us to substantially weaken assumptions about the evolutionary distance, by using an interval rather than a point estimate. It is shown how under these weaker assumptions a good alignment still can be found, through a generalisation of the well-known Needleman-Wunsch algorithm. In doing so, we rely on an extension of dynamic programming to the case where the gain is described by an imprecise probability model. Our approach also identifies those cases in which insufficient information is available in order to construct a good alignment.

Keywords: bioinformatics, imprecise probabilities, Needleman-Wunsch algorithm, substitution matrix, molecular evolution, dynamic programming, sensitivity analysis

1 Introduction

Aligning genetic sequences is a very widely used and important technique in bioinformatics [6]. To give a few examples, through sequence alignment we can determine evolutionary relationships among species, and in particular, we can reconstruct phylogenetic trees. An alignment may also reveal functional regions in genetic sequences. Such information may for example lead to the discovery of new or improved drug treatments, or may help in deciding what treatment is best fitted for a particular patient genotype. Sequence alignment is also a handy tool in predicting structural and biochemical properties of sequences.

The alignment problem is usually formulated as an *optimisation problem*. Basically, positive scores are assigned to matches, and negative scores are assigned to mismatches and gaps. These scores are summarised in what is called a *score matrix*. We aim to find the alignment with the highest total score. This approach has two benefits: (i) it allows us to characterise the optimal ("best") alignment from all possible alignments in an objective way, and (ii) the highest score, corresponding to the best alignment, provides us with an objective measure of the quality of this alignment. Moreover, an efficient algorithm to calculate the optimal alignment of a small number of sequences (say, two or three sequences) can be constructed through dynamic programming [7]. In this article, we will focus on *pair-wise* sequence alignment, that is, the alignment of only two sequences.

Clearly, aligning genetic sequences relies heavily on the choice of the score matrix: how should we reward matches, and how should punish gaps and mismatches? In practice, a large number of score matrices are being used, and precise choice of the score matrix relies on additional assumptions about the sequences under study. For example, when using PAM score matrices [3], on which we will focus in this paper, the following assumptions are made:[1]

- the evolutionary distance of the sequences to their closest common ancestor is known,
- evolution is in an equilibrium point,
- in this equilibrium point, there is evolutionary reversibility—any point mutation is as probable as its reverse,
- point mutations at different locations in the sequence are i.i.d., and
- point mutations at different times are i.i.d.

Different evolutionary distances induce different score matrices. These matrices are denoted by PAM(T), where T denotes the evolutionary distance between the sequences under study and their closest common ancestor.

Obtaining this evolutionary distance is a major issue in molecular evolution, especially when comparing short sequences. Indeed, 'estimation bias usually occurs when the sequence length is short so that stochastic effects are strong' [14]. In many cases, one can only rely on the sequences under study to estimate evolutionary distance—no additional information is available.

One approach is somehow to guess the evolutionary distance from the similarity of the two sequences. Typically, PAM250 is chosen if the sequences are 20% similar, PAM120 if they are 40% similar, PAM60 if they are 60% similar, etc. It is however not entirely clear how in general similarity percentages can be derived from two sequences, prior to alignment.

Another approach to solve the optimisation problem not for one, but for a set of PAM matrices, or even with different other methods, and then choose the method that returns the highest optimal score. The performance of different alignment methods has been studied, and one of the interesting results that have come out of such studies is that 'for different pairs many different methods create the best alignments', and hence, that 'if a method that could select the best alignment method for each pair existed, a significant improvement of the alignment quality could be gained' [5]. However, in practice it is computationally unfeasible to try out a large numbers of methods and to tune all parameters (such as evolutionary distance, gap penalty, etc.) for each one of them.

In this paper, it is investigated whether a bias in the evolutionary distance also leads to a bias in the optimal alignment. In particular, a generalisation of the well-known Needleman-Wunsch algorithm [7] is proposed in order to determine whether an alignment, or parts of it, are insensitive to the evolutionary distance in an interval. In order to do so, we rely on an extension of the dynamic programming formalism to the case where the gain is described by an imprecise probability model [4].

The paper is organised as follows. Section 2 discusses the standard approach to amino acid sequence alignment, using score matrices, gap penalties and dynamic programming. In Section 3 we introduce and motivate a robustified notion of sequence alignment, based on a simple imprecise probability model for evolutionary distance. Section 4 deals with generalising the dynamical programming approach in order to determine a robust optimal alignment.

2 Optimal Sequence Alignment

2.1 What is Sequence Alignment?

A sequence alignment consists of writing two (or more) sequences in rows, and writing similar characters in the same column. In do-

[1] This is not meant to be the current state of the art. A lot of research in molecular evolution is on generalising these assumptions.

ing so, one is allowed to introduce so-called *gaps*, denoted by a dash '-' in either one of the sequences. Assuming that the sequences are derived from a common ancestor sequence, matches correspond to conserved regions, mismatches correspond to mutations and gaps correspond to deletions or insertions, briefly called *indels*, in either one of the sequences. Figure 1 gives an example of an amino acid alignment.

```
         X        10        20
H-alpha  V-LSPADKTNVKAAWGKVGAHAGEYGAEA
         | | |  |  | | ||||     | | ||
H-beta   VHLTPEEKSAVTALWGKV--NVDEVGGEA
         X       10          20
```

Figure 1: An extract from a possible alignment of hemoglobin alpha and beta chains [8].

It is convenient to represent alignments in a grid, as depicted in Figure 2. All paths from the upper left corner to the lower right corner represent possible alignments. The path drawn in Figure 2 corresponds to the alignment given in Figure 1. A diagonal move introduces no gaps, a downwards move introduces a gap in the upper sequence, a rightwards move introduces a gap in the lower sequence.

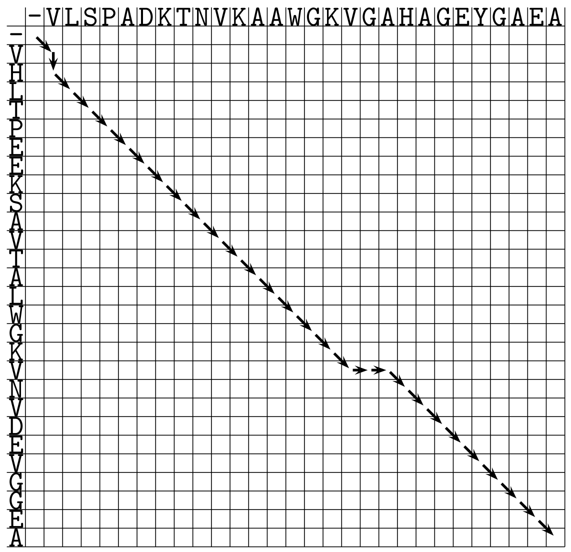

Figure 2: Alignments can be conveniently represented in a grid.

When trying to explain evolutionary relationships between sequences, we should identify the alignment that has the highest chance of being the result of an evolutionary process. That is, we try to explain the alignment as the result of evolution from a common ancestor.

We first show how evolutionary dynamics can be described on the level of genetic sequences. Then we show how a score matrix is obtained from these dynamics, and how the resulting optimisation problem indeed identifies the alignment that has the highest chance of being the result of evolution from a common ancestor.

2.2 Evolutionary Sequence Dynamics

The PAM ('point accepted mutation') matrices are widely accepted as the standard scoring system when looking for evolutionary relationships in protein sequences. They are related to the evolution of amino acid sequences described by a Markov model for amino acid substitution [3]. Indels, which introduce alignment gaps, are not modelled by PAM and are treated separately. We will only give a very brief description of the basic ideas underlying the dynamics. For a more extensive discussion and improvements of this approach the reader is referred to the references [3, 11, 1, 12].

Let $A_t(i)$ denote the amino acid at site i at (discrete) time t of a sequence of length N. It is first assumed that amino acids mutate independently at each site of the sequence. This implies that the probability of the sequence A_t to evolve to the sequence A_{t+s} is equal to

$$P[A_{t+s}|A_t] = \prod_{i=1}^{N} P[A_{t+s}(i)|A_t(i)]. \qquad (1)$$

Hence, it suffices to know only the probabilities $P[A_{t+s}(i)|A_t(i)]$ at each site i of the sequence. It is also assumed that amino acids mutate independently in time,

$$P[A_{t+s}(i)|A_t(i)] = \prod_{r=t}^{t+s-1} P[A_{r+1}(i)|A_r(i)]. \qquad (2)$$

We thus only need to know the probabilities $P[A_{r+1}(i)|A_r(i)]$ at each site i and time r.

Finally, assuming that the transition probabilities are identically distributed in time and space, $P[A_{r+1}(i)|A_r(i)]$ does not depend on the actual values of r and i, but only on the amino acids $A_r(i)$ and $A_{r+1}(i)$. Hence, if we know for any pair (a, b) of amino acids the probability $P[b|a]$ of a being substituted by b after one unit of time, then we also know the probability of any sequence A_t evolving to A_{t+s}, through Eqs. (1) and (2). Under the assumptions made so far, this establishes that we can model evolution of amino acid sequences through a Markov model.

It is convenient to assume that evolution from ancestors to descendants is modelled by the same Markov process as the evolution from descendants to ancestors, that is, that the Markov process is *time-reversible*. Assuming $P[b|a] > 0$ for all amino acid pairs (a, b), the Markov process attains a stationary distribution π after a sufficient long time. Moreover, π is independent of the initial distribution, and is the unique solution of

$$\sum_a P[b|a]\pi[a] = \pi[b]. \tag{3}$$

Assuming we attained this equilibrium, the process is time-reversible if and only if [10]

$$P[b|a]\pi[a] = P[a|b]\pi[b]. \tag{4}$$

Consider two amino acid sequences, B and C, that have evolved from a common ancestor A in t time units. Assuming time-reversibility, and assuming that all amino acids in A are i.i.d. according to the stationary distribution π, evolution from A to B and C in t time units is equivalent to evolution from B to A in t time units, and then from A to C in t time units. But this is equivalent to evolution from B to C in $2t$ time units. Hence, we can calculate the probability of B and C having evolved from a common ancestor in t time units simply by calculating the probability of C having evolved from B in $2t$ time units.

In practice, the transition probabilities $P[b|a]$ of the Markov model are estimated using a large dataset of sequences that are already aligned (originally, sequences from closely related species were considered, that is, sequences of at least 85% similarity). Many generalisations of this model have been developed, dropping stationarity of the transition probabilities, allowing different transition probabilities on different sites, etc.

2.3 A Log Likelihood Ratio Scoring

Using the Markov model for amino acid evolution, a scoring matrix is derived that has the interpretation of a log likelihood ratio. The entries of the matrix are roughly given by (up to a normalisation factor)

$$s_t(a, b) = \log \frac{L_{\text{evol}}[a, b](t)}{L_{\text{rand}}[a, b]}, \tag{5}$$

that is, the logarithm of the likelihood that a and b are aligned as a consequence of the evolutionary Markov process from a common ancestor t time units ago, divided by the likelihood that a and b are aligned 'by chance', that is, as a consequence of a multinomial process, where amino acid frequencies are obtained from the same data used to construct the Markov model. A positive score $s_t(a, b)$ means that a and b are more likely to be aligned by evolution than by chance, a negative score means the opposite. Remark that $s_t(a, b) = s_t(b, a)$.

To obtain a score for sequences, recall that we assumed different sites on sequences to be independent. Hence, the log likelihood ratio of two aligned sequences B and C—of equal length and without gaps—is obtained by adding the log likelihood ratios at each site of the sequences:

$$S_t(B, C) = \sum_{i=1}^{N} s_t(B(i), C(i)) \tag{6}$$

2.4 Gap Scoring

More general, let B be a sequence of length N, and let C be a sequence of length M. Consider any alignment u of B and C, and denote the characters (amino acids or gaps) at site i in the alignment by $B_u(i)$ and $C_u(i)$. The score of the alignment is given by

$$S_t(B,C)(u) = \sum_{i=1}^{K} s_t(B_u(i), C_u(i)), \qquad (7)$$

where K is the length of the alignment. If both $B_u(i)$ and $C_u(i)$ are amino acids, the $s_t(B_u(i), C_u(i))$ is given by the log likelihood ratio (Eq. (5)). If either one of them, say $B_u(i)$, is a gap, then the score is given by minus the *gap opening penalty* g if $B_u(i-1)$ is not a gap, and by minus the *gap extension penalty* r if $B_u(i-1)$ is a gap (g and r are positive).

2.5 Choice of Score Matrix and Gap Penalties

As argued before, the score for a pair of amino acids is given by Eq. (5). This score rewards alignments that are more likely by evolution than 'by chance', and punishes alignments that are less likely by evolution than 'by chance'.

Gap openings are less likely than gap extensions, and therefore the gap opening penalty g is chosen substantially higher than the gap extension penalty r. The gap penalties should also be chosen relative to the range of scores in the score matrix. If the gap penalty is too high, gaps will never appear in the optimal alignment. And if it is too low, too many gaps will appear in the optimal alignment.

Much research has been devoted to analysing how the score matrix and gap penalties should be chosen. The choice of the score matrix is based mainly on the evolutionary dynamical model and estimates of the evolutionary distance. Through statistical analysis, appropriate gap opening and extension penalties have been motivated for various score matrices (see for instance [9]).

One result is that a good choice for the score matrix, and consequently also a good choice for the gap penalties, can be made based on the evolutionary distance between sequences and their closest common ancestor.

2.6 Needleman-Wunsch Algorithm

Finding the optimal alignment is at first sight an extremely hard computational task. The number of possible alignments of two sequences of length N grows exponentially with N. Even for sequences of modest length, computing power is far from able to compare that many sequences in a reasonable amount of time.

Dynamic programming provides a method for exponentially reducing the number of alignments that need to be considered in order to find the optimal one [7]. Due to lack of space the original algorithm is not discussed here. A generalised version of the algorithm will be discussed in Section 4 further on.

3 An Imprecise Probability Model for Evolutionary Distance

In Section 2, it was argued that a good choice of the score matrix and the gap penalties can be made based on the evolutionary distance between the sequences under study and their closest common ancestor. Unfortunately, for short sequences, estimation of evolutionary distance is subject to serious bias due to stochastic effects [14]. Instead of somehow trying to improve evolutionary distance estimates between short sequences by reducing stochastic effects—this may well be impossible—we propose a different approach.

Instead, does a bias in the evolutionary distance also leads to a bias in the optimal alignment? Or, how sensitive is the alignment to the evolutionary distance? It is well-known that optimal alignment is quite sensitive to the choice of the score matrix, especially for long sequences [5]. But for short sequences, this does not need to be the case. To give an extreme example: if we would find that the optimal alignment is independent of the evolutionary distance, we also should not have to worry about it.

Recent developments in imprecise probability theory provide the perfect tool for performing such analysis. Let us briefly touch upon those results and apply them to the alignment problem.

Let $\mathcal{T} = \{t \in \mathbb{R} : t \geq 0\}$ be the space of possible evolutionary distances t between two sequences B and C and their closest common ancestor. Assume that the only information we have about $t \in \mathcal{T}$ is that it takes a value in the interval $[t_1, t_2]$, for some $t_1 \leq t_2$. In imprecise probability theory, this information can be used in order to construct a partial preference ordering on alignments [13]:

Definition 1 (Preference). *Let u and v be two alignments (of B and C). Then, u is said to be* strictly preferred to *v, and we write $u >_{[t_1,t_2]} v$, if*

$$\inf_{t \in [t_1,t_2]} [S_t(B,C)(u) - S_t(B,C)(v)] > 0. \quad (8)$$

If $u >_{[t_1,t_2]} v$ then there is an $\epsilon > 0$ such that $S_t(B,C)(u) > S_t(B,C)(v) + \epsilon$ for every $t \in [t_1, t_2]$. This means that, independently of the evolutionary distance in $[t_1, t_2]$, u is (uniformly) a strictly better alignment of B and C than v. In such a case, we should of course prefer u over v.

The optimisation problem can now also be restated. Usually, the partial order $>_{[t_1,t_2]}$ will not have a greatest element. Therefore, it makes more sense to look for undominated, or maximal elements.

Definition 2 (Maximality). *Say an alignment u is* maximal *with respect to $[t_1, t_2]$ if $v \not>_{[t_1,t_2]} u$, that is, if*

$$\sup_{t \in [t_1,t_2]} [S_t(B,C)(u) - S_t(B,C)(v)] \geq 0, \quad (9)$$

for all possible alignments v of B and C.

The idea behind this definition is that, if we do not prefer any other alignment v over u, then we should consider u as a good alignment candidate. The information we have does not allow us to make a better choice than u. An efficient algorithm for finding all maximal alignments will be given in Section 4. But let us first make a few important remarks.

Firstly, the notion of maximality generalises the classical notion of optimality. Indeed, if $t_1 = t_2 = t$ then any maximal alignment actually maximises the score $S_t(B,C)(v)$ over all possible alignments v.

Secondly, it is often argued that it is important to find *the* best alignment. But, when looking for maximal alignments, we do not obtain a single solution, but rather a set of solutions—perhaps even a pretty large set. At first sight, this may seem undesirable. Nevertheless, I believe even a set of best possible alignments can be useful:

- If we obtain a large set, then this simply means that we have insufficient information in order to construct the best alignment.
- We might be lucky and find that there is only one maximal alignment. If that is the case, we actually also know that this alignment is insensitive to assumptions made about evolutionary distance in the interval $[t_1, t_2]$.
- More generally, there may be certain constant patterns in the set of maximal alignments, i.e., it may happen that certain regions are consistently aligned over the whole set of maximal alignments. We then do not only know that these regions are optimally aligned, but also that they are insensitive to assumptions made about evolutionary distance in the interval $[t_1, t_2]$.

4 Finding Maximal Alignments Through Dynamic Programming

Recently, the algorithm of dynamic programming [2] has been generalised in order to find all maximal paths of a dynamical system, in case maximality is defined in the sense of Definition 2 [4]. We briefly discuss how the algorithm is implemented.

More general, let B be a sequence of length N, and let C be a sequence of length M. First, finding maximal alignments of B and C is restated in terms of finding the maximal paths of a dynamical system. This is done by interpreting alignments as paths of a dynamical system, and scores as gains associated with that path. Figure 2 shows how we can do that. The grid represents the state space. At each point in the grid we can move either rightwards, downwards or diagonal (except at the right and bottom borders). The gain associated with a move from position (i, j) if the previous move was p, is given by

$$G_t(i,j,p,\downarrow) = \begin{cases} r_t, & \text{if } p = \downarrow \\ g_t, & \text{otherwise} \end{cases}$$

$$G_t(i,j,p,\rightarrow) = \begin{cases} r_t, & \text{if } p = \rightarrow \\ g_t, & \text{otherwise} \end{cases}$$

$$G_t(i,j,p,\searrow) = S_t(B(i), C(j))$$

The gain associated with a path is simply given by the sum of the gains of each move.

The gain depends on the evolutionary distance t. Since the gain also depends on the previous move we must extend the state space with an additional state variable p at each point (i, j) in order to remember our previous move. Otherwise, we cannot apply the dynamical programming formalism.

Let $\mathcal{P}(i, j, p)$ denote the set of all paths from (i, j, p) to the right bottom corner. Observe that p denotes the previous move, $p \in \{\downarrow, \rightarrow, \searrow\}$, which is needed in order to calculate the gain (in order to tell the difference between a gap opening and a gap extension). Let $\mathcal{M}(i, j, p)$ denote the set of maximal paths from (i, j, p) to the bottom right corner, that is,

$$\mathcal{M}(i,j,p) = \max\nolimits_{>_{[t_1,t_2]}} \mathcal{P}(i,j,p) \qquad (10)$$

It is convenient to define $\mathcal{M}(i, j, p) = \emptyset$ whenever $i > N$ or $j > M$. Observe that $\mathcal{P}(i, j, p)$ is a finite set for every state (i, j, p). Hence, the compactness condition under which the generalised Bellman equation holds is trivially satisfied [4].

Theorem 1 (generalised Bellman equation). *For any state (i, j, p) the following equality holds:*

$$\begin{aligned} \mathcal{M}(i,j,p) = \max_{>_{[t_1,t_2]}} & \\ & (i,j,p;\downarrow) \oplus \mathcal{M}(i+1,j,\downarrow) \\ & \cup (i,j,p;\rightarrow) \oplus \mathcal{M}(i,j+1,\rightarrow) \\ & \cup (i,j,p;\searrow) \oplus \mathcal{M}(i+1,j+1,\searrow), \end{aligned} \qquad (11)$$

where $(i, j, p; \downarrow) \oplus \mathcal{M}(i+1, j, \downarrow)$ denotes the set of all concatenations of the downward move from state (i, j, p), with a maximal path from state $(i + 1, j, \downarrow)$, etc.

Eq. (11) yields an efficient recursive algorithm to calculate the set of all maximal paths $\mathcal{M}(0, 0, \searrow)$, and hence, all maximal alignments. It solves a global maximisation problem by solving $3MN$ smaller maximisation problems (see Figure 3).

```
** initialisation **
for p=|,-,\
  MAX(N,M,p)={(M,N,p)}
  for i=0 to N
    MAX(i,M+1,p)={}
  next i
  for j=0 to M
    MAX(N+1,j,p)={}
  next j
next p
** dynamic programming **
for i=N to 0
  for j=M to 0
    for p=|,-,\
      if (i<N) or (j<M)
        ** Bellman **
        MAX(i,j,p)=max{
          (i,j,p;|)+MAX(i+1,j,|),
          (i,j,p;-)+MAX(i,j+1,-),
          (i,j,p;\)+MAX(i+1,j+1,\)
        }
    next p
  next j
next i
```

Figure 3: The algorithm for calculating maximal alignments.

5 Discussion and Future Research

We demonstrated one possible way of how imprecise probabilities can be applied in bioinformatics. Imprecise probability theory allows us to substantially weaken assumptions we have to make about data, for instance about the evolutionary distance. In this paper, we did that by means of an interval rather than using a point estimate. It turns out that a good alignment still can be found in an efficient way, through a generalisation of the well-known Needleman-Wunsch algorithm, and relying on an extension of dynamic programming to the case where the gain is described by an imprecise probability model. This generalisation could be particularly useful in cases where the sequences under study are rather short, or other cases where point estimates of evolutionary distance are unreliable or hard to obtain. An implementation of the algorithm is currently in preparation.

Acknowledgements

This paper presents research results of project G.0139.01 of the Fund for Scientific Research, Flanders (Belgium), and of the Belgian Programme on Interuniversity Poles of Attraction initiated by the Belgian state, Prime Minister's Office for Science, Technology and Culture. The scientific responsibility rests with the author.

References

[1] Benner S. A., Cohen M. A., and Gonnet G. H. Amino-acid substitution during functionally constrained divergent evolution of protein sequences. *Protein Engineering*, 7(11):1323–1332, 1994.

[2] Richard Bellman. *Dynamic Programming.* Princeton University Press, Princeton, 1957.

[3] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt. A model of evolutionary change in proteins. In M. O. Dayhoff, editor, *Atlas of protein sequence and structure*, volume 5, pages 345–352. National biomedical research foundation, 1978.

[4] Gert de Cooman and Matthias C. M. Troffaes. Dynamical programming for deterministic discrete-time systems with uncertain gain. In Jean-Marc Bernard, Teddy Seidenfeld, and Marco Zaffalon, editors, *ISIPTA '03 – Proceedings of the Third International Symposium on Imprecise Probabilities and Their Applications*, pages 162–176. Carleton Scientific, July 2003.

[5] A. Elofsson. A study on protein sequence alignment quality. *Proteins-structure function and genetics*, 46(3):330–339, 2002.

[6] David W. Mount. *Bioinformatics. Genome and Sequence Analysis.* Cold Spring Harbor Laboratory Press, New York, 2001.

[7] S. B. Needleman and C. D. Wunsch. An efficient method applicable to the search for similarities in the amino acid sequences of two proteins. *Journal of Molecular Biology*, 48:443–453, 1970.

[8] Craig G. Nevill-Manning, Cecil N. Huang, and Douglas L. Brutlag. Pairwise protein sequence alignment using Needleman-Wunsch and Smith-Waterman algorithms, 1997. personal communication.

[9] W. R. Pearson. Empirical statistical estimates for sequence similarity searches. *Journal of Molecular Biology*, 276(1):71–84, 1998.

[10] Sheldon M. Ross. *Stochastic Processes.* John Wiley & Sons, New York, 2nd edition, 1996.

[11] Jones D. T., Taylor W. R., and Thornton J. M. The rapid generation of mutation data matrices from protein sequences. *Computer Applications in the Biosciences*, 8(3):275–282, 1992.

[12] Müller Tobias and Vingron Martin. Modeling amino acid replacement. *Journal of Computational Biology*, 7(6):761–776, 2000.

[13] Peter Walley. *Statistical Reasoning with Imprecise Probabilities.* Chapman and Hall, London, 1991.

[14] Gu X. and Li W. H. Estimation of evolutionary distances under stationary and nonstationary models of nucleotide substitution. *Proceedings of the National Academy of Sciences of the United States of America*, 95(11):5899–5905, 1998.