uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Predrag Mizdrak

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.C.S.

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Novel iterative approach to joint sequence alignment and tree inference under maximum likelihood: a critical assessment

TITRE DE LA THÈSE / TITLE OF THESIS

Marcel Turcotte

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Stéphane Aris-Broson

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Yongyi Mao

Frank Dehne

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Novel iterative approach to joint sequence alignment and tree inference under maximum likelihood: a critical assessment

by

Predrag Mizdrak

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the MCS degree in
Computer Science

School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-59887-0
Our file Notre référence
ISBN: 978-0-494-59887-0

Acknowledgement

It is difficult to overstate my gratitude to my supervisors, Dr. Marcel Turcotte and Dr. Stéphane Aris-Brosou. Without their guidance, energy, enthusiasm in research and inspiration, this work would not have been possible, and I thank them so much for everything they have done for me.

Abstract

Multiple sequence alignment (MSA) and phylogeny tree reconstruction are two imporant problems in bioinformatics. In some respect, they represent "two sides of the same coin", since solving either of the two problems would be easier if the solution to the other problem was given. However, most of the currently available algorithms present a solution to only one of these two problems, either completely ignoring the other problem or assuming that its solution is known in advance. Attempts have been made to solve these two problems simultaneously, but they are either too computationally intensive or inappropriate to analyze divergent sequences. Here we derive a new method that addresses these shortcomings by iteratively improving the starting alignment and its corresponding evolutionary tree based on maximum likelihood scores. We show that the method produces trees with significantly better likelihood scores for fairly to highly divergent sequences. Yet, this improvement does not translate directly into an improvement of the tree and alignment quality.

Contents

List of Tables

List of Figures

List of Abbreviations

Baliphy	Algorithm for joint estimation of alignments and phylogenies
DNA	Deoxyribonucleic acid
F84	Felsenstein evolutionary model
HKY	Hasegawa-Kishino-Yano evolutionary model
JC69	The simplest evolutionary model, named after the authors Jukes and Cantor
K2P	Kimuras two parameter evolutionary model
LSD	Least significant difference
MCMC	Markov chain Monte Carlo
ML	Maximum likelihood
MSA	Multiple sequence alignment
Muscle	Multiple sequence alignment algorithm
NJ	Neighbor-Joining tree reconstruction method
Pandit	Public repository of molecular sequences, alignments and trees
PhyML	Phylogenetic tree inference algorithm which uses maximum likelihood
r8s	Algorithm for estimating absolute rates and divergence times on trees
RF	The Robinson-Foulds tree distance
RNA	Ribonucleic acid
Rose	An algorithm for generating sequence families
TreeBase	Public repository of molecular sequences, alignments and trees
SP	Sum of pairs, an alignment similarity score
UPGMA	Unweighted Pair Group Method with Arithmetic Mean reconstruction method

Chapter 1

Introduction

It is believed that the Earth is around 4.6 billion years old [1] and it is home to more than 1.75 million documented, and over 10 million total number of different species [5], ranging from plants, bacteria to animals. It is hard to imagine what a bacteria smaller than 5×10^{-4} mm [39] has in common with a giant whale that weighs over 100 tons, or what a human being has in common with a plant. The landmark work of Darwin, *The Origin of Species* [4], for the first time satisfactorily answered these questions. Darwin argued that through the process of natural selection, species evolve over generations, and that evolution was the main factor contributing to the wide variety of life, and hence, all living forms on Earth are related in a way that their ancestors can be traced back to one common source.

Given a set of species, one of the fundamental problems in biology is to reconstruct their evolutionary tree. The first attempts at solving this problem were based on morphological characters of the species. For example, one would consider a set of binary characters, such as "can fly", "has feathers", "lays eggs" and then infer their evolutionary relationship based on this set of characters. Those species sharing most of the characters are considered to be closer relatives than those that do not. This approach has been proven to be inconsistent [46] when the species under consideration are not very closely related. The same morphological characters can also evolve independently in dif-

ferent lineages and that further complicates the inference of evolutionary relationship using this method.

Advancements in molecular biology in the last century have paved the way for molecular phylogeny: the study of evolutionary relationships among different living organisms based on molecular evolution. At the macromolecular level, the hereditary information of all forms of life, with the exception of some viruses, is carried by deoxyribonucleic acid (DNA) molecule, which consists of four bases or nucleotides: adenine (A), guanine (G), thymine (T) and cytosine (C). Two other macromolecules that are the subject of intensive research are ribonucleic acid (RNA) molecules which consist of an alphabet of four letters, and protein molecules whose alphabet consist of twenty amino acids. All three types of molecules are linear structures and the order of the bases is significant. We will consider only DNA sequences in this thesis.

The widespread availability of molecular sequences has shifted the focus of inferring the evolutionary trees between species (which leads to species trees) to inferring the evolutionary trees between sequences. The topology of species and sequence trees are not necessarily the same [30], and the species tree can be inferred based on multiple inferred sequence trees from the same set of species or from gene order. There are several advantages to deriving evolutionary relationship using molecular information, and some of the most convincing ones are that all living organisms, no matter how distantly related, share the same bases (in DNA sequences, they are A, C, G and T), and methods based on molecular evolution facilitate easier quantification of gene (and subsequently species) similarity.

A DNA molecule is a double helical structure consisting of two complementary strands. During the process of cell division, a DNA molecule replicates into two identical molecules. This process is quite reliable and the replicated DNA molecules are identical to the original molecule. However, even the perfect nature makes a misstep from time to time, and in the case of DNA replication, these missteps can fall into either large scale or point events. The large scale mutations involve inversions (copying a part of the sequence

in reverse order), translocations (insertion of segments from different sequences), duplications (repetition of two or more contiguous nucleotides in the sequence) and deletions. The point mutations include insertions (inserting a new base into a replicated DNA that was not present in the original sequence), deletions (skipping a particular base) and substitutions (replacing a given base with a different base). Over a long period of time, that can be measured in millions of years, these molecular imperfections along with other factors (such as geographic separation between species) can cause a species to split into two or more descendent species.

There are numerous applications of molecular phylogeny. Other than aiding the reconstruction of the Tree of Life and inferring the divergence times between species [29], molecular phylogenies are also used to help predict the function of unknown genes based on their similarity with the genes of known function [8], inferring horizontal gene transfer [27], tracking the source of disease infection [45], inferring gene duplication events [26], predicting the secondary structure of proteins [3]. It plays such an important role in the whole field of biology that Theodosius Dobzhansky, a prominent biologist of the 20th century, once said: *"Nothing in biology makes sense except in the light of evolution"* [6].

In the next chapter we are going to present a background review of phylogenetic tree inference and sequence alignment methods, both of which are the key techniques for better understanding of evolutionary relationships between sequences. Following the background section, we will present our method of iterative improvement of the two activities of sequence alignment and tree reconstruction. Then, we will present our results and wrap up with concluding remarks.

Chapter 2

Background

In this section, we will review three broad areas in bioinformatics that are related to this study: sequence alignment, evolutionary models and phylogenetic tree inference. Sequence alignment is usually the first step in inferring evolutionary relationship between sequences. It is used as an input for most of the algorithms that perform the inference of evolutionary trees.

2.1 Sequence alignment

During the process of DNA replication, one or more of several different types of mutations may happen. Within the context of sequence alignment, traditionally only point mutations (insertions, deletions and substitutions) are considered.

If the two sequences differ due to an insertion or deletion event, it is not possible to know if the two sequences' differences are due to one or the other event. For example, the two sequences ACCT and ACGCT shown in Figure 2.1, could have occurred by deletion of the nucleotide G from the sequence ACGCT (the left side of the figure) or they could have occurred by inserting the nucleotide G between the second and the third nucleotide in their ancestor sequence (right side of the figure). Since we cannot differentiate these

Figure 2.1: Deletion event from the root node on the left side (ACGCT) and insertion event from the root on the right side (ACCT) result in the same observed sequences (ACCT and ACGCT)

two events unless a third sequence is considered, we often refer to them as a single event and call it an indel event.

The pairwise sequence alignment problem takes two sequences S_1 and S_2, of length $|S_1|$ and $|S_2|$, respectively, and arranges them into a matrix of size $2 \times k$, where $k \geq max(|S_1|, |S_2|)$. The matrix columns represent pairs of nucleotides that have evolved from the same ancestor nucleotide, or a nucleotide that was inserted into, or deleted from its ancestor sequence at a given position. These indel events are represented by a "nil" nucleotide, which is often denoted by "-" character. Matrix rows represent one of the two sequences (S_1 or S_2) with possible insertions of indel nucleotides. A pairwise alignment in each column is assigned a score and, in the simplest case, the score of the whole alignment of the two sequences, also called the similarity score, is calculated as the sum of alignment scores of all pairs of nucleotides in the matrix. The objective of the pairwise alignment problem is to find an alignment with the best similarity score with the hope that this will reflect the most probable homology hypothesis. For example, aligning the above two sequences $S_1 = ACGCT$ and $S_2 = ACCT$ may produce a matrix of size 2×5, as shown in Figure 2.2.

The similarity score of the above alignment is calculated as $S(S_1, S_2) = s(A, A) + s(C, C) + s(G, -) + s(C, C) + s(T, T) = v_{11} + v_{22} + v_{35} + v_{23} + v_{45}$, where v_{ij} is the

```
ACGCT
AC-CT
```

Figure 2.2: Sample pairwise alignment with a single indel event

alignment score of two nucleotides and is usually taken from a similarity score matrix, a general form of which is shown in Table 2.1.

	A	C	G	T	-
A	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}
C	v_{21}	v_{22}	v_{23}	v_{24}	v_{25}
G	v_{31}	v_{32}	v_{33}	v_{34}	v_{35}
T	v_{41}	v_{42}	v_{43}	v_{44}	v_{45}
-	v_{51}	v_{52}	v_{53}	v_{54}	v_{55}

Table 2.1: A general form of a similarity score matrix

Several variations of the above problem exist. Global pairwise alignment optimizes the similarity score over the whole length of the two sequences, while local pairwise alignment tries to locate two substrings of the two sequences that have the highest similarity score. The first deterministic optimal solution for the global pairwise alignment problem was given by the Needleman-Wunsch algorithm [24] which used dynamic programming to optimize the similarity function. Also using dynamic programming algorithm, one can obtain the optimal solution for the local sequence alignment. The first solution to that problem was presented in Smith-Waterman algorithm [41].

The value of a similarity score does not give a definitive answer as to whether two sequences are related or not. After obtaining a similarity score, one uses statistical methods in order to find out if the alignment score is the result of pure chance or relatedness between two sequences [22].

While pairwise sequence alignment seeks to answer the question of how similar two

sequences are, multiple sequence alignment (MSA) assumes that the sequences are related and tries to locate their conserved regions. An MSA of l sequences $S_1, S_2, \ldots, S_l$ produces a matrix of size $l \times k$, where $k \geq max(|S_1|, |S_2|, \ldots, |S_l|)$. A column of an MSA is called a site and it corresponds to one or more nucleotides which have evolved from the same position in a common ancestral sequence. Similar to the alignment score for pairwise alignment, there is an alignment score for an MSA and the objective of an MSA algorithm is to find an MSA which optimizes that score. One of the most popular MSA scores is called the sum-of-pairs score or the SP-score. For a given MSA with l sequences and k columns, the SP-score is calculated as follows:

$$S(M) = \sum_{c=1}^{k} \sum_{i=1}^{l-1} \sum_{j=i+1}^{l} s(base_{ic}, base_{jc}) \tag{2.1}$$

where $s(base_{i,c}, base_{j,c})$ is the alignment score between nucleotides i and j at site c.

An alignment of two of more sequences is called a profile. For example, in Figure 2.3, aligning ACGGTG and CCGTGG produces a profile. The alignment of two profiles produces another profile, and an MSA is just a profile with all input sequences included. When aligning two profiles, one uses a method for pairwise sequence alignment and treats profile sites as single nucleotides. For example, in Figure 2.3, the third site in the MSA on the right, was obtained by inserting a new site between the second and the third site of the second profile. This new site being inserted into the second profile consists of two indel nucleotides, one for each sequence in the profile.

Since optimizing the SP score function is NP-hard [49], aligning more than ten sequences is computationally infeasible and most of the practical solutions are heuristic in nature. They convert an MSA problem into a progressive pairwise alignment (see [19] for a review), where first, the most similar sequences are aligned using a pairwise sequence alignment algorithm (for example, aligning S1 and S2, or S3 and S4 in Figure 2.4). Then, these profiles are aligned into new profiles (for example, aligning P1 and P2 into P5 in Figure 2.4) and the procedure continues recursively until the last two profiles are aligned into one profile, which represents an MSA of the input sequences.

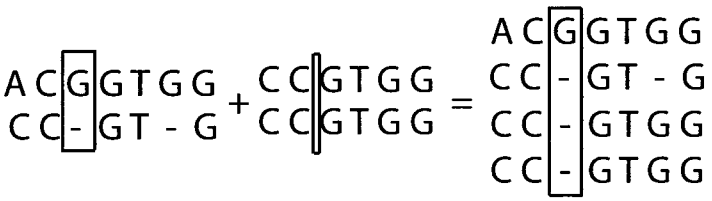

Figure 2.3: Alignment of two profiles showing a site (on the right) which was obtained by inserting a new site in the second profile

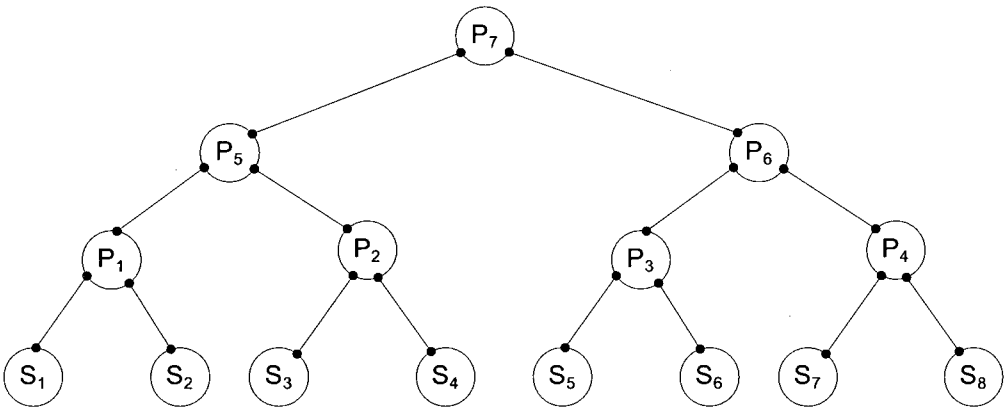

Figure 2.4: A sample guide tree for sequences $S_1, ..., S_8$. Internal nodes represent ancestor sequences and the root node (P_7) represents a sequence from which all sequences have evolved.

The tree in Figure 2.4 showing the order of sequence and profile alignment steps, is called a guide tree. To produce an MSA, given a guide tree, all one has to do is to apply a pairwise alignment algorithm to input sequences and then later on profiles in the order indicated by the guide tree, until only one profile is left. The problem with this approach, however, is that one does not know the guide tree ahead of time, and it is often constructed using heuristic methods from the set of input sequences. After the next section, we provide a section of phylogenetic tree inference.

2.2 Evolutionary models

Evolution can take millions of years and often cannot be observed directly during the lifespan of a scientist. Therefore, mathematical models have been developed in order to better understand the evolutionary process. Most studies, especially those in probabilistic domain, require an evolutionary model. Here we describe the simplest evolutionary model in more detail and mention several, more advanced models.

In all evolutionary models, the nucleotide substitution rate (u) plays a central role. It plays such a role because what is observed are differences between sequences, and these differences translate into rates of evolution, defined as a number of changes per site per some unit of time. Two neighboring sequences in a tree representing evolutionary relationship between sequences are in ancestor-descendant relationship. The weight of an edge (w) connecting them is interpreted as the expected number of nucleotide substitutions per site. So, if t units of time separate two sequences, then $w = u \times t$.

One of the simplest evolutionary models for DNA sequence evolution is the Jukes-Cantor (or the JC69) model [17]. It is often represented as in Figure 2.5. Under the assumptions of the JC69 model, the probability of a nucleotide changing to any other nucleotide is the same. If u denotes the rate of a nucleotide getting substituted by a different nucleotide, then each nucleotide will mutate to a particular (different) nucleotide with a rate of $\frac{u}{3}$, as shown in Figure 2.5. However, a nucleotide can change to itself with

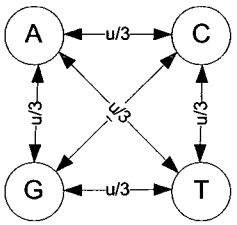

Figure 2.5: Diagram of the JC69 model. Each nucleotide can change to another nucleotide with the same probability.

the same rate. That makes the total substitution rate equal to $4 \times \frac{u}{3}$.

The JC69 model assumes that the nucleotide substitution events follows a Poisson distribution with the parameter of that distribution being equal to $4 \times \frac{u}{3}$ (this parameter is often denoted by α). Using the properties of the Poisson distribution, we know that for some time interval t, we can expect, on average $4 \times u \times \frac{t}{3}$ substitutions per site. According to the Poisson distribution, for a given parameter α, the probability of n events occurring is given by:

$$P(n, \alpha) = \frac{\alpha^n \times e^{-\alpha}}{n!}, \quad n = 0, 1, 2, 3, \cdots . \tag{2.2}$$

The probability of no substitution occurring along a branch of length t is $e^{-\alpha} = e^{-\frac{4}{3} \times u \times t}$ and the probability of having at least one substitution is $1 - e^{-\alpha} = 1 - e^{\frac{4}{3} \times u \times t}$. What is the probability that a nucleotide changes along a branch of length t in a model with a substitution rate of $4 \times \frac{u}{3}$? It is the probability of two events: the event E_1 that the nucleotide changes to a different nucleotide (this is equal to $\frac{3}{4}$), and the event E_2 that at least one substitution occurs along the branch (this is equal to $1 - e^{-\frac{4}{3} \times u \times t}$). The product of these two probabilities gives us the probability that a nucleotide will change to a different nucleotide during a time interval t along some branch, and it is given by:

$$P(E_1 \cap E_2) = \frac{3}{4}(1 - e^{-\frac{4}{3} \times u \times t}). \tag{2.3}$$

The graph of the above equation is shown in Figure 2.6. It shows how branch length affects the site difference and *vice versa*. We see that for small branch lengths ($u \times t < 1$),

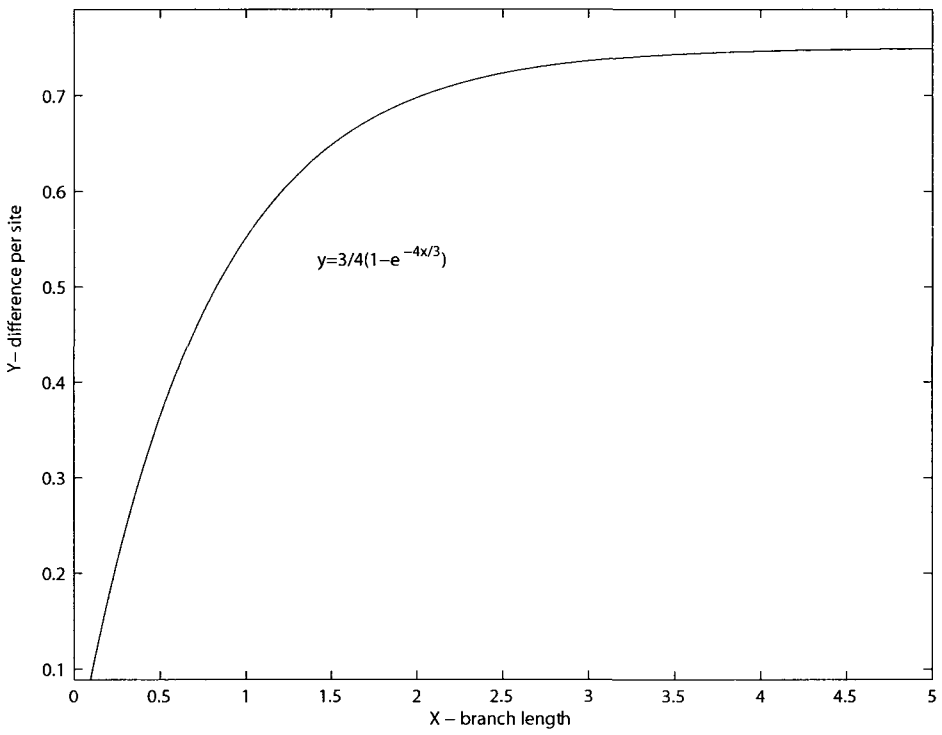

Figure 2.6: Site difference vs. branch length in JC69 model. As the branch length approaches 5, the site difference approaches 0.75, and no matter how divergent the sequences are, their difference will never exceed 0.75.

the difference per site increases approximately linearly as the branch length increases. Having a site difference value allows one to infer the length of a branch separating the two nucleotides in the phylogenetic tree. However, as the branch length become larger, the site difference approaches 0.75 and does not exceed it. This means that for branches longer than some given length (for example $u \times t > 5$), the child sequence will always have 25% of its sites identical to its immediate ancestor sequence. Conceptually, this can be explained as follows: on long branches, a nucleotide can change to a different nucleotide which in turn can change back to the original nucleotide, and while we may observe the same nucleotide at two ends of a branch, it does not mean that no change has taken place on it. This phenomenon is often called saturation. Inferring branch lengths based on large site difference (the values closer to 0.75) has to account for these multiple changes per site and introduces errors.

The assumption of the JC69 model of equal substitution rate from any nucleotide to any other nucleotide is known to be biologically implausible. For example, the transition rates (substitutions between A and G, or C and T) are often different from transversion rates (substitutions between A and C, or G and T) and this difference is taken into consideration in Kimura's two parameter, or the K2P model [18]. Other models, like the Hasegawa-Kishino-Yano, or the HKY model [15], and the Felsenstein or the F84 model [9], allow for arbitrary frequencies of nucleotides. Parameters of these evolutionary models are constant across all sequences and along all sites in each sequence. More complex models, on the other hand, allow for variation of substitution rates among different sequences in the same tree or even along different sites within the same sequence. These model extensions can be constructed on top of any model and are denoted as "base+I" (for those allowing for invariant sites [15]) or as "base+Γ" [50] (for those models allowing for across-site rate variation), where "base" is the basic model like the JC69, HKY or any simple evolutionary model.

Most of the evolutionary models are nested (one model is a special case of another model). For example, the JC69 model is a special case of the K2P model, which is a

special case of HKY model, which in turn, is a special case of HKY+I. A model with more parameters will always fit a given data set better than a model with fewer parameters. The disadvantage of estimating a model with more parameters is that estimates of those parameters will have a larger variance. It is a common practice to penalize models which use more parameters and that way find a model that fits the input sequences with the fewest parameters possible [28].

2.3 Phylogenetic tree reconstruction

A phylogenetic tree shows the evolutionary relationship of a given set of sequences. A tree may be rooted, in which case the root node represents the ancestor node from which the observed sequences and their ancestors have evolved. The direction in which time flows is indicated on a rooted tree by designating one of the internal nodes as the ancestor of all the other nodes. A rooted phylogenetic tree is called ultrametric if the distance between the root node and any of its leaf nodes is the same. Most of the sequences observed in practice have not evolved according to an equal rate and their tree is said to be non-ultrametric. We make the assumption that phylogenetic trees are strictly binary. An unrooted phylogenetic tree only shows the relationships between the sequences and does not choose any particular direction in time.

Phylogenetic tree reconstruction methods may be classified based on various criteria. For example, methods that operate on nucleotide sites must start with a multiple sequence alignment and generally fall into two categories: maximum parsimony methods and maximum likelihood (ML) methods. Other methods include distance based methods and they do not require an MSA, but instead calculate distances between pairs of sequences.

The parsimony methods are based on the hypothesis that evolutionary changes are very rare events and that those trees that explain the observed sequences with the least number of changes must be closest to the true evolutionary tree. They consist of two

steps: finding the minimum number of changes needed to explain the data for a given tree topology, and second, traversing through the tree space. Given a site and a tree, the parsimony method (such as [11] and [37]) places the site nucleotides as the leaf nodes of the tree and infers the nucleotides of all internal nodes so that the total number of substitutions from the root node to all leaf nodes is minimal. The method repeats that step for all sites and adds their substitution distances to get the tree distance. Then, it tries different trees and chooses the one with the smallest distance. Due to the very large size of tree space, this method becomes computationally very expensive and using it on more than twenty sequences is practically infeasible.

Maximum likelihood (ML) methods assume that the sequences have evolved according to some evolutionary model (described in the previous section) and try to find a tree that would have most likely produced the observed alignment. Assuming that the sites evolve independently of each other, the likelihood of the alignment is computed as a product of likelihoods across all of its sites.

For example, given a site of six nucleotides (A, C, C, A, G, T) and a possible evolutionary tree T shown in Figure 2.7, the likelihood of the site, under some assumed evolutionary model M, is computed as follows:

$$P(ACCAGT|T, M) = \sum_{I_1,I_2,I_3,I_4,I_5 \in \{A,C,G,T\}} P(A, C, C, A, G, T, I_1, I_2, I_3, I_4, I_5|T, M) \tag{2.4}$$

In the above tree, the leaf nodes are observed variables and nodes I_i are their ancestors that have to be identified. Using properties of conditional probabilities, the above expression can be simplified and identifying the ancestor nodes can be done optimally. After calculating an ML score for many trees, the ML algorithm chooses a tree that produces the best ML score. But similar to parsimony methods, or any other method that optimizes a function over a set of all trees, this method runs into the same difficulty of traversing all possible trees.

Traversing only tree topologies (without considering branch lengths) is prohibitively

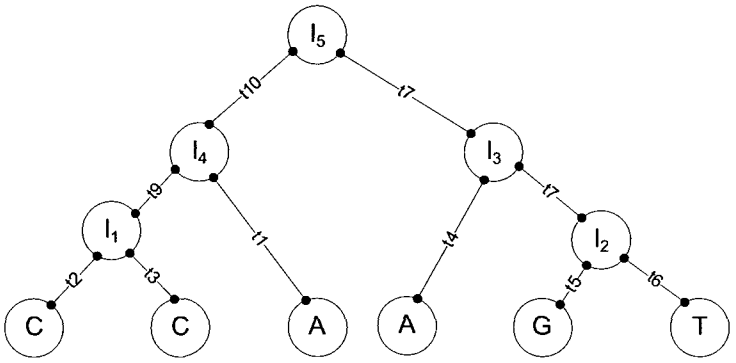

Figure 2.7: Phylogenetic tree of a site consisting of six nucleotides, and their ancestor nodes (I_i). Branch lengths are also shown (tn).

expensive for all but smallest trees (less than a dozen of leaf nodes). For example, the number of different rooted trees with n leaf nodes is given by the following expression [10]:

$$\frac{(2n-3)!}{2^{n-1}(n-1)!} \tag{2.5}$$

Most of the practical methods are based on heuristics, in which one chooses a starting tree and then explores the tree space by perturbing the starting tree. The tree with the best score from the neighborhood of the starting tree is chosen as the starting point for further perturbation and the process continues until no tree with better score can be found, or some fixed number of perturbation steps has been reached. Several methods of perturbing a starting tree have been proposed (Nearest-Neighbor Interchange, Subtree Pruning and Regrafting, Tree Bisection and Reconstruction [10], to name a few) but none of them provide an optimal solution. These methods often run into the local maxima problem, where a tree with the best likelihood is found (for small perturbations of the neighboring trees) but its likelihood is smaller than the likelihood of the most likely tree. All practical solutions try to balance accuracy with computational costs.

Distance matrix methods, on the other hand, calculate distances between all pairs of sequences and then look for a tree that best predicts those distances. For example,

starting with a set of sequences, one first builds a square matrix D in which each $(i,j)^{th}$ element (D_{ij}) represents the distance between sequence S_i and S_j. Given a fixed tree topology, the distance between any two sequences can also be calculated as the sum of weights of all edges connecting them in the tree (we denote these distances by d_{ij}). Using a least square method (such as [2]), we can find lengths of these internal edges whose sums equal d_{ij}, by differentiating the function Q below. The parameters w_{ij} represent weight factors and are specific to the algorithm. By solving the system of linear equations, one can obtain the lengths of all internal edges.

$$Q = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}(D_{ij} - d_{ij})^2 \tag{2.6}$$

The above approach optimizes branch lengths for a fixed topology. However, since the topology is also unknown, the methods continue by iteratively modifying topology and re-optimizing the branch lengths.

Another popular class of algorithms for reconstructing phylogenetic trees is based on clustering the input sequences. Many algorithms are based on Unweighted Pair Group Method with Arithmetic Mean or UPGMA algorithm [21] and Neighbor-Joining or NJ algorithms [35]. UPGMA first finds the smallest value in the pairwise matrix, which corresponds to the shortest distance between two sequences, and then joins the two sequences into a new sequence, which represents their parent sequence and which is located in the tree half way between the two sequences. It then removes the two closest sequences from the set and replaces them with their ancestor sequence. The procedure then rebuilds the pairwise matrix of the new (by one smaller) set and the process continues until only one sequence is left, in which case it is the root sequence of the input data set. UPGMA methods perform very well when the equal rate assumption holds.

The steps in NJ algorithm are similar to UPGMA except that it uses a different method for choosing the pair of sequences to merge, the branch lengths and re-computing the distance between the new (merged) sequence and the rest of sequences. Trees produced by NJ algorithm are not ultrametric in general and are unrooted.

2.4 Simultaneous sequence alignment and phylogenetic inference

In this section we present an overview of the problem of simultaneous multiple sequence alignment and phylogenetic tree inference. This section lays the ground for our method which will be presented in the next chapter.

Progressive multiple sequence alignment methods require a guide tree, which is not known prior to the sequence alignment. It has been suggested, and we also prove that in our results section, that the quality of a guide tree can affect the quality of the alignment [44]. As outlined in the previous section, various heuristic algorithms have been proposed for building a guide tree. Many phylogenetic tree inference methods (maximum parsimony and maximum likelihood methods, in particular) require an existing MSA for their input, while many MSA methods require a guide tree for their input. Therefore, to circumvent this circular dependence of the two problems, attempts have been made to solve the two problems simultaneously. The first attempts at simultaneous MSA and phylogenetic inference were based on parsimony approach [38] but they have not scaled well on all but very small number of input sequences (often less than ten).

Even the methods with better performance, such as [16] still suffer from the general disadvantages that are common to all parsimony methods - poor performance on more dissimilar input sequences. Phylogenetic methods that require an MSA for their input take the input as the true alignment and then infer a tree based on that alignment. They discard all other alignments which may be very close to the one provided in the input. However, different alignment algorithms may provide different alignments and no algorithm currently exists that would align sequences correctly. That may result in an alignment of a smaller alignment score to be better (or more correct) alignment and yet the phylogenetic inference algorithm would not give any consideration to it. Bayesian methods consider many alignments and trees by sampling from the joint posterior distribution of alignments and trees, conditional only on the input sequences.

Another method for the simultaneous alignment and phylogenetic inference was proposed by [34]. It is based on an iterative approach, in which it first creates an alignment and then uses that alignment to construct a guide tree. Following that, it then iteratively tries to improve the alignment using the inferred guide tree, and also, it attempts to improve the guide tree using the latest alignment. The algorithm in [34] uses maximum parsimony criterion to construct a guide tree and the authors show that the method converges after only several iterations. This iterative approach is not limited to the maximum parsimony optimization criteria used for constructing the guide tree and any other optimization criterion could be used as well.

The main hypothesis of the above iterative approach is that given the true alignment and the true tree, A_{true} and T_{true}, respectively, an alignment at i^{th} iteration would be better than the alignment at $(i-1)^{th}$ iteration, because the guide tree at the i^{th} iteration should be better than the guide tree at the earlier iteration (since it was inferred from the improved alignment).

Assuming that the distance measure between two alignments is sensitive enough to the alignment change produced by the last guide tree, ideally, we should have that $d(A_i, A_{true}) \leq d(A_{i-1}, A_{true})$, where A_i is the alignment at the i^{th} iteration, and $d(,)$ is a distance measure between two alignments. Similarly for trees, we would like to have that $d_t(T_i, T_{i-1}) < d_t(T_{i-1}, T_{i-2})$, where $d_t(,)$ is a distance measure between two trees. In an ideal scenario, one would like to have that $d(A_{true}, A_n) \rightarrow 0$ for n very large, however, due to the local maxima/minima problem that is always present in optimizing multidimensional functions, it would be over-optimistic to expect such convergence in practice.

The problem of the simultaneous alignment and phylogenetic tree inference has also been approached from the Bayesian perspective. Several solutions have been proposed (such as [20] and [31]), and while they provide a more rigorous and formal setting, they come at a very high computational cost.

Point estimation methods (whether deterministic - like parsimony, or probabilistic -

like likelihood methods) map an input into a single output value that best optimizes the given function. The output value could be a single object (such as an alignment or a tree) or a pair of two objects (like an alignment and a tree). What makes an output single is not how many output values it has but rather how many outputs of the same type it has.

The result of a Bayesian approach, on the other hand, is not a single point estimate but rather a distribution which accounts for a level of uncertainty. For example, given a set of molecular sequences X, produced by evolution that we simulate using a model with a set of model parameters Θ, [31] produces a conditional distribution function of the tree T and the alignment A, given the observed set of sequences X. Using the Bayes formula, that conditional distribution can be expressed as follows:

$$P(T, A|X) = \int_{\Theta} P(T, A, \theta|X)d\theta = \int_{\Theta} \frac{P(X|T, A, \theta)P(T, A)P(\theta)}{P(X)}d\theta. \qquad (2.7)$$

In the above equation, $P(T, A|X)$ is called the posterior probability of the tree T and the alignment A. The likelihood of observing the set of sequences given the tree T and the alignment A is described by $P(X|T, A, \theta)$. The prior information, $P(T, A)P(\theta)$, specifies what the modeler believes about the processes that generated the data X. The last term $P(X)$ is called the marginal probability of the observed sequences. Solving the posterior distribution of 2.7 represents a challenge since in all but the simplest cases, the analytic solution to it is not known. The approximate solution is often provided by Markov chain Monte Carlo (MCMC) simulations [12]. We use [31] to validate our iterative procedure on the real data set.

Chapter 3

Experimental Design

In this section, we first describe the iterative procedure and its simulation settings. Then, we introduce several distance measures that we use for comparing different evolutionary trees and sequence alignments. In the last part, we describe the procedure of validating our algorithm on both simulated and real data sets.

3.1 Iterative procedure

Maximum likelihood methods take a single alignment and try to infer its evolutionary tree as if the alignment was correct. However, as already mentioned, no algorithm reconstructs the true alignment and the success rate of the tree reconstruction method depends on how well the MSA algorithm performs. We designed the iterative procedure on the hypothesis that realigning the sequences using the inferred evolutionary tree as the new guide tree, and then reconstructing a new guide tree from the "improved" alignment, would improve both the final MSA and the evolutionary tree.

Figure 3.1 shows the diagram of the iterative procedure. It takes a set of input sequences and then performs multiple iterations, with each iteration consisting of an MSA and tree inference steps. The procedure uses `Muscle` [7] to align sequences and

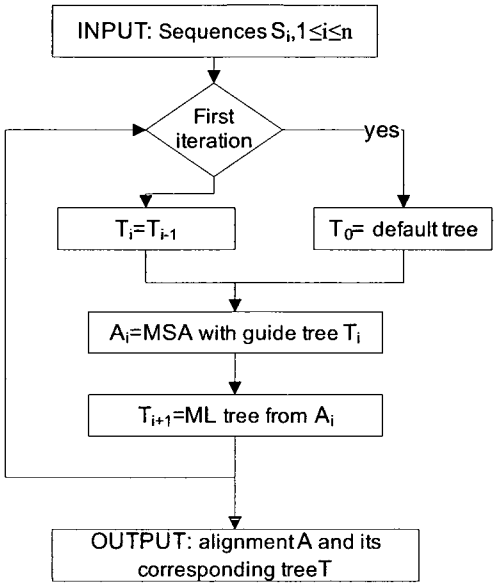

Figure 3.1: Diagram of the iterative procedure

`PhyML` [14] to infer a phylogeny tree, however, any third-party programs could be used for these tasks, as long as their outputs are compatible or can be converted to the expected format. In the first iteration, `Muscle`'s default guide tree is used as the guide tree, but in every other iteration, the inferred ML tree from the previous iteration is used as the guide tree for new MSA. The procedure remembers the alignment with the best ML score. After a fixed number of iterations has been performed, it outputs the pair consisting of an MSA with the best ML score and its corresponding guide tree.

In our study, the above iterative procedure always performs a fixed number of iterations and assumes the simplest evolutionary model (JC69) in tree inference step. The procedure could be extended so that these two parameters are provided in the input along with a set of sequences that it currently requires.

3.2 Simulation study

Evolution is a process that may take millions of years and often cannot be directly observed. Therefore, to test phylogenetic tree reconstruction algorithms or sequence alignment algorithms in a general way, one often performs a simulation study. In such a study, an algorithm simulates the evolutionary events, such as deletion, insertion and substitution. Since the algorithm simulates, or controls, these events, it can produce not only the set of sequences, but also their evolutionary tree and their alignment. The tree generated during the simulation is called the "true" evolutionary tree, and similarly, the simulated alignment is called the "true" alignment. We denote them by T_{true} and A_{true}, respectively. True evolutionary trees and true alignments exist only in simulation studies; in practice, they are not known. Given A_{true} and T_{true}, the algorithm accuracy is determined by comparing the alignment and evolutionary tree that it produces with the true alignment A_{true} and the true tree T_{true}.

We test our iterative procedure for different values of evolutionary rates and indel probabilities. For each particular setting, we generate 400 trees, with each tree having 50 leaf nodes, which correspond to 50 sequences that have to be aligned and whose phylogenetic tree is to be inferred. We set the average length of each sequence to 750 nucleotides. These settings provide a good balance between the computational speed and the processing of a larger number of sequences, which allows us to validate our program in an environment in which it could be used in practice.

3.2.1 Experimental setup

In this section we describe the setup of our simulation study. We start off with the description of the initialization step in which we generate evolutionary trees, sequences and alignments.

Figure 3.2 shows the diagram of the initialization step of the algorithm. First, we generate the true tree T_{true}, which is then used to construct the set of sequences and

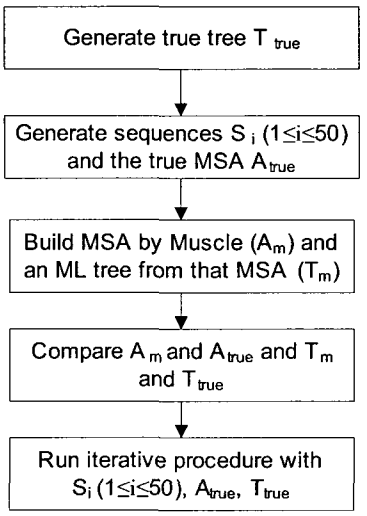

Figure 3.2: Initialization diagram: generate the true tree and the true alignment and calculate their distance from the `Muscle` alignment and its corresponding evolutionary tree.

their true alignment A_{true}. Following that, we perform an MSA on the generated set of sequences using `Muscle` and we denote the resulting MSA by A_m. Then, we use `PhyML` to infer a phylogenetic tree based on the A_m alignment. We will denote this tree by T_m. Given A_m and T_m, the procedure then calculates the distance between A_{true} and A_m, and T_{true} and T_m, respectively. Comparison of the iterative procedure and `Muscle` is based on how close their alignments and trees are to A_{true} and T_{true}, respectively.

At the end of the initialization, we run a slightly modified version of our iterative procedure. In practice, the iterative procedure would be used as explained above. However, in our simulation study, we had to pass both the true tree and the true alignment so that the iterative procedure can calculate the distances between the MSA that it constructs and the true MSA that it receives in input. Similarly, it has to calculate the distance between the inferred tree and the true tree T_{true}. These distances are calculated at the end of each iteration in the iterative procedure (Figure 3.2).

3.2.2 Generating true trees, sequences and true alignments

We use the `r8s` package [36] to simulate the evolutionary tree. Each tree is generated according to the Yule stochastic process [42], conditioned on the number of leaf nodes and also, conditioned on the time elapsed between the root node and the leaf nodes. The Yule process is a pure-birth process that models speciation.

Since the tree simulated by `r8s` is ultrametric by default and most of the trees encountered in practice are not, in the next step we perturb tree's branch lengths so that the tree is not ultrametric. Similar to [34], we do this in two steps. First, we multiply each branch's length by a number that is uniformly drawn from the set $\{1, 2, 3\}$. Then, to account for different evolutionary rates, we multiply all branches of a tree by a scale parameter which is drawn from the following set $\{1, 8, 16, 32, 64\}$. The resulting evolutionary rate of a tree is denoted by scale parameters: $s = 1$, $s = 8$, $s = 16$, $s = 32$ and $s = 64$.

We study separately ultrametric and non-ultrametric trees. Branches of ultrametric trees are perturbed only in the second step, however, branches of non-ultrametric trees are perturbed in both steps: the first step to make them non-ultrametric, and the second step to account for different evolutionary rates. So, the scale parameter of 1 ($s = 1$) can be used for both ultrametric and non-ultrametric trees but it has different meaning. In ultrametric case, it corresponds to exactly the same tree that was produced by `r8s`, while in non-ultrametric case it corresponds to a tree of small evolutionary rates whose branches were perturbed only in the first step above.

Given the true evolutionary tree, we then generate a family of sequences using the `Rose` program [43] under the simplest evolutionary model (JC69) that is described in the previous chapter (Section 2.2). We assign the frequency of each nucleotide in our sequences to be the same (or by definition of JC69, it is 25%). These frequencies are used by `Rose` to generate the root sequence. `Rose` then iteratively generates the root's descendant sequences and all other sequences for "true" tree generated under the Yule process. The internal nodes/sequences are only used to construct their child sequences

and are not used in the algorithm.

To generate a sequence S_{child} from its immediate ancestor sequence S_{parent}, `Rose` first copies S_{parent} to S_{child} and then mutates each nucleotide of S_{child} into another nucleotide according to the given model and mutation parameters. In our simulation, these parameters are derived from the default value of the mean substitution parameter.

The following step taken by `Rose` is to simulate the deletion and insertion evolutionary events, which are controlled by "deletion threshold" and "insertion threshold" parameters, respectively. Similar to [34], we set these probabilities to either 5×10^{-4} or 5×10^{-5}. Under these rates, the models with very small evolutionary rate result in very few insertions and deletions. The sequences do not differ much and ML procedures are particularly weak on such inputs. Manual inspection of the alignments showed that the number of indels was low compared to usual alignments. Hence, in order to test the ML procedure in more realistic cases, we have introduced higher indel probability rate of 5×10^{-3}.

A single deletion or insertion event may result in removing or inserting one or more contiguous nucleotides at the same time. This may result in high variance of the resulting sequences and for that reason, we have opted to limit the insertion and deletion events to only one nucleotide at a time. When `Rose` determines that a deletion or insertion event should occur (based on the insertion/deletion parameters mentioned above), it looks at the deletion function or insertion function. These functions are just lists of probabilities of deleting or inserting one, two, three or more contiguous nucleotides. To limit these events to only one nucleotide, we set the first value in these lists to 0.99999 and the value of the remaining three lengths to $(1 - 0.99999)/3 \approx 2.5 \times 10^{-6}$.

3.2.3 Sequence alignment and tree inference

As already mentioned, for multiple sequence alignment we use `Muscle`, which is one of the most popular and efficient multiple sequence alignment algorithms. It consists of three stages: the draft progressive stage, the improved progressive stage, and the refinement

stage. The first stage calculates a matrix of k-mer distances between all pairs of input sequences. The k-mer distance is a squared sum of the difference between the frequency of each k-word in two sequences. It can be computed in linear time and is often used for that reason. Once the matrix is built, `Muscle` uses the UPGMA algorithm to construct the guide tree, which it then uses to progressively align the input sequences.

The second `Muscle` stage improves the alignment from the first stage by creating a new guide tree and re-aligning the input sequences using the new guide tree. The new guide tree is generated using the same clustering algorithm (UPGMA), however the matrix of distances is not based on k-mer distance but rather on Kimura distance [18]. Kimura distance is a distance between two sequences generated in an evolutionary Kimura two-parameter model, and to calculate it, one needs to have an MSA. The MSA from the first stage is used as an input MSA for the second stage. The last stage is iterative and consists of randomly removing an edge from the guide tree of the second stage, which results in splitting the multiple sequence alignment into two alignments (or profiles). These two profiles are then aligned anew and compared to the alignment from the second stage (in terms of the SP score), and the better of the two is kept. This iterative process is repeated until convergence or until the fixed number of iterations is performed.

We align the input sequences using the default `Muscle` parameters and obtain the alignment A_m. This is the `Muscle` alignment against which we will compare alignments produced by our procedure. We denote an alignment produced by our procedure in iteration i by A_i. In each iteration except the first, we provide `Muscle` with our guide tree, which is then used to create a new MSA. Similar to [34], we perform alignment using only the first phase of `Muscle`. Tests with alignments that were constructed using the first two phases of `Muscle` did not produce any improvement (these results are not shown). Comparing `Muscle` with our algorithm makes sense since both algorithms have an iterative component: `Muscle` optimizes the MSA by iteratively splitting it into two profiles and realigning these profiles, while our procedure iteratively builds a guide tree

from the current MSA, which it then uses to builds a new MSA.

Given a multiple sequence alignment, we then use `PhyML` in order to infer a new guide tree using the same evolutionary model (JC69) under which the sequences in the alignment were generated. `PhyML` is one of the fastest tree inference methods based on the maximum likelihood principle.

3.3 Distance measures

One of the advantages of evolutionary studies on a molecular level, as opposed to morphological level, is that one can quantify and compare different results. However, comparing these results, whether they are phylogenetic trees or alignments, poses a difficult challenge in bioinformatics. For example, given the two phylogenetic trees, how do we quantify their difference? Similarly, given two MSAs obtained by aligning the same set of input sequences, how do we quantify their similarity? The difficulty arises due to the fact that the objects being compared live in a multidimensional space. The comparison algorithms which would take all of their dimensions into consideration would be very difficult to derive and/or use. For that reason, most of the comparison algorithms used in practice do not consider all (or most of the) dimensions that the object lives in. Instead, they somewhat arbitrarily choose a set of dimensions (usually one) along which to compare the objects. Depending on the chosen dimension, the distance between the objects is then calculated without considering any of the remaining dimensions. In doing so, the algorithms loose important information about the objects and the results may be incomplete and often ambiguous.

For example, an MSA is at least a two-dimensional object and a unary SP function takes an MSA and turns it into a scalar value. Two different MSA algorithms aligning the same set of sequences could produce two different alignments whose SP scores might actually be the same. This is so because the SP function looses information about the MSA while mapping it to a scalar value.

We perform multiple replicates to evaluate performance of our method in average case scenario. We now formally introduce several functions that we use to either calculate a score of a particular MSA or tree, or a distance between two MSAs or trees.

Definition 1: Maximum Likelihood (ML) is a function which takes an alignment and maps it to a real number. Assuming that all sites in an alignment are independent, which is a common assumption as it facilitates the likelihood computation, the ML score of an alignment is calculated as the product of ML scores of all of its sites. The ML score of a site is calculated according to Equation 2.4.

The value of the ML function, as given in Equation 2.4, can be very small. In order to avoid underflow, it is common to perform its optimization in the log domain, in which case, the ML function is called the maximum log-likelihood function. However, for simplicity, we will still refer to it as a maximum likelihood function. Assuming that the underlying evolutionary model has been fixed, choosing the best MSA from a finite set of alignments is then as simple as finding an alignment with the largest ML score. In the case of the ML function, the best score would be the highest score, while in the case of the maximum log-likelihood distance, the best score would be the score with the smallest absolute value (since $log(x) < log(y)$ for $0 < y < x < 1$).

Definition 2: The Robinson-Foulds distance [33], or the RF distance, is a function that takes two trees and maps them to a whole number which represents the number of edges for which the two trees differ. In this definition, an edge is fully specified by the two nodes that it connects. The RF distance is often called the symmetric difference distance.

For example, consider the two trees T_1 and T_2, shown in Figure 3.3. To calculate the RF distance, we have to find the edges that are not shared between the two trees. Each tree has 6 edges, but only two of them can be found in both trees (these are edges AB and BG). Therefore, $RF(T_1, T_2) = 6 - 2 = 4$.

Definition 3: The Branch Score distance [32] is the sum of absolute values of the differences between lengths of common edges in the two trees. If an edge is not shared

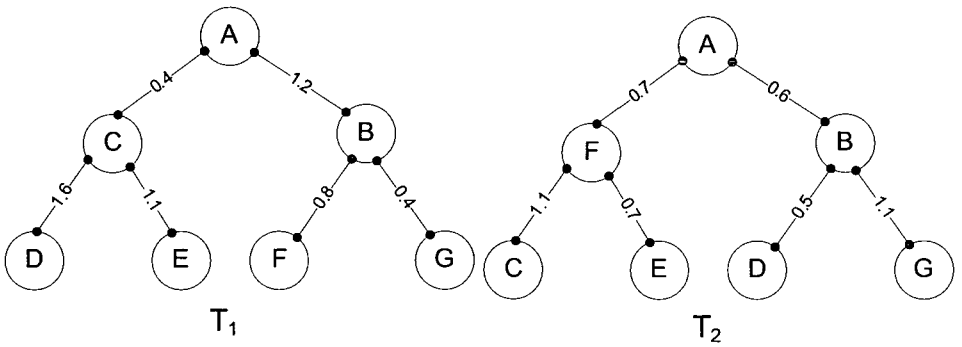

Figure 3.3: Two sample trees T_1 and T_2. Branch lengths shown on the trees are product of time and evolutionary rate.

by the two trees, then the distance of that edge is added to the result.

Using the example from Figure 3.3, the two shared edges contribute to the total Branch Score distance only by the absolute value of the difference of their length, while those edges that are not shared contribute with their total length. If we denote the length of an edge connecting two nodes (node A and node B) in tree T by $d_T(A, B)$, then the Branch Score distance between the tree T_1 and the tree T_2, as shown in Figure 3.3, is calculated as follows: $BSD(T_1, T2) = |d_{T_1}(A, B) - d_{T_2}(A, B)| + |d_{T_1}(B, G) - d_{T_2}(B, G)| + d_{T_1}(A, C) + d_{T_1}(C, D) + d_{T_1}(C, E) + d_{T_1}(B, F) + d_{T_2}(A, F) + d_{T_2}(F, C) + d_{T_2}(F, E) + d_{T_2}(B, D) = 8.3$

The RF distance takes only the tree topology into consideration and completely ignores branch lengths. The Branch Score distance, on the other hand, considers both the tree topology and branch lengths. For two trees of exactly the same topology, the value of the Branch Score distance depends only on branch lengths, however, if topologies of two trees are different, then whether a branch will contribute to the Branch Score value by its whole length or not, depends on whether the two nodes it connects are the same in both trees. It is easy to come up with pairs of trees with very different topologies and branch lengths that have the same Branch Score or RF distance. Both of them give us the "measure" of distance between two trees only from a very narrow angle.

The definition of the next distance requires that we introduce an additional concept: pruning. For a given tree T, pruning is the operation of removing one or more leaf nodes from T and suppressing all internal nodes of degree 2. A tree produced by the pruning of one or more nodes is called a pruned tree. For example, by pruning the node D from tree T_1 in Figure 3.3 (which results in removing the node C since it is the only internal node of degree 2), we get the pruned tree shown in Figure 3.4.

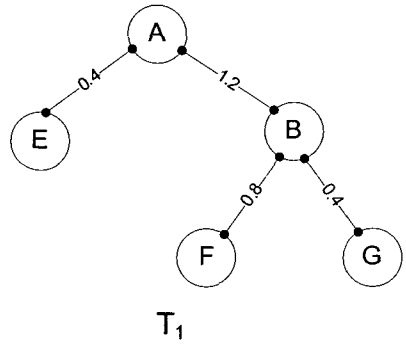

Figure 3.4: Tree produced by pruning node C from tree T_1 in Figure 3.3

Definition 4: The agreement distance between two trees T_1 and T_2 is the minimum number of leaf nodes that have to be pruned from the two trees so that their pruned trees are the same [13].

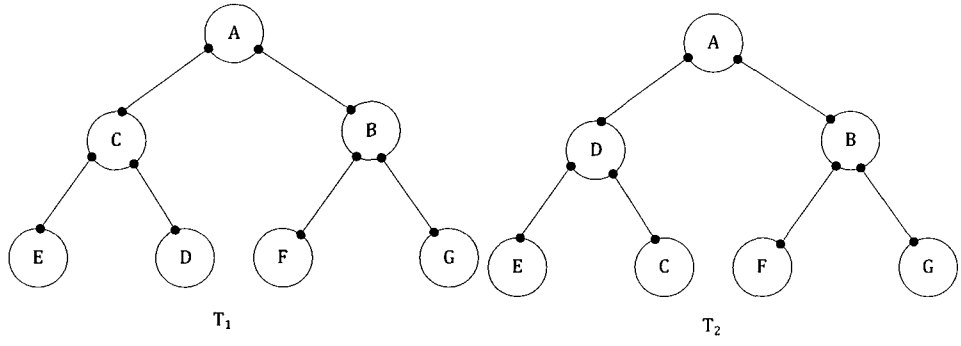

Figure 3.5: Trees used for demonstrating the agreement distance

Trees T_1 and T_2, shown in Figure 3.5, differ only in the pair of nodes (C,D). In the tree T_1, the node C is chosen as the ancestor of E and D, while in the tree T_2, the node D is chosen as the ancestor of E and C. Removing only one node from each tree (this would be node D in tree T_1 and the node C in tree T_2) would result in the pruned tree shown in Figure 3.4. Since we are removing only one leaf node from the trees, the agreement distance between T_1 and T_2 is 1, $Agreement(T_1, T_2) = 1$.

The following is a definition of a measure for comparing two alignments.

Definition 5: Given an MSA S, let $\tilde{S}$ denote the list of all pairs of nucleotides across all sites in the alignment S, and let $|\tilde{S}|$ denote the size of that list. Also, given two lists $\tilde{S_1}$ and $\tilde{S_2}$, let $\tilde{S_1} - \tilde{S_2}$ be the list of all elements found in the list $\tilde{S_1}$ and not in the list $\tilde{S_2}$. Then, the alignment similarity score (*AlignScore*) between two alignments S_1 and S_2 is defined as follows:

$$AlignScore(S_1, S_2) = 1 - \frac{|\tilde{S_1} - \tilde{S_2}|}{|\tilde{S_1}|}. \tag{3.1}$$

```
ACGT        A-CGT
A-GT        AG--T
AC-T        AC-T-
```

Figure 3.6: Two sample MSAs used for calculating the alignment similarity score

In general, the *AlignScore* function does not have to be a symmetric function because the list $\tilde{S_2} - \tilde{S_1}$ is not necessarily the same as the list $\tilde{S_1} - \tilde{S_2}$. As an example, given the two alignments in Figure 3.6, we have the following:

$$\tilde{S_1} = ((A,A),(A,A),(A,A),(C,C),(G,G),(T,T),(T,T),(T,T)) \tag{3.2}$$

$$\tilde{S_2} = ((A,A),(A,A),(A,A),(G,C),(G,T),(T,T)) \tag{3.3}$$

$$\tilde{S}_1 - \tilde{S}_2 = ((C,C),(G,G),(T,T),(T,T)) \tag{3.4}$$

$$\tilde{S}_2 - \tilde{S}_1 = ((G,C),(G,T)) \tag{3.5}$$

Then,

$$|\tilde{S}_1| = 8, |\tilde{S}_2| = 6, |\tilde{S}_1 - \tilde{S}_2| = 4, |\tilde{S}_2 - \tilde{S}_1| = 2, \tag{3.6}$$

$$AlignScore(S_1, S_2) = 1 - 4/8 = 0.5, AlignScore(S_2, S_1) = 1 - 2/6 = 0.66. \tag{3.7}$$

The above concludes the description of the distances that we will be using for comparing alignments and evolutionary trees.

3.4 Framework for analyzing results

In this section we outline the method that we use to test the improvement of our iterative procedure. For any given set of experimental conditions, we simulate the true tree (T_{true}) and generate a set of input sequences and their true alignment (A_{true}). We then obtain the `Muscle` alignment (A_m) and infer tree based on that alignment (T_m). In the iterative procedure, we run $n = 400$ replicates and for each replicate, we choose an alignment with the best ML score (A_{ML}) and its corresponding evolutionary tree (T_{ML}). For each derived MSA or tree, obtained either by `Muscle` or the iterative procedure, we calculate its distance from A_{true} or T_{true}, using distance metrics described above (Section 3.2.3). Now, for any particular distance, at the end of the iterative procedure, we have two samples: $X = \{X_1, X_2, \ldots, X_n\}$ and $Y = \{Y_1, Y_2, \ldots, Y_n\}$, where the sample X contains values of distances between A_m (or T_m) and A_{true} (or T_{true}), and the sample Y contains the values of distances between A_{ML} (or T_{ML}) and A_{true} (or T_{true}). For example, if

we consider the RF distance, then X_i and Y_i would be the RF distances between the tree inferred from `Muscle` alignment at the i^{th} replicate and the true tree, and the RF distance between the tree inferred from iterative alignment at i^{th} replicate and the true tree, respectively. Given the two samples X and Y, how do we compare performances of the two algorithms?

Running multiple replicates allows us to compare the average value of one distance function against the average value of the other distance function. We would like to find out if the average value of the sample Y, denoted by $\mu(Y)$, is significantly "better" than the average value of the sample X, which we denote by $\mu(X)$. The "better" value depends on the actual distance function. In case of ML, RF and Branch Score distances, "better" means lower, while in case of similarity score, "better" means higher. In case of ML, RF and Branch Score distance, we can use the Student t-test, that translates into testing the null hypothesis $H_0 : \mu(X) > \mu(Y)$, or equivalently, $H_0 : \mu(X) - \mu(Y) > 0$, against the alternative hypothesis $H_a : \mu(X) \leq \mu(Y)$. The assumptions of the t-test are that the samples follow normal distribution, they have the same variance and X and Y are mutually independent. In our testing we use a version of t-test, called Welch's two sample t-test [47], which eliminates the assumption that the samples have the same variance. For testing normality conditions, we use Shapiro-Wilk test [40].

Under the assumptions of normality and mutual independence, the random variable

$$V = \frac{\mu(X) - \mu(Y)}{std(\mu(X) - \mu(Y))} \tag{3.8}$$

follows the Student's t distribution with $df = 2n - 2$ degrees of freedom, where n is the sample size. In Welch's $t-$test, the formula of df is more complex and generally not an integer. The standard error of the difference between two means, $std(\mu(X) - \mu(Y))$, is calculated as:

$$std(\mu(X) - \mu(Y)) = \sqrt{\frac{\sigma_X^2}{n} + \frac{\sigma_Y^2}{n}} \tag{3.9}$$

where $\frac{\sigma_X^2}{n}$ and $\frac{\sigma_Y^2}{n}$ are estimated sample variances of X and Y, respectively.

We say that the mean of the sample Y is significantly lower than the mean of the

sample X, if the value of the random variable V is greater than $t(0.975, df)$, where $t(0.975, df)$ is the critical value of t distribution at $\alpha = 0.05$ confidence level. The t-test calculates the probability of observing a pair (X, Y) whose test statistic would be more extreme than the one observed, given that the null hypothesis is correct. This probability is called the p-value. Very small p-values indicate that it is very unlikely that the given sample was generated under H_0. In all our hypothesis testing, we use the confidence level of 5% (i.e. $\alpha = 0.05$).

For simplicity, consider the following two-sided test with null hypothesis $H_0 : \mu(X) = \mu(Y)$, where X and Y are same as in the above scenario. In this case, we reject the null hypothesis if $|V| < t_c$, where t_c is the critical value of t distribution at $\alpha = 0.05$. We say that the means of the two samples X and Y are significantly different if $|\mu(X) - \mu(Y)| > t_c \times std(\mu(X) - \mu(Y))$. The smallest critical value t, which we denote here by t_{lsd}, which makes the means of the two samples to be statistically different gives rise to the quantitative difference between two samples that is often called the least significant difference (LSD). So, the means of two samples are significantly different if and only if the difference between them is larger than LSD value, where

$$LSD = t_{lsd} \times std(\mu(X) - \mu(Y)). \tag{3.10}$$

Ideally, we would like to present the results as in Figure 3.7, where the error bars are of LSD length and significance of improvement (or not) can be read from the graph. If error bars of two sample overlap (like A and B), then the samples do not have significantly different means. On the other hand, if error bars do not overlap (like X and Y), then the two samples have significantly different means.

The graphical presentation of the results, illustrated by Figure 3.7, is justified only if the two samples satisfy the assumptions of the t-test (or Welch's two sample t-test). However, if the assumptions are not met, then the LSD quantity, as calculated in Equation 3.10, is not justified. As we will see in the next section, our samples X and Y do not always satisfy the normality conditions. In order to present results in the same way,

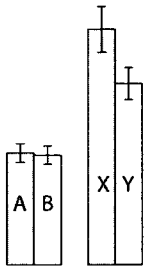

Figure 3.7: General format of presenting results

whether the t-test assumptions are met or not, we set length of all error bars on our graph to two standard deviations.

Assuming the variances are the same and the normality conditions are satisfied, we have the following

$$LSD = t_{lsd} \times \sigma_X \times \sqrt{\frac{2}{n}} < 3 \times \sigma_X \times \frac{\sqrt{2}}{20} < 0.3 \times \sigma_X < \sigma_X \tag{3.11}$$

where σ_X is the standard deviation of the sample X. Given that $n = 400$, the above equation also holds when variances are significantly different.

When reading a graph, similar to the one in Figure 3.7, on which the length of error bars are set to two standard deviations, we apply the following rules:

- if X and Y satisfy normality conditions and their error bars do not overlap, then the samples X and Y have significantly different means (by 3.11);
- if X and Y satisfy normality conditions and their error bars overlap, then they still may have significantly different means and to test if that is correct, we perform Welch's two sample t-test;
- if X or Y do not satisfy normality conditions, their error bar do not overlap, and $\mu(Y)$ is "better" than $\mu(X)$, we say that iterative method provides an improvement over `Muscle`; and

- if X or Y do not satisfy normality conditions, their error bars overlap, then we say that there is no improvement, even if $\mu(Y) < \mu(X)$.

The improvement of our iterative procedure will be at least as good as the above set of rules indicate. For example, if no improvement was found using the above rules, a non-parametric statistical method may still show that $\mu(Y)$ is significantly better than $\mu(X)$. That makes our analysis conservative.

3.5 Analysis of real data sets

The difficulty in comparing two algorithms on real data set is that we do not know the true alignment and the true evolutionary tree, and therefore, cannot directly quantify the accuracy of the algorithm under study. However, in order to provide at least some comparison between our iterative procedure and `Muscle`, we use the result produced by `Baliphy` as the reference result. In our analysis, we still have two samples X and Y, but the meaning of their variables is slightly different. In the previous section, the variable X_i contained the distance between two trees (true tree and the tree inferred based on `Muscle` alignment) for a replicate i. Here, X_i denotes the distance between the tree inferred based on the `Muscle` alignment (we have only one such tree for the whole experiment) and the tree produced by `Baliphy` at iteration i. Similarly, Y_i denotes the distance between the tree inferred based on the iterative alignment and the `Baliphy` tree at the same iteration i. To test the improvement of our procedure with respect to `Muscle`, we will perform Welch's two sample t-test to find out if $\mu(X)$ is significantly different from $\mu(Y)$.

The authors of `Baliphy` have shown in their simulation study that posterior distribution of MSAs and evolutionary trees converges for small data sets. This convergence takes place after some number of iterations called the "burn-in" period. Their study was conducted on two small data sets. One set contained only five sequences with each sequence being shorter than 130 nucleotides, and another larger data set which consisted

of twelve sequences, with each sequence being less than 500 nucleotides long. For a real data set, we have chosen a set of plant sequences from TreeBase public repository [23] (Matrix accession number: M1200). Our data set consists of 62 sequences and each sequence is longer than 2500 nucleotides.

Chapter 4

Results

Before presenting any of our results, we would like to justify the use of our iterative procedure. Given an improvement of our method with respect to the result obtained by `Muscle`, we would like to have a convincing argument that the improvement is due to the iterative nature of the algorithm and not the result of our algorithm using the ML optimization function. To show that, we have to show that the ML score of the iterative alignment (the alignment produced by our iterative procedure at some iteration $i, i > 1$) is better than the ML score produced by the standard ML algorithm which performs only one iteration ($i = 1$). We start our results section with that argument. Following that, we present the main results of our study on ultrametric trees, and we do so not because the evolutionary trees on real data are ultrametric (most of them are not), but because it helps us better understand our results on non-ultrametric trees. We then compare the results on ultrametric and non-ultrametric trees. In the last section, we present the results of our study on a set of real sequences.

4.1 Is an iterative ML algorithm better than a single-step ML algorithm?

Figure 4.1 shows ML scores of alignments produced by our iterative procedure and ML scores produced by a standard single-step ML algorithm in various settings (indel probabilities and evolutionary rates).

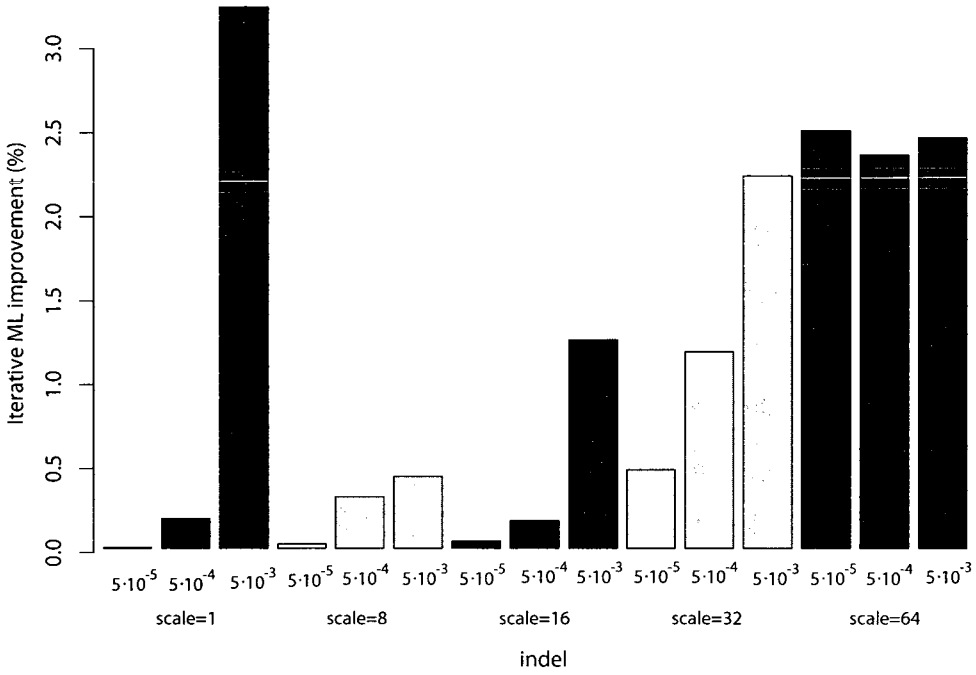

Figure 4.1: The average improvement of the iterative ML method over the standard ML procedure. The improvement is represented as the percentage by which the ML score of the standard procedure is improved (lowered) by the iterative ML method. Scores from models with different evolutionary rates (scale) are colored differently.

The improvement of the iterative procedure in Figure 4.1 is presented as the percentage by which the ML score of the standard procedure has been lowered. For example, given a particular simulation settings, if $\mathcal{A}_m$ represents the average ML score of the `Muscle` alignment, and $\mathcal{A}_{it}$ the average ML score of the best iterative alignment, then

the value on the y axis is given by the following expression

$$\frac{\mathcal{A}_{it} - \mathcal{A}_m}{\mathcal{A}_m} \tag{4.1}$$

Figure 4.1 shows that this improvement ranges from no improvement to an improvement of slightly over 3%. The no-improvement scenario is observed on models with smallest evolutionary rates and smallest indel probabilities ($(\text{indel}, \text{scale}) = (5 \times 10^{-5}, 1)$). Sequences originating from these models are easy to align and it is likely that alignments derived after only several iterations do no differ greatly from the true alignemnts. In such cases, there is very little improvement that the iterative method can produce. Alignments in scenarios with larger evolutionary rates and indel probabilities, are more difficult to reconstruct, are further from the true alignments, and therefore, there is more room for improvement.

Both standard and iterative maximum likelihood procedures optimize exactly the same problem. This problem consists of finding a set of parameter values of the underling probabilistic model which would produce the observed data more likely than any other set of parameter values. The observed data are more likely to be generated by the set of parameter values produced by the iterative procedure than the set of parameters produced by the standard procedure. That represents an improvement of the iterative procedure.

4.2 Results on ultrametric trees

In this section we compare the results of our iterative procedure with results produced by the `Muscle` procedure on ultrametric trees. We start by comparing the ML scores of the two methods and then investigate how this improvement in ML score translates into improvements in other distance measures, such as RF distance, branch score distance, alignment similarity score and the agreement distance.

4.2.1 Improvement in ML score

The two samples $X = \{X_1, \ldots, X_n\}$ and $Y = \{Y_1, \ldots, Y_n\}$ contain the absolute values of ML scores obtained from `Muscle` and iterative procedure alignments, respectively. Figure 4.2 shows how the average values of X and Y, as well as their standard deviation, change as a function of simulation parameters. In all cases but the last $((\text{indel}, \text{scale}) = (5 \times 10^{-3}, 64))$, both X and Y satisfy normality conditions, and testing if $\mu(X)$ differs significantly from $\mu(Y)$ can be performed using Welch's t-test.

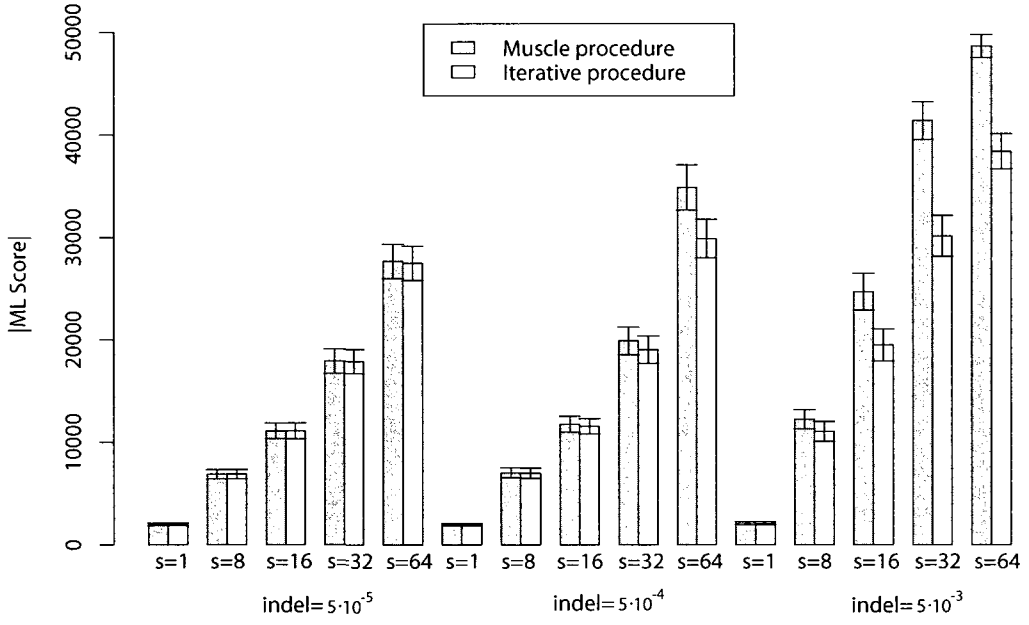

Figure 4.2: Average ML scores of `Muscle` and iterative procedure for ultrametric trees. Smaller |ML score| corresponds to better ML scores.

The iterative procedure does not produce any improvement over `Muscle` on sequences that have evolved from models with very low indel probabilities (indel $= 5 \times 10^{-5}$). This holds true even for models with very high evolutionary rates $(\text{indel}, \text{scale}) = (5 \times 10^{-5}, 64)$. It is not surprising to see these results since the evolutionary models with very small indel

probabilities produce sequences that vary very little in length. Aligning such sequences is very easy for both `Muscle` and maximum likelihood procedures, especially when the evolutionary trees are ultrametric, as they are in this case. As mentioned in the previous sections, `Muscle` uses the UPGMA clustering algorithm for constructing its guide tree, and the UPGMA algorithm performs best when evolutionary relationship between those sequences is ultrametric. The ML method, on the other hand, does not incorporate indel probability rate parameters in their model, and these rates are very small in this case anyway. The site variation is due only to the substitution rates. These parameters are incorporated into the JC69 model and can be inferred from the data.

As for models with medium indel probabilities (indel $= 5 \times 10^{-4}$), Figure 4.2 shows that the improvement of our iterative procedure is significant on models that are derived from very high evolutionary rates (scale = 64). However, Welch's two sample t-test also reveals that the improvement on smaller scales is significant. For example, in experiments with $(\text{indel}, \text{scale}) = (5 \times 10^{-4}, 32)$ settings, the critical value of Welch's test is $t_{797.97} = 8.97$, and p-value $< 2.2 \times 10^{-16}$. Also, for experiments with $(\text{indel}, \text{scale}) = (5 \times 10^{-4}, 16)$ settings, the two values are $t_{797.47} = 4.08$, p-value $= 4.9 \times 10^{-5}$, and the iterative procedure outperforms `Muscle` in that scenario.

For models with the largest indel probability (indel $= 5 \times 10^{-3}$), Welch's two sample t-test reveals that the improvement of our iterative procedure is significant even on smallest evolutionary rates (scale = 1) ($t_{792.7} = 2.35$, p-value = 0.01875). The improvement is also significant for larger evolutionary rates. This significance on scale = 8 setting is obtained by Welch's test ($t_{796.88} = 17.73$, p-value$< 2.2 \times 10^{-16}$). On larger evolutionary rates (scale = 16 and scale = 32), the significance of the improvement is obtained by the fact that both X and Y satisfy normality conditions and that their standard deviation bars do not overlap.

For experiments with largest evolutionary rates and largest indel probabilities, the sample X does not satisfy normality conditions. However, since the standard deviation bars do not overlap, we conclude that the iterative procedure provides an improvement

over `Muscle`. The histogram of ML scores of the sample X, Figure 4.3), shows that the distribution of sample X is slightly skewed to the left.

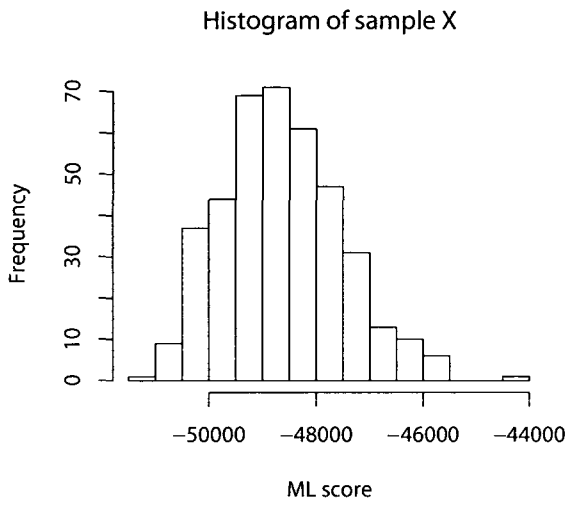

Figure 4.3: Histogram of `Muscle` ML score for $(\text{indel}, \text{scale}) = (5 \times 10^{-3}, 64)$

Figure 4.2 shows that, with respect to the ML score produced by the iterative procedure and `Muscle`, the iterative procedure provides a significant improvement over `Muscle` on models with medium indel probability and medium-to-high evolutionary rates, $(\text{indel}, \text{scale}) \in \{(5 \times 10^{-4}, 16), (5 \times 10^{-4}, 32), (5 \times 10^{-4}, 64)\}$, and on all models with high indel probabilities.

Figure 4.2 also shows that the absolute values of the ML score of both `Muscle` and ML alignments increases at least linearly (the alignment scores get at least linearly worse) as the evolutionary rate increases. The variance also increases as the function of evolutionary rate, except for the case of $(\text{indel}, \text{scale}) = (5 \times 10^{-2}, 64)$. We have seen in Figure 2.6 that as the site difference approaches the saturation threshold (0.75), it becomes very difficult to reconstruct branch lengths. Models with high evolutionary rates result in high site saturation and the ML method fails to recover the true branch lengths, lowering the absolute value of the ML score.

4.2.2 Likelihood improvement and RF distance to true trees

In this section we compare the iterative procedure and `Muscle` in their ability to reconstruct the true evolutionary tree. Each algorithm produces a tree representing its best effort at reconstructing the true tree, and we use the RF distance to calculate the distance between each derived tree and the true tree. The two samples $X = \{X_1, \ldots, X_n\}$ and $Y = \{Y_1, \ldots, Y_n\}$, represent these distances (RF distance between the tree inferred based on the `Muscle` and iterative alignment, respectively, and the true evolutionary tree). We also investigate how the improvement in ML score translates into an improvement in RF distance. The two samples X and Y do not satisfy normality conditions in any of the simulation settings.

Figure 4.4 shows how the average RF distance between inferred trees and true trees changes as a function of different evolutionary rates and indel probabilities. Given that the distribution of the two samples does not satisfy the normality condition, we cannot test, using parametric methods, if the improvement of the iterative procedure is significantly better than that of `Muscle` (even in the case where the standard deviation error bars overlap the least, $(\text{indel}, \text{scale}) = (5 \times 10^{-3}, 64)$). Even if $\mu(Y)$ is significantly lower from $\mu(X)$ in this case, other cases suggest that the improvement of the iterative procedure, with respect to the ML score from the previous section, does not translate into an improvement in RF distance. For example, the RF score of the iterative procedure for the experiment with $(\text{indel}, \text{scale}) = (5 \times 10^{-4}, 64)$ settings is worse than the RF score of the `Muscle` algorithm. However, in the previous section, we have seen that in the case of the ML score, for the same simulation settings, the iterative procedure had an ML score that was significantly better that `Muscle`'s.

While the |ML score| increases as the evolutionary rate increases (Figure 4.2), the RF distance, as the function of evolutionary rate and indel probability, shown in Figure 4.4, is concave. Considering only the models with the smallest evolutionary rates (scale = 1), we see that the accuracy of predicting the true evolutionary tree is by far worse than it is for models with larger evolutionary rates (scale = 8 or scale = 16). This phenomenon occurs

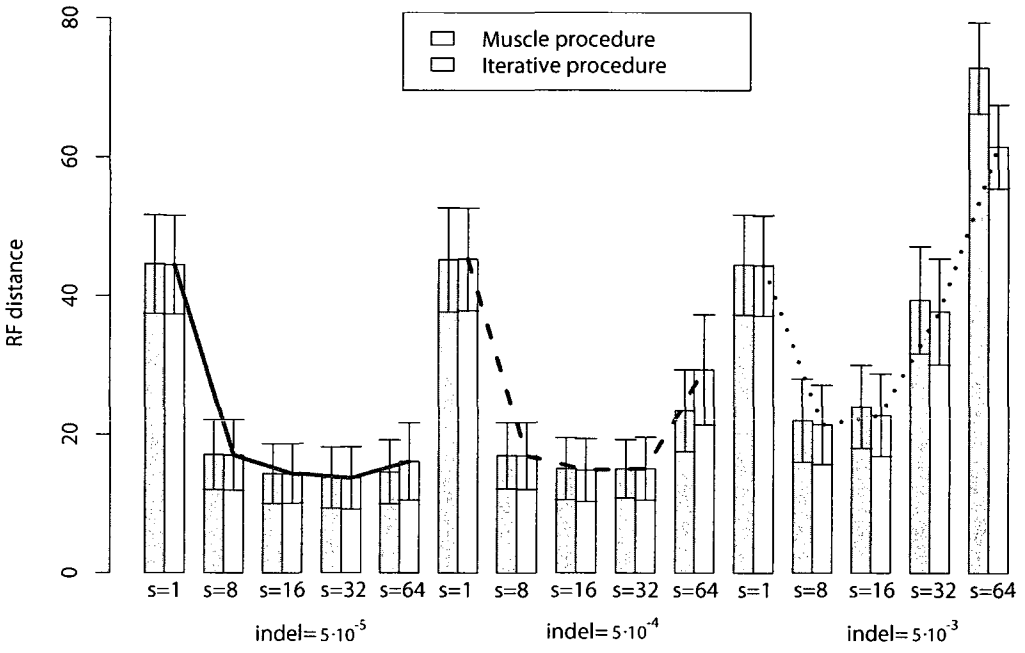

Figure 4.4: Average RF distance between the true tree and the tree produced based on `Muscle` alignments (gray bars), and the average RF distance between the true tree and the tree derived from the best iterative alignments (white bars).

because the models with the smallest evolutionary rates (scale = 1) tend to produce evolutionary trees with branches of very small length. Predicting the length of these branches is a challenge for phylogenetic methods. `PhyML`, for example, sets the inferred length of many small branches to zero, in which case the trees are not binary and tree distance between such trees can be very large, as shown in Figure 4.4.

The tree in Figure 4.5 is generated by `Rose` with indel probability of 5×10^{-5} and an evolutionary rate of 1. The tree is not ultrametric. Figure 4.6 shows the tree inferred by `PhyML` based on the alignment of the sequences generated by the tree in Figure 4.5. The

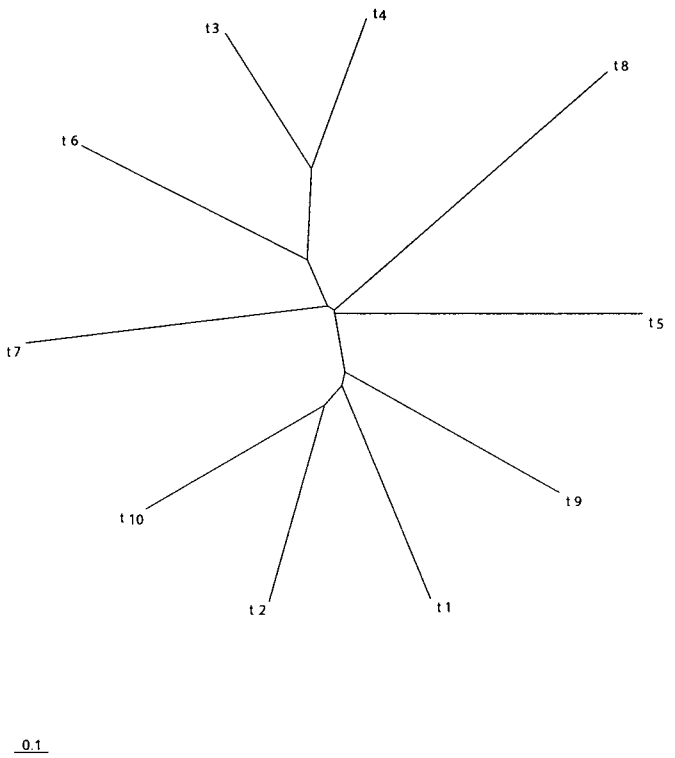

Figure 4.5: Sample non-ultrametric tree generated by Rose with (indel $= 5\times10^{-5}$, scale $=$ 1) parameters

structure of the two trees differs significantly: the true tree is binary, while the inferred tree is not binary (it is multifurcating). Four nodes of the inferred tree ($t5, t6, t7$ and $t8$) share the same ancestor, with one of these nodes ($t6$) overlapping with the actual ancestor due to the fact that its distance to its immediate ancestor is zero. The RF distance between the two trees is 13, although the "true" tree has only 17 edges. By visual inspection it may seem that the trees are more similar than the RF metric indicates (both $((t2, t10), t1)$ and $((t4, t3), t6)$ subtrees are present in the two trees) but the branch length of zero, in the inferred tree, impacts very heavily on the RF distance.

The above example demonstrates a problem of reconstructing evolutionary trees with very short branch lengths: short branch lengths often correspond to too short a time for

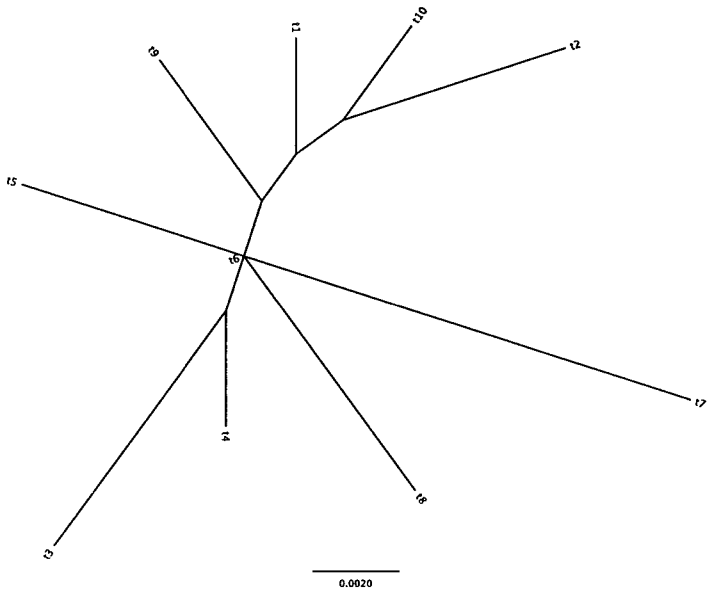

Figure 4.6: Reconstructed tree from alignments with small evolutionary rate. The inferred tree is not binary.

an evolutionary event to occur. That makes the reconstruction of short branches very difficult. This result is consistent with a previous study that showed that both parsimony and ML-based methods are less successful in reconstructing trees of very small branch lengths than trees with larger branch lengths [51].

```
                    10        20        30        40        50        60
t8/1-750 ATAGGTCTCAGGATAACCGGGCTGAGCTCAGACCATCGTCACTCATCAATCGCAGAGATAAC
t2/1-750 ATAGGTCTCAGGAAAACCGGGCTGAGCTCAGACCATGGCCACTCATCAATCGCAGAGATAAC
t10/1-75 ATAGGTCTCAGGATAACCGGGCTGATCTCAGACCATGGCCACTCATCAATCGCAGAGATAAC
t1/1-750 ATAGGTCTCAGGATAACCGGGCTGAGCTCAGACCATGGCCACTCATCAATCGCAGAGATAAC
t9/1-750 ATAGGTCTCAGGATAACCGGGCTGAGCTCAGACCATGGCCACTCATCAATCGCAGAGATAAC
t5/1-750 ATAGGTCTCAGGATAACCGGGCTTAGCTCAGACCATGGTCACTCATCAATCGCAGAGATAAC
t6/1-750 ATAGGTCTCAGGATAACCGGGCTGAGCTCAGACCATGGTCACTCATCAATCGCAGAGATAAC
t3/1-750 ATAGGTCTCAGGATAACCGGGCTGAGCTCAGACCATGGTCACTCATCAATCGCAGAGATAAC
t4/1-750 ATAGGTCTCAGGATAACCGGGCTGAGCTCAGACCATGGTCACTCATCAATCGCAGAGATAAC
t7/1-750 ATAGGTCTCAGGATAACCGGGCGGAGCTCAGACCATGGTCACTCATCAATCGCAGAGATAAC
```

Figure 4.7: Sample alignment of ten sequences generated by a model with parameters $(\text{indel}, \text{scale}) = (5 \times 10^{-5}, 1)$

Figure 4.7 shows the alignment of the sequences produced by the evolutionary tree shown in Figure 4.5. The tree shown in Figure 4.6 was generated based on this alignment. We see that the sites contain very few substitutions. Many site columns contain only one nucleotide and they provide no information to ML methods.

	scale=1	scale=8	scale=16	scale=32	scale=64
indel=5×10^{-3}	22.3	3.3	2.1	1.6	1.5
indel=5×10^{-4}	23.4	3.2	2	1.6	1.3
indel=5×10^{-5}	23.4	2.9	2.1	1.6	1.4

Table 4.1: Percent of branch lengths set to zero by the tree inference algorithm

The percentage of branch lengths that the tree inference algorithm sets to zero is shown in Table 4.1. As expected, for models with the smallest evolutionary rates, a large number of branch lengths are set to zero (almost a quarter of them). This number drops quickly as the evolutionary rate increases. But even for models with the largest evolutionary rates, an original tree may still contain few branches of very short length that are very hard to reconstruct.

Another interesting observation from Figure 4.4 is that for a fixed evolutionary rate, the error of reconstructing the true tree increases as the indel probability increases. This is due to the fact the ML method does not have a parameter that models the indel probability. The ML algorithm is an optimization algorithm in which the optimization device is tightly coupled with the underling model assumption and the further our assumed model is from the real model, the worse our ML optimization result will be.

During the n iterations of the same replicate, our procedure goes through a sequence of pairs of MSAs and their corresponding trees, $(A_j, T_j), 1 \leq j \leq n$, and it picks the pair $(A_i, T_i), 1 \leq i \leq n$, whose MSA A_i has the best ML score. The alignment A_i is closest to the true alignment (with respect to the ML distance and under the assumed model) but its corresponding guide tree T_i does not necessarily correspond to the tree that is closest (in any given distance) to the true evolutionary tree. The ML optimization criterion is a

unary operator which depends only on the alignment at hand and we can always choose the pair with the best ML score. However, the RF distance is a binary operator, one of whose operands is the true evolutionary tree, which we do not know in practice and so cannot compare against. In this respect, the ML distance is the primary distance (since it determines if a pair (A_j, T_j) will be chosen or not), while the RF distance is the secondary distance: the best tree was not chosen because it was closest to the true tree but because the alignment it was built from had the best ML score. Making the RF distance primary (by choosing a pair (A_j, T_j) whose tree is closest to the true evolutionary tree and not whose alignment has the best ML score) would still not result in significant improvement of the iterative procedure in any experimental settings (see appendix A). It is not possible to make the RF distance primary in practice, for the reasons mentioned above. However, if by making the RF distance primary, the improvement of the iterative procedure, with respect to the RF distance, was significant in any case, then it would have been worth investigating how to choose the best tree of all pairs (A_i, T_i) whose alignments A_i have the same ML score. For example, if in a given run, we have k alignments with the same best ML score A_{best}, $(A_{best}, T_j), 1 \leq j \leq k \leq n$, the current iterative procedure chooses the first tree as the best tree. If making the RF distance primary resulted in significant improvement of reconstructing the tree tree, then one could investigate how to choose not the first, but the best tree of all k trees.

4.2.3 Likelihood improvement and alignment quality score

In this section we compare the two methods with respect to their ability to reconstruct the true multiple sequence alignment. The two samples $X = \{X_1, \ldots, X_n\}$ and $Y = \{Y_1, \ldots, Y_n\}$ represent similarity score between the `Muscle` alignment and the true alignment, and between the iterative alignment and the true alignment. The alignment similarity score is calculated according to Equation 3.1.

Figure 4.8 shows that as the evolutionary rate increases, both methods are less able to reconstruct the true alignment and the alignment quality ranges from 100% for models

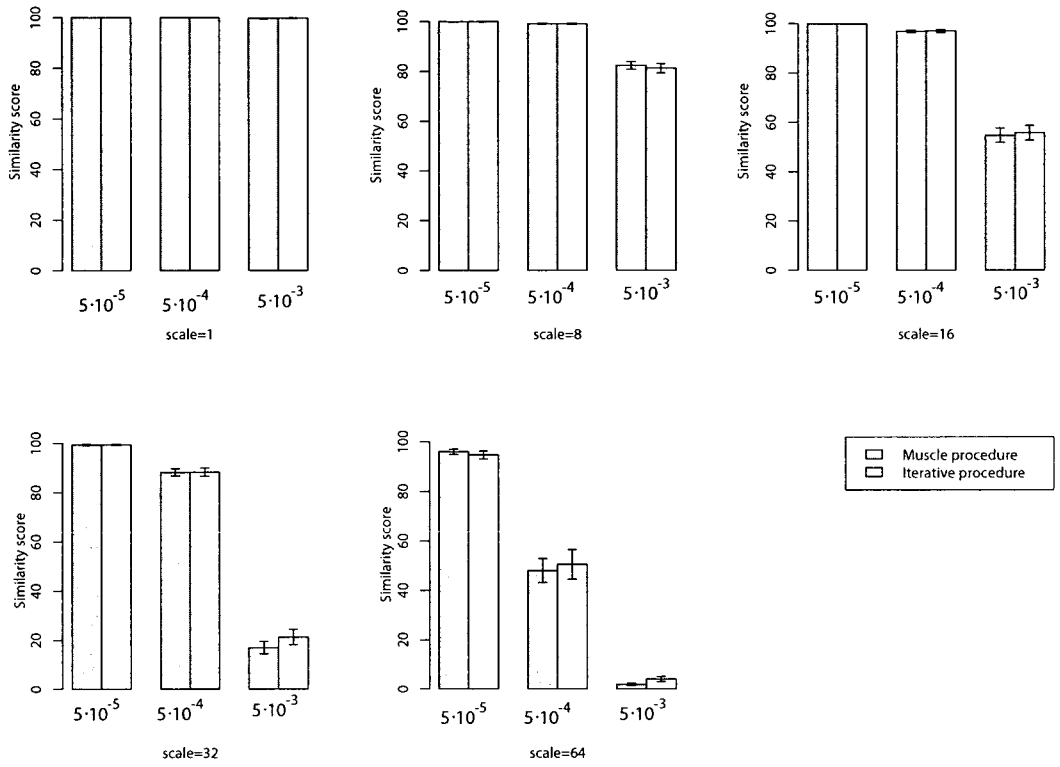

Figure 4.8: Alignment similarity score between the true alignments and the alignments constructed by `Muscle` algorithm (gray bars). White bars indicate the similarity score between true alignments and the alignments constructed by the iterative procedure

with smallest and medium evolutionary rates (scale = 1 and scale = 8) to single digit percentage for alignments from models with very high evolutionary rate. Figure 4.8 also shows that the improvement of the iterative procedure, with respect to the alignment similarity score, lags behind the improvement with respect to the ML score distance. For example, for models with scale = 1, there can be no improvement, since the true alignments are perfectly reconstructed. No improvement is obtained on models with scale = 8 evolutionary rate. The improvement of the iterative procedure is significant for $(\text{indel}, \text{scale}) = (5 \times 10^{-3}, 16)$ settings. At confidence level of $\alpha = 0.05$, the critial value of Welch's test is $t_{797.55} = -5.0329$ and p-value $= 5.976 \times 10^{-7}$. The samples X and Y do

not satisfy normality assumptions for models with larger evolutionary rates (scale = 32 and scale = 64) and therefore, no significance of the improvement can be obtained using parametric tests.

The alignment similarity scores from Figure 4.8 clearly demonstrate the degree to which the ML methods perform poorly on reconstructing the phylogeny trees with smallest evolutionary rates (which result in evolutionary trees having smaller internal branches). Given a set of sequences which have evolved according to the model with very small indel probability (indel = 5×10^{-5}), we can reconstruct their true alignment perfectly (for scale < 64) and yet the reconstructed phylogeny is very distant from the true tree (with RF distance greater than 40, as shown in Figure 4.4). One can expect no further improvement in the alignment (after obtaining the perfect alignment) and so no justifiable improvement in phylogeny reconstruction can be expected using the iterative procedure.

	scale=1	scale=8	scale=16	scale=32	scale=64
indel=5×10^{-3}	4.64	3.1	4.45	14.66	28.98
indel=5×10^{-4}	1.33	1.95	2.09	2.53	10.59
indel=5×10^{-5}	1.26	1.23	1.43	1.62	2.91

Table 4.2: Average number of iterations required in iterative procedure in order to obtain the best MSA and its corresponding guide tree

From the Table 4.2, we see that to reconstruct the "true" alignment with sequences from scenarios with very small evolutionary rates (scale = 1), we have to perform, on average, at least five iterations (rounded value of 4.64). This number grows as the indel probability increases, which is expected since it is harder to align such sequences. Also, for a fixed indel probability, the average number of iterations required to obtain alignment with the best ML score, grows as the evolutionary rate increases and ranges from five iterations for models with smallest evolutionary rates and large indel probability, to 29 iterations for models with largest evolutionary rate. Larger numbers of required iterations

justify the use of the iterative procedure because it shows that we can get better result, with respect to the ML score, than we could get by using the standard ML methods. On the other hand, smaller values of the average number of required iterations also justify the use of the iterative procedure because it shows that we can get better results than we could get with the standard ML procedure without having to perform a large number of iterations. Inferring a phylogeny using the maximum likelihood methods can be computationally expensive. The results shown in Table 4.2 also suggests a possible stopping criterion for the iterative procedure. One could find a 95% confidence interval for the average number of iterations required to find the best score and then use the upper bound of such interval as the stopping rule.

4.2.4 Likelihood improvement and Branch Score distance

In this section we investigate how the Branch Score distance changes as we optimize the ML function. The two samples $X = \{X_1, \ldots, X_n\}$ and $Y = \{Y_1, \ldots, Y_n\}$ contain the values of Branch Score distance between the tree inferred based on the `Muscle` alignment and the true tree (X), and the distance between the tree inferred based on the iterative alignment and the true tree (Y).

Figure 4.9 shows that the two procedures produce the same results. Neither the average Branch Score distance between true trees and inferred trees, nor its variance, seem to be sensitive to either an evolutionary rate (for a fixed indel probability) nor indel probability (for a fixed evolutionary rate). An improvement in Branch Score distance would require that the reconstructed tree (or its sub-trees) match the original tree both in tree topology and branch lengths and this is a more rigorous requirement than the requirements set by other distances that we use here. Figure 4.9 also shows that as we double evolutionary rate parameter, so does the Branch Score distance between the inferred trees and the true trees. This is so because the ML algorithm fails to recover lengths of branches that are greater than 1 (this will be shown in Section 4.5). We will not be considering the Branch Score distance in the rest of out study.

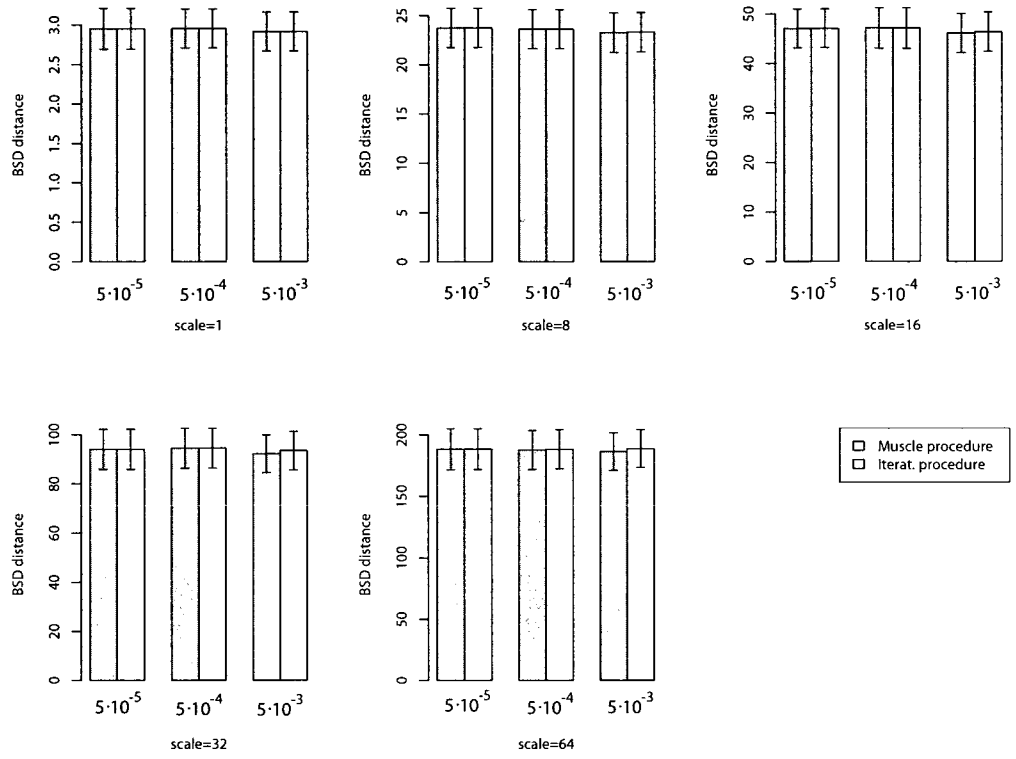

Figure 4.9: Average Branch Score distance between the true evolutionary tree and the tree inferred from `Muscle` and iterative procedure, respectively.

4.2.5 Likelihood improvement and Agreement distance

The two samples $X = \{X_1, \ldots, X_n\}$ and $Y = \{Y_1, \ldots, Y_n\}$ contain values of the Agreement distance between the inferred tree based on the `Muscle` alignment and the true tree, and the distance between the inferred based on the iterative alignment and the true tree. Normality conditions are not met in any experimental settings, for both samples.

Figure 4.10 shows $\mu(X)$ and $\mu(Y)$ and their standard deviations, for all experiments. Similar findings from the previous section are also present with the Agreement distance. Namely, the Agreement distance on trees from models with small evolutionary rate (scale = 1) is larger than it is on models with larger evolutionary rates (scale $\in \{8, 16, 32\}$). This is not surprising because the Agreement distance suffers from

the same shortcomings of tree inference as the RF distance does, which is comparing inferred multifurcating trees with true binary trees.

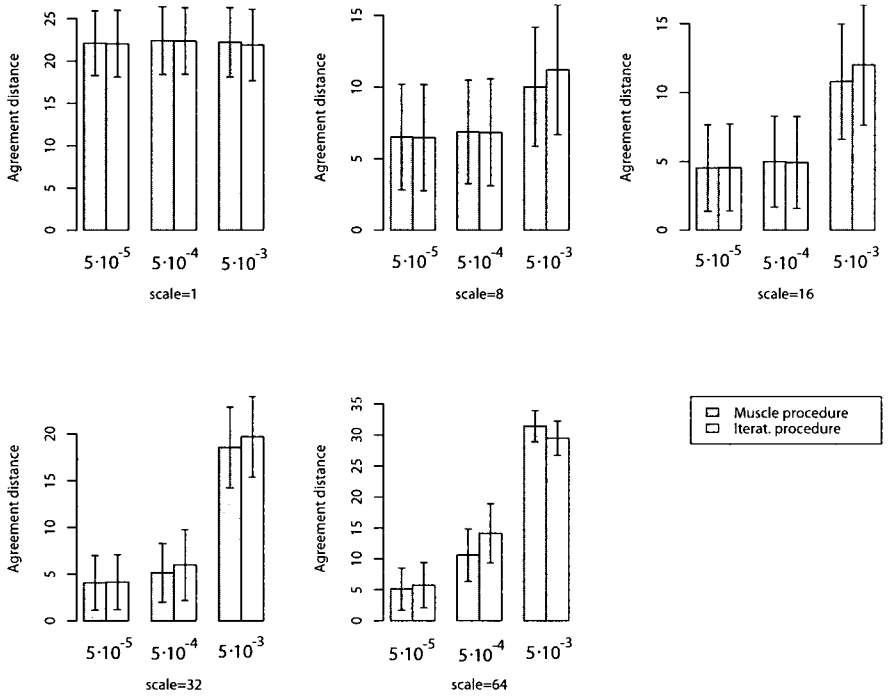

Figure 4.10: Average Agreement distance between the true evolutionary tree and the tree inferred from `Muscle` and iterative procedure, respectively.

However, multifurcating trees are not the sole cause that drive the results of the Agreement distance. To see this, it is necessary to view leaf nodes of a tree differently from its internal nodes. For example, leaf nodes of any inferred tree are observations, and there is no uncertainty as far as the nucleotide that they represent is concerned. However, their ancestor nodes are inferred properties of the tree and there is uncertainty associated with them. The immediate ancestors of observed sequences have some uncertainty (even though their child nodes are observations and they are "certain"), but the ancestors of the second generation of the observed sequences have even more uncertainty associated with them, because they are inferred based on nodes that already have some level of

uncertainty with them. Hence, the tree root is the least certain node of the tree. Since the Agreement distance starts removing the observed sequences from the leaf nodes, pruned trees representing the common structure between two trees may contain more uncertain than certain nodes. This implies that the Agreement distance may not be best suitable for calculating the distance between two trees if the certainty of node properties (such as sequence content in our study) is highest at leaf nodes and decreases as the depth of internal nodes increases, with the root node having the highest level of uncertainty.

4.3 Result comparison on ultrametric and non-ultrametric trees

In this section we investigate the results of our procedure on non-ultrametric trees. We do so by comparing them to the results on ultrametric trees which we have presented in the previous section.

4.3.1 Comparison of ML distances

Figure 4.11 shows the ML scores for both ultrametric and non-ultrametric trees. Each column in the graph represents the average ML score of non-ultrametric trees and the error bars at its top represent the standard deviation for the given sample. The error bars located completely within a column, represent the standard deviation of the corresponding sample when taken on ultrametric trees.

Several observations can be made from Figure 4.11. First, the average ML score for ultrametric trees is always better (since its absolute value is smaller) than the ML score for non-ultrametric trees. This holds equally true for both `Muscle` method and our iterative procedure.

The two samples X and Y do not always satisfy normality conditions and comparing the two methods (within the context of parametric statistics) becomes a lot harder on

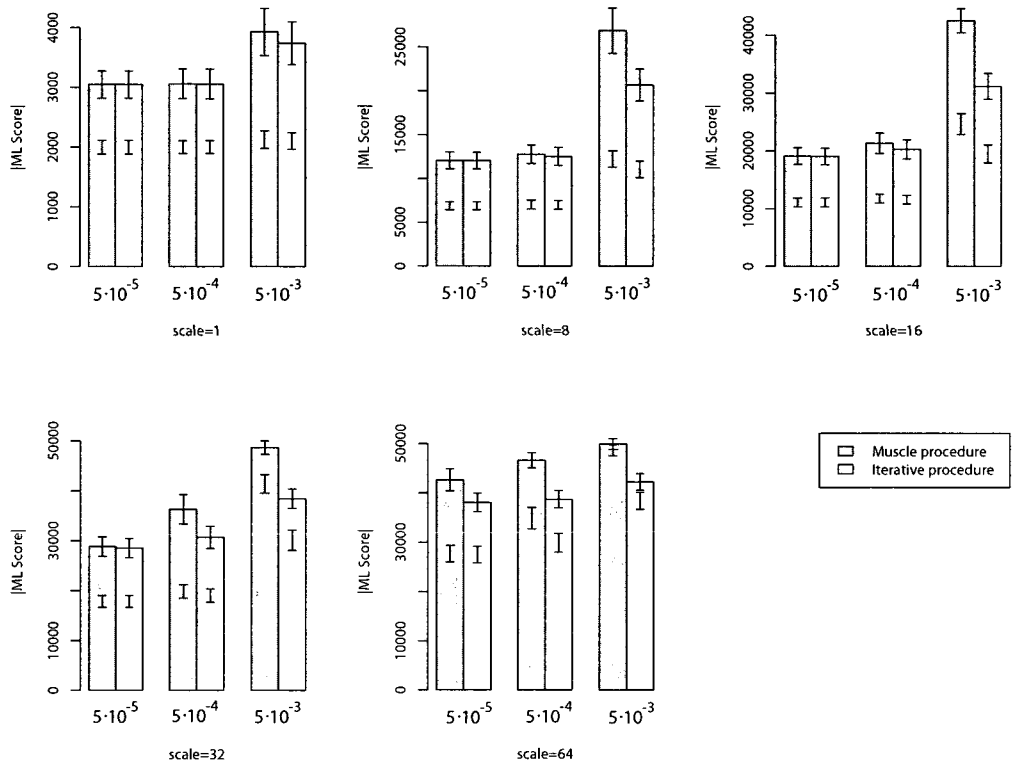

Figure 4.11: Average ML scores for the alignments constructed by `Muscle` and our iterative procedure for sequences generated by both ultrametric and non-ultrametric evolutionary trees. The error bars for ultrametric trees are shown inside bars (since their ML score is smaller) and error bars for non-ultrametric trees are shown at the top of bars.

non-ultrametric than on ultrametric trees. For example, the iterative procedure was significantly better (with respect to the ML score) than `Muscle` on ultrametric models with the settings of $(\text{indel}, \text{scale}) = (5 \times 10^{-3}, 1)$. However, on non-ultrametric trees, the samples X and Y do not satisfy the normality conditions in that particular case. Even though the difference between $\mu(Y)$ is $\mu(X)$ is greater on non-ultrametric than ultrametric trees, this still does not tell us if this improvement is significant or not. The scenario with $(\text{indel}, \text{scale}) = (5 \times 10^{-3}, 16)$ setting manifest the same phenomenon.

On the other hand, for experiments with $(\text{indel}, \text{scale}) = (5 \times 10^{-3}, 8)$ settings, the improvement of the iterative procedure was not significant on ultrametric trees, and on non-ultrametric trees it is significant ($t_{710.15} = -38.9744$, p-value $< 2.2e - 16$). On larger evolutionary models (scale = 32 and scale = 64), samples X and Y do not satisfy normality conditions and the improvement may not be significant.

In summary, the iterative procedure on non-ultrametric trees widens the gap with `Muscle`, in quantitative terms, and we hypothesize that this improvement may be significant even when the normality conditions are not met. However, this improvement cannot be verified using parametric statistical tests.

4.3.2 Comparison of RF distances

Figure 4.12 shows the RF scores for non-ultrametric trees (solid error bars). Error bars for ultrametric trees are shown as dashed lines so that the RF distances on two types of trees can be compared. These results are similar to the results shown above on ultrametric trees. The closest we get to improvement is for the case with the largest evolutionary rate and the largest indel probability ($(\text{scale}, \text{indel}) = (64, 5 \times 10^{-3})$). However, since the normality assumption is not met, the significance of the improvement of the iterative procedure cannot be assumed with parametric tests.

The RF distances on non-ultrametric trees are considerably lower for models with smallest evolutionary rates than they are on ultrametric trees. In all other cases, RF distances on non-ultrametric trees are larger than they are on ultrametric trees. As we have pointed out in the section on ultrametric trees, the RF distances between the true and inferred trees are rather large on such models due to the fact that many internal branches of the inferred tree may be of length zero. We generate non-ultrametric trees from ultrametric trees (by multiplying the weight of each tree edge with 1, 2 or 3). This leads to fewer internal edges having very small length and therefore fewer internal edges of the inferred tree end up being of length zero. That, in turn, results in smaller RF distance between the true and the inferred phylogeny tree.

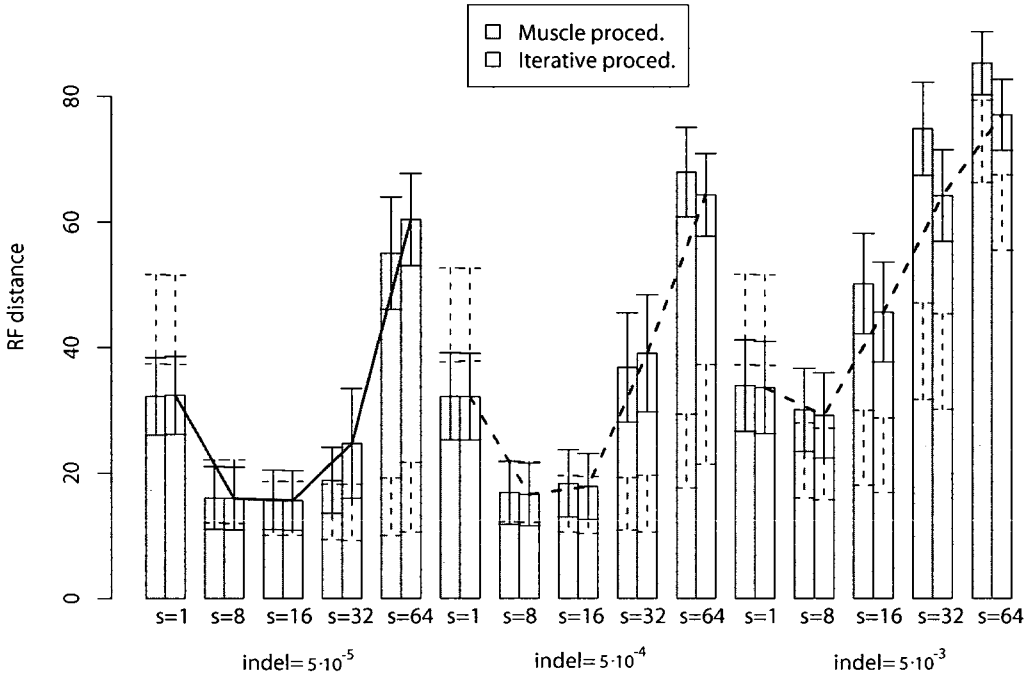

Figure 4.12: Average RF distance between true trees and trees inferred based on `Muscle` alignments, and the average RF distance between true trees and trees inferred based on iterative alignments. The standard deviation error bars for ultrametric trees are shown inside bars (since their RF distance is smaller) and standard deviation error bars for non-ultrametric trees are shown at the top of bars.

As the evolutionary rate increases, so does the difficulty of reconstructing the true tree. This is best demonstrated on the models with smallest indel probabilities. For such models, Figure 4.12 shows the effect of both saturation (high evolutionary rate) and small branch lengths (small evolutionary rate). Both cases result in large RF distances, with saturation being more difficult to resolve than the lack of information.

4.3.3 Comparison of alignment similarity scores

Similar to graphs in the previous two sections, Figure 4.13 shows the average values of alignment similarity scores for both methods on ultrametric trees (error bars right above columns) and non-ultrametric trees (error bars drawn on the top of columns).

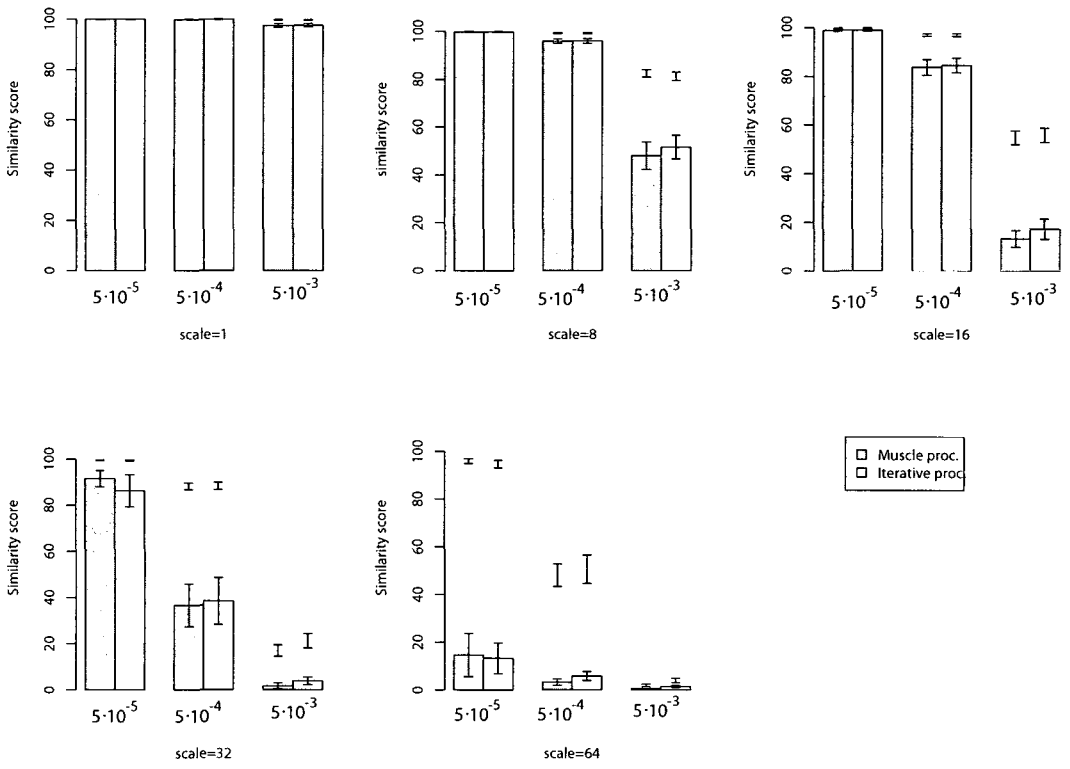

Figure 4.13: Similarity scores between the true alignment and the alignment produced by `Muscle` and the iterative procedure, respectively. The error bars for ultrametric trees are located above the bars

Again, no improvement in alignment score is obtained with the iterative procedure over the `Muscle` algorithm. Both methods start with almost perfect reconstruction of the true alignment (on models with smallest evolutionary rates) and end with completely different alignments (very small alignment score) for models with highest evolutionary rates. This holds true for both ultrametric and non-ultrametric trees; however, the drop

from perfect reconstruction to complete miss in alignment is faster for non-ultrametric trees. For example, the success of reconstructing the true alignment for models with second largest evolutionary rate (scale = 32) on non-ultrametric trees seems to be comparable to the success of reconstructing the alignment with the largest evolutionary rate (scale = 64) for ultrametric trees. Also, the alignment similarity score for models with very high indel probability does not exceed 20% in all but on those models with very small evolutionary rates (scale = 1 or scale = 8).

The difference between any two error bars for a particular evolutionary model in Figure 26 can be used to see the models for which we get the largest drop in alignment similarity score. The largest such drop occurs for model with (scale, indel) = $(64, 5\times10^{-5})$ parameters. This is not surprising because under this model, our inferred evolutionary trees, when measured by RF distance, were furthest from the true evolutionary trees, as shown in Figure 25.

4.4 Testing the algorithm on real data

As mentioned in section 3.5, we ran `Baliphy` program on a data set of 62 sequences with each sequence containing at least 2500 nucleotides. We ran the program with two different seed numbers, resulting in two runs that we call here Run 1 and Run 2. As the Figure 4.14 shows, by plotting only the values of posterior distribution for these two runs, both distributions seem to converge. However, by plotting the two data sets starting from iteration 500, as shown in Figure 4.15, we observe that only Run 1 may have reached stationarity and Run 2 does not converge since its likelihood values seem to follow an increasing trend. The number 500 was chosen just to illustrate the convergence idea.

Closer inspection of the graph in Figure 4.15 shows that even Run 1 does not converge. For example, every log-likelihood value in Run 1 is smaller than −35700 (horizontal bar on the graph), while most of the latest iterations in Run 2 produce log-likelihood

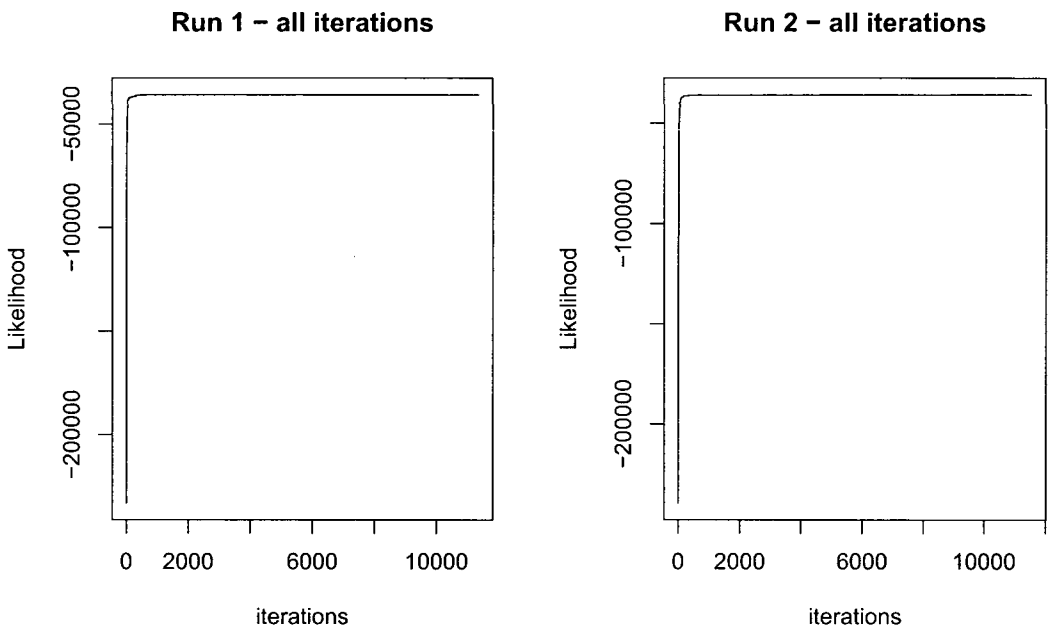

Figure 4.14: Posterior distribution values of a tree and an alignment during the `Baliphy` run for all iterations

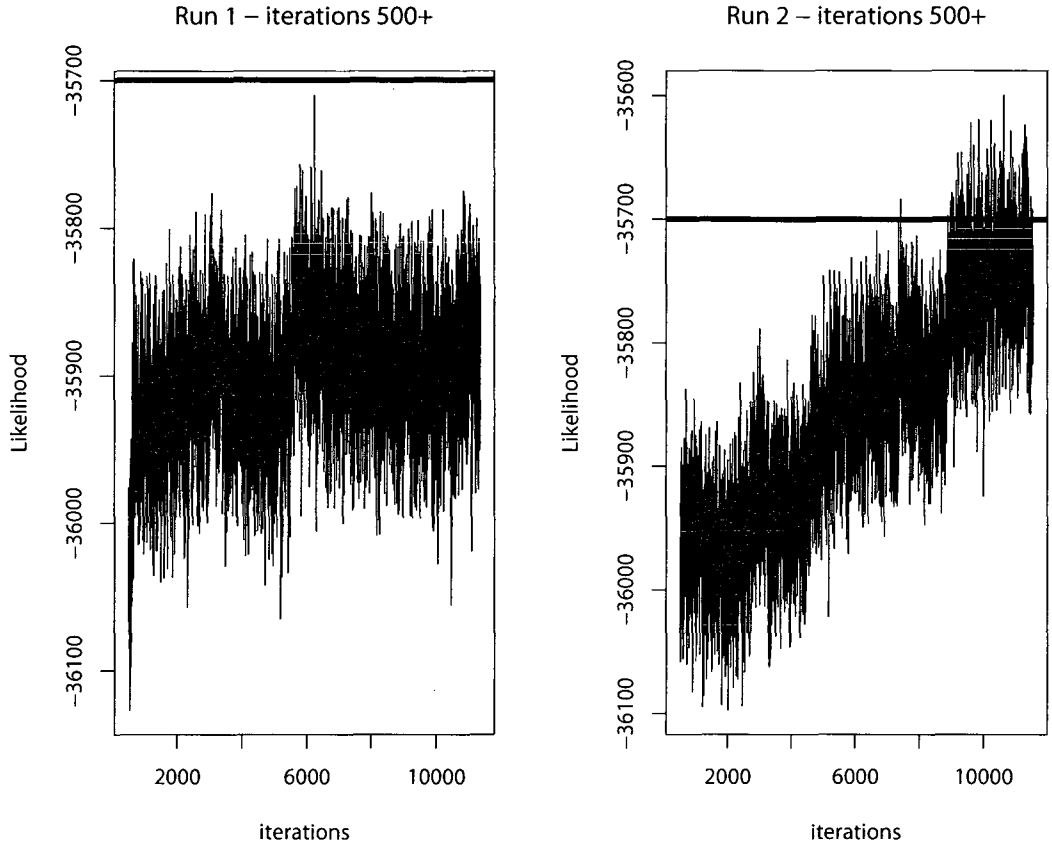

Figure 4.15: Posterior distribution values of a tree and an alignment during the `Baliphy` run, starting from iteration 500.

values greater than that threshold. This clearly shows that neither of the two runs is converging. While the authors of `Baliphy` [31] have shown in their simulation study that the likelihood of sequence alignments and evolutionary trees converges for small data sets, our simulation study shows that it may take a very long time for that convergence to take place on larger data sets. The above results are obtained by `Baliphy` after leaving it running for more than five months. Given the trend of the two likelihood graphs, it is unclear that the result in the next few months would answer the convergence question.

If X and Y are two samples as explained in section 3.5, then for Run 1, we have $\mu(X) = 27.2$ and $\mu(Y) = 24.12$. By using Welch's two sample t-test, we obtain the critical value of t-test ($t_{443.09} = 18.55$), and p-value (p-value $< 2.2 \times 10^{-16}$), which show that $\mu(Y)$ is significantly lower than $\mu(X)$. If convergence criteria was met, then it would indicate that the iterative procedure performs significantly better than `Muscle`, however, due to lack of the convincing convergence argument, our results on the real data set are inconclusive.

4.5 Validating simulation parameters

In every simulation study, the question arises about how realistic the simulation settings are. Here, we followed [34] for choosing parameter values (indel probability and evolutionary rates) for our simulation study. We have also added an additional (larger) indel probability parameter ($indel = 5 \times 10^{-3}$) for two reasons. First, because our method is targeted toward more divergent sequences, we wanted to test our method on such sequences. Second, by visual examination of alignments generated with very small evolutionary rates, we noticed that there were very few (sometimes less than five) gaps in the alignments, and sequences lengths differed by only a few nucleotides. Models with small evolutionary rates are particularly difficult for ML methods (due to lack of information) and we wanted to introduce more evolution in the sequences.

As explained in Section 3.2.2, we followed [34] and performed perturbation of branch

lengths in two steps. During the first step branch lengths are perturbed in order to convert an ultrametric into a non-ultrametric tree. In the second step, [34] multiplies each branch length by 16, 32 or 64. We added one more scale level ($s = 8$) in order to investigate a case between very small evolutionary rate ($s = 1$) and medium-to-high evolutionary rates ($s = 16$, $s = 32$, $s = 64$).

To test how realistic our simulation settings are, we have compared the observed average indel probabilities of our alignments with the observed average indel probabilities of known alignments found in the public repository Pandit [48]. We have also compared the average branch length of our inferred trees with the average branch lengths of the trees found at the same source. However, these comparisons have to be taken with caution. Multiple sequence alignment and phylogeny tree inference tasks are often performed on a set of related sequences. The degree of their relatedness depends on the kind of problem that one may be interested in solving. For example, aligning protein sequences that would help determine a secondary structure of a particular protein, would require a greater level of relatedness between those sequences than aligning sequences in order to infer an evolutionary tree between distantly related sequences.

It should be noted that the inferred (or observed) indel probabilities are not the same as indel parameters used in our simulation. The inferred indel probability is equal to the percentage of indels in the alignment and its value is always greater than the simulation indel parameter. This is so because aligning two profiles may result in multiple insertions of indels in one or both profiles. For example, aligning two profiles in Figure 2.3, both of which have a total of two indels, results in an alignment that has four indels.

Most of the alignments found in Pandit are very similar. For example, 7.94% of them contain no indels. Excluding these alignments, the average percent of indels is 12.78%, and it ranges from 0% to almost 83%, with only 0.75% of them having 60% or more indels. Alignments generated in our simulation with largest indel probability (5×10^{-2}) and largest evolutionary rate (scale = 64), as shown in Table 4.3, result in 87% of indels on average, which may be unrealistic compared to sequences found in Pandit. This might

explain very low accuracy in alignment similarity score in Figure 4.13. Even alignments with smaller evolutionary rates ($16 \leq$scale≤ 64) for the same indel probability (5×10^{-2}) seem to be rather uncommon in the Pandit database.

	scale=1	scale=8	scale=16	scale=32	scale=64
indel=5×10^{-3}	9.6	49	65	78	87
indel=5×10^{-4}	1	9	16.5	28	44
indel=5×10^{-5}	0.09	0.1	1.9	3.8	7.5

Table 4.3: Average percentage of indels across different simulations

We find that our iterative procedure does not perform well in reconstructing branch lengths of the true tree, with the average inferred branch length being smaller than 1 even when the average branch length of the true tree is well above 1. For example, Figure 4.16 shows that the inferred branch lengths of trees built from both true alignments and inferred alignments are smaller than 1, while the average branch length of the true tree is 8.3. The reason for this is not entirely clear. Even with saturation effect, the method should still be able to infer branches whose length is in low single digits. Figure 2.6 shows that the site difference reaches 0.75 as the branch length approaches 4.

Models with evolutionary rate of 32 have branches of average length of 16, while the average length of branches of all trees found in Pandit is only 0.37, with variance of 4. Only 0.3% of the edges of all trees found at Pandit have length of 16 or larger. Therefore, further research is required to find out why `PhyML` performs so poorly reconstructing tree branch lengths.

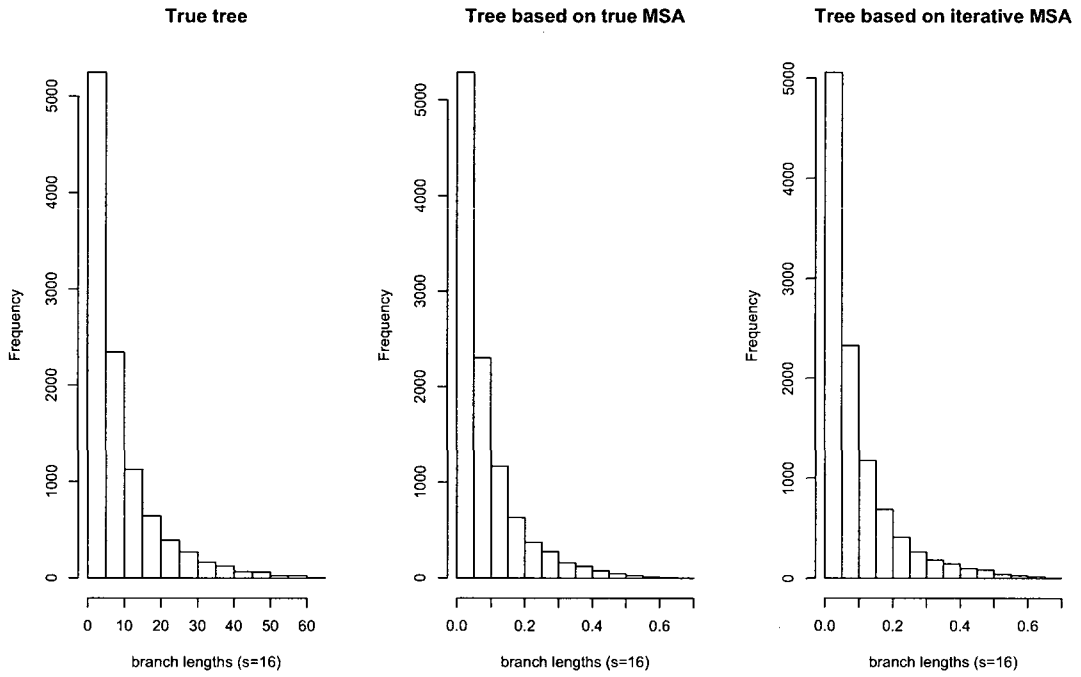

Figure 4.16: Distribution of branch lengths both for true trees and inferred trees (the trees inferred based on the true MSA and the iterative MSA). The trees were taken from models with medium evolutionary rate (scale = 16) and medium indel probability (indel = 5×10^{-4})

Chapter 5

Discussion

Multiple sequence alignment and phylogeny inference are two closely related problems that have historically been addressed separately. A parsimony-based approach was suggested in [34] for solving these two problems simultaneously. On the other hand, [31] has attempted to solve the two problems using Bayesian framework and it is very computationally expensive. Our goal was to develop an algorithm that would be faster than [31] and be better suited for divergent sequences than [34].

In this study, we have shown that the iterative procedure, with respect to maximum likelihood, can produce improvement over a single-step maximum likelihood algorithm, and that this improvement is statistically significant on medium to very divergent sequences. This improvement is not translated into improvements with respect to other distances, like a tree distance or an alignment similarity score distance. There are at least two reasons for this. First, the ML method fails to reconstruct branches of very small length and produces multifurcating trees. This weighs very heavily on tree distance function. Second, all of the distance functions are heristic and their results, at best, measure only one aspect of the concept under study. Some distances may not be well suited in phylogeny studies. For example, the Agreement distance starts removing nodes whose properties are most certain, and ends up with nodes that we are least confident about

(see Section 3.3).

The tree inference algorithm seems to have a considerable weight on the overall success of the iterative procedure. Not only is it unable to reconstruct branches of very small length, but it is also unable to reconstruct branches of length greater than 1, and this goes beyond the limitations imposed by the underling evolutionary model (as can be seen in Figure 2.6). The reasons for this are not entirely clear.

The SP-score is often used as a "measure of goodness" of an MSA. Since our study was conducted in probabilistic terms, we have used the ML score instead; however, to compare two alignments, we have used the alignment similarity score (as defined in Equation 3.1). Both of these scores are based on the same underlying principle, which is to count all pairs of nucleotides across all sites in the MSA. A recent study has suggested that the effect of a guide tree on MSAs and phylogenetic trees may be limited [25] when the alignment is measured in terms of the SP-score. However, in our study we have shown that the alignment score, when measured in terms of the maximum likelihood function, can be significantly improved with better guide trees. Therefore, if a guide tree has a limited effect on alignments and phylogenetic trees, when measured in terms of the SP-score, then it is so not because of guide trees but because of the limitations that are inherent not only in SP-score, but also in other heuristic distances.

Our work also exposes the difficulty that more formal methods, such as `Baliphy`, face when given more realistic data sets. For example, on a data set of only 62 sequences, `Baliphy` did not reach the convergence point even after five months of computations. Given the times series plots of the likelihood sampled from the posterior, there is no any indication that they may start converging soon.

We cannot perform direct comparison of our results and the results obtained by [34], since [34] did not perform statistical significance analysis of the improvement of its algorithm. In case of ML distance, our method always produces numerically better results, however, the improvement is statistically significant only in some cases. On the other hand, the RF distance produces numerically better result in most of the cases, but

they are not necessarily significantly better than those of `Muscle`.

Chapter 6

Conclusion and Future work

The two main algorithms that we have used in our study (`Muscle` for alignment and `PhyML` for tree reconstruction), have been developed independently, each to solve only one of the two problems under study (either an MSA problem or a phylogeny inference problem). We hypothesize that the full advantage of the iterative procedure could be realized only by integrating these algorithms. For example, in current MSA algorithms, only the topology of a guide tree is used as provided in the input, while branch lengths are either completely ignored or are inferred using some ad hoc technique. The inferred branch lengths are often different from the lengths of the input tree. What would be the effect on the result of the iterative procedure if MSA algorithms used both tree topology and branch lengths of the guide tree as they were provided in the input?

Most of the current ML methods consist of a single-routine optimization algorithm. `Muscle` has an interative component but it is different from the iterative component of our algorithm. That raises the following question: how should the ML optimization routine be updated to better adapt to the iterative paradigm? For example, currently the ML methods take an alignment and try to infer a guide tree that best predicts the alignment, and they do so as if the given alignment was correct. In an iterative approach, while inferring an ML tree of the current alignment, an algorithm knows that there is

another alignment coming in the next iteration which may be somewhat similar to the current alignment. It may be possible to adjust the ML optimization routine to better accommodate the iterative approach. This may include earlier stopping criteria (for at least some iterations) or adjusting tree perturbation steps based on the perturbations done during the previous iterations.

In light of `Baliphy`, it may also be worth investigating the optimization problem of simultaneous alignment and phylogeny inference using simulated annealing method. This could be still done within the ML framework and that is why closer integration between ML methods and MSA algorithms is required. Our results in Appendix B show that the iterative procedure may often get stuck between two or more $(MSA, tree)$ pairs, neither of which have the MSA with the best ML score (see the oscillating cases in Figures B.1-B.4). Having a pool of candidates of such pairs at each iteration may increase the chances of the procedure to avoid such traps.

Current distance measures lack conceptual interpretation (in the context of evolutionary studies). For example, the RF distance of 5 does not tell us much about the difference between two trees. We may have two very different evolutionary trees and yet their RF distance may be the same. There may be many different MSAs with the same SP score, and the best one of them, based on the SP score, is usually chosen arbitrarily. Integrating these distance functions across different methods and assuming the same underling assumptions, should improve the overall performance of the iterative procedure. For example, assuming that an evolutionary model has been set, one could calculate the uncertainty of having a nucleotide at a given node and then, in case of the Agreement distance, prune the tree starting with the most uncertain node.

Most of the existing distance functions have one thing in common: mapping a multi-dimensional object (such as an MSA, which is a two-dimensional structure that has evolved from a single sequence over some period of time) into the real line. Comparison of such results is very easy, though most of the time not meaningful, because by the very nature of any such mapping, the result loses so much information about the object

it represents that the end result may be of very limited value. Could the objects of interest be mapped into a space of a dimension larger than 1 but still smaller than the dimension of the space in which the actual object resides? Mapping an MSA of size $k \times n$ (k sequences with each sequence having n nucleotides) into a space of dimension $l \times n$, where $1 \leq l < k$, would result in a mapped value that is more representative of the given MSA, and comparing two MSAs through such values would be more meaningful.

Appendix A

Choosing the best alignment and tree based on RF distance

For a given replicate consisting of t iterations, each producing an MSA and a tree $((A_1, T_1) \cdots (A_t, T_t))$, in our study, we choose a pair whose MSA has the best ML score. This is very easy to implement in practice since one can always calculate an ML score of the alignment. Similar to our notation in the previous chapters, we will denote the set of these values by a sample $X = \{X_1, \cdots, X_t\}$. On the other hand, let sample $Y = \{Y_1, \cdots, Y_t\}$ contain ML scores of alignments whose derived trees are closest to the true evolutionary tree. An optimal solution would produce the same values in the two samples, because the most likely alignment would correspond to the correct evolutionary tree (or a tree that is closest to the true tree). However, no such method exists, and these two samples are different.

As can be seen in the Figure A.1, the average values of the two samples do not change extensively. In fact, they differ significantly only in one case (scale=8, indel)=5×10^{-3}. In this case, the critical value of Welch's two sample t-test is $t_{797.7} = 2.07$, and p-value= 0.039.

To test the improvement in RF distance, the two samples would contain distances

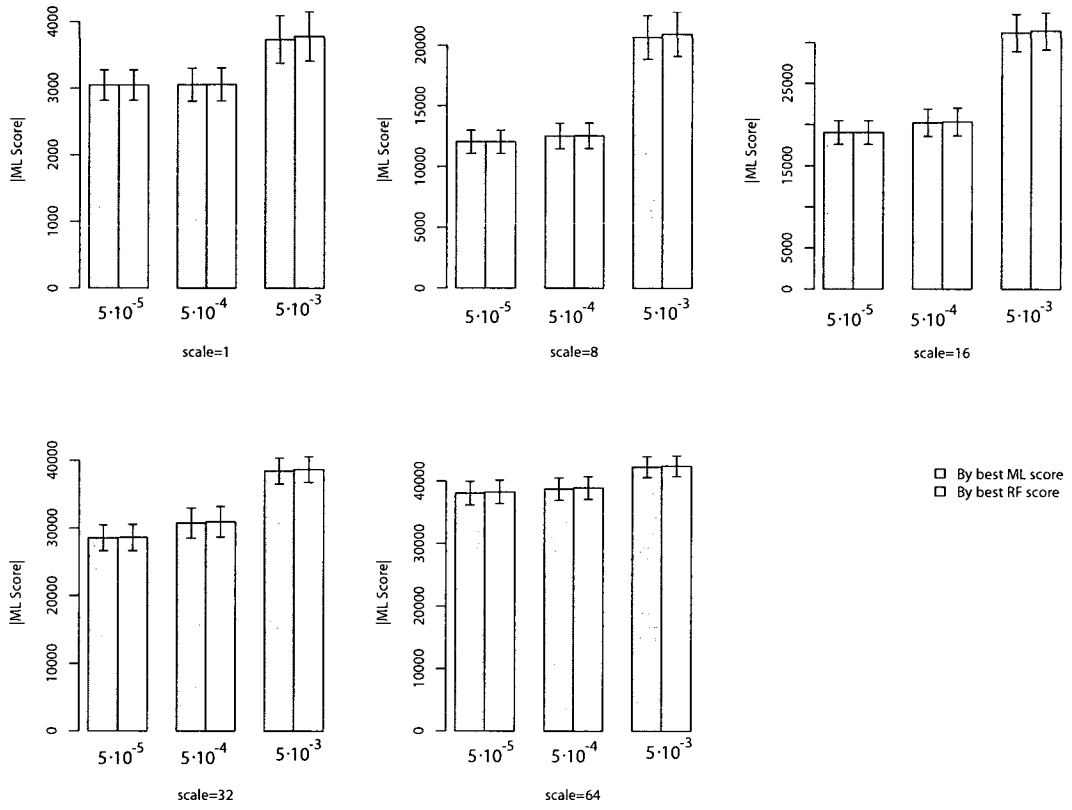

Figure A.1: Average ML scores of alignments whose guide trees are closest to the true tree. Only one case results in significantly worse alignment $(\text{scale}, \text{indel}) = (8, 5 \times 10^{-3})$.

between the true tree and the tree that was inferred based on the alignment with the best ML score (X), and the distance between the true tree and the inferred tree that is closest to the true evolutionary tree (Y). The average values of these two samples are shown in Figure A.2.

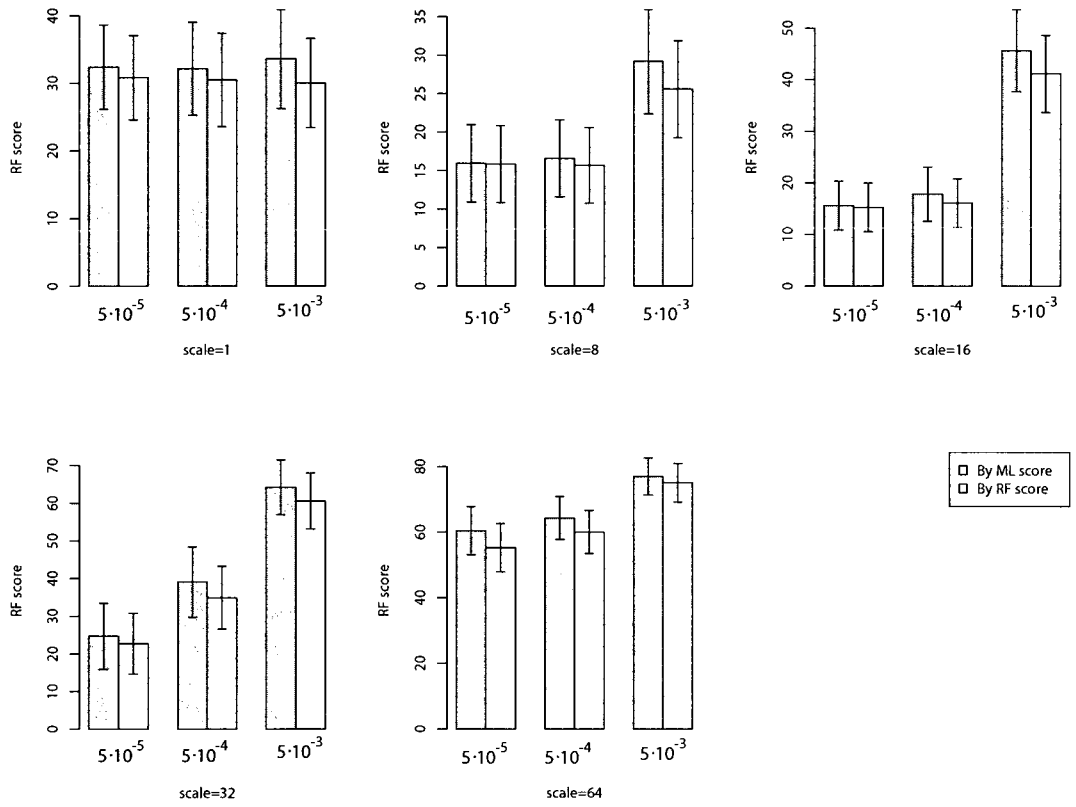

Figure A.2: Average RF distances between the best inferred trees and the true trees.

Two samples X and Y do not satisfy normality conditions, and within the context of parametric statistics, there is no significant improvement between their average values. This shows that choosing an (MSA, tree) pair, based on the best derived tree, even if such a method was available, would still not produce a tree that is significantly closer to the true evolutionary tree.

Appendix B

Likelihood convergence trends

This section presents graphs showing how the ML score changes from one iteration to another. In order to be able to see the trend of that change by visual inspection, we plot ML scores for only 25 randomly selected replicates. The ML score of each replicate is adjusted so that they all start from zero. For example, if t iterations produce the ML scores: $y_1, y_2, \cdots, y_t$, then the adjusted ML scores are: $0, y_2 + |y_1|, y_3 + |y_1|, \cdots, y_t + |y_1|$, where $|y_1|$ is the absolute value of the first ML score (ML scores are negative). This allows us to observe not only the convergence trend of the ML score, but it also shows us how the best result in the iterative procedure is often obtained. For example, on models with highest evolutionary rate scale = 64 (Figure B.5) an ML score at any iteration $i > 1$ will always be better than the ML score of the standard ML procedure ($i = 1$). For other models, this does not always hold. Figure B.2 contains many ML lines that are below zero, indicating that the iterative procedure, for the number of iterations we have performed, may end up with worse result at some iteration $i > 1$ then the score that it started with at the first iteration. This still may result in an improvement because the iterative procedure does not select the last ML score, but instead it chooses the best ML score that it encounters over all of t iterations.

The graphs show that ML score does not always converge. It is likelier that it will

converge on models with smaller indel probabilities than it is to converge on models with larger indel probabilities. For example, for models with evolutionary rate of 16, the ML score converges on all models with indel probabilities of 5×10^{-5} and 5×10^{-4} but not on models with indel probability of 5×10^{-3} (see Figure B.3). The figures below also show that for smaller indel probabilities, the ML score may oscillate between two or more ML scores. This may be due to the fact that `Muscle` uses only the topology of its guide tree (and ignores branch lengths). Then, the topology of two guide trees (in consecutive iterations) may be so similar that aligning profiles according to them, will not result in alignment difference that is big enough to affect derivation of a guide tree with different topology in the following iteration.

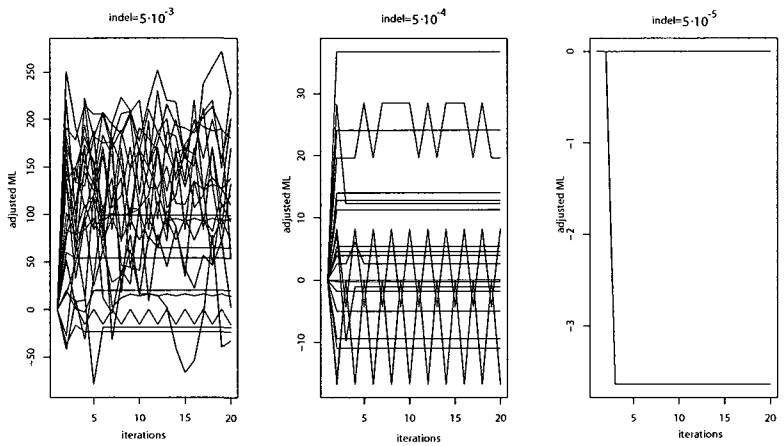

Figure B.1: ML convergence trend for models with evolutionary rate of 1.

The graphs in this section show that one cannot rely on the convergence of ML score in order to find the best alignment for a given replicate, since the ML score function does not always converge, and when it does, it does not necessarily converge to the best value.

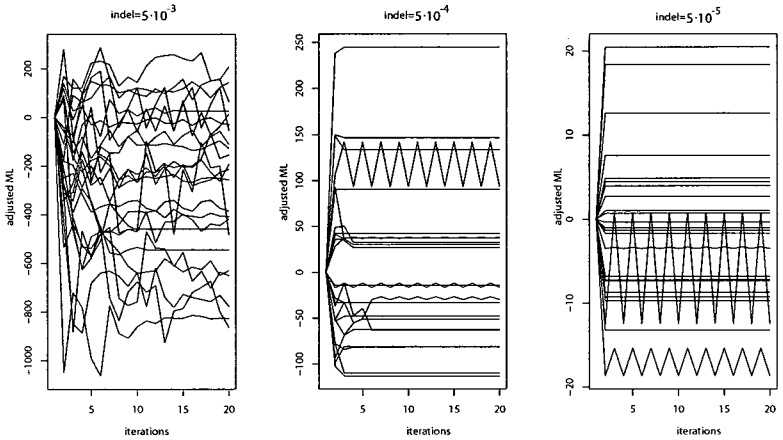

Figure B.2: ML convergence trend for models with evolutionary rate of 8.

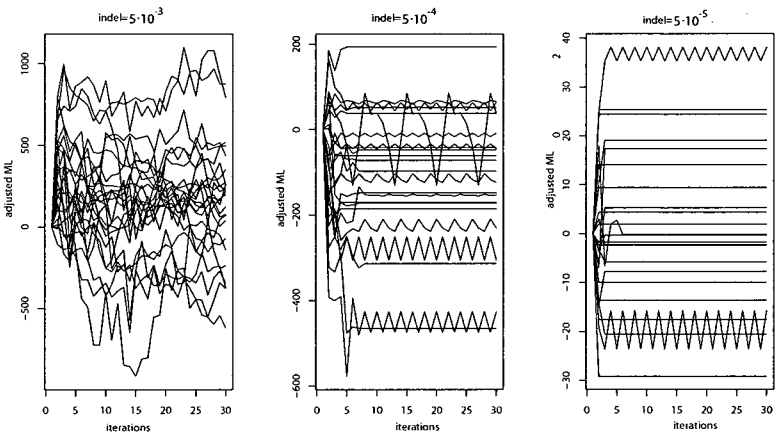

Figure B.3: ML convergence trend for models with evolutionary rate of 16.

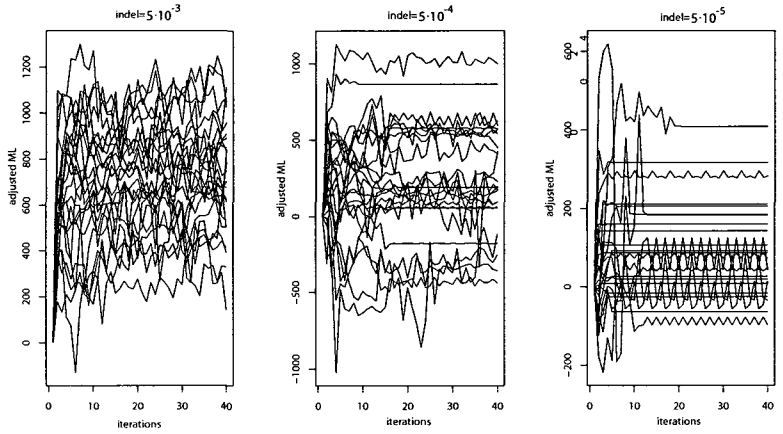

Figure B.4: ML convergence trend for models with evolutionary rate of 32.

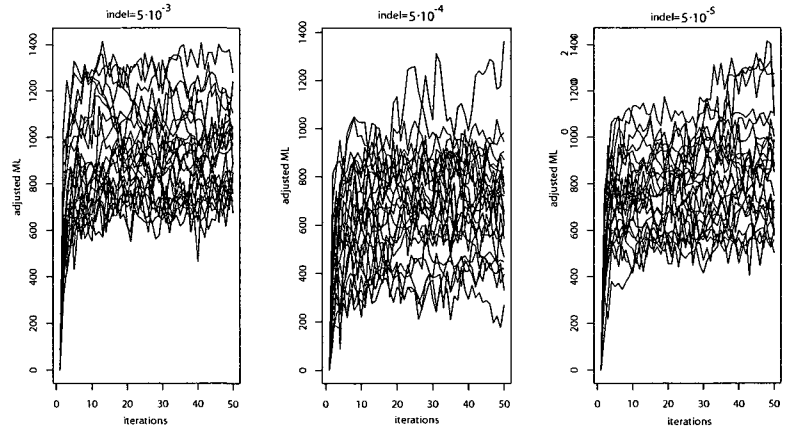

Figure B.5: ML convergence trend for models with evolutionary rate of 64.

Bibliography

[1] Y. Amelin, A. N. Krot, I. D. Hutcheon, and A. A. Ulyanov. Lead isotopic ages of chondrules and calcium-aluminum-rich inclusions. *Science*, 297(5587):1678–1683, Sep 2002.

[2] L. L. Cavalli-Sforza and A. W. Edwards. Phylogenetic analysis: Models and estimation procedures. *American Journal of Human Genetics*, 19(3 Pt 1):233–257, May 1967.

[3] P. Y. Chou and G. D. Fasman. Prediction of the secondary structure of proteins from their amino acid sequence. *Advances in enzymology and related areas of molecular biology*, 47:45–148, 1978.

[4] C. Darwin. *The origin of species by means of natural selection.* John Murray, 1859.

[5] Global Diversity. *Earth's living resources in the 21st century.* World Conservation Monitoring Centre, 2000.

[6] T. Dobzhansky. Nothing in biology makes sense except in the light of evolution. *The American Biology Teacher*, 35:125–129, 1973.

[7] R. C. Edgar. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797, 2004.

[8] J. A. Eisen and M. Wu. Phylogenetic analysis and gene functional predictions: phylogenomics in action. *Theoretical Population Biology*, 61(4):481–487, Jun 2002.

[9] J. Felsenstein. Distance methods for inferring phylogenies: a justification. *Evolution*, 38:16–24, 1984.

[10] Joseph Felsenstein. *Inferring Phylogenies*. Sinauer Associates, Inc., 2004.

[11] W. M. Fitch. Toward defining the course of evolution: Minimum change for a specified tree topology. *Systmatic Zoology*, 20:406–416, 1971.

[12] A. Gelman, J. B. Carlin, and H. S. Stern. *Bayesian Data Analysis*. Chapman & Hall/CRC, second edition, 2003.

[13] W. A. Goddard, E. Kubicka, G. Kubicki, and F. R. McMorris. The agreement metric for labelled binary trees. *Mathematical Biosciences*, 123:215–226, 1994.

[14] S. Guindon and O. Gascuel. A simple, fast, and accurate algoritm to estimate large phylogenies by maximum likelihood. *Systematic Biology*, 52(5):696–704, 2003.

[15] M. Hasegawa, H. Kishino, and T. Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial dna. *Journal of Molecular Evolution*, 22(2):160–174, 1985.

[16] J. Hein. A new method that simultaneously aligns and reconstructs ancestral sequences for any number of homologous sequences, when the phylogeny is given. *Molecular Biology and Evolution*, 6(6):649–668, Nov 1989.

[17] T. H. Jukes and C. R. Cantor. Evolution of protein molecules. *Academic Press, New York*, pages 21–132, 1969.

[18] M. Kimura. A simple method for estimating evolutionary rate of base substitution through comparative studies of nucleotide sequences. *Journal of Molecular Evolution*, pages 111–120, 1980.

[19] F. Lewitter. Recent evolutions of multiple sequence alignment algorithms. *Computational Biology*, 3(8):e123, 2007.

[20] G. Lunter, I. Miklos, A. Drummond, J. L. Jensen, and J. Hein. Bayesian coestimation of phylogeny and sequence alignment. *BMC Bioinformatics*, 6(83), 2005.

[21] C. D. Michener and R. R. Sokal. A quantitative approach to a problem of classification. *Evolution*, 11:490–499, 1957.

[22] A. Y. Mitrophanov and M. Borodovsky. Statistical significance in biological sequence analysis. *Briefings in Bioinformatics*, 7(1):2–24, Mar 2006.

[23] V. Morell. Treebase: the roots of phylogenety. *Science*, 273:569, 1996.

[24] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, Mar 1970.

[25] S. Nelesen, K. Liu, D. Zhao, R. C. Linder, and T. Warnow. The effect of the guide tree on multiple sequence alignments and subsequent phylogenetic analyses. *Pacific Symposium on Biocomputing*, 13:25–36, 2008.

[26] B. E. Pfeil, J. A. Schlueter, R. C. Shoemaker, and J. J. Doyle. Placing paleopolyploidy in relation to taxon divergence: a phylogenetic analysis in legumes using 39 gene families. *Systematic Biology*, 54(3):441–454, Jun 2005.

[27] H. Philippe and C. J. Douady. Horizontal gene transfer and phylogenetics. *Current Opinion in Microbiology*, 6(5):498–505, Oct 2003.

[28] D. Posada and T. R Buckley. Model selection and model averaging in phylogenetics: advantages of akaike information criterion and bayesian approaches over likelihood ratio tests. *Systematic Biology*, 53(5):793–808, Oct 2004.

[29] S. R. Ramrez, B. Gravendeel, R. B. Singer, C. R. Marshall, and N. E. Pierce. Dating the origin of the orchidaceae from a fossil orchid with its pollinator. *Nature*, 448(7157):1042–1045, Aug 2007.

[30] B. Rannala and Z. Yang. Phylogenetic inference using whole genomes. *Annual Review of Microbiology*, 9:217–231, Sep 2008.

[31] B. D. Redelings and M. A. Suchard. Joint bayesian estimation of alignment and phylogeny. *Systematic Biology*, 54(3):401–418, Jun 2005.

[32] D. F. Robinson and L. R. Foulds. Comparison of weighted labelled trees. *In Lecture notes in mathematics, Springer Verlag,Germany*, 1979:119–126.

[33] D. F. Robinson and L. R. Foulds. Comparison of phylognetic trees. *Mathematical Bionsciences*, 53:131–147, 1981.

[34] U. Roshan and D. R. Livesay adn S. Chikkagoudar. Improving progressive alignment for phylogeny reconstruction using parsimonious guide-trees. *Bioinformatics and BioEngineering, Sixth Symposium on*, pages 159–164, 2006.

[35] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, Jul 1987.

[36] M. J Sanderson. r8s: inferring absolute rates of molecular evolution and divergence times in the absence of a molecular clock. *Bioinformatics*, 19(2):301–302, Jan 2003.

[37] D. Sankoff. Minimal mutation trees of sequences. *SIAM Journal of Applied Mathematics*, 28:35–42, 1975.

[38] D. Sankoff and R. Cedergren. Simultaneous comparison of three or more sequences related by a tree. *Addison-Wesley, Reading, Mass, USA*, pages 253–264, 1983.

[39] H. N. Schulz and B. B. Jorgensen. Big bacteria. *Annual Review of Microbiology*, 55:105–137, 2001.

[40] S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3):591–611, 1965.

[41] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Computational Biology*, 147(1):195–197, Mar 1981.

[42] David Stirzaker. *Stochastic processes and Models.* Oxford University Press, 2005.

[43] J. Stoye, D. Evers, and F. Meyer. Rose: generating sequence families. *Bioinformatics*, 14(2):157–163, 1998.

[44] J. D. Thompson, D. G. Higgins, and T. J. Gibson. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673–4680, Nov 1994.

[45] G. Vogel. Hiv strain analysis debuts in murder trial. *Science*, 282:851–852, 1998.

[46] W. C. Warren, L. W. Hillier, J. A. M. Graves, E. Birney, C. P Ponting, F. Grtzner, K. Belov, W. Miller, L. Clarke, A. T. Chinwalla, S. Yang, A. Heger, D. P. Locke, P. Miethke, P. D. Waters, F. Veyrunes, L. Fulton, B. Fulton, T. Graves, J. Wallis, X. S. Puente, C. Lpez-Otn, G. R. Ordez, E. E. Eichler, L. Chen, Z. Cheng, J. E. Deakin, A. Alsop, K. Thompson, P. Kirby, A. T. Papenfuss, M. J. Wakefield, T. Olender, D. Lancet, G. A. Huttley, A. F. A. Smit, A. Pask, P. Temple-Smith, M. A. Batzer, J. A. Walker, M. K. Konkel, R. S. Harris, C. M. Whittington, E. S. W. Wong, N. J. Gemmell, E. Buschiazzo, I. M. V. Jentzsch, A. M., J. Schmitz, A. Zemann, G. Churakov, J. O. Kriegs, J. Brosius, E. P. Murchison, R. Sachidanandam, C. Smith, G. J Hannon, E. Tsend-Ayush, D. McMillan, R. Attenborough, W. Rens, M. Ferguson-Smith, C. M. Lefvre, J. A. Sharp, K. R. Nicholas, D. A. Ray, M. Kube, R. Reinhardt, T. H. Pringle, J. Taylor, R. C. Jones, B. Nixon, J. Dacheux, H. Niwa, Y. Sekita, X. Huang, A. Stark, P. Kheradpour, M. Kellis, P. Flicek, Y. Chen, C. Webber, R. Hardison, J. Nelson, K. Hallsworth-Pepin, K. Delehaunty, C. Markovic, P. Minx, Y. Feng, C. Kremitzki, M. Mitreva, J. Glasscock, T. Wylie, P. Wohldmann, P. Thiru, M. N. Nhan, C. S. Pohl, S. M. Smith, S. Hou, M. B. Renfree,

E. R. Mardis, and R. K. Wilson. Genome analysis of the platypus reveals unique signatures of evolution. *Nature*, 453(7192):175–183, May 2008.

[47] B. L. Welch. The generalization of student's problem when several different population variances are involved. *Biometrika*, 34:28–35, 1947.

[48] S. Whelan, P. I. W. de Bakker, and N. Goldman. Pandit: a database of protein and associated nucleotide domains with inferred trees. *Bioinformatics*, 19(12):1556–1563, Aug 2003.

[49] J. Winfried. Computational complexity of multiple sequence alignment with sp-score. *Journal of Computational Biology*, 8(6):615–623, 2001.

[50] Z. Yang. Estimating the pattern of nucleotide substitution. *Journal of Computational Biology*, 39(1):105–111, Jul 1994.

[51] Z. Yang. On the best evolutionary rate for phylogenetic analysis. *Systematic Biology*, 47(1):125–133, 1998.