Stochastic Modeling of Biological Sequence Evolution

by

Keyuan Xu

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master's of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2005

Author .....
Department of Electrical Engineering and Computer Science
May 6, 2005

Certified by .....
George C. Verghese
Professor
Thesis Supervisor

Certified by .....
Peter C. Doerschuk
Professor
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Stochastic Modeling of Biological Sequence Evolution

by

Keyuan Xu

Submitted to the Department of Electrical Engineering and Computer Science
on May 6, 2005, in partial fulfillment of the
requirements for the degree of
Master's of Engineering in Electrical Engineering and Computer Science

Abstract

Markov models of sequence evolution are a fundamental building block for making inferences in biological research. This thesis reviews several major techniques developed to estimate parameters of Markov models of sequence evolution and presents a new approach for evaluating and comparing estimation techniques. Current methods for evaluating estimation techniques require sequence data from populations with well-known phylogenetic relationships. Such data is not always available since phylogenetic relationships can never be known with certainty. We propose generating sequence data for the purpose of estimation technique evaluation by simulating sequence evolution in a controlled setting. Our elementary simulator uses a Markov model and a binary branching process, which dynamically builds a phylogenetic tree from an initial seed sequence. The sequences at the leaves of the tree can then be used as input to estimation techniques. We demonstrate our evaluation approach on Arvestad and Bruno's estimation method, and show how our approach can reveal performance variations empirically. The results of our simulation can be used as a guide towards improving estimation techniques.

Thesis Supervisor: George C. Verghese
Title: Professor

Thesis Supervisor: Peter C. Doerschuk
Title: Professor

Acknowledgments

I thank Professor George C. Verghese and Professor Peter C. Doerschuk for their excellent guidance and constant motivation to excel. Their mastery of mathematics and insight into engineering helped me develop valuable new perspectives on our research.

I thank Professor Sanjoy K. Mitter for supporting my research through NSF Grant CCR-0325774.

I thank my wonderful parents Minzhen Xu and Shouying Wang for their care and guidance. They have supported me through every challenge I've faced. All of my achievements are also theirs.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The field of bioinformatics has gained widespread popularity due largely to efforts such as the genome projects, which have yielded an abundance of biological sequence data for analysis. This has led to the development and enhancement of many computational techniques for making inferences in biology and medicine. For example, predictive network models have been built to describe regulatory mechanisms in cellular processes. Advancements have been made in sequence alignment techniques and three-dimensional protein structure prediction algorithms to allow researchers to more confidently infer the functionality of newly discovered proteins.

A fundamental building block in the development of many tools and techniques in bioinformatics is a mathematical model of the evolution of genetic information, and in particular, a model of DNA and protein sequence evolution. Common uses for models of sequence evolution include building scoring systems for sequence alignment algorithms and predicting branch lengths for phylogenetic tree reconstruction algorithms. The most widely accepted model of sequence evolution is the Markov model of residue substitution [12]. In its simplest form, this model ignores insertions and deletions, and assumes that substitution at each site of a sequence proceeds independently of other sites and according to a common continuous-time Markov chain. Thorne [50] and Liò and Goldman [31] provide reviews of research on models of sequence evolution.

Many techniques have been developed to estimate the parameters of a Markov model of sequence evolution from observations of contemporary sequence data. However, in order to confidently make inferences from estimated Markov models, it becomes necessary to evaluate the accuracy of these estimation techniques. Yang [55] and Whelan and Goldman [53] used likelihood ratio tests to compare nested parametric models by fitting them to sequence data sets from populations with well-known phylogenetic relationships. Müller and Vingron [36] and Müller et al. [35] tested their estimation techniques by randomly generating alignments at given divergences using a given Markov model and then attempting to reconstruct the model parameters using the alignments. Devauchelle et al. [9] tested their estimation technique by simulating realizations of sequences along well-known phylogenetic trees with a given Markov model and then attempting to reconstruct the model parameters from these sequences.

All of these evaluation methods require *a priori* accurate knowledge of sequence alignments, time divergences, or phylogenetic tree structure. However, historical evolutionary events are generally not known with certainty, and there is a lack of data with which estimation techniques can be evaluated. Thus, current evaluation methods can test only a small sampling of an estimation technique's performance and only for species with phylogenetic relationships that are known with high levels of confidence.

The similar problem of evaluating of phylogenetic tree reconstruction algorithms when known phylogenies are unavailable was discussed by Hillis et al. [21]. Their solution was to generate phylogenies in the laboratory, which they did for bacteriophage T7 by elevating their mutation rates through the use of mutagens [20]. Then, they simulated sequence evolution along their generated phylogenetic trees with a given model and used the sequences at the leaves to evaluate various phylogenetic tree reconstruction algorithms [21]. However, this evaluation method is limited to organisms that can be properly cultivated and manipulated in laboratory settings. Kuhner and Felsenstein [29] similarly evaluated phylogenetic tree reconstruction accuracy. They randomly generated phylogenetic trees by simulating a branching process. Sequence

evolution was then simulated along these phylogenetic trees with a given model. A drawback of this evaluation method is that the generation of phylogenetic trees and the simulation of sequence mutation take place as independent processes, whereas one would expect them to be closely coupled in real life.

We propose using a new simulation technique to generate evolutionary events, in a controlled setting with specifiable parameters, for the purpose evaluating techniques that estimate parameters of Markov models of sequence evolution. Our simulation is different from [21] and [29] in that phylogenies and sequence mutations are generated as part of a single process. This eliminates the problem of having to determine phylogenies in advance. Estimation techniques can be applied to the realized sequences of the simulation, and the estimated parameters can be compared to the underlying specified parameters for evaluating accuracy. Using simulations, data sets can be repeatedly generated under varying sets of parameters, and estimation accuracy can be repeatedly observed. This allows users to gain a sense of a technique's performance distribution and performance under varying evolutionary conditions. Our method can also be applied to evaluating the accuracy of phylogenetic tree reconstruction algorithms, but we focus on Markov model estimation techniques in this thesis.

The layout of this document is as follows. For the remainder of this chapter, we provide a mathematical characterization of the Markov model of sequence evolution and provide examples of its application in bioinformatics. Chapter 2 provides a review of some well-known techniques that have been developed for estimating the parameters of Markov models from observed sequence data. We also discuss generalizations of the simplifying assumptions made for the elementary Markov model and point to research that attempt to use structural information to estimate Markov model parameters. Chapter 3 introduces our evolution simulation algorithm and describes how it can be used to evaluate estimation techniques. Chapter 4 describes a series of experiments where we evaluated the accuracy of Arvestad and Bruno's estimation technique [4] using our simulation approach. We also considered modifications to their technique and our simulator and examined their effects on estimation accuracy.

1.2 Markov Model of Evolution

The characterization of a Markov model begins with the definition of an alphabet of residues. In the case of DNA, for example, the alphabet consists of the set $S = \{A, C, G, T\}$. Let $X(t)$ denote a random process that takes values from the set S and represents the evolution of a particular site in the sequence through time. The model then consists of a Markov chain whose transition probabilities are assembled in the matrix $\mathbf{P}(t) = \{p_{ij}(t)\}$, defined for $t \geq 0$, where $p_{ij}(t) = Pr[X(s+t) = j | X(s) = i]$ refers to the probability that residue i will be replaced by residue j after a time of t. Note the assumption that $Pr[X(s+t) = j | X(s) = i]$ is only dependent on the lag t and not on the absolute time point s. All sites in a sequence evolve independently according to the same Markov chain.

This definition specifies $\mathbf{P}(t)$ as a row-stochastic matrix; hence, the properties $p_{ij}(t) \geq 0$ and $\sum_j p_{ij}(t) = 1$ must hold. In addition, $\mathbf{P}(0) = \mathbf{I}$ must be true, since all residues must remain unchanged over a time lag of zero units. Furthermore, we have $\mathbf{P}(t)\mathbf{P}(s) = \mathbf{P}(t+s)$, which is known as the Chapman-Kolmogorov equation.

Markov matrices $\mathbf{P}(t)$ can be generated from an instantaneous rate matrix $\mathbf{Q} = \{q_{ij}\}$. Using the forward Kolmogorov equation $\frac{d}{dt}\mathbf{P}(t) = \mathbf{P}(t)\mathbf{Q}$ and the initial condition $\mathbf{P}(0) = \mathbf{I}$, we get the relationship

$$\mathbf{P}(t) = \exp(t\mathbf{Q}) = \sum_{n=0}^{\infty} \frac{\mathbf{Q}^n t^n}{n!}. \tag{1.1}$$

The off-diagonal terms q_{ij} for $i \neq j$ are positive and represent the instantaneous transition rates of the process. The diagonal terms q_{ii} are defined such that each row of $\mathbf{Q}$ sums to zero.

The Markov chain is commonly assumed to be equilibrium, with stationary distribution vector π, and to be time-reversible. Time-reversibility implies that a Markov chain running forward in time is indistinguishable probabilistically from the same Markov chain running backwards in time. Formally, reversibility holds for a process in equilibrium when the detailed balance equations $\pi_i q_{ij} = \pi_j q_{ji}$ hold, or equivalently,

$\pi_i p_{ij}(t) = \pi_j p_{ji}(t)$ hold for all $t \geq 0$, and all possible i, j.

For simplicity, we will refer to $\mathbf{Q}$ as a rate matrix and $\mathbf{P}(t)$ as a mutation matrix for the remainder of this thesis.

1.3 Applications of the Markov Model

Markov models of sequence evolution provide a basis for making phylogenetic inferences, making them useful in several areas of bioinformatics research. Two popular applications, in which Markov models of sequence evolution are applied, are sequence alignment and phylogenetic tree reconstruction. There are large volumes of literature devoted to these two topics. We present a brief overview here and point to chapters 6 and 14 of Ewens and Grant [12] and Durbin et al. [11] for more comprehensive reviews.

1.3.1 Sequence Alignment

Sequence alignment is site-by-site arrangement of sequences intended to highlight their evolutionary similarities. In particular, the arrangement is made to maximize the likelihood that corresponding sites of an alignment evolved from a common ancestor, given some model of sequence evolution. Alignments are often used to reveal information regarding gene functionality and the classification of protein families.

There are several varieties of sequence alignments, including pairwise alignments, multiple alignments, and database searches for sequence similarity. Pairwise alignments further break down into global alignments, local alignments, and fitting one sequence into another. Figure 1-1 shows an example of a global pairwise alignment of 2 DNA sequences. The "_" symbol represents an indel which are positions where insertions or deletions are hypothesized to have occurred in the evolutionary path relating the 2 sequences.

Several algorithms have been developed for finding optimal sequence alignments. The most commonly used pairwise alignment algorithms are the dynamic programming algorithms of Smith and Waterman [45] and Needleman and Wunsch [37]. A

commonly used algorithm to construct multiple alignment is CLUSTAL W [49], and a common algorithm used for database searches for similarity is BLAST [3].

Sequence alignment algorithms typically require a scoring system, which can be derived from evolutionary relationships. In pairwise sequence alignment, a score is attributed to the residue pairing at each site of the alignment. Summing the residue pairing scores over all sites gives the score of an alignment. These pairwise scores can be conveniently assembled in a matrix. Early scoring matrices for pairwise amino acid sequence alignment include: the Unitary Protein Matrix (UPM), which simply scored each residue pairing a 1 if the residues matched and a 0 if the residues mismatched; the Genetic Code Matrix (GCM), which gave each residue pairing a score between 0 and 3 depending on the number of nucleotides the two amino acids had in common; and the Structure Genetic Matrix (SGM), developed by Mclachlan [32], which was constructed using observed amino acid sequences and physio-chemical properties.

The most commonly used scoring matrices today are the log-odds matrices derived from Markov models of sequence evolution [44]. In deriving log-odds matrices, prior biological knowledge must first be used to select a time t, which represents the most likely time divergence between the set of sequences to be compared. A mutation matrix $\mathbf{P}(t)$ can then be used to construct a scoring matrix $\mathbf{S} = \{s_{ij}\}$ using the transformation

$$s_{ij} = C \log(\frac{p_{ij}(t)}{\pi_j}), \tag{1.2}$$

where C is an arbitrary scaling constant. The score s_{ij} is attributed to pairing residue i with residue j; it gives the ratio of the likelihood that residue i aligned with j as a result of the model, given by $\mathbf{P}(t)$, and the likelihood that the alignment occurred randomly. Taking the logarithm of this likelihood provides a scoring system that is

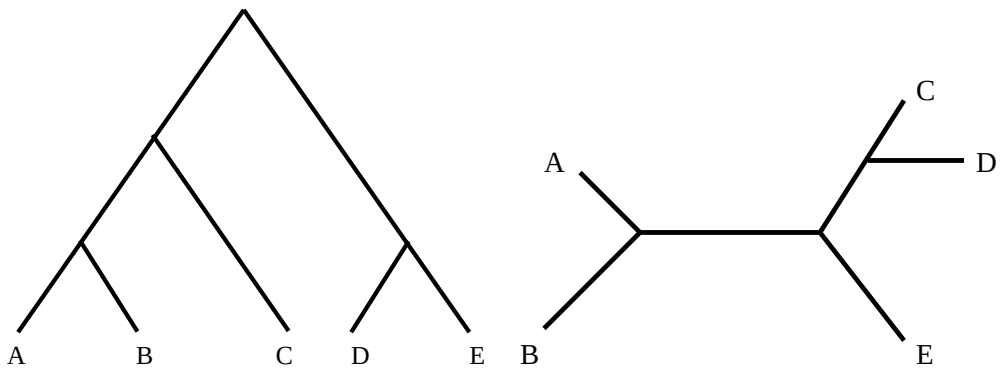
```
G A A T C T
  |  |    |
C A A _ C A
```

Figure 1-1: Global pairwise alignment example.

additive. Hence, a high alignment score indicates that the sequences involved are closely related, given the particular Markov model. Clearly, an accurate model of sequence evolution is essential for generating confident alignments.

1.3.2 Phylogenetic Tree Construction

Inferring a phylogenetic tree that describes the evolutionary relationships between a set of sequences has long been a topic of interest in bioinformatics and is also useful in the prediction of gene functionality. There are two types of phylogenetic trees: rooted trees, where the root represents the common ancestor of all sequences related by the tree; and unrooted trees, where the direction of evolutionary time flow is not specified. Figure 1-2 show examples of rooted and unrooted trees relating the hypothetical sequences A, B, C, D, and E.



(a) Rooted tree **(b)** Unrooted tree

Figure 1-2: Phylogenetic tree examples.

Constructing a phylogenetic tree for a given set of sequences involves first, calculating the evolutionary distance between every pair of sequences, and then, specifying the tree topology. Evolutionary distances are normally calculated using maximum likelihood techniques with models of sequence evolution. Chapter 13 of Ewens and Grant [12] shows how maximum likelihood measures of distance can be estimated when simple parametric Markov models of evolution are assumed. More rigorous techniques used to predict distances from more general Markov models are described

by Baake and von Haeseler [5]. Note that since we can only observe contemporary sequences, there is no way to determine the absolute scale on time. Hence, we can only estimate relative evolutionary distances.

Pairwise distances are then passed as input to algorithms that determine the topology of a phylogenetic tree. Well-known algorithms for topology reconstruction include the neighbor-joining algorithm [43] and the Unweighted Pair Group Method with Arithmetic Mean (UPGMA) algorithm [46].

A popular tree reconstruction algorithm that does not use a model of sequence evolution is the method of maximum parsimony. In this algorithm, the optimal topology of a tree is found by enumerating all possible topologies relating a given set of sequences and selecting the tree that minimizes some "cost" function. When constructing a phylogenetic tree for a population of sequences, cost is typically defined as the total number of observed substitutions across all branches of the tree. While this method is advantageous in that it does not require any pre-derived knowledge, its branches do not have a well-defined time structure. This method is further discussed in chapter 2 in the context of the PAM estimation technique [8].

Chapter 2

Estimation of Markov Model Parameters

2.1 Overview

Several techniques have been developed to empirically estimate Markov models of sequence evolution through analysis of observed sequences. Markov models have also been characterized parametrically using knowledge of biochemical properties. Parametric models are naturally more popular for describing DNA evolution. There are much fewer nucleotides than amino acids, making it relatively easier to define a manageable set of parameters to characterize properties in nucleotide substitution. Models of protein evolution are typically estimated empirically without any prior parameterizations. Since evolutionary events between neighboring sites in a sequence are generally not independent as the Markov model assumes, modeling protein evolution is often preferred over modeling DNA evolution, because it can potentially characterize dependence between adjacent sites at the nucleotide level.

We present, in this chapter, a review of some of the most widely-used empirical estimation techniques and parameterizations of Markov models of DNA evolution. We focus on estimation techniques that use observations of sequence data, and also introduce estimation techniques that use the structural properties of biological sequences. The estimation techniques are described for sequences with a general alphabet of

residues S and presented under a common mathematical framework. We also discuss some shortcomings of the elementary Markov model of sequence evolution and present some generalizations that have been proposed.

2.2 Parametric Models

The most general rate matrix $\mathbf{Q}$ of a Markov model of sequence evolution has only the requirement that its off-diagonal terms are positive, and the sum of each of its rows are zero. Several parametric models have been proposed to simplify the characterization of the rate matrix for models of DNA evolution. The following sections introduce some of the most widely-used parameterizations.

2.2.1 Jukes-Cantor

The most simple parametric model of nucleotide substitution was proposed by Jukes and Cantor [25]. Their model assumed that all nucleotides are equally likely to undergo substitution, and given that a substitution has taken place, any other nucleotide is equally likely to be the replacing nucleotide. Hence, the rate matrix of this model can be parameterized as

$$\mathbf{Q} = \begin{bmatrix} . & \alpha & \alpha & \alpha \\ \alpha & . & \alpha & \alpha \\ \alpha & \alpha & . & \alpha \\ \alpha & \alpha & \alpha & . \end{bmatrix}. \tag{2.1}$$

The diagonal terms are all -3α since the sum of the rows of $\mathbf{Q}$ must be zero. It can be easily verified that the Jukes-Cantor model has a uniform stationary distribution and is reversible. In practice, this model is generally found to be too simple and unable to sufficiently model the rates of nucleotide substitution.

Note that we have ordered the indices of this matrix and all subsequent DNA mutation matrices discussed in section 2.2 such that $1 = \mathrm{A}, 2 = \mathrm{G}, 3 = \mathrm{C}, 4 = \mathrm{T}$.

2.2.2 Kimura

Kimura [27] introduced a slightly more complex model of nucleotide evolution that allows for differences between transition and transversion rates. A transition is the substitution of a purine by a purine or the substitution of a pyrimidine by a pyrimidine, while a transversion is the substitution of a purine by a pyrimidine or vice versa. Nucleotides A and G are purines, and nucleotides C and T are pyrimidines. Transitions and tranversions are expected to differ, because purines and pyrimidines have different molecular structures. The rate matrix of this model is parameterized as

$$\mathbf{Q} = \begin{bmatrix} . & \alpha & \beta & \beta \\ \alpha & . & \beta & \beta \\ \beta & \beta & . & \alpha \\ \beta & \beta & \alpha & . \end{bmatrix}. \tag{2.2}$$

The parameter α controls the rate of transitions, while the parameter β controls the rate of transversions. It can also be easily shown that Kimura's model has a uniform stationary distribution and is reversible. The Kimura model generalizes to the Jukes-Cantor model when $\alpha = \beta$.

Kimura's model is still overly simplistic since it possesses many assumptions of symmetry and uniformity. Chapter 13 of Ewens and Grant [12] and Liò and Goldman [31] describe several generalizations of Kimura's two-parameter model, including the contributions by Blaisdell, Schadt et al. , Takahata and Kimura, and Gojobori et al.

2.2.3 Felsenstein

Felsenstein [13] introduced an alternate generalization of the Jukes-Cantor model. Felsenstein's model specifies that the rates of substitution are proportional to the stationary distributions of the replacing nucleotides. The rate matrix is parametrized

as

$$\mathbf{Q} = \begin{bmatrix} . & \alpha\pi_G & \alpha\pi_C & \alpha\pi_T \\ \alpha\pi_A & . & \alpha\pi_C & \alpha\pi_T \\ \alpha\pi_A & \alpha\pi_G & . & \alpha\pi_T \\ \alpha\pi_A & \alpha\pi_G & \alpha\pi_C & . \end{bmatrix}. \tag{2.3}$$

The parameters π_A, π_G, π_C, and π_T specify the stationary distribution, and α in this case is a scaling parameter. This model of substitution allows for arbitrary stationary distribtions but is still restricted to be reversible. Felsenstein's model generalizes to Jukes and Cantors model when $\pi_A = \pi_G = \pi_C = \pi_T = 0.25$.

2.2.4 HKY

Hasegawa et al. [18] introduced a model that combined the features of both the Kimura and Felsenstein models. This model, called the HKY model, can be described as the Felsenstein model with an extra parameter to characterize the difference between transitions and transversions. The rate matrix can be written as

$$\mathbf{Q} = \begin{bmatrix} . & \alpha\pi_G & \beta\pi_C & \beta\pi_T \\ \alpha\pi_A & . & \beta\pi_C & \beta\pi_T \\ \beta\pi_A & \beta\pi_G & . & \alpha\pi_T \\ \beta\pi_A & \beta\pi_G & \alpha\pi_C & . \end{bmatrix}. \tag{2.4}$$

The HKY model generalizes to the Kimura model when $\pi_A = \pi_G = \pi_C = \pi_T = 0.25$ and generalizes to the Felsenstein model when $\alpha = \beta$. Like the Felsenstein model, it allows for arbitrary stationary distributions but is restricted to be reversible.

2.2.5 General Reversible Model

The general reversible (REV) model only parameterizes the assumption that the process of nucleotide substitution is reversible. This model can be generalized to each of the previous four models upon proper setting of its free parameters. Its rate

matrix takes the form

$$\mathbf{Q} = \begin{bmatrix} . & \alpha\pi_G & \beta\pi_C & \gamma\pi_T \\ \alpha\pi_A & . & \zeta\pi_C & \eta\pi_T \\ \beta\pi_A & \zeta\pi_G & . & \mu\pi_T \\ \gamma\pi_A & \eta\pi_G & \mu\pi_C & . \end{bmatrix}. \tag{2.5}$$

Yang [55] and Yang et al. [57] evaluated these nested parametric models by fitting them to homologous DNA sequences from families with well-known phylogenies. Maximum likelihood techniques were used to determine how well the various models explained the observed sequences. In [57], it was found that the HKY model was superior the Jukes-Cantor, Kimura, and Felsenstein models. In [55], it was found that the REV model provides a better fit for observed sequence data than the HKY model when substitution rates were assumed to be homogeneous across all sites in a sequence. Furthermore, the general unconstrained model was not found to improve upon the REV model in these experiments. Yang thus recommended the REV model to be used in phylogenetic analysis [55].

2.2.6 The Codon-Based Model

A parametric Markov model that describes sequence evolution at the codon level was first introduced by Goldman and Yang [14] and later simplified by Nielsen and Yang [38]. There are 64 different codons, 3 of which are stop codons and were not considered. The substitution rates of the remaining 61 codons were assembled in a 61-by-61 continuous-time Markov matrix. The rate matrix took the form $\mathbf{Q} = \{q_{ij}\}$ where

$$q_{ij} = \begin{cases} 0, & i \text{ and } j \text{ differ at 2 or 3 positions} \\ \mu\pi_j, & i \text{ and } j \text{ differ by 1 synonymous transversion} \\ \mu\kappa\pi_j, & i \text{ and } j \text{ differ by 1 synonymous transition} \\ \mu\omega\pi_j, & i \text{ and } j \text{ differ by 1 nonsynonymous transversion} \\ \mu\omega\kappa\pi_j, & i \text{ and } j \text{ differ by 1 nonsynonymous transition} \end{cases} \tag{2.6}$$

The parameters π_j for $j = 1 \ldots 61$ represents the stationary distributions of the 61 non-stop codons; μ represents a scaling factor; κ represents the transition/transversion ratio; and ω represents the ratio of synonymous to nonsynonymous substitutions. Models of amino acid substitution are generally more popular in practice than the codon-based model.

2.3 Maximum Parsimony Techniques

We begin our exploration of empirical estimation techniques for Markov models of sequence evolution by discussing the Point Accepted Mutations (PAM) technique [8]. PAM and its variants are often call maximum parsimony techniques, since they use the method of maximum parsimony to construct phylogenetic trees as part of their technique.

Dayhoff and coworkers developed PAM to estimate amino acid substitution rates from observed sequences. Their original technique made no explicit mention of a Markov model of sequence evolution. In this document, we present their technique for a general alphabet of residues S and in the framework of a Markov model.

2.3.1 PAM

The aim of the PAM technique is to infer all individual residue substitutions, i.e. point mutations, from a set of related sequences organized into multiple alignments. The residue mutations are inferred by constructing phylogenetic trees for each multiple alignment and observing the residue exchanges across each branch of the inferred trees from root to leaf. The relative frequencies of these point mutations are then used to estimate a reversible mutation matrix $\mathbf{P}(\tau)$, which is used to estimate a reversible rate matrix $\mathbf{Q}$.

Maximum Parsimonious Phylogenetic Trees

Phylogenetic trees are inferred from each multiple alignment using the method of maximum parsimony. This method works by enumerating all possible phylogenetic

trees for a given multiple alignment, and selecting the tree with the minimum number of total substitutions across all branches. The observed residue exchanges across all phylogenetic trees of all multiple alignments are tallied in a count matrix $\mathbf{N}$. A two-way counting scheme was used in the tallying, whereby if residue i is aligned with residue j across a tree branch, then $\mathbf{N}_{ij}$ and $\mathbf{N}_{ji}$ are both incremented by 1. If $i = j$, then $\mathbf{N}_{ii}$ is incremented by 2. This counting method is equivalent to assuming reversibility. It assumes that each of the two sequences spanning a branch can be treated as the ancestor; thus time can flow in either direction. When more than one most parsimonious tree exists for a multiple alignment, the contribution of that multiple alignment to the count matrix is averaged over all of its most parsimonious trees. The rows of the count matrix $\mathbf{N}$ are normalized to stochastic vectors to obtain an estimate $\mathbf{P}(\tau)$.

Closeness Criterion

The sequences of each multiple alignment must be "sufficiently close" to each other to reduce the probability of having two or more consecutive substitutions at any site. Having unobserved mutations puts biases into estimates of the substitution rates. To satisfy this requirement, Dayhoff et al. used multiple alignments, where each sequence in the alignment was no more than 15% different from any other sequence in the alignment. Having "sufficiently close sequences also provides a small divergence time τ. Hence, the estimated mutation matrix can be related to the mutation rate matrix by the expression

$$\mathbf{P}(\tau) = \exp\{\tau\mathbf{Q}\} \approx \mathbf{I} - \tau\mathbf{Q}. \tag{2.7}$$

Time Scale

Absolute time scales cannot be estimated because all sequences observed in the estimation are assumed to be from one point in time. Dayhoff also defines a meaningful relative time scale by letting the distance of 1PAM denote the amount of evolutionary time necessary for 1% of the residues to mutate. Let Δt represent the distance of

1PAM; $\mathbf{P}(\Delta t)$ represents the 1PAM mutation matrix. Assuming Δt is also small, $\mathbf{P}(\Delta t)$ can be estimated from $\mathbf{P}(\tau)$ as follows:

$$\begin{aligned} \mathbf{P}(\tau) &\approx \mathbf{I} - \tau\mathbf{Q} && (2.8) \\ \Delta t\mathbf{Q} &\approx \frac{\Delta t}{\tau}\mathbf{I} - \frac{\Delta t}{\tau}\mathbf{P}(\tau) && (2.9) \end{aligned}$$

$$\begin{aligned} \mathbf{P}(\Delta t) &\approx \mathbf{I} - \Delta t\mathbf{Q} && (2.10) \\ &\approx \mathbf{I} - \frac{\Delta t}{\tau}\mathbf{I} + \frac{\Delta t}{\tau}\mathbf{P}(\tau) && (2.11) \\ &\approx (1-c)\mathbf{I} + c\mathbf{P}(\tau). && (2.12) \end{aligned}$$

The constant $c = \Delta t/T$ must be selected such that the expected number of substitutions after 1 step of the mutation matrix $\mathbf{P}(\Delta t)$ is 1% of the total number of residues. Hence,

$$\begin{aligned} \sum_i \sum_{i \neq j} \pi_i \mathbf{P}(\Delta t)_{ij} = \sum_i \sum_{i \neq j} c\pi_i \mathbf{P}(\tau)_{ij} = 0.01 && (2.13) \\ c = \frac{0.01}{\sum_i \sum_{i \neq j} \pi_i \mathbf{P}(\tau)_{ij}}, && (2.14) \end{aligned}$$

where π_i is the frequency of residue i, which can be estimated from the observed frequency of residue i sequence data.

2PAM mutation matrices can be extrapolated using

$$\mathbf{P}(2\Delta t) \approx \mathbf{P}(\Delta t)\mathbf{P}(\Delta t). \quad (2.15)$$

Since the 1PAM matrix is itself an estimation, this extrapolation technique losses accuracy in estimating nPAM matrices for large values of n. Dayhoff uses this method for extrapolating matrices up to a distance of 250PAM. Note that the nPAM matrix, for $n > 1$, is defined as n steps of the 1PAM matrix, and n% of residues do not mutate with one step of the nPAM matrix.

$\mathbf{Q}$ [1] can be estimated up to an unknown multiplicative scale factor using $\tau\mathbf{Q} \approx \mathbf{I} - \mathbf{P}(\tau)$ or $\tau\mathbf{Q} = \exp\{\mathbf{P}(\tau)\}$. The unknown scale factor corresponds to the fact that absolute time scales cannot be estimated.

2.3.2 Extensions of PAM

Since the development of the PAM technique in the 1970s, genome projects have yielded an abundance of amino acid sequence data. By the early 1990s, some tens of thousands of protein sequences were available for analysis.

Applying the PAM technique to this data would likely improve estimates of Markov models of protein evolution. However, the PAM technique in its original form would require inordinate amount of processing time if applied to protein sequence databases of such size. In separate efforts, Jones et al. [24] and Gonnet et al. [16] developed algorithms that efficiently estimate Markov models of sequence evolution from large databases of amino acid sequences using the PAM formalism. The resulting Markov models are expected to perform better in sequence analysis applications.

2.3.3 Disadvantages of PAM

The PAM technique has two main disadvantages. First, PAM estimates are only based on sequences that are "sufficiently close" to each other. This criterion leads to two consequences: 1) an abundance of sequence data is omitted from being used as estimation data, and 2) the estimated rates of substitution are only accurate at short evolutionary divergences. Substitution rates at longer evolutionary divergences are extrapolated by iterated application of the 1PAM matrix and lose accuracy for increasingly high divergences.

Second, the method of maximum parsimony yields phylogenetic trees with no time structure; the branches of the resulting tree have no specified lengths. The mutation matrix $\mathbf{P}(\tau)$ is estimated by uniformly averaging observed substitutions across each

[1] The rate matrix $\mathbf{Q}$ was not considered in the original work of Dayhoff et al.

branch, which implicity assumes that they are all of equal length. Ideally, if the true phylogenetic tree was known and has branch lengths $t_1 \dots t_n$, $\mathbf{Q}$ would be estimated by first estimating the mutation matrices $\mathbf{P}(t_1) \dots \mathbf{P}(t_n)$ for each branch, then taking a weighted average of their logs:

$$\mathbf{Q} = \frac{\log\{\mathbf{P}(t_1)\}}{t_1} + \cdots + \frac{\log\{\mathbf{P}(t_n)\}}{t_n}. \tag{2.16}$$

Thus, the PAM technique could yield distorted results by using a phylogenetic tree with no specified branch lengths. The following estimation techniques address these issues.

2.4 Maximum Likelihood Techniques

Maximum likelihood estimation techniques improve upon the shortcomings of the PAM technique. These techniques generally aim to simultaneously find the phylogenetic tree (complete with branch lengths) and the Markov model of evolution that maximizes the likelihood of a given set of observed sequence alignments. Mathematically, the goal is to maximize the likelihood $L(\mathbf{Q}, \mathbf{T}|\mathbf{A})$, subject to varying the parameters $\mathbf{Q}$ and $\mathbf{T}$, where $\mathbf{T}$ represents the tree topology and branch lengths and $\mathbf{A}$ represents the observed alignments. A parameter space consisting of phylogenetic tree topology, branch lengths, and a rate matrix $\mathbf{Q}$ is searched to find the optimal setting. Typically, iterated techniques characteristic of the expectation maximization algorithm are used for optimization over several parameters.

Maximum likelihood techniques have been used to estimate Markov models for different families of amino acid sequences by Adachi and Hasegawa [1], Yang et at. [58], Adachi et al. [2], Whelan and Goldman [54], and Müller et al. [35]. Variations of the basic methodology have been explored by these authors in attempt to improve computational efficiency.

These techniques improve upon the PAM estimation technique, since they present a model that allows multiple substitutions to occur across any site in an alignment.

However, maximum likelihood techniques are also very computationally expensive. Usually, they are only applied to small data sets.

2.5 Faster Empirical Techniques

Several empirical techniques have been published in recent years, which address the shortcomings of both the PAM and the maximum likelihood techniques. These methods typically take as input a set of pairwise aligned sequences and aim to estimate the mutation matrix $\mathbf{P}(t_k)$ and the time divergence t_k implied by each alignment k. The estimates of $\mathbf{P}(t_k)$ and t_k are manipulated in various ways to obtain estimates of the rate matrix $\mathbf{Q}$.

These methods are expected to be more accurate than the methods of maximum parsimony because they can incorporate information from alignments with all degrees of evolutionary divergences. The divergence times t_k allow the estimates $\mathbf{P}(t_k)$ to be weighted in various ways to account for different substitution rates at different divergence times. These methods are also typically faster than the maximum likelihood techniques, since they do not directly infer phylogenetic tree topologies.

We introduce four of these techniques, namely the Arvestad and Bruno [4], Resolvent [36], Devauchelle et al. [9], and PMB [52] techniques.

2.5.1 Arvestad and Bruno

Arvestad and Bruno [4] developed a technique for estimating a reversible rate matrix $\mathbf{Q}$ [2] from a given collection of pairwise aligned sequences. Their technique aims to reconstruct $\mathbf{Q}$ by estimating its eigenvectors and eigenvalues from the observed alignments. We write the eigen-decomposition of $\mathbf{Q}$ as $\mathbf{Q} = \mathbf{V\Lambda V^{-1}}$, where the columns of $\mathbf{V}$ are the eigenvectors, and the (diagonal) entries of the diagonal matrix $\mathbf{\Lambda}$ are the eigenvalues.

[2] $\mathbf{Q}$ was unnecessarily assumed to be normal in [4]. We remove this assumption.

Estimating $\mathbf{P}(t_k)$

Observed residue exchanges in the aligned sequences are used to obtain estimates $\mathbf{P}(t_k)$ with some unknown t_k for each alignment k. Before we describe the estimation procedure, notice that the reversibility assumption is necessary for the estimate $\mathbf{P}(t_k)$ to be valid. The sequences of each alignment k are the leaves of a phylogenetic tree, and the matrix $\mathbf{P}(t_k)$ estimates the process that travels from one sequence, backwards in time to the common ancestor, and forward in time to the other sequence, with a total time separation of t_k.

One way to ensure reversibility is to employ the two-way counting scheme in estimating $\mathbf{P}(t_k)$. This scheme was described in the PAM section for counting residue exchanges across all branches of a phylogenetic tree.

Another way to ensure reversibility is to use a one-way counting scheme to estimate $\mathbf{P}(t_k)$, and then symmetrize the matrix quantity

$$\Pi^{1/2} \sum_k \mathbf{P}(t_k) \Pi^{-1/2} \tag{2.17}$$

by replacing it with

$$\frac{\Pi^{1/2} \sum_k \mathbf{P}(t_k) \Pi^{-1/2} + (\Pi^{1/2} \sum_k \mathbf{P}(t_k) \Pi^{-1/2})^T}{2}, \tag{2.18}$$

to make the detailed balance equations hold for all $\mathbf{P}(t_k)$. Here, Π denotes a diagonal matrix of the steady-state residue frequencies. In the one-way counting scheme, one sequence in each pairwise alignment is arbitrarily chosen as the ancestor, and the other is the descendant. Since reversibility is assumed, the choice of ancestor can be arbitrarily made. The i, j-th element of a count matrix $\mathbf{N^{(k)}}$ is estimated as the number of times residue i in the ancestor aligns with residue j in the descendant. The rows of $\mathbf{N^{(k)}}$ are normalized to stochastic vectors to obtain an estimate $\mathbf{P}(t_k)$.

Estimating the Eigenvectors of Q

The matrix $\mathbf{P}(t_k)$ can be factored as follows:

$$\begin{aligned}
\mathbf{P}(t) &= \exp(\mathbf{Q}t) && (2.19)\\
&= \mathbf{I} + \mathbf{Q}t + \frac{1}{2!}\mathbf{Q}^2t^2 + \cdots && (2.20)\\
&= \mathbf{I} + \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-\mathbf{1}}t + \frac{1}{2!}(\mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-\mathbf{1}})^2t^2 + \cdots && (2.21)\\
&= \mathbf{V}(\mathbf{I} + \mathbf{\Lambda}t + \frac{1}{2!}\mathbf{\Lambda}^2t^2 + \cdots)\mathbf{V}^{-\mathbf{1}} && (2.22)\\
&= \mathbf{V}\exp(\mathbf{\Lambda}t)\mathbf{V}^{-\mathbf{1}} && (2.23)
\end{aligned}$$

Notice from the decomposition that $\mathbf{Q}$ and $\mathbf{P}(t_k)$ have the same eigenvectors. Furthermore, any linear combination of the matrices $\mathbf{P}(t_k)$ have the same eigenvectors as $\mathbf{Q}$. Hence, we get the eigenvectors of $\mathbf{Q}$ by calculating the eigenvectors of $\sum_k \mathbf{P}(t_k)$. A weighted linear combination of the $\mathbf{P}(t_k)$'s can also be used to maximize the effect of the least noisy estimates.

Estimating the Eigenvalues of Q

The eigenvalues of $\mathbf{Q}$ can be estimated up to an unknown scale factor using the fact that

$$\begin{aligned}
\mathbf{P}(t_k) &= \mathbf{V}\exp(\mathbf{\Lambda}t_k)\mathbf{V}^{-\mathbf{1}} && (2.24)\\
\mathbf{V}^{-\mathbf{1}}\mathbf{P}(t_k)\mathbf{V} &= \exp(\mathbf{\Lambda}t_k) && (2.25)\\
\log(\mathbf{V}^{-\mathbf{1}}\mathbf{P}(t_k)\mathbf{V}) &= \mathbf{\Lambda}t_k. && (2.26)
\end{aligned}$$

Each alignment k yields four observations of five unknown variables, namely the four eigenvalues and the time divergence t_k. This corresponds to the fact that absolute time scales cannot be estimated. Thus it is impossible to determine the eigenvalues with certainty. Rather, the eigenvalues can be determined up to an unknown time scale by estimating their ratios λ_r/λ_s. This can be done using a linear regression

through the origin:

$$\lambda_r/\lambda_s = \frac{\sum_{\text{all alignments } k}(\lambda_r t_k)(\lambda_s t_k)}{\sum_{\text{all alignments } k}(\lambda_s t_k)^2}. \tag{2.27}$$

A weighted linear regression can be also used to minimize noise. One of the eigenvalues of a rate matrix must be 0. For the remaining eigenvalues, one is arbitrarily set to -1, and the others are estimated according to the ratios λ_r/λ_s.

The eigenvalues are guaranteed to be real because the rate matrix $\mathbf{Q}$ is reversible. Reversibility implies that the detailed balance equations $\pi\mathbf{Q} = \mathbf{Q^T}\pi$, written here in matrix form, must hold. Hence, the quantity

$$\mathbf{M} = \pi^{\mathbf{1/2}}\mathbf{Q}\pi^{\mathbf{-1/2}} \tag{2.28}$$

must be symmetric. This guarantees the matrix $\mathbf{M}$ to have real eigenvalues. To show that $\mathbf{Q}$ and $\mathbf{M}$ have the same eigenvalues, we factor

$$\mathbf{M} = \pi^{\mathbf{1/2}}\mathbf{Q}\pi^{\mathbf{-1/2}} = \mathbf{V}\mathbf{\Lambda}\mathbf{V^T}, \tag{2.29}$$

where $\mathbf{\Lambda}$ is a diagonal matrix of real eigenvalues and $\mathbf{V}$ is an orthogonal matrix of eigenvectors. Isolating $\mathbf{Q}$, we get

$$\mathbf{Q} = \pi^{\mathbf{-1/2}}\mathbf{V}\mathbf{\Lambda}\mathbf{V^T}\pi^{\mathbf{1/2}}, \tag{2.30}$$

Hence, the eigenvalues of $\mathbf{Q}$ are $\mathbf{\Lambda}$

Once the eigenvalues and eigenvectors are known, $\mathbf{Q}$ can be constructed using the eigen-decomposition expression, up to an unknown multiplicative factor, corresponding to the arbitrary value of -1 used for one of the eigenvalues.

2.5.2 Resolvent

Müller and Vingron developed the Resolvent technique [36], which takes as input a collection of pairwise alignments and aims to estimate a reversible rate matrix $\mathbf{Q}$ by

first, estimating its resolvent and then, expressing $\mathbf{Q}$ in terms of the resolvent. The resolvent is a matrix quantity equivalent to the component-wise Laplace transform of $\mathbf{P}(t)$ and is written as $\mathbf{R}(s) = \{r_{ij}(s)\}$, where

$$r_{ij}(s) = \int_0^\infty e^{-st} p_{ij}(t) dt. \tag{2.31}$$

To express $\mathbf{Q}$ in terms of its resolvent, we take Laplace transforms on both sides of the Chapman-Kolmogorov equation

$$\frac{d}{dt}\mathbf{P}(t) = \mathbf{P}(t)\mathbf{Q}, \tag{2.32}$$

and algebraically manipulate the result. The rate matrix $\mathbf{Q}$ can be written as

$$\mathbf{Q} = s\mathbf{I} - \mathbf{R}(s)^{-1}. \tag{2.33}$$

The resolvent technique uses an iterative estimation approach. Mutation matrices $\mathbf{P}(t_k)$ are first estimated for each alignment k. Using these estimates and an initial rate matrix $\mathbf{Q}_0$, time divergences t_k are estimated for each alignment k using the maximum likelihood technique. These time divergences are then used, along with the mutation matrices, to estimate the resolvent by approximating the right side of equation 2.31. Equation 2.33 is then used to calculate a rate matrix $\mathbf{Q}_1$. The entire procedure then repeats until convergence and yields an estimate of the rate matrix.

Estimating $\mathbf{P}(t_k)$ and t_k

Estimates of the mutation matrix $\mathbf{P}(t_k)$ and its corresponding time divergence t_k are obtained from each pairwise alignment k. Müller and Vingron used the two-way counting scheme, which was used in the PAM technique, to get reversible estimates of $\mathbf{P}(t_k)$.

Consider the m-th iteration of estimating t_k. Let $\mathbf{N^{(k)}} = \{n_{ij}^{(k)}\}$ represent the matrix of two-way counts obtained from estimating $\mathbf{P}(t_k)$. Let π represent a vector of steady-state residue frequencies, which is estimated from the observed residue

frequencies in all alignments. Given an the Markov model estimated from the previous step, $\mathbf{P}_{m-1}(t) = \exp\{t\mathbf{Q}_{m-1}\}$, the likelihood of alignment k to have an evolutionary time divergence of t_k is

$$\mathbf{L}(t_k|\text{Alignment } k) = \prod_i \prod_j (\pi_i p_{m-1,ij}(t_k))^{n_{ij}^{(k)}}. \tag{2.34}$$

Taking logs on both sides yields

$$\log \mathbf{L}(t_k|\text{Alignment } k) = \sum_i \sum_j n_{ij}^{(k)} \log(\pi_i p_{m-1,ij}(t_k)). \tag{2.35}$$

The value of t_k that maximizes this likelihood is the solution to the expression

$$\begin{aligned}
0 = \frac{d}{dt} \log \mathbf{L}(t_k|\text{Alignment } k) &= \sum_i \sum_j n_{ij}^{(k)} \frac{d}{dt} \log(\pi_i p_{m-1,ij}(t_k)) && (2.36)\\
&= \sum_i \sum_j n_{ij}^{(k)} \frac{d}{dt} \log(\pi \mathbf{P}_{m-1}(t_k))_{ij} && (2.37)\\
&= \sum_i \sum_j n_{ij}^{(k)} \frac{\frac{d}{dt}(\pi \mathbf{P}_{m-1}(t_k))_{ij}}{(\pi \mathbf{P}_{m-1}(t_k))_{ij}} && (2.38)\\
&= \sum_i \sum_j n_{ij}^{(k)} \frac{(\pi \mathbf{P}(t_k)\mathbf{Q}_{m-1})_{ij}}{(\pi \mathbf{P}_{m-1}(t_k))_{ij}} && (2.39)\\
&= \sum_i \sum_j n_{ij}^{(k)} \frac{(\mathbf{P}_{m-1}(t_k)\mathbf{Q}_{m-1})_{ij}}{p_{m-1,ij}(t)}. && (2.40)
\end{aligned}$$

The solution can be found by applying Newton's method for example.

Estimating the Resolvent

Consider the m-th iteration of estimating the rate matrix. The resolvent is first calculated by approximating equation 2.31, and equation 2.33 is used to calculate $\mathbf{Q_m}$.

Assuming there n pairwise alignments, each element of the resolvent can be esti-

mated using the piece-wise linear integration

$$r_{ij}(s) = \int_0^\infty e^{-st} p_{ij}(t)dt \tag{2.41}$$

$$\approx \int_0^{t_1} e^{-st} p_{ij}(t)dt + \cdots + \int_{t_n}^\infty e^{-st} p_{ij}(t)dt, \tag{2.42}$$

where

$$\int_{t_k}^{t_{k+1}} e^{-st} p_{ij}(t)dt \approx \int_{t_k}^{t_{k+1}} e^{-st}(p_{ij}(t_k) + \frac{p_{ij}(t_{k+1}) - p_{ij}(t_k)}{t_{k+1} - t_k}(t - t_k))dt. \tag{2.43}$$

The choice of s is independent of $\mathbf{Q}$. However, to minimize the effect of approximation errors, s is chosen to maximize the likelihood of the alignments

$$\log \mathbf{L}(s|n \text{ Alignments}) = \sum_k \sum_i \sum_j n_{ij}^{(k)} \log(\pi_i (e^{t_k(s\mathbf{I} - \mathbf{R}(s)^{-1})})_{ij}). \tag{2.44}$$

2.5.3 Devauchelle et al.

Devauchelle et al. [9] developed a novel method for analyzing how well a single reversible rate matrix $\mathbf{Q}$ describes a set of pairwise aligned sequences, i.e., given a set of pairwise alignments whose sequences are related by some phylogenetic tree, they analyzed how well evolution along every branch of that tree can be modeled by a single reversible $\mathbf{Q}$. Their analysis also yielded an approach for estimating a reversible $\mathbf{Q}$ from a given set of pairwise alignments, which we present here.

Estimating Matrix Logarithms

Given a set of alignments, a mutation matrix $\mathbf{P}(t_k)$ was estimated from the observed substitution frequencies of each alignment k. The estimation procedure can be carried out using a two-way counting scheme or a one-way counting scheme with subsequent symmetrization, as described in the Arvestad and Bruno section. The matrix logarithms of these mutation matrices are then calculated:

$$\mathbf{L}(t_k) = \log(\mathbf{P}(t_k)) \tag{2.45}$$

Note that numerical methods for calculating matrix logarithms may not work under certain circumstances for reasons discussed in [9] and [52]. For this reason, Devauchelle et al. considers only pairwise alignments in which the two sequences are "sufficiently related," defined to mean that their corresponding mutation matrix estimate $\mathbf{P}(t_k)$ admits a logarithm. A sufficient condition for the sequences of alignment k to be sufficiently related is if there exists some positive integer n such that

$$||(\mathbf{P}(t_k) - \mathbf{I})^n||_F < 1, \tag{2.46}$$

where the Frobenius norm $||\mathbf{A}||_F$ of a matrix $\mathbf{A} = \{a_{ij}\}$ is defined as

$$||\mathbf{A}||_F = \sqrt{\sum_i \sum_j a_{ij}^2} = \sqrt{tr(\mathbf{A}\mathbf{A}^T)}. \tag{2.47}$$

Principal Components Analysis

Given that all alignments k are related by a single reversible rate matrix $\mathbf{Q}$. We can write

$$\mathbf{L}(t_k) = t_k \mathbf{Q} \tag{2.48}$$

for all k. Before continuing, we introduce a change in our notations to make the following mathematical derivations more intuitive. Instead of representing quantities of the Markov model as $|S|$-by-$|S|$ matrices, where S is the alphabet of residues, and $|S|$ is its corresponding size, we represent the quantities as $|S|^2$-by-1 vectors. We let the log vector $\tilde{\mathbf{L}}(t_k)$ and the rate vector $\tilde{\mathbf{Q}}$ represent the vectorized versions of $\mathbf{L}(t_k)$ and $\mathbf{Q}$ respectively.

The rate vector $\tilde{\mathbf{Q}}$ and the divergence times t_k can be estimated simultaneously by choosing them to maximize the quantity

$$\sum_k ||\tilde{\mathbf{L}}(t_k) - t_k \tilde{\mathbf{Q}}||^2, \tag{2.49}$$

where the norm $||\cdot||$ of a vector is the square root of the sum of the squares of its entries. It is easy to see that equation 2.49 is maximized when $\tilde{\mathbf{L}}(t_k) = t_k\tilde{\mathbf{Q}}$ for all alignments k. Manipulating these equations, we get

$$\begin{aligned}
\tilde{\mathbf{L}}(t_k)^T &= t_k\tilde{\mathbf{Q}}^T && (2.50)\\
\tilde{\mathbf{L}}(t_k)^T\tilde{\mathbf{Q}} &= t_k||\tilde{\mathbf{Q}}||^2 && (2.51)\\
(\tilde{\mathbf{L}}(t_k)^T\tilde{\mathbf{Q}})\tilde{\mathbf{L}}(t_k) &= t_k^2||\tilde{\mathbf{Q}}||^2\tilde{\mathbf{Q}}. && (2.52)
\end{aligned}$$

Summing these equations yields

$$\sum_k(\tilde{\mathbf{L}}(t_k)^T\tilde{\mathbf{Q}})\tilde{\mathbf{L}}(t_k) = \sum_k t_k^2||\tilde{\mathbf{Q}}||^2\tilde{\mathbf{Q}} \quad (2.53)$$

Equation 2.53 shows that $\tilde{\mathbf{Q}}$ is an eigenvector of the linear mapping

$$\mathbf{A} \rightarrow \sum_k(\mathbf{A}(t_k)^T\tilde{\mathbf{Q}})\tilde{\mathbf{L}}(t_k), \quad (2.54)$$

with eigenvalue $\sum_k t_k^2||\tilde{\mathbf{Q}}||^2$. Note that $\mathbf{A}$ represents a $|S|^2$-by-1 vector. It was found in [9] that $\tilde{\mathbf{Q}}$ is the eigenvector corresponding to the top eigenvalue of the linear mapping.

The rate matrix is found through a principal component analysis of the given pairwise aligned sequences. Devauchelle et al. constructed a matrix $\mathbf{K}$ which rows are made up of the vectors $\tilde{\mathbf{L}}(t_k)^T$. Thus, $\mathbf{K}$ has dimensions n-by-$|S|^2$, where n is the number of "sufficiently related" alignments used in the estimation. In computing the singular value decomposition of $\mathbf{K}$, we compute the matrix $\mathbf{C} = \mathbf{K}^T\mathbf{K}$,

$$\begin{aligned}
\mathbf{K}^T\mathbf{K} &= \sum_k \tilde{\mathbf{L}}(t_k)\tilde{\mathbf{L}}(t_k)^T && (2.55)\\
&= \sum_k t_k^2\tilde{\mathbf{Q}}\tilde{\mathbf{Q}}^T && (2.56)\\
\mathbf{K}^T\mathbf{K}\tilde{\mathbf{Q}} &= \sum_k t_k^2||\tilde{\mathbf{Q}}||^2\tilde{\mathbf{Q}}. && (2.57)
\end{aligned}$$

This shows that the matrix $\mathbf{C}$ represents the linear mapping of equation 2.54. Thus, $\tilde{\mathbf{Q}}$ can be estimated by calculating the eigenvector corresponding to the largest eigenvalue of $\mathbf{C}$. $\tilde{\mathbf{Q}}$ can then be rearranged into matrix form to get the matrix $\mathbf{Q}$.

Notice that $\mathbf{Q}$ is only estimated up to a multiplicative scale factor, since eigenvectors are only defined up to a multiplicative scale factor. This corresponds once again to the fact that an absolute time scale cannot be estimated.

Alternate Algorithm

We present an alternate method for deriving an estimate $\tilde{\mathbf{Q}}$ given the estimates $\tilde{\mathbf{L}}(t_k)$. Consider first a matrix $\mathbf{A}$ with eigenvalues $\lambda_1, \ldots, \lambda_n$, arranged in decreasing order by magnitude, $|\lambda_1| > \cdots > |\lambda_n|$, and corresponding eigenvectors $\mathbf{v_1}, \ldots, \mathbf{v_n}$. Assuming that a unique largest eigenvalue (by magnitude) exists, then we have

$$\mathbf{A}^r \mathbf{v_1} \approx \lambda_1^r \mathbf{v_1} \tag{2.58}$$

when r is large. Hence, we can estimate the largest eigenvector $\mathbf{v_1}$ up to an unknown multiplicative scale factor by first guessing some initial vector and then repeatedly multiply it by the matrix $\mathbf{A}$.

We use the same approach to estimate $\tilde{\mathbf{Q}}$. We first make an initial guess $\tilde{\mathbf{Q}}_\mathbf{0}$; then, we iteratively apply the mapping of equation 2.54 to our guess. After each iteration m, we normalize our estimate to unit Frobenius norm to prevent the estimate from blowing up or decaying to zero, since it is being scaled by the largest eigenvalue at each step. Hence, the m-th iteration of the algorithm becomes

$$\tilde{\mathbf{Q}}_\mathbf{m} = \sum_k (\tilde{\mathbf{L}}(t_k)^T \tilde{\mathbf{Q}}_{\mathbf{m-1}}) \tilde{\mathbf{L}}(t_k) \tag{2.59}$$

$$\tilde{\mathbf{Q}}_\mathbf{m} = \tilde{\mathbf{Q}}'_\mathbf{m} / ||\tilde{\mathbf{Q}}'_\mathbf{m}||_F \tag{2.60}$$

This iteration converges assuming there is a unique largest eigenvalue by magnitude.

2.5.4 PMB

Veerasamy et al. [52] developed the Probability Matrix from Blocks (PMB) technique, which estimates a reversible rate matrix $\mathbf{Q}$ from the BLOSUM series of scoring matrices [19]. The BLOSUM technique, which was developed by Henikoff and Henikoff, estimates a series of scoring matrices for amino acid alignment. Veerasamy et al. used data from these scoring matrices to derive the corresponding Markov model of amino acid substitution.

BLOSUM Clustering Percentages

The BLOSUM technique derives scoring matrices from residue exchange counts observed in multiple alignments of sequences. The technique involves a clustering scheme to reduce biases from over-represented protein families. At the c% clustering level, each multiple alignment is partitioned into clusters such that each sequence in a cluster has c% or higher sequence identity to at least one other sequence in that cluster. Residue exchange counts are then taken from each cluster of sequences; a two-way counting scheme is used and the observed exchanges in each pairwise alignment of every cluster are tallied in the count matrix $\mathbf{N^{(c)}}$. The residue exchange counts are weighted such that each cluster contributes uniformly toward the substitution rate estimates regardless of the number of sequences it contains.

This clustering scheme results in scoring matrices sensitive at different evolutionary divergence times. When c is low, the sequences of each cluster are guaranteed to have very little identity to sequences in all other clusters of the same multiple alignment. The resulting residue exchange counts are then representative of a large divergence time. The opposite is true when c is high. The PMB technique takes advantage of this clustering scheme to obtain mutation matrix estimates $\mathbf{P}(t)$ for a wide range of divergence times t.

Estimating $\mathbf{P}(t_c)$

A mutation matrix $\mathbf{P}(t_c) = \{p_{ij}(t_c)\}$ is estimated for each BLOSUM clustering level c using the observed residue exchange count matrix $\mathbf{N^{(c)}} = \{n_{ij}^{(c)}\}$, where t_c represents the unknown time divergence implied at the $c\%$ clustering level,

$$p_{ij}(t_c) = \frac{n_{ij}^{(c)}}{\sum_j n_{ij}^{(c)}}. \tag{2.61}$$

Since the matrices $\mathbf{N^{(c)}}$ are constructed using a two-way counting scheme, the resulting mutation matrix estimates are reversible.

A vector of steady-state residue frequencies $\pi^{(c)} = \{\pi_i^{(c)}\}$ is also estimated for each clustering percentage from the observed residue frequencies,

$$\pi_i^{(c)} = \frac{\sum_j n_{ij}^{(c)}}{\sum_{i'} \sum_j n_{i'j}^{(c)}}. \tag{2.62}$$

Estimating t_c

To estimate the divergence time t, the PMB technique defines a quantity $D(t)$, defined to be the average probability of substitution implied by one step of the matrix $\mathbf{P}(t)$ and written as

$$D(t) = 1 - \sum_i \pi_i p_{ii}(t). \tag{2.63}$$

The quantity $D(t)$ can be directly calculated from observed sequence data, and its behavior with respect to t is fairly predictable. For small values of t, we expect $D(t)$ to increase linearly with t. For increasingly larger values of t, we expect $D(t)$ to level out to a constant. The PMB technique estimates an explicit equation relating $D(t)$ and t. The divergence times t_c are then estimated by calculating the quantities $D(t_c)$ and applying the estimated equation.

The relationship between $D(t)$ and t is estimated as follows. An expression for

$\frac{dD(t)}{dt}$ is first estimated using the five-point formula for numerical differentiation,

$$\frac{dD(t)}{t} = \frac{D(t-2h) - 8D(t-h) + 8D(t+h) - 12D(t+2h)}{12h}. \tag{2.64}$$

With h chosen to be $0.01t$, the expression simplifies to

$$t\frac{dD(t)}{t} = \frac{D(0.98t) - 8D(0.99t) + 8D(1.01t) - 12D(1.02t)}{0.12}. \tag{2.65}$$

The quantities $D(nt)$ can be calculated by applying the Chapman-Kolmogorov equation,

$$\begin{aligned} D(nt) &= 1 - \sum_i \pi_i p_{ii}(nt) && (2.66) \\ &= 1 - \sum_i \pi_i p_{ii}(t)^n. && (2.67) \end{aligned}$$

Using equations 2.65 and 2.67, the coordinate pair $(D(t_c), t_c\frac{dD(t)}{dt}|_{t=t_c})$ can be calculated for each clustering percentage c. The resulting coordinate points can be plotted on a coordinate plane of $D(t)$ versus $t\frac{dD(t)}{t}$ and fitted to a curve. Veerasamy et al. found that a cubic polynomial was a sufficient fit for the data from the BLOSUM count matrices, giving the resulting curve the expression

$$t\frac{dD(t)}{dt} = a_3 D(t)^3 + a_2 D(t)^2 + a_1 D(t) + a_0. \tag{2.68}$$

We can infer the boundary conditions

$$\lim_{t\to 0} D(t) = 0 \tag{2.69}$$

and

$$\lim_{t\to 0} \frac{dD(t)}{dt} = 1, \tag{2.70}$$

From these boundary conditions, it can be concluded that $a_0 = 0$ and $a_1 = 1$. This

simplifies the resulting curve to have the expression

$$\frac{dt}{t} = \frac{dD(t)}{a_3 D(t)^3 + a_2 D(t)^2 + D(t)}, \tag{2.71}$$

which can be easily solved since it is separable. This solution provides the relationship between $D(t)$ and t that was sought. The time divergences t_c are estimated by calculating $D(t_c)$ and applying the relationship given by the solution to equation 2.71.

Estimating Q

A rate matrix is estimated at each clustering level using the fact that

$$\mathbf{Q^{(c)}} = \frac{\log \mathbf{P}(t_c)}{t_c}. \tag{2.72}$$

Note again that logarithm of a matrix may not exist under certain circumstances.

From this collection of rate matrix estimates, the matrix that minimizes the expression

$$\sum_c \frac{||\exp(t_c \mathbf{Q^{(c)}}) - \mathbf{P_c}(t)||}{||\mathbf{P_c}(t)||} \tag{2.73}$$

is selected to be the universal rate matrix $\mathbf{Q}$. Veerasamy et al. defined matrix norm $||\mathbf{A}||$ to be the largest eigenvalue of $\mathbf{A}$.

Adaptation to Pairwise Alignment Data

The PMB method could also have been applied to a collection of pairwise aligned sequences. The estimates $\mathbf{P}(t_c)$, originally obtained from the observed residue exchanges at each clustering percentage c, can be replaced by $\mathbf{P}(t_k)$ obtained from observed frequency exchanges in each alignment k. The remainder of the technique can proceed exactly as described above. However, the differential equations yielded from different sets of observed alignments may be difficult to solve.

2.6 Estimation Techniques Based on Structural Information

All of the estimation techniques presented thus far use observed sequence data for calculating rate matrix estimates. For amino acid sequences, several estimation techniques that use protein structural information to derive models of sequence evolution and the related sequence alignment scoring matrices have also been developed. Some researchers have recommended the use of structural information to judge evolutionary relationships for protein sequences, because protein structures are better conserved than their corresponding sequences [7]. These estimation techniques are based more on analysis of biochemical properties of amino acids, in contrast to estimation techniques using only sequence data, which are based almost entirely on quantitative techniques. This thesis focuses on techniques using sequence data, but presents a brief introduction of techniques using structural data in this section.

2.6.1 Contact-Based Model of Protein Evolution

Lin et al. [30] developed the Contact Accepted mutatiOn (CAO) model, which describes a Markov model of amino acid side-chain contact mutation. This model considers not the substitution of individual amino acids, but rather, the interchanging of amino acids side-chain contacts within a protein. The CAO model consists of a 400 by 400 rate matrix corresponding to the 20 by 20 possible combinations of amino acids that make up the 400 possible contact combinations. This description of side-chain contact evolution is advantageous because it incorporates sequence and structural information into a single model.

A rate matrix for the CAO model has been estimated in [30] using Müller and Vingron's resolvent technique [36] and protein data from public databases. To further elaborate upon the idea of characterizing proteins by their side-chain contacts, Kleinjung et al. [28] have developed algorithms to perform pairwise sequence alignments based on contact information.

2.6.2 Scoring Matrices from Structural Alignments

We have shown that substitution scoring matrices can be calculated from observed residue exchanges in sequence alignments either directly, through the BLOSUM technique [19], or indirectly, through log odds-matrices [44]. Substitution scoring matrices have also been calculated through observed amino acid substitutions in the structural superposition of pairs of similar proteins by Risler et al. [42], Johnson and Overington [23], and Prlic et al. [39]. Amino acids on separate proteins that have been structurally superimposed are usually considered to be aligned if their C^{α} and/or C^{β} molecules are closer than some specified threshold. Each alignment serves as an observed amino acid substitution.

2.6.3 Scoring Matrices from Structural Properties

Various techniques have also been developed to use structural properties of amino acids derive substitution scoring matrices or enrich scoring matrices derived from observed sequence data.

Rao [41] developed a scoring matrix, called the Exchange Matrix derived from PARameters (EMPAR), based on physical characteristics of amino acid residues. The physical parameters used include the alpha helical, beta strand, and reverse turn propensity parameters, the residue polarity, and the consensus hydrophobicity.

Miyazawa and Jernigan [34] derived a scoring matrix by measuring the average stability of a protein's structure caused by an amino acid exchange. The degree of stability of a protein's structure was estimated by examining a protein's residue-residue contacts.

Dosztányi and Torda [10] estimated substitution scoring matrices based on force fields. Their technique involved associating an energy score to each residue within the context of some protein for an entire database of proteins. All sites with residue i in the database were then computationally mutated to residue j, and the average resulting energies at those sites were used to calculate a similarity score between residues i and j. Note that this estimation technique requires no evolutionary arguments.

Teodorescu et al. [48] derived a mixed scoring matrix through a linear combination of a BLOSUM matrix and threading energy information. Threading is the alignment of a protein sequence with a three-dimensional protein structure and is associated with an energy function that measures the quality of fit [33].

2.7 Extensions of the Elementary Markov Model

The elementary Markov model makes many simplifying assumptions on the process of sequence evolution. The realism of some of these assumptions has been studied and generalizations of the elementary model have been proposed and analyzed.

2.7.1 Insertions and Deletions

The Markov model of sequence evolution only considers the process of substitution and ignores insertions and deletions (collectively referred to as indels); indels have proven to be difficult to model [50]. However, models for insertions and deletions are important since they can provide a basis for scoring gap penalties in dynamic programming algorithms that infer optimal gapped alignments between a set of sequences [37], [45].

An analysis of indels was introduced in Gonnet et al. [16] and further elaborated on, by the same group, in Benner et al. [6]. They used results from an exhaustive matching of the entire sequence database to empirically describe indel probabilities in the context of pairwise alignments of homologous sequences. In particular, they characterized gap length distribution and the probability of having a gap as a function of evolutionary distance. These results were used to refine the scoring of gap penalties.

2.7.2 Rate Heterogeneity

The Markov model assumes that all sites in a sequence evolve according to the same Markov chain. This property has been studied in detail, and it has been widely recognized that such an assumption is unrealistic. Several studies have proposed

generalizations of the elementary Markov model where this assumption has relaxed to allow all sites to evolve according to the same Markov chain, but scaled to run at different rates. This property is commonly called rate heterogeneity.

Kelly and Rice [26] developed a model of sequence evolution that allows heterogeneous rates of substitution by defining a continuous nonnegative random variable to represent substitution rate. The relative rate of substitution at each site is a realization of this random variable. In their analysis, Kelly and Rice assumed gamma and log-normal distributions for this random variable and estimated the distribution parameters from observations of sequence data and phylogenetic trees using the maximum likelihood technique. They also considered the case where no distribution was assumed and calculated bounds on the mean and variance of the substitution rate.

Holmes and Rubin [22] modeled residue substitution in a sequence using a hidden Markov model. Hidden states were used to model a site's biophysical context to account for different selective pressures and allow for substitution rate heterogeneity. They also derived an expectation maximization algorithm for the maximum likelihood training of their model from observed sequence alignments.

2.7.3 Site Independence

The Markov model assumes that all sites in a sequence evolve independently of all other sites. This property has also been found to be unrealistic and has been partially overcome by modeling molecular evolution at the protein or codon level to account for site dependence at the DNA level. Several further generalizations have also been considered.

Gonnet et al. [15] tested this independence assumption by developing a Markov model of sequence evolution for dipeptides. Their model consists of 400-by-400 substitution matrices that report the transition probabilities between all pairs of dipeptides. Empirical results in their study revealed that amino acid substitution rates are correlated to the substitution rates of neighboring amino acids; the degree of correlation depends on the amino acid type.

Yang [56] developed a space-time model for DNA substitution that allows for

heterogeneity and correlated rates of substitution over sites. The time process consists of the elementary Markov model for describing nucleotide substitution, and the space process consists of a serially correlated gamma function that characterizes the variation and correlation of substitution rates over sites.

Chapter 3

Sequence Evolution Simulator

3.1 Overview

We developed an elementary model that simulates sequence evolution according to the preceding Markov model in a controlled setting. The simulation uses a Markov rate matrix $\mathbf{Q}^* = \{q^*_{ij}\}$ [1] and proceeds for time T, beginning from a given seed sequence. Substitutions and a binary branching process are simulated to construct a phylogenetic tree with the realized sequences at the leaves. The realized sequences can then be passed as input to techniques that estimate Markov model parameters to evaluate their accuracy in reconstructing $\mathbf{Q}^*$. Users can adjust the sequence length L, simulation time T, extinction rate x, and rate matrix $\mathbf{Q}^*$ to allow the simulation to represent different evolutionary conditions.

Classical methods of simulating sequence evolution (used in [9], [21], [29], and [40]) require as input a phylogenetic tree with specified branch lengths. Typically, the branch lengths are modified to denote the mean number of substitutions per site that will be realized along that branch. A root sequence and a Markov model of sequence evolution are also given. Evolution is simulated down each branch, beginning with the root and ending at the tips of the leaves. For each branch, a mutation matrix $\mathbf{P}(t)$ is constructed from the given Markov model and scaled such that the mean number

[1] We take $\mathbf{Q}^*$ to denote the underlying specified model and later use $\mathbf{Q}$ to denote the estimated model.

of substitutions per single application of the matrix matches the branch length. $\mathbf{P}(t)$ is then applied to the sequence at the top of the branch to create a sequence for the bottom of the branch.

Rambaut and Grassly introduced Seq-Gen in [40], and Grassly et al. introduced PSeq-Gen in [17], which are two software packages that implement DNA and protein sequence evolution respectively in this fashion. These software packages can both be downloaded at

http://evolve.zoo.ox.ac.uk/software.html.

The advantage of our simulator is that it does not require as input a phylogenetic tree. Instead, it generates a phylogenetic tree as part of the simulation according to a process that is coupled with the process of sequence mutation simulation, thus reflecting their expected dependencies in real life. Users of our simulator do not need to discover phylogenies in advance. In addition, with classical simulators, a simulation run always uses the same phylogenetic tree unless the user manually inputs a new one. With our simulator, the phylogenetic tree varies for each simulation run, since it is generated by a random process.

3.2 Simulation Algorithm

3.2.1 Simulating Point Mutations

Sequence mutations are simulated assuming an elementary Markov model of sequence evolution. All sites evolve independently, but governed by the same rate matrix $\mathbf{Q}^*$. Suppose a given site is currently in state i, i.e., residue i is currently in that position. A continuous-time Markov process states that the site will remain in state i for a time period that is exponentially distributed with parameter $|q^*_{ii}|$. Our simulator mimics this action. Since software simulations take place in discrete-time, we use Bernoulli trials to approximate the Poisson arrivals that define the exponentially distributed holding times at each site. Suppose Δt represents the time elapsed during each discrete simulation step. The Poisson process can then be approximated by a

Bernoulli process with parameter $|q^*_{ii}|\Delta t$. Thus, at each simulation step for each site in a sequence, a random number is uniformly generated between 0 and 1. If this random number is less than $|q^*_{ii}|\Delta t$, then a substitution occurs at that site.

Given that a substitution has occurred, the replacing residue is selected according to the embedded discrete-time Markov chain $\mathbf{D}^* = \{d^*_{ij}\}$ defined by

$$d^*_{ij} = \left\{ \begin{array}{ll} 0, & i = j \\ q^*_{ij}/|q^*_{ii}|, & i \neq j \end{array} \right\}. \tag{3.1}$$

Residue replacement is simulated by generating a random number that is uniformly distributed between 0 and 1. If the random number is less than d^*_{i1}, residue 1 replaces residue i. Otherwise, if the random number is less that $d^*_{i1} + d^*_{i2}$, residue 2 replaces residue i. This process repeates until a replacing residue is found. Note that since $\sum_j d^*_{ij} = 1$, a replacing residue must be found, and since $d^*_{ii} = 0$, the replacing residue cannot be the same residue.

3.2.2 Simulating Extinction

At each simulation step, for an entire sequence, a Bernoulli trial with parameter x is simulated to determine whether or not that sequence becomes extinct. If a sequence becomes extinct, it is removed from the simulation. If the sequence survives, it ia then considered site-by-site for point mutations. We assume that extinction occurs at a uniform rate x for all time and over all sequences, for simplicity. Extinction can be similarly simulated by generating a random number uniformly distributed between 0 and 1. If this random number is less than x, extinction occurs.

3.2.3 Phylogenetic Tree Generation

A phylogenetic tree is constructed during the simulation through a binary branching process. Each sequence is represented by a branch extending from a node. The starting seed sequence is represented by a branch extending from the root. During each simulation step, all branches are extended by length Δt. If a sequence undergoes

mutation at any of its sites during a step, a split is generated in the corresponding branch; the original parent sequence occupies one branch and the mutated sequence occupies the other. Both the original and mutated sequences continue involving independently along their respective branches. If a sequence becomes extinct during a step, the corresponding branch ceases to grow. The leaves of the phylogenetic tree at the bottom-most layer will thus represent the surviving sequences. Figure 3-1 illustrates this branching process.

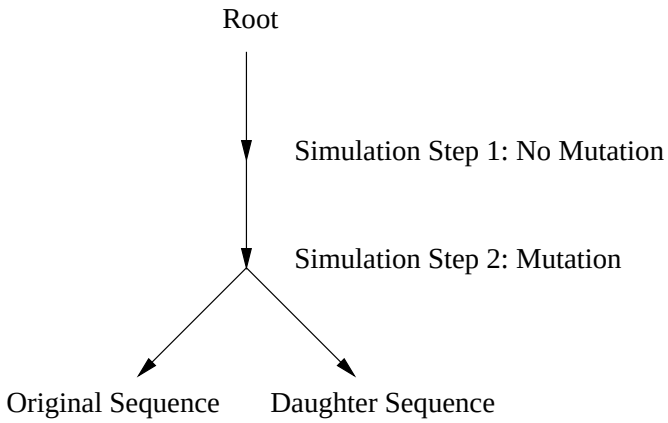


Figure 3-1: Phylogenetic tree generation.

3.2.4 The Evolution Process

Figure 3-2 shows the pseudocode that implements our sequence evolution simulation algorithm.

```
for time <- T/△t
    for i <- all sequences
        SIM_EXTINCT(seq(i))
        if extinct -> remove seq(i)
        for j <- all sites
            SIM_MUTATION(seq(i), site(j))
            if mutate -> add mutated seq
        end
    end
end
```

Figure 3-2: Pseudocode for Sequence Evolution Algorithm.

The elementary model used here can be refined in various ways to better represent

features of evolution in different contexts. For example, an interesting question to consider is whether or not it is appropriate to branch at a simulation step when substitution takes place at only one site in a sequence. The answer is dependent on the semantics of the simulation. Having branching even for single site mutations is representative of a microscopic context, where perhaps, each branch represents a single strand of DNA or protein in a cell, and each simulation step represents DNA replication or protein synthesis. In this context, it seems appropriate to differentiate even single residue differences.

On the other hand, one might also consider a macroscopic context where each branch represents a species, and each branching event represents the process of speciation. In this context, branching should only occur when sequences are sufficiently different. The Extensions section discusses algorithms more representative of this context. For our initial evaluations, the elementary simulator is sufficiently rich.

3.3 Evaluation of Accuracy

The realized sequences of the simulation are passed as input to various estimation techniques to evaluate their accuracy in reconstructing $\mathbf{Q}^*$. Rate matrix estimates $\mathbf{Q}$ are typically calculated with arbitrary time scales, corresponding to the fact that absolute time scales cannot be estimated by observing sequences from one point in time. Hence the estimate $\mathbf{Q}$ is usually on a different time scale than the true rate matrix $\mathbf{Q}^*$ assumed in the simulation. In order to compare $\mathbf{Q}$ with $\mathbf{Q}^*$, a scale factor ρ must be determined such that $\rho\mathbf{Q}$ and $\mathbf{Q}^*$ are on the same time scale. We use a Frobenius-norm matching to determine the appropriate scale factor ρ. The Frobenius-norm of a matrix is given by equation 2.47.

We select the optimal scale factor as the choice of ρ that minimizes the quantity $||\rho\mathbf{Q} - \mathbf{Q}^*||_F^2$. It can easily be determined that the optimal choice of ρ is calculated using the expression

$$\rho = \frac{\sum_{ij} q_{ij} q^*_{ij}}{\sum_{ij} q_{ij}}. \tag{3.2}$$

We use the relative error

$$f = \frac{||\rho\mathbf{Q} - \mathbf{Q}^*||_F}{||\mathbf{Q}^*||_F} \tag{3.3}$$

as a metric to denote the accuracy of the estimated rate matrix $\mathbf{Q}$ with respect to the rate matrix $\mathbf{Q}^*$ used in the simulation.

3.4 Order of Growth

Sequence evolution in our simulator follows a Galton-Watson branching process, which is discussed in [47]. The simulation begins with a population of size $Z_0 = 1$ sequence of length L. The simulation proceeds for time T in $T/\Delta t$ independent simulation steps. During a simulation step, an offspring generating function is applied to each sequence, which yields either 0, 1, or 2 sequences depending on whether or not extinction or mutations occurred. The event tree of a single simulation step is depicted in figure 3-3. Note that extinction occurs with probability x for an entire sequence, and is independent of site-by-site mutations. We let p denote the probability that a single sequence splits into two sequences. Since a split occurs if and only if at least one residue in a sequence mutates, p has the value

$$p = 1 - (1 - \sum_i \pi_i |q^*_{ii}| \Delta t)^L, \tag{3.4}$$

where $\mathbf{Q}^* = \{q^*_{ij}\}$ is the assumed Markov model of sequence evolution.

Let Z_n denote the size of the population after n steps of simulation. Using the techniques in [47], we can characterize the survival probability of the population and the order of growth of Z_n in terms of its mean and variance. We begin by calculating

the expected number of sequences after one simulation step:

$$\begin{aligned} \mathrm{E}[Z_1] &= \sum_{z=0}^{2} sP(Z_1 = z) && (3.5) \\ &= (1-p)(1-x) + 2p(1-x) && (3.6) \\ &= (1+p)(1-x). && (3.7) \end{aligned}$$

In the terms of the Galton-Watson process, $(1+p)(1-x)$ is the mean number of offspring produced per parent. If $(1+p)(1-x)$ is ≤ 1, then this population will eventually die out with certainty. On the contrary, if $(1+p)(1-x)$ is > 1, then there is a non-zero probability that this population will survive forever.

To calculate the mean and variance of Z_n for all $n \geq 2$, we must calculate the probability generating function (pgf) of the offspring generation function. The probabilities of yielding 0, 1, or 2 offspring at each simulation step are given by $p_0 = x$, $p_1 = (1-p)(1-x)$, and $p_2 = p(1-x)$ respectively. The pgf of applying the offspring generating function for one simulation step can then be expressed as

$$\begin{aligned} G_1(s) &= \sum_{k=0}^{2} p_k s^k && (3.8) \\ &= x + (1-p)(1-x)s + p(1-x)s^2. && (3.9) \end{aligned}$$

The pgf of applying offspring generating function for two simulation steps is given by $G_2(s) = G_1(G_1(s))$, and the pgf of applying offspring generating function for three simulation steps is given by $G_3(s) = G_1(G_1(G_1(s)))$. This technique allows us to calculate the pgf, $G_n(s)$, after any number of simulation steps n. Note that the k-th factorial moment of a discrete random variable X, $\mathrm{E}[X(X-1)\cdots(X-k+1)]$, can be calculated by taking the k-th derivative of its pgf with respect to s and evaluating the resulting quantity at $s = 1$. Hence, the mean and variance of Z_n can be obtained from the expressions

$$\mathrm{E}[Z_n] = \frac{d}{ds} G_n(s)|_{s=1} \qquad (3.10)$$

and

$$\mathrm{Var}[Z_n] \;=\; \frac{d^2}{ds^2}G_n(s)|_{s=1} + \frac{d}{ds}G_n(s)|_{s=1} - (\frac{d}{ds}G_n(s)|_{s=1})^2. \qquad (3.11)$$

We can qualitatively see that increasing sequence length L or increasing the elements of the rate matrix $\mathbf{Q}^*$ increases the single step mutation probability p, which in turn increases the population's survival probability and the mean number of sequences after n simulation steps. Increasing extinction rate x has the opposite effect.

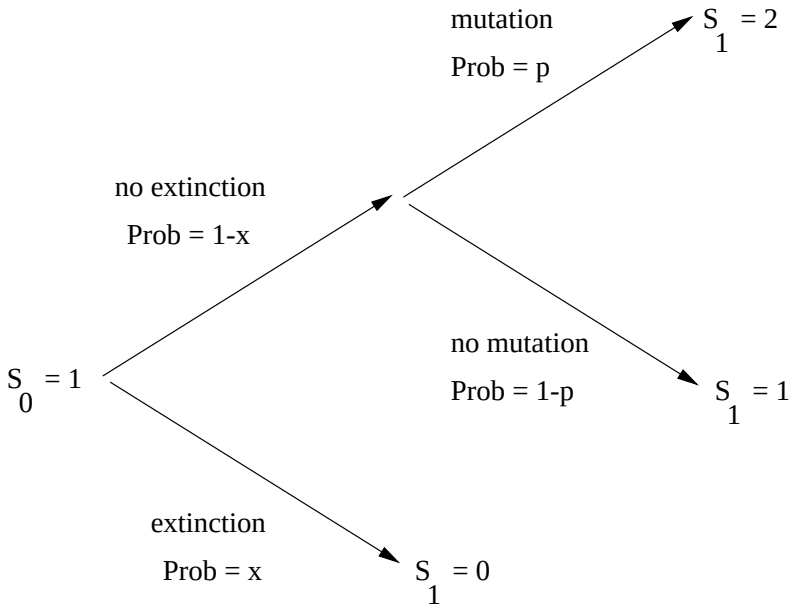


Figure 3-3: Events of a single simulation step.

Chapter 4

Evaluation of Arvestad and Bruno's Technique

4.1 Overview

We used our simulation method to evaluate Arvestad and Bruno's estimation technique [4]. Multiple datasets of sequences were generated using the preceding simulation algorithm under varying set of parameters, which represent varying evolutionary conditions. The Arvestad and Bruno technique was used to estimate Markov models from these datasets. The relative error metric shown in expression 3.3 was calculated for each Markov model estimate. The empirical distributions of the resulting relative errors were observed to evaluate the accuracy of the Arvestad and Bruno technique.

In addition, we considered adaptations to the Arvestad and Bruno technique and our simulation algorithm and examined their impact on the relative error metric. An advantage of a simulation approach to evaluation is that it allows us to empirically evaluate the effect of even minor modifications on estimation accuracy. The effects of such modifications are often difficult or tedious to characterize analytically.

4.2 Accuracy Across Simulation Parameters

We used Arvestad and Bruno's technique [4] to estimate the rate matrix $\mathbf{Q}^*$ used in our simulation in four separate experiments. In each experiment, a single simulation parameter was varied across a specified range, and all other parameters were held constant at their default values. In the first experiment, sequence length L was varied from 25 to 150 in steps of 25; in the second experiment, the extinction rate x was varied from 0% to 15% in steps of 3%; in the third experiment, simulation time T was varied from 3 to 7 in steps of 1; in the forth experiment, time scale c was varied from 0.1 to 0.9 in steps of 0.1. These experiments allowed us to observe the sensitivity of the technique's accuracy across various simulation parameters.

The time scale variable was implemented in the simulator to provide an easy way to uniformly scale the mutation rates. Upon changing the time scale, the maximum off-diagonal element of the rate matrix is set to the new time scale, and all other elements are scaled according to their ratios with the maximum off-diagonal term.

Table 4.1 shows the parameters used. Note that the rate matrix $\mathbf{Q}^*$ is a randomly generated reversible rate matrix, and the initial distribution of residues corresponds to its steady state distribution. The seed sequence was randomly generated for each experiment; the residue at each site of the seed sequence was randomly chosen, independently of all other sites, and according to the probabilities of the initial distribution. The $\mathbf{Q}^*$ displayed in table 4.1 was scaled according to the time scale value $c = 0.3$. In experiment 4, $\mathbf{Q}^*$ was rescaled appropriately for each value of c.

For each experiment, 100 simulations were run under each set of parameters. For each simulation sun, all possible pairs of realized sequences were (trivially) aligned site-by-site and passed to the Arvestad and Bruno technique to estimate $\mathbf{Q}^*$. Note that alignments can be constructed in this manner, because corresponding sites in two sequences are known to have evolved from a common ancestor given that no insertions and deletions were simulated. If the number of surviving sequences was less than a certain threshold for any simulation, the run was repeated to avoid calling the estimation algorithm on a sparse dataset. The threshold was set to 5 sequences

Table 4.1: Simulation Parameters.

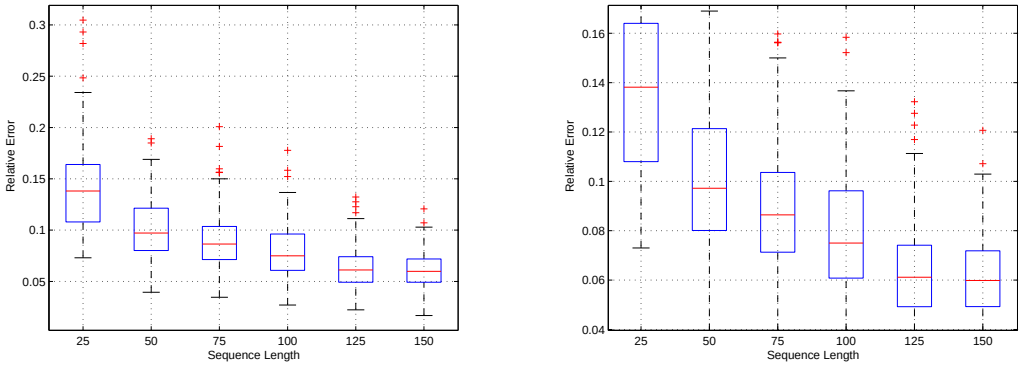
Sequence Type	DNA
Sequence Length L	Expt. 1: 25-150; Default: 100
Extinction Rate x	Expt. 2: 0%-15%; Default: 2%
Total Evolution Time T	Expt. 3: 3-7; Default: 6
Time Scale c	Expt. 4: 0.1-0.9; Default: 0.3
Discrete Step Time Δt	1
Initial Distribution π^*	$\begin{bmatrix} 0.443 & 0.195 & 0.188 & 0.175 \end{bmatrix}$
Rate Matrix $\mathbf{Q}^*$	$\begin{bmatrix} -0.278 & 0.065 & 0.095 & 0.119 \\ 0.148 & -0.349 & 0.114 & 0.088 \\ 0.224 & 0.118 & -0.404 & 0.062 \\ 0.300 & 0.097 & 0.067 & -0.464 \end{bmatrix}$

for these experiments.

Figures 4-1 to 4-4 show boxplots of the distributions of the relative errors for each experiment. Note that the box represents the interquartile range, the line inside the box represents the median, and the data points outside the whiskers are considered outliers. The plots were generated using the *boxplot* command in Matlab version 7. Boxplots are described in detail by Vanderviere and Huber in [51]. For each figure, the left plot is zoomed to show the entire boxplot distribution and the outliers, while the right plot is zoomed for a closer view of the interquartile ranges.

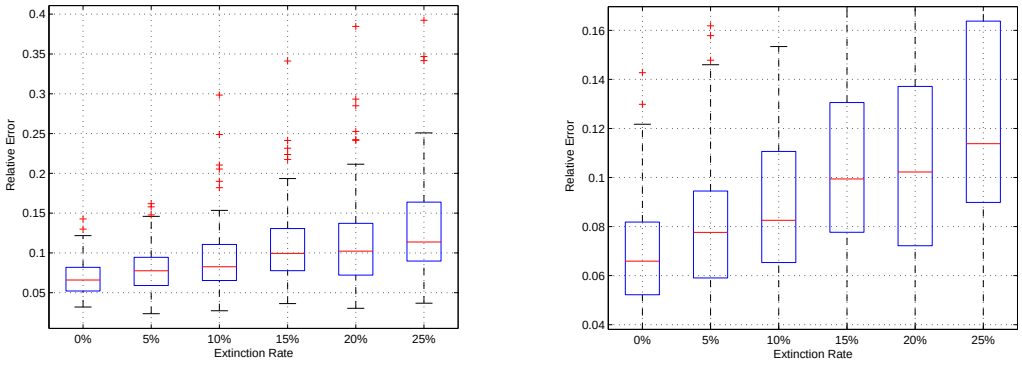
Experiments 1 through 3 show that the accuracy of Arvestad and Bruno's technique improves as the amount of input data increases. Increasing sequence length and simulation time increases the amount of realized sequences from the simulation. Note that an increase in sequence length leads to an increase in input data in two ways: first, it leads to an increase in the number of sites in each sequence; second, it increases the probability of branching at each simulation step, which can be calculated as $p = 1 - (1 - \sum_i \pi_i |q^*_{ii}| \Delta t)^L$. An increase in extinction rate conversely leads to a decrease in the amount of input data.

An increase in input data improves mutation matrix estimates $\mathbf{P}(t_k)$ taken from pairwise alignments k. Using the two-way count scheme, each element of the exchange matrix $\mathbf{P}(t_k)$ is estimated as $p^{(k)}_{ij} = n^{(k)}_{ij}/n^{(k)}_i$, where $n^{(k)}_{ij}$ is the number of times in



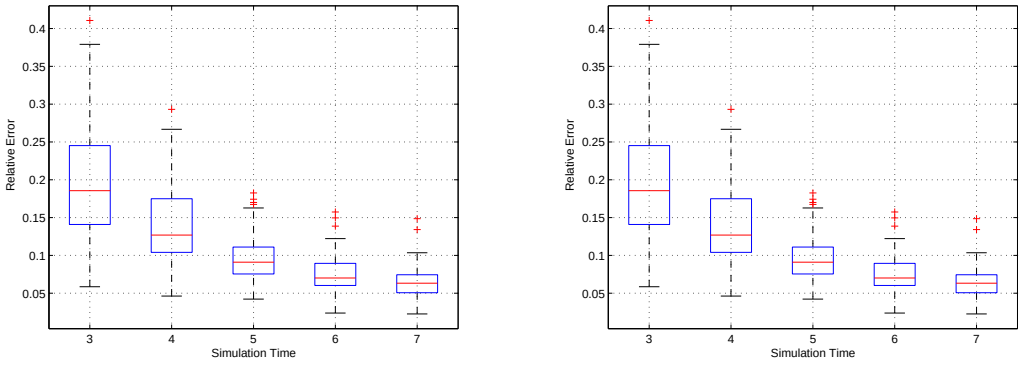
(a) full view **(b)** zoomed in

Figure 4-1: Relative errors across sequence length L.



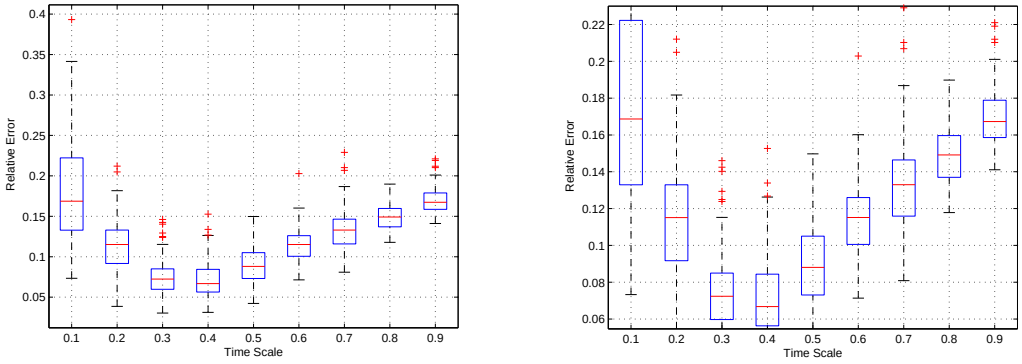
(a) full view **(b)** zoomed in

Figure 4-2: Relative errors across extinction rate x.



(a) full view **(b)** zoomed in

Figure 4-3: Relative errors across total simulation time T.



(a) full view **(b)** zoomed in

Figure 4-4: Relative errors across time scale c.

alignment k that residue i aligns with residue j in any order, and $n_i^{(k)}$ is the total number of occurrences of residue i in both sequences of alignment k. The variance of each estimate $p_{ij}^{(k)}$ is

$$\frac{p_{ij}^{(k)}(1 - p_{ij}^{(k)})}{n_i^{(k)}}. \tag{4.1}$$

With more input data, this variance becomes smaller, and we can expect more accurate estimates of $\mathbf{P}(t_k)$. Each estimate $\mathbf{P}(t_k)$ is directly used to estimate the eigenvectors and eigenvalues of the rate matrix. Hence, improving the accuracy of $\mathbf{P}(t_k)$ is likely to improve the accuracy of the rate matrix estimate, which the experimental results demonstrate.

From our results in experiment 2, we can see that the performance of Arvestad and Bruno's technique decreases steadily with increasing extinction rate. However, note that, since a simulation was rerun if it yielded less than the threshold of 5 sequences, we are unable to evaluate and compare estimation results from parameter sets that are expected to yield less than 5 realized sequences per simulation run. If we were to further increase extinction rate, thus reducing the expected number of surviving sequences, we would reveal less and less information regarding its effect on estimation accuracy, because the true expected number of surviving sequences would be masked by the threshold.

The effect of time scale on estimation accuracy varies, as shown in experiment 4. One might expect, at first, that increasing time scale increases the probability of branching at each simulation step p, which in turn increases the amount of sequences available for input data, which increases estimation accuracy. The results support this hypothesis for low values of c. For high values of c, however, estimation accuracy actually becomes worse. A likely reason for this behavior is that, since substitution rates increase for high values of c, the expected number of substitutions at internal nodes of the phylogenetic tree increases. Recall that, in Arvestad and Bruno's estimation technique, each estimate $\mathbf{P}(t_k)$ is taken from an alignment composed of two sequences at the leaves of the tree and represents the evolutionary process that travels

from one sequence, up to the common ancestor, and down to the other sequence. The estimate $\mathbf{P}(t_k)$ becomes less accurate as the number of residue mutations along its path increases, because the estimate is based only on observing the end points of the path. The substitutions at the internal nodes are unobserved. Thus, the quality of our estimates becomes worse when substitution rates increase.

This argument also suggests that estimation accuracy eventually decreases if total simulation time is further increased. At higher simulation times, the depth of the phylogenetic tree increases; hence, the expected length of the path between the two sequences of every alignment increases; hence, the expected number of mutations along every path increases. Estimates of $\mathbf{P}(t_k)$ would thus be more biased. This argument will be examined in future simulation experiments.

4.3 Accuracy Benchmarks

To gauge the quality of the relative error distributions observed in the previous experiments, we performed two benchmark experiments. One set of experiments involved randomly generating 300 reversible instantaneous rate matrices $\mathbf{Q}$ and observing how well $\rho\mathbf{Q}$ matched $\mathbf{Q}^*$ for the optimum ρ. One would anticipate that the relative error in this case would be quite poor.

The other set of experiments involved applying Dayhoff's PAM technique [8] to the phylogenetic trees constructed from the simulations. We generated 300 simulation runs using the default parameters in table 4.1, i.e. $L = 100$, $x = 2\%$, $T = 6$, and $c = 0.3$. For each resulting phylogenetic tree, observed residue frequencies were counted across each branch from root to leaf. All counts were summed together to obtain an estimate $\mathbf{P}(t)$ for each tree. A scaled rate matrix was estimated using $t\mathbf{Q} = \log(\mathbf{P}(t))$. We then observed how well $\rho t\mathbf{Q}$ matched $\mathbf{Q}^*$ for the optimum ρ. We call this technique PAM*. We expect this technique to yield an accurate estimate since we use knowledge of the actual phylogenetic tree and sequences at its internal nodes.

Note that the simulation parameters used imply high probabilities of mutation,

$p \approx 1$, and thus yield phylogenetic trees that branched at nearly every step and have almost all branches of equal length. Hence, the observed exchange frequency counts across each branch can be appropriately summed together for a single $\mathbf{P}(t)$ estimate.

Fig. 4-5 shows the distribution of the relative errors (on a log scale) of the benchmark experiments. For comparison, it also shows the distribution of relative errors of the Arvestad and Bruno technique evaluated under the same parameter set as the one used for the PAM* experiments. This distribution was compiled from the results of experiments 1-4, described in the previous section, and includes experiment 1 with $L = 100$, experiment 3 with $T = 6$, and experiment 4 with $c = 0.3$.

We can see from these plots that the median values of the benchmark experiments are just below 0.50 and 0.06 respectively. The Arvestad and Bruno technique achieves a median value of just more than 0.07 under the same parameter set. Thus, the Arvestad and Bruno technique's accuracy is close to the accuracy of the PAM* technique, which uses knowledge of the simulated phylogenetic tree and all sequences at the tree's internal nodes. This demonstrates the merit of the Arvestad and Bruno technique!

4.4 An Alternate Way of Estimating Eigenvalues

We developed an alternate version of Arvestad and Bruno's technique, where we modified the method by which the eigenvalues were estimated. We evaluated the

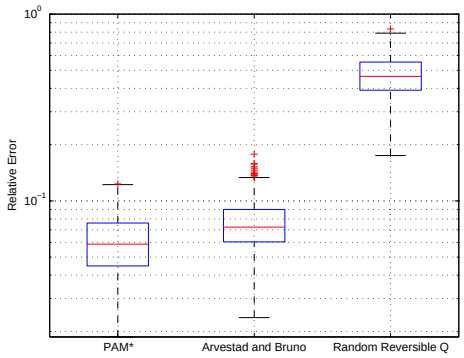


Figure 4-5: Relative errors of the benchmark experiments.

alternate form of the technique with our simulator and compared its performance with the performance of the original technique.

Recall from chapter 2 that the original Arvestad and Bruno technique yields weighted eigenvalue estimates $\mathbf{\Lambda} t_k$, for each alignment k. These weighted eigenvalue estimates are then used to estimate eigenvalue ratios using a linear regression through the origin:

$$\lambda_r/\lambda_s = \frac{\sum_{\text{all alignments } k}(\lambda_r t_k)(\lambda_s t_k)}{\sum_{\text{all alignments } k}(\lambda_s t_k)^2}. \tag{4.2}$$

One of the eigenvalues is known to be 0. For the remaining eigenvalues, one is arbitrarily chosen to be -1, and the others are estimated according to the ratios λ_r/λ_s.

Thus, Arvestad and Bruno obtained estimates of eigenvalue ratios by averaging over all times t_k. Our modification involves first estimating time ratios t_u/t_v by averaging over all eigenvalues. This can be similarly done using a linear regression through the origin:

$$t_u/t_v = \frac{\sum_{i=1}^{4}(\lambda_i t_u)(\lambda_i t_v)}{\sum_{i=1}^{4}(\lambda_i t_v)^2}. \tag{4.3}$$

We arbitrarily set one of the time variables to unity, $t_1 = 1$. The remaining times t_k are set according to the time ratios t_k/t_1. The weighted eigenvalue estimates $\mathbf{\Lambda} t_k$ are then divided by the corresponding estimate of t_k to normalize all eigenvalue estimates to the same time scale. The individual eigenvalue estimates are then averaged to obtain an aggregate estimate. Assuming there are n total alignments, each eigenvalue estimate can be expressed as

$$\mathbf{\Lambda} = \frac{\sum_k \frac{\mathbf{\Lambda} t_k}{t_k}}{n}. \tag{4.4}$$

To compare the alternate Arvestad and Bruno technique with the original technique, we ran 100 iterations of our simulation with the default parameters shown in table 4.1. We used both versions of the technique to estimate $\mathbf{Q}^*$ for each simulated

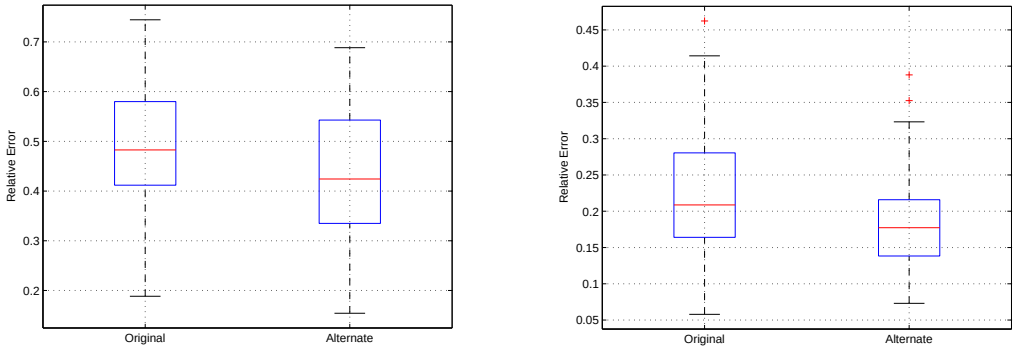
dataset and recorded their resulting relative errors. This experiment was repeated for four different time scales, $c = \{0.01, 0.03, 0.05, 0.3\}$, to gauge the effect of the modification under varying substitution rates. Figure 4-6 shows the distribution of the relative errors for the original and alternate techniques under the different time scales.

The results show that the alternate version of the technique has a lower median relative error for the three experiments with the smaller time scale values. This suggests that the alternate version of the technique improves estimation accuracy at sufficiently low substitution rates. Analytically, it would be quite tedious to determine the exact source of improvement of the alternate technique over the original. One possibility is that the alternate technique involves two stages of averaging, thereby eliminating more noise in the eigenvalue estimate. Using our simulator, however, performance variations from even subtle modifications can be identified through empirical analysis.

To further characterize the improvement of the alternate version of the technique over the original one, figures 4-7 through 4-10 show the distributions of the eigenvalue estimates yielded for each version of the technique at each time scale considered. Each axis on the 3-D scatter plot represents one of the non-zero eigenvalues ranked in increasing order. The * marks the position of the eigenvalues of the true rate matrix, and every other point marks the eigenvalue positions of the rate matrix estimates. These plots show that the eigenvalue estimates of the alternate version of the technique are a bit more condensed around the true eigenvalue position at the three lower time scales. The improvement is most noticeable for $c = 0.03$ and $c = 0.05$.

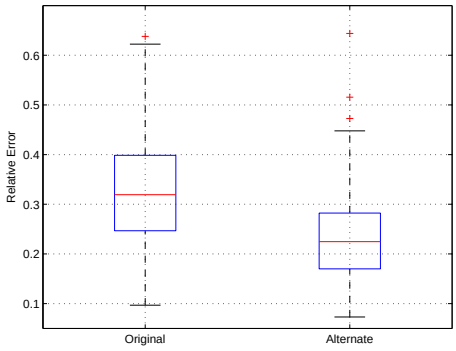
4.5 Incorporating Population Frequencies

We modified our simulation algorithm to account for population frequencies in sequences and examined its impact on the accuracy of Arvestad and Bruno's technique. Originally, we only simulate branching when at least one site in a sequence mutates during a simulation step. In the modified simulator, we branch at every simulation

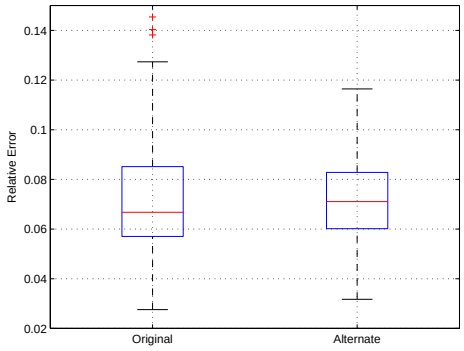


(a) $c = 0.01$

(b) $c = 0.03$

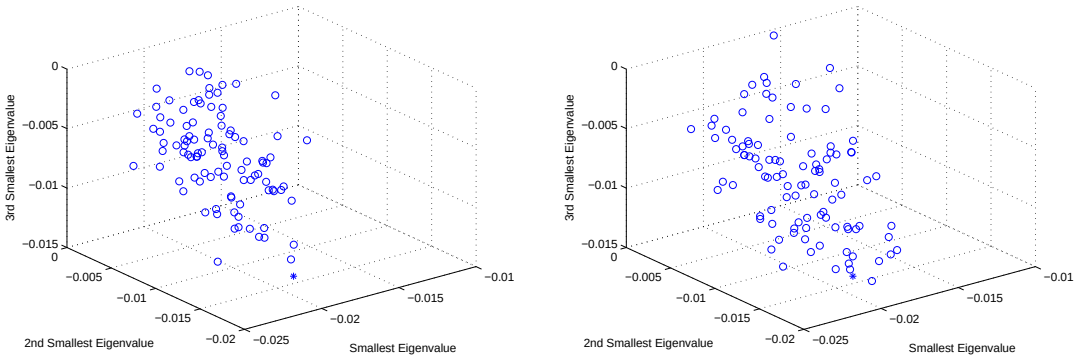


(c) $c = 0.05$



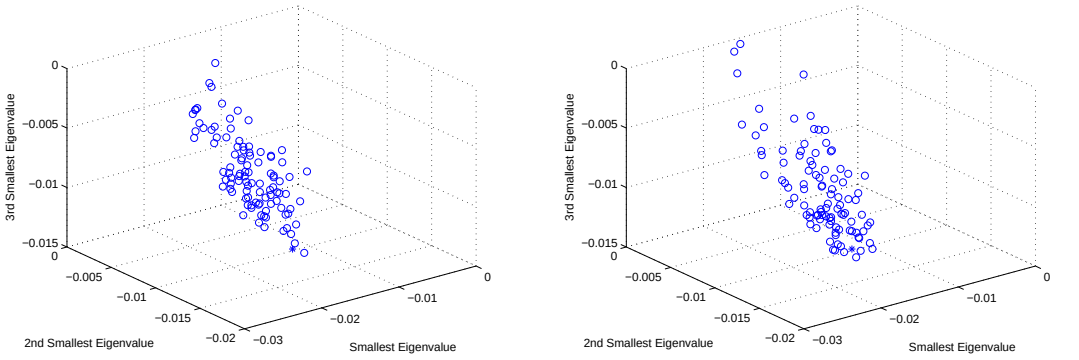
(d) $c = 0.3$

Figure 4-6: Relative errors of the alternate version of Arvestad and Bruno technique.



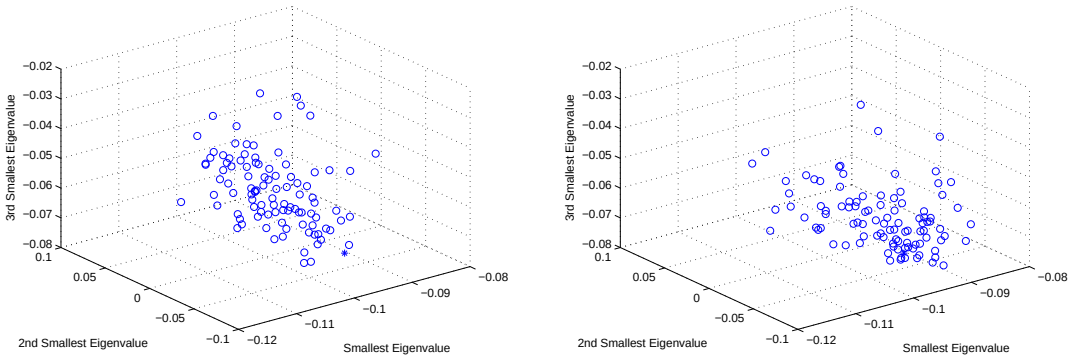
(a) Original **(b)** Alternate

Figure 4-7: Eigenvalues distributions yielded by the alternate and original techniques at $c = 0.01$.



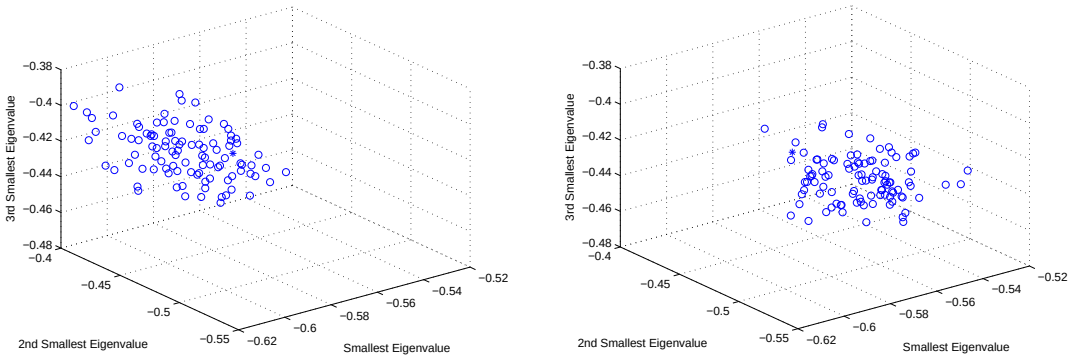
(a) Original **(b)** Alternate

Figure 4-8: Eigenvalues distributions yielded by the alternate and original techniques at $c = 0.03$.



(a) Original **(b)** Alternate

Figure 4-9: Eigenvalues distributions yielded by the alternate and original techniques at $c = 0.05$.



(a) Original **(b)** Alternate

Figure 4-10: Eigenvalues distributions yielded by the alternate and original techniques at $c = 0.3$.

step regardless of whether there is a mutation or not. The modified algorithm seems to be more characteristic of real life scenarios such as DNA replication; when proliferation occurs, certain daughter sequences are identical to the parent sequence, and other daughters are mutated versions of their parents. The modification also allows the realized datasets to account for frequencies in our realized population of sequences. We are interested in whether or not this additional information can lead to improvements in estimation accuracy.

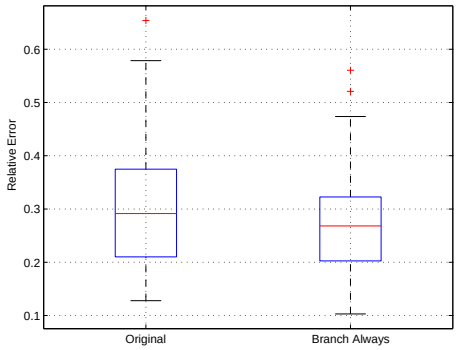
We ran 200 iterations of our simulation; the first 100 iterations were run under the original algorithm, and the second 100 iterations were run under the modified algorithm where branching was always simulated. The simulation parameters used in this experiment were the default parameters shown in table 4.1, except time scale, for which we used $c = 0.01$. We needed to pick a low time scale for this experiment to ensure that the probability of branching for a single simulation step p is sufficiently low. If p is high, branching would almost always occur, and there would be virtually no differences between the data yielded from both simulations. A time scale of $c = 0.01$ yields a rate matrix

$$\mathbf{Q}^* = \begin{bmatrix} -0.0093 & 0.0022 & 0.0032 & 0.0040 \\ 0.0049 & -0.0116 & 0.0038 & 0.0029 \\ 0.0075 & 0.0039 & -0.0135 & 0.0021 \\ 0.0100 & 0.0032 & 0.0022 & -0.0155 \end{bmatrix}, \tag{4.5}$$

and a p value of 0.6888, whereas a time scale of $c = 0.3$ yields a p value that is very close to 1.

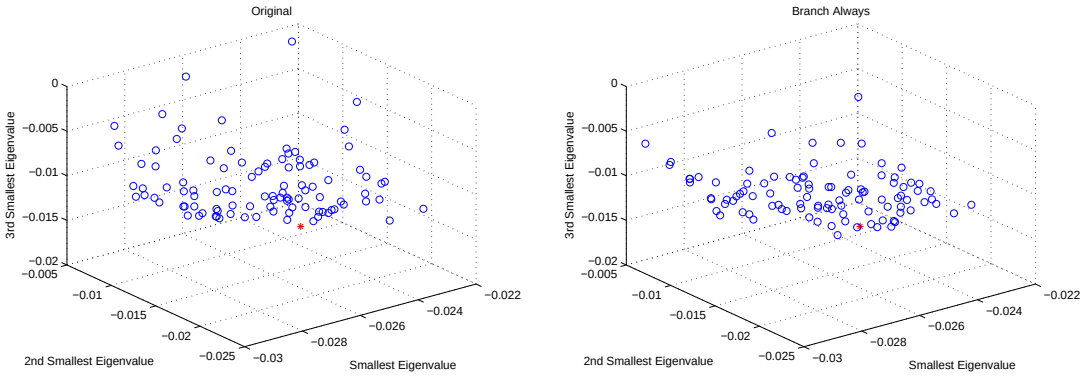
Figure 4-11 shows the distribution of the relative errors under each simulation algorithm. We can see that the relative errors under the branch always version of the simulation algorithm are lower. Figure 4-12 shows the 3-D scatter plots of the distribution of the eigenvalue estimates under each simulation algorithm. As before, the * indicates the eigenvalue position of the true rate matrix, and each axis represents one eigenvalue ranked in increasing order. We can see that the eigenvalues of the estimates under the simulation algorithm that always branches are more condensed

around the true rate matrix. These initial experiments to indicate that populations frequencies can help to improve Markov model estimation accuracy. Further experiments are necessary to determine whether or not these improvements are simply because the branch always algorithm yields more estimation data.



(a)

Figure 4-11: Relative errors of the original and branch always simulation algorithms.



(a) Original **(b)** Branch Always

Figure 4-12: Eigenvalue distributions yielded by each simulation algorithm.

Chapter 5

Conclusion

We have presented the Markov model of sequence evolution and provided a review of several important techniques that have been developed to estimate its parameters from observed alignments of sequence data. We have also introduced techniques that incorporate structural properties of proteins into estimates of amino acid substitution rates and discussed generalizations of the elementary Markov model, which relax the assumptions of rate homogeneity, site independence, and no indels. Such a model is useful in inferring protein functionality and genetic relationships, and in understanding the process of evolution at the molecular level.

We introduced a new method of simulating sequence evolution under an elementary Markov model, which is different from classical methods that require as input a known phylogenetic tree and simulate sequence mutations along the branches of the tree. Our simulator makes no *a priori* assumptions about phylogenetic relationships and generates a phylogenetic tree through a branching process, whose rules are governed by simulated sequence mutations.

The key utility of such a simulator is that it enables the user to evaluate and compare different techniques that estimate parameters of a Markov model of sequence evolution. The simulator can be continuously re-run to generate sequence families to be used as estimation data. Estimation results allow us to gain a sense of a technique's performance distribution and performance under varying parameter sets. A major benefit of using our simulator in evaluations is that it eliminates the need to

determine phylogenies in advance, which is a problem that has been faced by several other evaluation techniques [21], [29].

We applied our evaluation technique to Arvestad and Bruno's estimation technique [4] and found that it performed quite well given sufficiently rich data. The accuracy values yielded by this technique approach the accuracy values yielded from the PAM* technique, which use full knowledge of the simulated phylogenetic tree. In addition, our simulator was able to empirically characterize performance differences in the modifications made to Arvestad and Bruno's estimation technique and the evolution simulation algorithm. Such characterizations are usually quite tedious using a purely analytical approach. Similar experiments can be designed to study estimation performance under different circumstances or for other estimation techniques discussed in chapter 2.

In conclusion, we are delighted to present our findings in this exciting field of research. We hope our initial analyses and experimentation can lead to new insight and spark new ideas into the way Markov models of sequence evolution are estimated and used.

Extensions

Several extensions stem from this project. Our elementary simulator of sequence evolution can be improved in many ways to more accuracy model more complex evolutionary processes. All generalizations of the elementary Markov model, such as modeling insertions and deletions and allowing for substitution rate heterogeneity and independence between adjacent sites, can be considered for implementation in the simulation algorithm. The extinction rate, which is currently modeled as a Bernoulli trial acting independently on all sequences, can be modified to be a sequence-dependent function to account for fitness in individuals. The rules of phylogenetic tree generation can also be modified to account for different evolutionary contexts.

In its current from, the sequence evolution algorithm is representative of a microscopic evolutionary context, such as DNA replication. The algorithm can be adapted

to represent a macroscopic evolutionary context, such as having sequences represent species, and branching represents the process of speciation. One way to realize such a context is to have each branch contain not one, but a collection of sequences. Each collection represents a species and its closely related homologs. Mutations and extinctions are to be simulated for each sequence in a collection. Branching occurs, when the sequences of a given collection form two distinct clusters on some spectrum. Each cluster would then be placed on its own branch, and evolution would proceed independently for each branch. An entire collection becomes extinct when all its individual sequences become extinct. Large scale disasters that cause the extinction of an entire collection can also be simulated.

Another extension is to consider the results of the evaluation experiments for developing improvements for Markov model estimation techniques. For example, we have found from empirical results that accounting for the frequencies of individual sequences in a population could lead to improved estimates. However, our experiments only considered passing all individuals of a given sequence as input into an estimation technique, such that more populous sequences would get more representation. The estimation technique itself does not explicitly account for population frequencies. Modifying techniques to account for frequencies would be an interesting experiment.

Availability

A software package called the Evolution Laboratory has been developed using Matlab version 7 the Bioinformatics Toolbox 2.0, and implements the sequence evolution simulation algorithm and a few estimation techniques discussed in this thesis. This software package is available for download from the URL

web.mit.edu/kashew/Public/EvolLab.zip.

Bibliography

[1] J. Adachi and M. Hasegawa. Model of amino acid substitution in proteins encoded by mitochondrial dna. *J. Mol. Evol.*, 42:459–468, 1996.

[2] J. Adachi, P.J. Waddell, W. Martin, and M. Hasegawa. Plastid genome phylogeny and a model of amino acid substitution for proteins encoded by chloroplast dna. *J. Mol. Evol.*, 50:348–358, 2000.

[3] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and J.D. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215:403–410, 1990.

[4] L. Arvestad and W.J. Bruno. Estimation of reversible substitution matrices from multiple pairs of sequences. *J. Mol. Evol.*, 45:696–703, 1997.

[5] E. Baake and A. von Haeseler. Distance measures in terms of substitution processes. *Theor. Pop. Biol.*, 55:166–175, 1999.

[6] S.A. Benner, M.A. Cohen, and G.H. Gonnet. Empirical and structural models for insertions and deletions in the divergent evolution of proteins. *J. Mol. Biol.*, 229:1065–1082, 1993.

[7] C.L. Clothia and A. Lesk. The relationship between the divergence of sequence and structure in proteins. *EMBO J*, 5:823–826, 1986.

[8] 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*, 5:345–352, 1978. Suppl. 3.

[9] C. Devauchelle, A. Grossman, A. Henaut, M. Holschneider, M. Monnerot, J.L. Risler, and B. Torresani. Rate matrices for analyzing large families of protein sequences. *J. Comput. Biol.*, 8:381–399, 2001.

[10] Z. Dosztanyi and A.E. Torda. Amino acid similarity matrices based on force fields. *Bioinformatics*, 17:686–699, 2001.

[11] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis, Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge, UK, 1998.

[12] W.J. Ewens and G.R. Grant. *Statistical Methods in Bioinformatics, an Introduction*. Springer, New York, 2001.

[13] J. Felsenstein. Evolutionary trees from dna sequences: A maximum likelihood approach. *J. Mol. Evol.*, 17:368–376, 1981.

[14] N. Goldman and Z. Yang. A codon-based model of nucleotide substitution for protein-coding dna sequences. *Mol. Biol. Evol.*, 11:725–736, 1994.

[15] G.H. Gonnet, M.A. Cohen, and S.A. Benner. Analysis of amino acid substitution during divergent evolution: the 400 by 400 dipeptide substitution matrix. *Biochem. Biophys. Res. Commun.*, 199:489–496, 1994.

[16] G.H. Gonnet, M.A. Cohen, and Benner S.A. Exhaustive matching of the entire protein sequence database. *Science*, 256:1443–1445, 1992.

[17] N.C. Grassly, Adachi J., and A. Rambaut. Pseq-gen: an application for the monte carlo simulation of protein sequence evolution along phylogenetic trees. *CABIOS*, 13:559–560, 1997.

[18] M. Hasegawa, H. Kishino, and T. Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial dna. *J. Mol. Evol.*, 22:160–174, 1985.

[19] S. Henikoff and J.G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. U.S.A*, 89:10915–10919, 1992.

[20] D.M. Hillis, J.J. Bull, M.E. White, M.R. Badgett, and I.J. Molineux. Experimental phylogenetics: Generation of a known phylogeny. *Science*, 255:589–592, 1992.

[21] D.M. Hillis, J.P. Huelsenbeck, and C.W. Cunningham. Application and accuracy of molecular phylogenies. *Science*, 264:671–677, 1994.

[22] I. Holmes and G.M. Rubin. An expectation maximization algorithm for training hidden substitution models. *J. Mol. Biol.*, 317:753–764, 2002.

[23] M.S. Johnson and J.P. Overington. A structural basis for sequence comparisons. *J. Mol. Biol.*, 233:716–738, 1993.

[24] D.T. Jones, W.R. Taylor, and J.M. Thornton. The rapid generation of mutation data matrices from protein sequences. *CABIOS*, 8:275–282, 1992.

[25] T.H. Jukes and C.R. Cantor. Evolution of protein molecules. *in Mammalian Protein Metabolism*, pages 21–132, 1969. H.N. Munro (ed.).

[26] C. Kelly and J. Rice. Modeling nucleotide evolution: a heterogeneous rate analysis. *Math. Biosci.*, 133:85–109, 1996.

[27] M. Kimura. A simple method for estimating evolutionary rate in a finite population due to mutational production of neutral and nearly neutral base substitution through comparative studies of neucleotide sequences. *J. Mol. Biol.*, 16:111–120, 1980.

[28] J. Kleinjung, J. Romein, K. Lin, and J. Heringa. Contact-based sequence alignment. *Nucleic Acids Res.*, 32:2464–2473, 2004.

[29] M.K. Kuhner and J. Felsenstein. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Mol. Biol. Evol.*, 11:459–468, 1994.

[30] K. Lin, J. Kleinjung, W.R. Taylor, and J. Heringa. Testing homology with contact accepted mutation (cao): A contact-based markov model of protein evolution. *Comput. Biol. Chem.*, 27:93–102, 2003.

[31] P. Lio and N. Goldman. Models of molecular evolution and phylogeny. *Genome Res.*, 8:1233–1244, 1998.

[32] A.D. Mclachlan. Repeating sequences and gene duplication in proteins. *J. Mol. Biol.*, 64:417–437, 1972.

[33] J. Meller and R. Elber. Protein recognition by sequence-to-structure fitness: Bridging efficiency and capacity of threading models. *Adv. Chem. Phys.*, 120:77–130, 2002.

[34] S. Miyazawa and R.L. Jernigan. A new substitution matrix for protein sequence searches based on contact frequencies in protein structures. *Protein Eng.*, 6:267–278, 1993.

[35] T. Muller, R. Spang, and M. Vingron. Estimating amino acid substitution models: a comparison of dayhoff's estimator, the resolvent approach and a maximum likelihood method. *Mol. Biol. Evol.*, 19:8–13, 2002.

[36] T. Muller and M. Vingron. Modeling amino acid replacement. *J. Comput. Biol.*, 7:761–776, 2000.

[37] S.B. Needleman and C.D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48:443–453, 1970.

[38] R. Nielsen and Z. Yang. Likelihood models for detecting positively selected amino acid sites and applications to the hiv-1 envelope gene. *Genetics*, 148:929–936, 1998.

[39] A. Prlic, F.S. Domingues, and J.S Manfred. Structure-derived substitution matrices for alignment of distantly related sequences. *Protein Eng.*, 13:545–550, 2000.

[40] A. Rambaut and N.C. Grassly. Seq-gen: an application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *CABIOS*, 13:235–238, 1997.

[41] J.K.M. Rao. New scoring matrix for amino acid residue exchanges based on residue characteristic physical properties. *Int. J. Peptide Protein Res.*, 29:276–281, 1987.

[42] J.L. Risler, M.O. Delorme, H Delacroix, and A. Henaut. Amino acid substitutions in structurally related proteins. *J. Mol. Biol.*, 204:1019–1029, 1988.

[43] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, 4:406–425, 1987.

[44] R.M. Schwartz and M.O. Dayhoff. Matrices for detecting distant relationships. *In M. O. Dayhoff, editor, Atlas of Protein Sequence and Structure*, 5:353–358, 1978. Suppl. 3.

[45] T.F. Smith and M.S. Waterman. The identification of common molecular subsequences. *J. Mol. Biol.*, 147:195–197, 1981.

[46] P.H. Sneath and R.R. Sokal. *Numerical Taxonomy*. W.H. Freeman and Company, San Francisco, 1973.

[47] D.E. Taneyhill, A.M. Dunn, and M.J. Hatch. The galton-watson branching process as a quantitative tool in parasitology. *Parasitol Today*, 15:159–165, 1999.

[48] O Teodorescu, T. Galor, J. Pillardy, and R. Elber. Enriching the sequence substitution matrix by structural information. *Proteins*, 54:41–48, 2004.

[49] J.D. Thompson, D.G. Higgins, and Gibson T.J. Clustal w: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nuc. Acids Res.*, 22:4673–4680, 2000.

[50] J.L. Thorne. Models of protein sequence evolution and their applications. *Curr. Opin. Genet. Dev.*, 10:602–605, 2000.

[51] E. Vanderviere and M. Huber. An adjusted boxplot for skewed distributions. *Physica-Verlag/Springer 2004*, pages 1933–1940, 2004. COMPSTAT Symposium 2004.

[52] S. Veerassamy, A. Smith, and E. Tillier. A transition probability model for amino acid substitutions from blocks. *J. Comput. Biol.*, 10:997–1010, 2003.

[53] S. Whelan and N. Goldman. Distribution of statistics used for the comparison of models of sequence evolution in phylogenetics. *Mol. Biol. Evol.*, 16:1292–1299, 1999.

[54] S. Whelan and N. Goldman. A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. *Mol. Biol. Evol.*, 18:691–699, 2001.

[55] Z. Yang. Estimating the pattern of nucleotide substitution. *J. Mol. Evol.*, 39:105–111, 1994.

[56] Z. Yang. A space-time process model for the evolution of dna sequences. *Genetics*, 139:993–1005, 1995.

[57] Z. Yang, N. Goldman, and A.E. Friday. Comparison of models for nucleotide substitution used in maximum liklihood phylogenetic estimation. *Mol. Biol. Evol.*, 11:316–324, 1994.

[58] Z. Yang, R. Nielsen, and M. Hasegawa. Models of amino acid substitution and applications to mitochondrial protein evolution. *Mol. Biol. Evol.*, 15:1600–1611, 1998.