A Birth-Death-Migration Model for Alignment Free Phylogeny Estimation using k-mer Frequencies

Siam Habib[†1], A.T.M Mizanur Rahman[†1], Md. Mohaiminul Islam[1], Khandaker Mushfiqur Rahman[1], Atif Rahman[*1]

[1]Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka

[†] These authors are co-first authors

***Corresponding author:** E-mail: atif@cse.buet.ac.bd

Associate Editor:

Abstract

Estimating phylogenetic trees from molecular data is an important challenge in bioinformatics and evolutionary biology. A widely used approach is to first perform a multiple sequence alignment and then to find a tree that maximizes likelihood computed under a model of nucleotide substitution. However, sequence alignment is computationally challenging for long sequences, especially in the presence of genomic rearrangements. To address this, methods of constructing phylogenetic trees without aligning the sequences i.e. alignment free methods have been proposed. They are generally fast and can be used to construct phylogenetic trees of a large number of species but they primarily estimate phylogenies by computing pairwise distances and are not based on any statistical model of molecular evolution. In this paper, we introduce a model for k-mer frequency change based on a birth-death-migration process which can be used to estimate maximum likelihood phylogenies from k-mer frequencies in an alignment free approach.

Key words: phylogeny estimation, alignment-free, k-mer frequency, likelihood, birth-death-migration process.

Article

Introduction

The phylogeny estimation problem is concerned with constructing an evolutionary tree of a set of species given their phylogenetic data and is considered one of the grand challenges in biology. The term phylogenetic data can refer to any data collected from the species in order to estimate their phylogeny. In the past, physiological and morphological traits were used to determine phylogenies but in modern times we primarily rely on genomic sequences. From a computational perspective, we can model the genomic sequences as strings. Thus from a computational point of view, the phylogeny problem takes as input a set of species and strings associated with each species, and returns a tree that "best" fits the data (genomic sequences). We decide which tree is the best tree by imposing an objective function on the tree space (the set of phylogenetic trees) and we wish to find the tree that maximizes our objective function.

There are two common approaches to phylogeny estimation based on how objective functions are constructed, distance based (Cavalli-Sforza and Edwards, 1967) (Fitch and Margoliash, 1967) (Kidd and Sgaramella-Zonta, 1971) and character based (Edwards, 1963) (Fitch, 1971). In distance based approaches, a distance metric is defined over the set of all available values for phylogenetic data. The goal is to place species with small distances between their phylogenetic data close together in the tree and species with large distances farther apart. This is done by computing a distance matrix containing the pairwise distance between the phylogenetic data for every pair of species and then constructing a weighted tree that minimizes the "deviation" between the distance matrix and the matrix containing the distances in the constructed tree between each pair of species.

In character based approaches, instead of constructing a distance matrix from the phylogenetic data, a "character matrix" is constructed. The individual characters are represented using the columns of the matrix and the rows represent the species. We assume that the characters change according to some rule. We try to place two species with similar characteristics close to one another in the phylogenetic tree. This is done by defining an objective function on the set of all phylogenetic trees given the character matrix and trying to find a tree that optimizes it. Two such candidates for objective functions are the parsimony function (Edwards and Cavalli, 1964) (Fitch, 1971) and the likelihood function(Fisher, 1922) (Fisher, 1921) (Edwards and Cavalli, 1964). Among all the methods of phylogenetic reconstruction, likelihood based ones are the most successful (Yang, 1994) (Felsenstein, 1981) (Felsenstein, 1978) (Gaut and Lewis, 1995) (Kuhner and Felsenstein, 1994) .

In likelihood based approaches, a statistical model is defined over how characters transition over time. Different characters are usually assumed to be independent of one another and how a character changes with time is modeled as a memoryless stochastic process i.e. a continuous time Markov process. Given the character matrix, a statistical model and a rooted weighted tree, one can thus find the likelihood of that tree (given the character matrix). The goal in this case is to find a tree with the maximum likelihood.

Successful statistical models used in likelihood based phylogeny estimations are based on statistical observations about molecular evolution, i.e. how genomic data (nucleotide or protein sequences) changes over time. Some of the notable ones among such models are Jukes-Cantor69 (Jukes et al., 1969), Kimura80 (Kimura et al., 1980), Kimura81 (Kimura, 1981), Felsenstein81 (Felsenstein, 1981), HKY85 (Hasegawa et al., 1985), and GTR (Tavaré, 1986). To use any such model in practice, first a multiple sequence alignment must be constructed from the genomic data and then the multiple sequence alignment

is used as the character matrix to the model. Computation of multiple sequence alignment is a hard problem in computer science both in theory (Wang and Jiang, 1994) and in practice (Kemena and Notredame, 2009; Thompson et al., 1999, 2011).

To overcome the computational cost of aligning the sequences, there has been a rising interest in alignment-free methods, where phylogenetic trees are constructed without explicitly aligning sequences. A multitude of alignment-free methods for phylogeny construction have been developed recently and there is a lot of variation and diversity in their approaches (Haubold, 2014; Zielezinski et al., 2019). Some notable alignment-free methods include kSNP (Gardner and Hall, 2013; Gardner and Slezak, 2010; Gardner et al., 2015), co-phylog (Yi and Jin, 2013), mash (Ondov et al., 2016), andi (Haubold et al., 2015), multi-SpaM (Dencker et al., 2018, 2020).

Alignment free methods also have additional benefits other than avoiding the computational cost of multiple sequence alignments. Sequence alignment generally assumes that changes to the genomic sequences only happen due to small local changes. But, in reality, alterations to genomic sequences also occur due to large scale genomic rearrangements such as reversals, translocations, duplications, deletions. Alignment based phylogeny estimation methods may lead to incorrect results in the presence of such rearrangements which can potentially be remedied by alignment free methods.

While alignment free methods seem very promising due to their scalability, performance, and ability to take into account non-local changes, most of them are distcance based and the best performing phylogeny construction tools are still likelihood based. Although, Höhl and Ragan (Höhl and Ragan, 2007), and Zahin et al. (Zahin et al., 2019) presented Bayesian and likelihood based alignment-free methods of phylogeny estimation using the absence or presence of k-mers, they rely on existing models for binary traits for likelihood computation, and do not explicitly model the process of changes to k-mer frequencies.

In this paper, we propose a method of constructing phylogenetic trees that tries to get the best of both worlds i.e. that is both alignment free and likelihood based. Unlike (Höhl and Ragan, 2007) and (Zahin et al., 2019), our model takes into account the frequencies of different k-mers adding further distinction between k-mers that are present. Our method counts the frequencies of different k-mers in the genomic data of each species and exploits the fact that if the k-word frequency profile of two species are similar they must be close to each other in the evolutionary tree. Furthermore, we introduce a linear birth-death model with constant positive migration to model how the k-mer frequency profile of a species can change with time. This

allows us to compare how likely a tree is, given the k-mer profiles of each species paving the way to combine the advantages of alignment free methods and the mathematical soundness of likelihood based methods.

We structure our paper into three parts. In part one of the paper we describe our model, the mathematical assumptions it makes, the motivation behind making such assumptions and also its limitations. In the second part, we show how we can compute likelihood of the phylogenetic trees using our model, and how we estimate parameters and the branch lengths. In the third section, we show our experimental results on simulated and real datasets.

Model

Birth-Death-Migration Process

A birth-death-migration (Bailey, 1991) process is a continuous time Markov process (Ross, 2014) where there are infinite states, each state is labeled with a non-negative integer. Every state i has a transition to state $i+1$ with transition rate λ_i that models population growth (a birth or a positive migration) and every state $i>0$ has a transition to state $i-1$ with transition rate μ_i that models a population decrease (death or negative migration). Figure 1 shows the Markov chain for a birth-death-migration process.

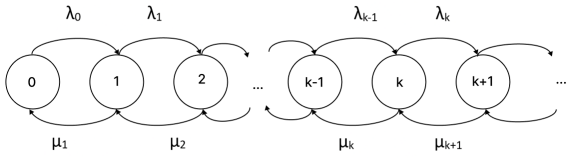

FIG. 1. A birth-death-migration process

Note that in a birth-death-migration process, a population growth event can happen in two ways, through a birth or through a positive or incoming migration. We will denote the birth rate at state i with b_i and the positive migration rate at state i with m_{i_+}. Then, we can write

$$\lambda_i = b_i + m_{i_+} \tag{1}$$

Similarly in a birth-death-migration process, a population decrease can happen in two ways, through a death or through a negative or outgoing migration event. We will denote the death rate at state i with d_i and the negative migration rate at state i with m_{i_-}. Then, we can write

$$\mu_i = d_i + m_{i_-} \tag{2}$$

When modeling populations in many scenarios, the rate of birth, the rate of death, and the rate of negative migration can be assumed to be linear in the size of the current population i.e.,

$$b_i = ib \tag{3}$$

$$d_i = id \tag{4}$$

$$m_{i_-} = im_- \tag{5}$$

where b, m and m_- are the rates of birth, death and negative migration when the population size is 1 respectively.

Modeling evolution of k-mer frequencies

Consider a set of k-mers and their occurrences in the genomic sequence of a species. We can consider the k-mer occurrences as distinct populations. Then, sizes of the populations represent the k-mer frequencies. The changes in k-mer frequencies

because of small scale substitutions, insertion-deletions, and genomic rearrangements parallel changes in the sizes of the populations due to births, deaths , and migration.

We can model how the population changes with time with a birth-death-migration process. Births model increase of a k-mer frequency due to a duplication event. Deaths model the decrease of a k-mer frequency because of a deletion event. Migrations are the increase or decrease of a k-mer due to substitutions of single or few nucleotides, small insertion-deletions, or at the breakpoints of genomic rearrangements.

For example, Figure 2 illustrates the process considering three populations, the occurrences of CGTG, CTTG, and CTTA. Initially, the sequence was ATCGTGCGTGTACTTG and CGTG, CTTG, and CTTA had populations of sizes 2, 1, and 0, respectively. First, a deletion event happens that deletes an occurrence of CGTG and reduces the population size to 1 (in the second frame from the top). This reduction in population can be modeled as a death event.

Then, a substitution of a single nucleotide happens that changes an occurrence of CTTG to CTTA. This can be thought of as a negative migration in the population of CTTG and a positive migration for CTTA (in Frame 3). Finally, in Frame 4, a duplication event happens for the occurrence of CTTA that increases the population size to 2. This is modeled as a birth event.

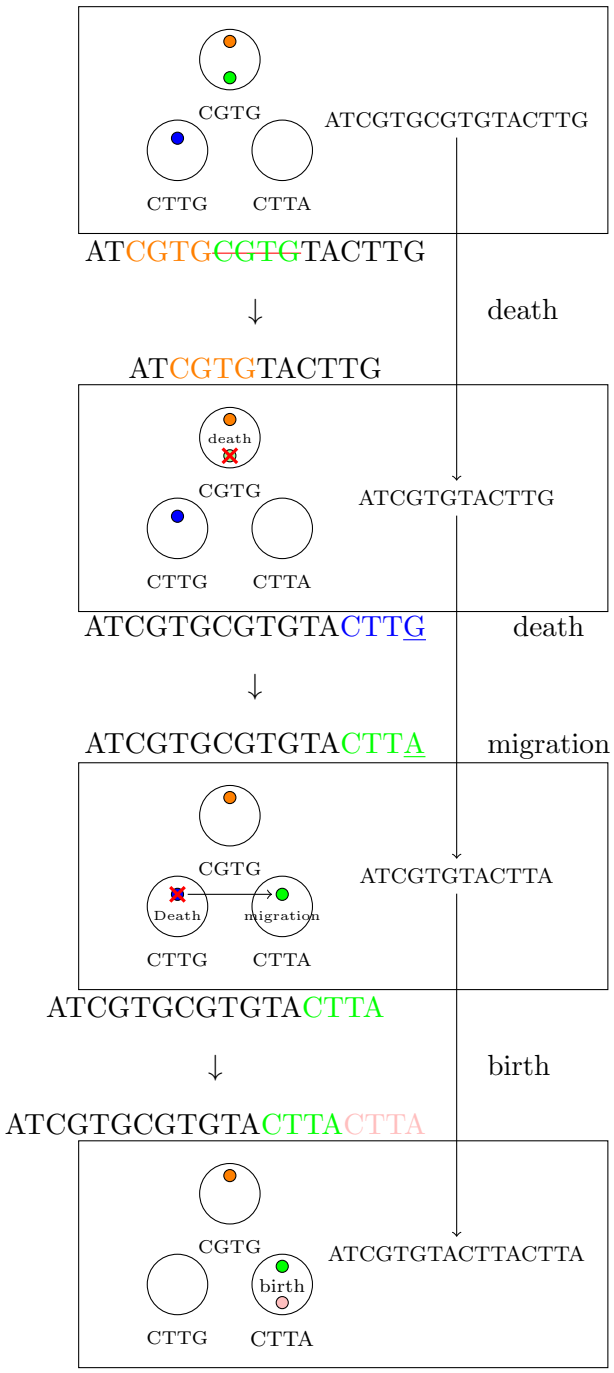

FIG. 2. How k-word frequency changes with time

From the above discussion we can see that we can model the following biological phenomena with birth-death and migration:

Gene Duplication In a gene duplication event (Frame 4 of Figure 2), occurrences of some

k-mer get duplicated thus increasing the population of the k-mers. We can model this as a birth. Assuming that gene-duplication is a memoryless stochastic process that is equally likely to happen at all sites in the genome, the rate of birth for any k-mer population at any given time is proportional to the size of the population i.e. the birth rate is linear.

Local Alterations We refer to substitutions of a single or few nucleotides, insertions-deletions, and changes at the breakpoints of genome rearrangements as local alterations (Frame 3). When a local alteration happens occurrences of a few k-mers change to others. This has two effects. It acts as negative migration for one population and positive migration for another. By the same argument for gene duplication, the rate of negative migration is linear. But positive migration is not. The rate at which positive migration to a k-mer l happens is proportional to the number occurrences of k-mers that are similar to l but not l. If we make the simplifying assumption that the rate of transition from any k-mer p to another k-mer q due to local alteration is roughly equal, we can model positive migration by a constant (during stationarity).

Gene Deletion In a gene deletion event, an occurrence of some k-mers get deleted (Frame 2). By similar arguments to the ones we presented above, this can be modeled as a linear death rate.

Thus we can model how the k-mer frequency of a single k-mer changes with time using a birth-death-migration process with linear rates of birth, death, and negative migration and a constant rate of positive migration i.e:

$$\lambda_i = ib + m_+ \tag{6}$$

$$\mu_i = i(d + m_-) \tag{7}$$

The model is simplified by denoting $\mu = d + m_-$ and by using the symbol m and λ in place of b and m_+, giving us the following equations:

$$\lambda_i = i\lambda + m \tag{8}$$

$$\mu_i = i\mu \tag{9}$$

We make further assumptions that the rates λ, μ and m are the same for all k-mers (for a fixed value of k) and independent of what the k-mer is i.e. mutations and rearrangements occur with the same rate at all sites in the genome. Furthermore, we assume that the k-mer frequency for different k-mers are independent and identically distributed .

Methodologies

In this section, we describe how we can use the model from the previous section to estimate a phylogeny of a set of species.

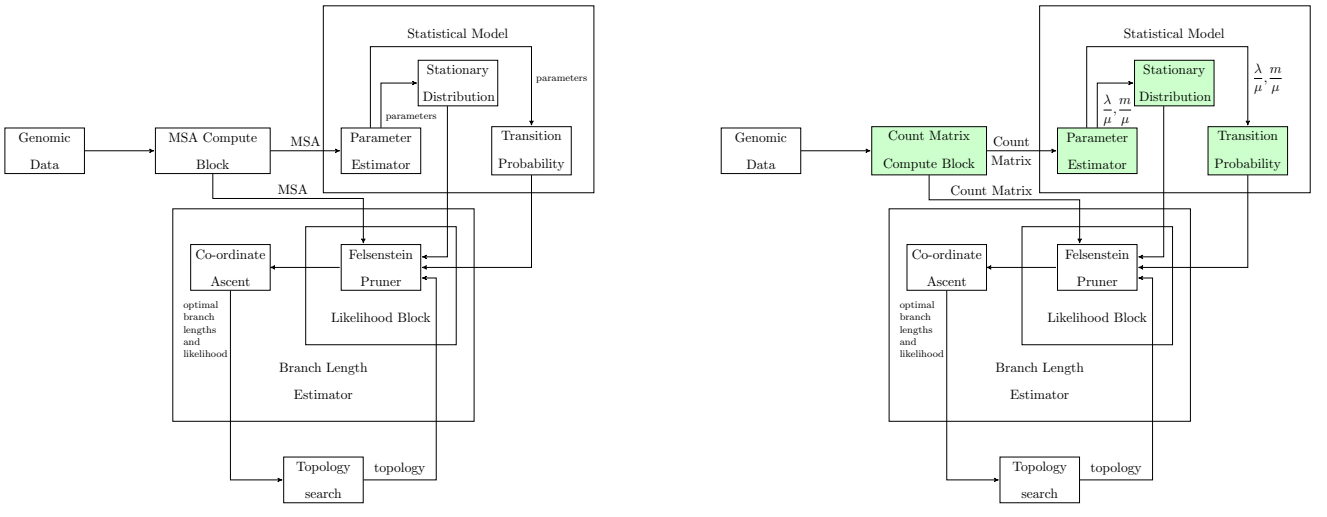

FIG. 3. Overviews of a) a conventional alignment-based and b) the proposed alignment-free likelihood based phylogeny estimation methods.

Overview of an alignment-free likelihood based phylogeny estimation method

The components in a conventional likelihood based phylogeny estimation schemes is shown in Figure 3a whereas Figure 3b illustrates the proposed alignment-free likelihood based method. The components and the modifications are described below:

1. **Character matrix computation:** First, the character matrix computation module takes as input the genomic data and computes a character matrix. For conventional likelihood based methods, this is done using multiple sequence alignment. In our method, it computes a count matrix where each column represents counts of a k-mer in each of the species.

2. **Parameter estimation:** The parameter estimator block takes as input the character matrix and the parametric statistical model the likelihood based phylogeny estimator is based on, and estimates the parameters of that model. In our case, it estimates the parameters for the linear birth-death process with constant positive migration.

3. **Likelihood computation:** A likelihood computation block takes as input the statistical model, the parameters of that model, the character matrix, and a weighted tree, and computes the likelihood of that tree given the parameters and the character matrix. This block uses Felsenstein's pruning algorithm (Felsenstein, 1973, 1981) to compute the likelihood in linear time (to the size of the tree). In the alignment-free setting, Felsenstein Pruner has to use the stationary distribution and the transition probabilities of our linear birth-death-migration model.

4. **Branch length estimation:** The branch length estimator block takes as input a

tree topology, the statistical model, and the parameters of that model as input and finds tree branch lengths that maximizes the likelihood for that tree. This module under the hood uses numerical optimizers (usually co-ordinate ascent or one of its variant) and repeatedly calls Felsenstein's pruning algorithm to estimate the branch lengths.

5. **Tree topology search:** The tree topology search unit searches over the tree topology space using some kind of local search algorithm. This module picks a tree topology and calls the branch length estimator to maximize the likelihood of that topology and then explores neighboring topologies. Our model can be incorporated in existing tree search methods to find the maximum likelihood tree.

We now describe how we compute the expressions in detail in the following sections.

Count Matrix Computation

The count matrix computation module takes as input the genomic sequences and computes a count matrix by counting the number of occurrences of different k-mers and their reverse compliments in the given genomic sequence. We have used Jellyfish (Marais and Kingsford, 2011) to implement our count matrix computation module.

Before generating the count matrix we must decide what is the value of k (the length of k-mers) for which we will be generating the count matrix and which k-mers to include in the count matrix. The reason why we should not include all k-mers of length k in the count matrix is two folds:

- Allowing all k-mers for a fixed size might make the matrix too large ($4^k/2$ k-mers of length k).
- We have made the assumption that the k-mer frequencies for any two different k-mers are independent and identically distributed. This will not be the case if we allow two k-mers that are close one another to be present in the matrix.

To resolve the issue of independence we uniformly sample num_k k-mers from all k-mers of length k (assuming that a k-mer and it's reverse compliment is equivalent) to decide which k-mers to include in our k-mer count matrix.

Thus our Count Matrix Computation module has two hyper-parameters:

- k, the length of the k-mers that we will count
- col, the number of k-mers to sample (which is also the number of columns in the k-mer count matrix)

We select the value of the hyper-parameters by picking pair (k,col) that produces the highest entropy (Shannon, 1948) (Sherwin, 2010) k-mer existence matrix, a 0-1 matrix where the entry of the ith row and jth column is 1 if and only if the jth k-word exists in the genome sequence of the

ith species. The entropy of the k-mer existence matrix is defined as the sum of the entropy of each columns of the matrix where the entropy of column j is defined as $-\frac{z}{z+o}\lg(\frac{z}{z+o})-\frac{o}{z+o}\lg(\frac{o}{z+o})$ where z and o denotes the number of zeros and ones present in the jth column of the k-mer existence matrix respectively.

Since the sum of entropies of each column is not normalized (increasing col will produce larger entropy), we normalize the entropy value by dividing the sum by the total number of columns present in the matrix.

We selected the hyper-parameters (k,col) by looping over an integer interval [LOW_K..HIGH_K] for k and setting col as min(max($\frac{4^k}{4k}$,MIN_COL),HIGH_COL) and finding the value of the pair (k,col) that generates the highest normalized entropy. The choice of setting col as $\frac{4^k}{4k}$ was motivated by the fact that there are $3k+1$ string that are atmost one edit distance apart from any k-mer and if we let col be greater than $\frac{4^k}{3k+1}$ we will be guaranteed to have a two closely related k-mers in our count matrix. The choice of dividing by $4k$ instead of $3k+1$ was just to round it up. The MIN_COL and MAX_COL was just there to make sure that col does not get too small or too large.

The choice for the numbers the numbers LOW_K, HIGH_K, MIN_COL, MAX_COL was 5,17,5000,50000 respectively.

Computing Stationary Distribution

We can find the expression for the stationary distribution of our linear birth-death-migration model by using the balance equations:

$$\begin{aligned} k\mu\pi_k &= ((k-1)\lambda+m)\pi_{k-1} \\ \Rightarrow \pi_k &= \left((k-1)\frac{\lambda}{\mu}+\frac{m}{\mu}\right)\frac{\pi_{k-1}}{k} \\ &= \prod_{i=0}^{k-1}\left(i\frac{\lambda}{\mu}+\frac{m}{\mu}\right)\frac{1}{i+1}\times\pi_0 \\ &= \frac{\prod_{i=0}^{k-1}\left(i\frac{\lambda}{\mu}+\frac{m}{\mu}\right)}{k!}\times\pi_0 \end{aligned} \tag{10}$$

By using $\sum_{k=0}^{\infty}\pi_k=1$ and the values from Equation 10, we get:

$$\begin{aligned} &\sum_{k=0}^{\infty}\pi_k=1 \\ &\Rightarrow\sum_{k=0}^{\infty}\frac{\prod_{i=0}^{k-1}\left(i\frac{\lambda}{\mu}+\frac{m}{\mu}\right)}{k!}\times\pi_0=1 \\ &\Rightarrow\left(1-\frac{\lambda}{\mu}\right)^{-\frac{m}{\lambda}}\pi_0=1 \text{ [Taylor Expansion]} \\ &\Rightarrow\pi_0=\left(1-\frac{\lambda}{\mu}\right)^{\frac{m}{\lambda}} \end{aligned} \tag{11}$$

By substituting the value of π_0 in Equation 10, we find the expression for π_k for all $k>0$.

$$\pi_k=\frac{\prod_{i=0}^{k-1}\left(i\frac{\lambda}{\mu}+\frac{m}{\mu}\right)}{k!}\left(1-\frac{\lambda}{\mu}\right)^{\frac{m}{\lambda}} \tag{12}$$

Thus we can find the value of π_k for any k in $\mathcal{O}(k)$ running time. Note that from Equation 12, we can see that for the stationary distribution to exist, $\lambda\leq\mu$. In practice, we are actually more interested in computing the log value of the stationary probabilities instead of the stationary probabilities themselves.

$$\ln(\pi_k) = \frac{m}{\lambda}\ln\left(1-\frac{\lambda}{\mu}\right) + \sum_{i=1}^{k}\ln\left(\frac{i-1}{i}\frac{\lambda}{\mu}+\frac{1}{i}\frac{m}{\mu}\right) \tag{13}$$

Parameter Estimation

First note that, the parameters of our model λ, μ, and m are dependent on the unit of time and thus any linear scaling of the terms will yield equal likelihood for the same tree (where the branch lengths are scaled down by the same factor). Thus instead of using λ, μ and m as parameters, we can use $\frac{\lambda}{\mu}$ and $\frac{m}{\mu}$. We will denote our parameters $\left(\frac{\lambda}{\mu}, \frac{m}{\mu}\right)$ using θ from now on.

Now assume that we are given a k-mer count matrix M_{cnt}. From the k-mer count matrix we can generate a count array C where C_i is defined as the number of occurrences of the integer i in the entire k-mer frequency matrix, M_{cnt}. Let N be the maximum entry in M_{cnt}. Then the length of the count array C is $N+1$ and contains the indices 0 to N. Under the assumptions that the entries of the k-word count matrix were sampled independently from the stationary distribution we can find the probability of the count array C given the parameters $\frac{\lambda}{\mu}$ and $\frac{m}{\mu}$ using the following equation:

$$P\left(C|\frac{\lambda}{\mu},\frac{m}{\mu}\right) = \left(\sum_{i=0}^{N}C_i\right)!\prod_{i=0}^{N}\left(\frac{1}{C_i!}\pi_i^{C_i}\right) \tag{14}$$

Using Equation 14, we can find the log likelihood of our parameters θ as follows:

$$\begin{aligned}\mathcal{L}(\theta|C) &= \ln\left(P\left(\frac{\lambda}{\mu},\frac{m}{\mu}|C\right)\right)\\ &= \ln\left(\left(\sum_{i=0}^{N}C_i\right)!\right) + \sum_{i=0}^{N}\ln\left(\frac{1}{C_i!}\pi_i^{C_i}\right)\\ &= K + \sum_{i=0}^{N}C_i\ln(\pi_i)\end{aligned} \tag{15}$$

where K is some constant dependent on the data (the count matrix which is constant).

Using Equation 15, we can find the expression for the gradient of the log-likihood:

$$\begin{aligned}\nabla\mathcal{L}(\theta|C) &= \nabla K + \sum_{i=0}^{N}C_i\nabla\ln(\pi_i)\\ &= \sum_{i=0}^{N}C_i\nabla\ln(\pi_i)\end{aligned} \tag{16}$$

We can also find the Hessian matrix of the log-likelihood function:

$$\nabla^2\mathcal{L}(\theta|C) = \sum_{i=0}^{N}C_i\nabla^2\ln(\pi_i) \tag{17}$$

Let us further simplify our notation to by using the symbols x and y respectively to denote the parameters $\frac{\lambda}{\mu}$, $\frac{m}{\mu}$. Then we can find the expressions for the gradient and the Hessian matrix for $\ln(\pi_k)$

for all values of k.

$$\begin{aligned}\nabla \ln(\pi_k) &= \begin{bmatrix} -\frac{y}{x^2}\ln(1-x)+\frac{y}{x^2-x} \\ \frac{\ln(1-x)}{x} \end{bmatrix} \\ &+\sum_{i=1}^{k} \begin{bmatrix} \frac{\frac{i-1}{i}}{\frac{i-1}{i}x+\frac{1}{i}y} \\ \frac{\frac{1}{i}}{\frac{i-1}{i}x+\frac{1}{i}y} \end{bmatrix} \\ &= \nabla \ln(\pi_{k-1}) + \begin{bmatrix} \frac{\frac{k-1}{k}}{\frac{k-1}{k}x+\frac{1}{k}y} \\ \frac{\frac{1}{k}}{\frac{k-1}{k}x+\frac{1}{k}y} \end{bmatrix} \end{aligned} \tag{18}$$

$$\begin{aligned}\nabla^2 \ln(\pi_k) &= \begin{bmatrix} a & b \\ c & 0 \end{bmatrix} \\ &-\sum_{i=1}^{k} \begin{bmatrix} \frac{\left(\frac{i-1}{i}\right)^2}{\left(\frac{i-1}{i}x+\frac{1}{i}y\right)^2} & \frac{\frac{i-1}{i}\frac{1}{i}}{\left(\frac{i-1}{i}x+\frac{1}{i}y\right)^2} \\ \frac{\frac{i-1}{i}\frac{1}{i}}{\left(\frac{i-1}{i}x+\frac{1}{i}y\right)^2} & \frac{\left(\frac{1}{i}\right)^2}{\left(\frac{i-1}{i}x+\frac{1}{i}y\right)^2} \end{bmatrix} \\ &= \nabla^2 \ln(\pi_{k-1}) - \begin{bmatrix} \frac{\left(\frac{k-1}{k}\right)^2}{\left(\frac{k-1}{k}x+\frac{1}{k}y\right)^2} & \frac{\frac{k-1}{k}\frac{1}{k}}{\left(\frac{k-1}{k}x+\frac{1}{k}y\right)^2} \\ \frac{\frac{k-1}{k}\frac{1}{k}}{\left(\frac{k-1}{k}x+\frac{1}{k}y\right)^2} & \frac{\left(\frac{1}{k}\right)^2}{\left(\frac{k-1}{k}x+\frac{1}{k}y\right)^2} \end{bmatrix} \end{aligned} \tag{19}$$

where

$$\begin{aligned} a &= \frac{2y}{x^3}\ln(1-x)+\frac{2y}{x^2(1-x)}-\frac{y}{x(1-x)^2} \\ b &= -\frac{1}{x^2}\ln(1-x)-\frac{1}{x(1-x)} \\ c &= -\frac{1}{x^2}\ln(1-x)-\frac{1}{x(1-x)} \end{aligned}$$

Notice that for the stationary distribution to exist, we must have that $0<x<1$ and $y\geq 0$. Thus, we have to find the most likely values for x and y under the constraints that $0<x<1$ and $y\geq 0$.

Note that the constrained region for our problem forms a rectangular region which is unbounded on one side. Finding the maximum value of a function in such a rectangle can be solved using trust-region method (Conn et al., 2000). We used scipy's trust-constr (scipy.org, 2023a) (scipy.org, 2023b) method to maximize the log likelihood of the parameters given the count array. Scipy's trust-constr implementation is based on(Byrd et al., 1999) and (Lalee et al., 1998) who themselves implement (Byrd, 1987), (Omojokun, 1989) and (Byrd et al., 1999).

Branch Length Estimation

Given a fixed tree topology and the model parameters, finding the optimal branch lengths that maximizes the tree topology exactly is considered hard for most phylogenetic models. Usually, branch length is estimated using co-ordinate ascent (Wright, 2015) based numerical methods.

Co-ordinate ascent is a numerical algorithm that maximizes an objective function $f:\mathbb{R}^n\to\mathbb{R}$

in an iterative manner. In each iteration, the algorithm picks a co-ordinate and optimizes that co-ordinate assuming all other co-ordinate values are fixed (Wright, 2015).

In the context of our problem, we want to maximize the likelihood of the tree given the topology T and parameters θ. Thus we may consider our likelihood function $\mathcal{L}_{T,\theta}:\mathbb{R}^{|E|}\rightarrow\mathbb{R}$ to take a $|E|$ dimensional vector as input, where $|E|$ is the number of branches in the tree. We will denote the vector of all branch lengths as $\mathbf{x}$. We initialize $\mathbf{x}$ with random non-negative integers. In each iteration, we loop over all the edges and update their branch lengths with the most likely length of that edge while keeping the rest of the branch lengths fixed.

Solving the problem of finding the most likely length of a branch while keeping rest of the branch lengths fixed is usually solved using numerical methods such as gradient ascent (Cauchy et al., 1847) (Polyak, 1987), Newton-Raphson method, Brent's method (Brent, 1973) (Dekker, 1969), trisection search, golden section search (Kiefer, 1953). We used scipy's `minimize_scalar` function with method set to `bounded` (scipy.org, 2023c) (scipy.org, 2023c).

Likelihood Computation

To compute the likelihood, we used Felsenstein's pruning algorithm. It queries the statistical model for the transition probabilities over the branch lengths. Let S be a continuous time Markov chain with a finite number of states. If n is the number of states in S then the rate matrix Q is an $n\times n$ matrix. We can compute the transition matrix function $P(t)$ of S using the following equation:

$$P(t)=e^{Qt} \tag{20}$$

However, in the case of our birth-death-migration model this can not be used because the birth-death-migration process has infinite states. Note that equation 20 is true regardless of whether the CTMC is finite or not. It is just no longer possible to compute the matrix efficiently. Therefore, we approximated the transition probability function for our model by generating an approximate rate matrix Q' by bounding the number of states to some large value. This is done by computing the stationary distribution the k-mer counts using our estimated parameters and finding the largest value i that has a higher stationary probability than some small cut-off probability (10^{-8}).

Result

In this section we present the results of our experiments. First, we evaluate the efficacy of our implementation of the parameter estimation block. Then, we assess the effectiveness of our model in finding the actual branch lengths and tree topologies. Finally, we benchmark our model using three datasets from the AF project (Zielezinski et al., 2019). The AF project has 3 datasets of fully assembled genetic data for genome based phylogeny, the E.coli dataset(Yi and Jin, 2013), the plant dataset (Hatje and

Kollmar, 2012) and the fish mtDNA dataset (Fischer et al., 2013). We used our model on the afproject datasets to compute the likelihood of different tree topologies in the afproject ranklist to show that our model does indeed assign higher likelihood to trees closer to the "original" tree topology when compared with the other topologies.

Parameter Estimation

We computed the parameters of our linear-birth-death-migration model using trust-region based numerical algorithms. In this section, we benchmark its accuracy using simulated data since it is not possible to know what the actual parameters are for any real dataset with certainty.

In our experiment, we generated synthetic datasets according to our model. We generated 60 different datasets in batches of 10. The datasets were generated using by generating random rooted trees with random branch lengths and then simulating the evolution of k-mer frequencies in accordance to a linear-birth-death migration models with randomly generated parameters.

Since, the parameters λ,m,μ do not uniquely identify the k-matrices, we benchmarked our algorithm by computing the values $\frac{\lambda}{\mu}$ and $\frac{m}{\mu}$ and comparing the estimated values against the real values. Figure 4 and 5 show the plot of the estimated vs true values of the parameters $\frac{\lambda}{\mu}$ and $\frac{m}{\mu}$ respectively. From the figure we can see that the value estimated values and the real values the straight line with slope 1 that goes throw

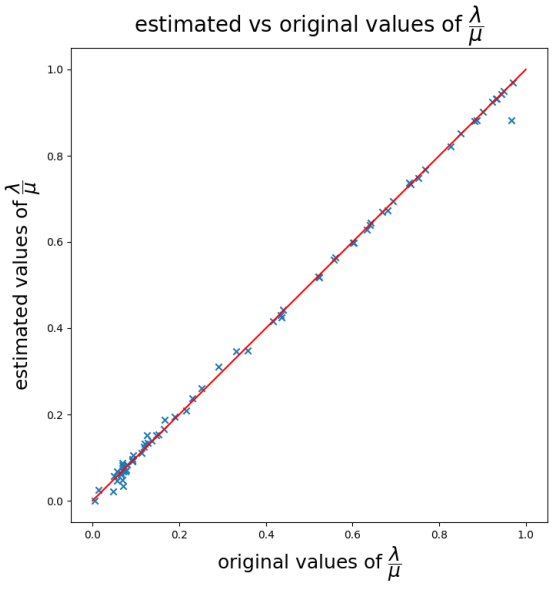

FIG. 4. parameter estimation for $\frac{\lambda}{\mu}$

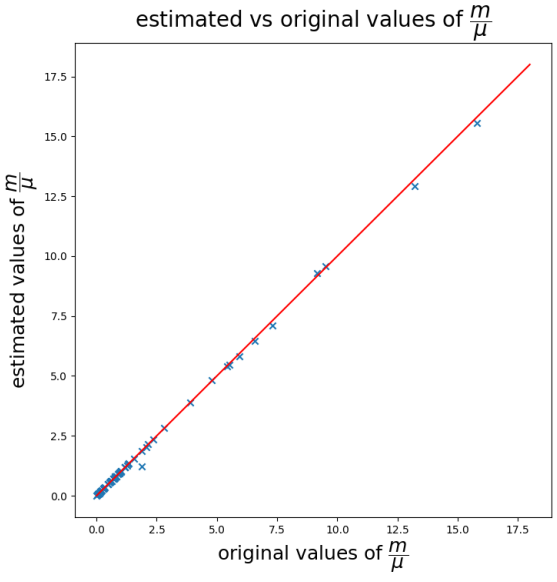

FIG. 5. parameter estimation for $\frac{m}{\mu}$

the center. The correlation coefficient between estimated and original parameters are 0.9991 for $\frac{\lambda}{\mu}$ and 0.9998 for $\frac{m}{\mu}$. This suggests that the estimated values are really close to the real value and our parameter estimator module can estimate the parameters with high enough precission.

Branch Length Estimation

We benchmarked our implementation of branch length estimation using two different batches of datasets, one synthetic and one real.

We generated our synthetic dataset in a similar manner as we discussed in the previous subsection. We used generated a batch of 10 trees with their birth-death-migration parameters normalized ($0<\lambda<1$ and $\mu=1$). We sampled the value of m uniformly from the interval (0,1). The branch lengths of the trees are between 0.1 and 0.9 units.

For the real dataset we used the seven primate dataset. We generated 7 different trees from the seven primate dataset. We generated the tree topology and branch lengths using RAxML-ng and the GTR+G model. We generated the k-word count matrices using jellyfish for all k in the integer interval (7,13).

For both datasets we benchmarked our implementation of the branch length estimation module by providing it the original tree topology without the branch lengths. We hide the original branch lengths by initializing the lengths of all the branches with a uniformly random number from the interval (1,5).

For the real dataset, we assume that the tree returned by RAxML using the GTR+G model is the original tree. For the simulated datasets we provide the module with the original parameters for the birth-death-migration process. For the real dataset, we use our parameter estimation module to estimate them before we pass it to the branch length estimation module.

In our model, the branch lengths does not uniquely identify the correct tree even when the topology is fixed since given any tree T, we can find a new tree T' with likelihood equal to that of T by scaling the branch lengths of T by a constant k and scaling the birth-death-migration parameters by $\frac{1}{k}$ for any $k>0$. Thus to analyze the results of our experiment we introduced the term BLDK(T,T',K) as a measure of distance between two trees.

$$\text{BLDK}(T,T',K)=\frac{\sum_{e\in E(T)}(l_T(e)-Kl_{T'}(e))^2}{|E(T)|} \quad (21)$$

Since for the simulated dataset, we have already provided the branch length estimation module with the original birth-death-migration process parameters, we will use BLDK with $K=1$ to analyze our results. For the real datasets, since we do not know the original birth-death-migration parameters, we analyze the results by setting the value of K in such a manner that it minimizes the value of BLDK for all K. We can determine the value of K that minimizes BLDK$(T_{real},T_{estimated},K)$ by taking its derivative and setting it to 0:

$$\begin{aligned}
&\frac{d}{dK}\text{BLDK}(T_{real}, T_{estimated}, K) = 0 \\
&\Rightarrow \sum (b_i^* - K b_i) \cdot b_i = 0 \\
&\Rightarrow \sum (b_i^* b_i) = K \cdot \sum b_i^2 \\
&\Rightarrow K = \frac{\sum (b_i^* b_i)}{\sum b_i^2}
\end{aligned} \tag{22}$$

Furthermore, for the real dataset, we find the value of k (k-mer length) that maximizes entropy and then generate diagrams for the original tree and the estimated tree and compare them. Since the branch lengths of both trees are not on the same scale (the original tree was generated using RAxML-ng and the estimated one using our model), we re-scale the branch lengths by dividing each tree with the sum of their branch lengths (thus the sum of the branch lengths for both trees are now 1).

Results of the Benchmark

Figure 6 shows the results of our benchmark with both simulated data (on the left) and real data(on the right). The figure shows the comparison between the BLDK of the tree estimated by our branch length computation module from the true topology (in red) against the BLDK of the tree with randomly initialized branch lengths (in blue). For simulated data, we set the value of K to one since we know that parameters were known and already normalized. For the real dataset we used the optimal value of K for each dataset that we obtained from equation 22.

We can see that for all datasets (both real and simulated), our estimated tree that we got from the branch length estimation module outperforms the trees with randomly initialized branch lengths.

Additional for the real trees, the trend can be observed that as the value of k gets larger, the value of the BLDK of the estimated tree becomes closer to the BLDK of the one initialized with random branch lengths. This is due to the fact that if the dataset is too small, for large values of k the k-word count matrix has a large number of 0s and loses most of it's information. This why we must use the value of k that maximizes the entropy of the k-word matrix in practice.

Comparing Diagrams of the Estimated and the Original Tree

In Figure 7 we compare the original tree topology (we got from RAxML) and our estimated tree that we estimated given the original topology using our model.

We can see that both trees look visually similar after re-scaling. There is an artifact close to the root, but its not an issue of branch length estimation, but rather one of rooting.

Another change that came from our tree was that the branch between baboon and it's ancestor shrunk while maintaining the relation that baboon is further from it's ancestor than gibbon.

Since the trees are very similar and agree upon which child is closer to the ancestor for all species (from visual inspection), this provides us evidence that our model is sound and the branch length estimation module is functioning.

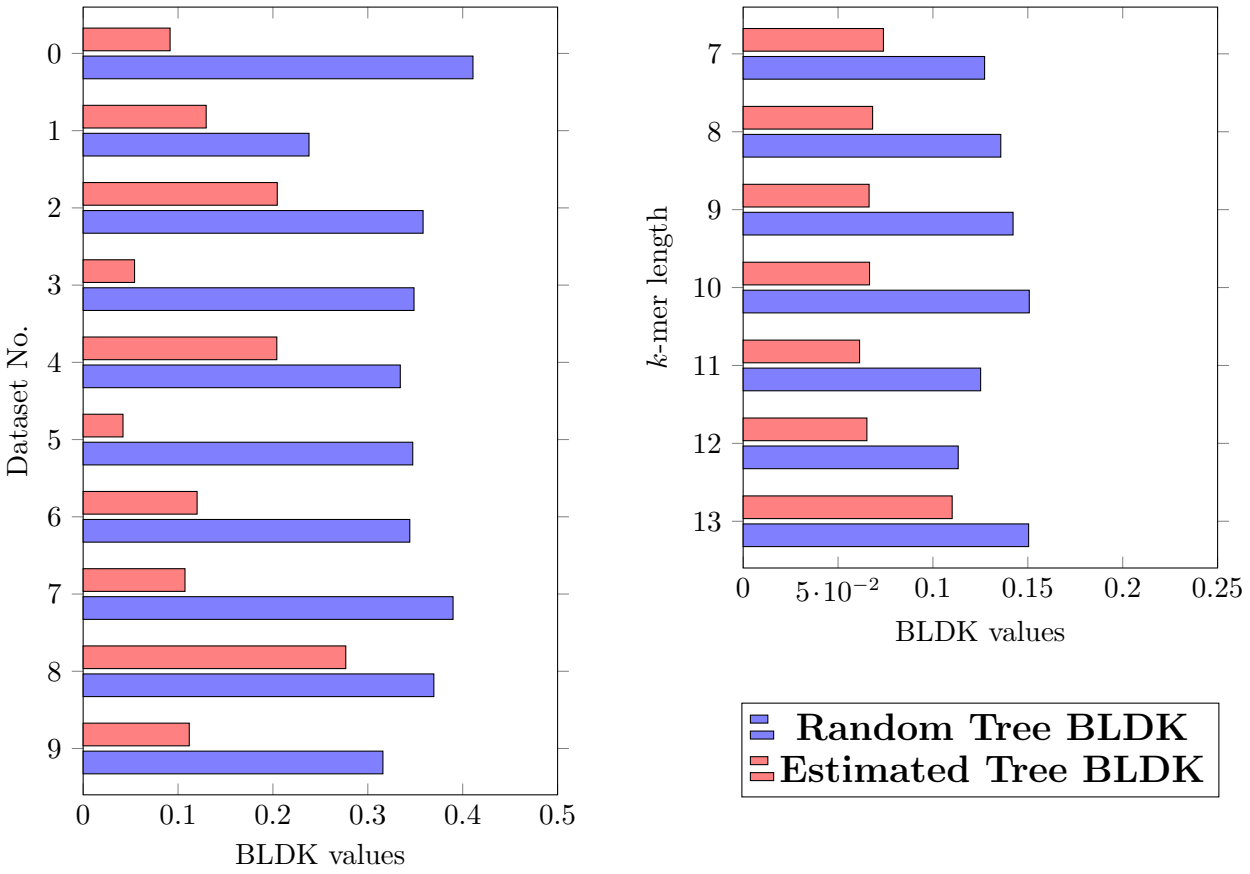

FIG. 6. Comparison of Random Tree BLDK and Estimated Tree BLDK for simulated data (on the left) and real data (on the right)

Tree Topology Search

To show that our model is indeed able to find a good candidate phylogenetic tree, we must show that the likelihood curve of a tree (given parameters and branch lengths) has desirable properties in the tree topology space. More precisely, we want the likelihood of the branch length optimized real tree to be higher than the likelihood of most other branch length optimized trees. Moreover we want trees with high likelihood to be topologically close to the real tree.

Since, the tree topology space is discrete, the problem of finding an optimal tree topology is combinatorial in nature. The lack of knowledge of what the actual tree topology is, the large number of tree topologies and the combinatorial nature of the problem makes it difficult to prove that any model has the said desirable properties in the tree topology space.

To validate our models efficacy in finding the actual tree topology we performed the two separate experiments:

SPR Experiments In the SPR Experiment we given an original tree topology, we generated a bunch of different tree topologies by performing one or more spr (subtree prunning and grafting) operations. We then compared the likelihood of those generated tree topologies with the original tree topology.

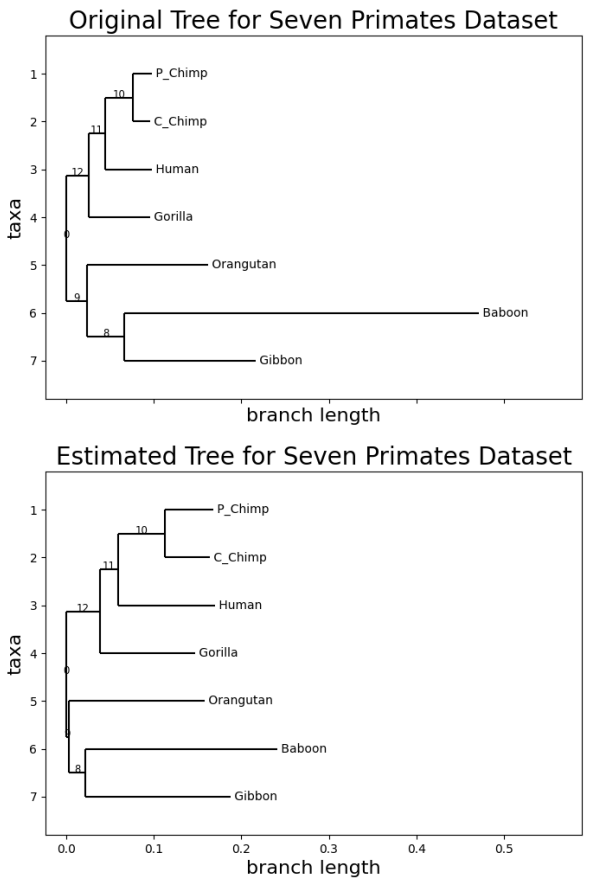

FIG. 7. Comparison Between the Original and Estimated Tree for the Seven Primate Dataset

For this we used two datasets. One synthetic and one real. The synthetic dataset was generated in a similar manner as mentioned in the previous sections. For the real dataset, we used the seven primate dataset. We assumed that the tree topology generated by raxml for the seven primate dataset is the original tree topology.

Likelihood Curve Experiment We generated a bunch of random tree topologies from the original tree generated by RAxML for the Seven Primates dataset and then computed their likelihood using our model. We then sorted the trees based on their RF distance from the original tree and generated a likelihood vs RF distance curve to investigate if our model possess desirable properties in the tree topology space.

Results of the SPR experiments

Table 1 and 2 show the results of our experiments for the SPR experiments for the synthetic and real dataset respectively. The rows of the table corresponds to different original topology. The changed topology column shows the maximum likelihood we got from all altered tree topologies for each tree.

Table 1. Comparison of Likelihood of Original Tree Topology and Altered Tree Topology for the Synthetic Dataset

	Original Topology Likelihood	Altered Topology (Best Likelihood)
Simulated tree-1	−133524.145	−133992.8641
Simulated tree-2	−120348.0918	−121041.3953
Simulated tree-3	−201296.3069	−202448.2381
Simulated tree-4	−139176.5926	−139951.6244

From the table we can see that the original tree topology outperforms the altered tree topology in all cases.

Table 2. Comparison of Likelihood of Original Tree Topology and Altered Tree Topology for the Synthetic Dataset

	Original Topology Likelihood	Altered Topology (Best Likelihood)
7-mer	−6020.860941	−6501.861718
8-mer	−3206.159255	−3369.410822
9-mer	−1290.828778	−1346.723154
10-mer	−460.5812113	−466.9038815
11-mer	−144.4945924	−147.5824894

Likelihood vs RF Distance

We generated some random trees with different RF distances from the original tree by repeatedly doing random SPR operations. And generated a likelihood curve.

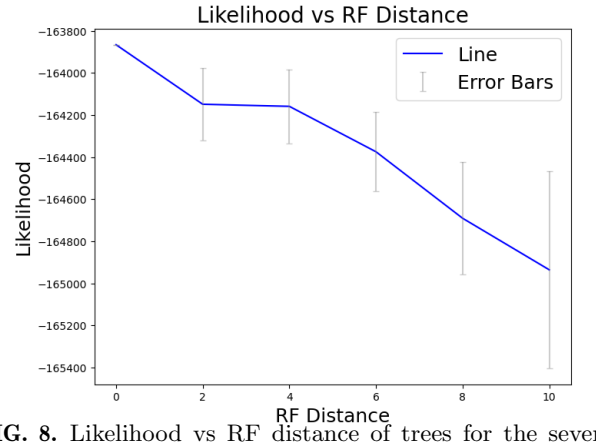

FIG. 8. Likelihood vs RF distance of trees for the seven primate dataset

From the figure we can see that as the RF distance from the original topology increases, the average likelihood decreases giving us definitive proof that our model does assign higher likelihood to trees topologically closer to the original tree than to ones that are further from it.

Benchmarking Our Model Using the AF Project Datasets

The AF project has 3 datasets of assembled genome sequence of the full genome for a set of species intended for phylogenetic tree construction. They are:

- The Ecoli Dataset
- The Plant Dataset
- The Fish mtDNA Dataset

The AF project also provides a "true tree" or a reference tree for each of the datasets. Different models can submit their generated trees to the afproject to be compared against the "true tree" and trees generated by other models.

In this experiment we do not search over all of the tree topology space using our model but instead compare the likelihood of different tree typologies already present in the AF project ranklists using our model. The goal of this experiment is to validate our hypothesis of whether or not our model does indeed assign higher likelihood to trees closer to the "original" tree topology when compared with the other topologies.

We selected the topologies in the following manner:

- We made sure that for each dataset, the topologies generated by mash, co-phylog, skmer, kSNP3, andi, phylonium are selected.
- For each dataset, we made sure that at least one topology was present from rank 1, rank 2 and rank 3 in the afproject ranklists.
- For each dataset, we kept the "original" or reference tree.
- For each dataset, additionally we generated a random topology with appropriate number of leaves and kept it for comparison as well.

Due to the limitations of our implementation, our code is only able to work with the newick format for rooted complete binary trees (each non-root internal node must have 3 children). Models that provided their topologies in tsv or phylip format were converted to newick format using Neighbor Joining Algorithm. If tree was not rooted, a random root was selected for rooting. If any of the tree had a node with more that two children, the node was split to multiple nodes arbitrarily.

Results of AF Project Benchmarks

Table 3. The Likelihood of Branch Length Optimized Tree Topologies from the E-coli Dataset

Model	Likelihood	RF	Rank
kSNP3	-586788.214	14	4
skmer, mash	-587693.661	16	4
andi	-589409.349	10	2
co-phylog	-589530.017	14	3
original topology	-590045.284	0	0
8/10_DNA_TWS	-590122.076	6	1
phylonium	-590122.075	10	1
FSWM/Read-SpaM	-590448.781	14	3
Random Topology	-808184.589	54	—

Table 3, 4 and 5 show the results of our experiments for the Ecoli dataset, the Plant dataset and the Fish mtDNA dataset respectively. Each table shows likelihood of each model in the descending order of their likelihoods along with their RF distance from the reference tree (not normalized) and their rank in the AF Project ranklist. Note that the since we had to split internal nodes with more than three children into multiple children, the RF distances between trees may now differ from the the value listed on AF Project.

We selected the value of k and the number of samples to compute the likelihood of our tree using the heuristic we mentioned in our methodologies. For value of k was 12, 14 and 8 and the number of samples were 50000, 50000 and 5000 for the Ecoli, the Plant and the Fish mtDNA dataset respectively.

Table 4. The Likelihood of Branch Length Optimized Tree Topologies from the Plant Dataset

Model	Likelihood	RF	Rank
skmer	-1082511.709	8	2
Original Topology	-1082712.868	0	0
mash	-1082715.367	6	1
co-phylog	-1083061.626	10	1
ASW(1,11,COS)	-1083217.839	10	3
kSNP3	-1083939.777	16	5
Multi-SpaM	-1084033.582	2	1
phylonium	-1088458.069	16	8
random topology	-1121801.549	24	—
andi	-1130283.135	20	10

We can see that in none of the tables, the original topology is assigned the highest likelihood. But in case of all three datasets, the random topology performs significantly worse the original topology and topologies with good ranks on the afproject ranklist which acts as an evidence supporting our hypothesis.

In case of the Ecoli dataset, the original topology was ranked 4th but the topologies that were assigned a higher likelihood by our model all came from good and reputable models. So we can claim that our model provides higher likelihood for good quality topologies over bad ones.

In case of the Plant dataset, the "original" tree got the second highest likelihood and the only topology that got higher likelihood was generated by skmer which is considered as a good model which again acts as evidence to our claim that our model provides higher likelihood for good

quality topologies over bad ones. Furthermore, if we observe the ranklist imposed by our model in Table 4, we can see that our ranklist obeys the expected ranklist with only two artifacts caused by skmer and Multi-SpaM. This again can be interpreted as evidence for our hypothesis.

In case of the Fish dataset, the "original" tree got quite a poor rank based on our model. But our ranklist obeys the afproject ranklist completely. Moreover an overwhelming number of models came to the conclusion that the topology generated by ASW1,9,EUC is the actual model (which can be seen do the sheer number of models in the afproject ranklist that agree with this topology). This suggests that topology preferred by our model may actually indeed be the "true" tree for the Fish dataset. The fact that the ranklist imposed our model is completely consistent with the one from afproject acts as a strong evidence for our hypothesis.

Table 5. The Likelihood of Branch Length Optimized Tree Topologies from the Fish mtDNA Dataset

Model	Likelihood	RF	Rank
ASW(1,9,EUC)	-96915.235	4	1
(4)SR(K)MER_FEM1	-96915.235	6	1
8/10_DNA_TWS	-96915.235	4	1
mash	-96915.236	4	1
AFKS–kl_conditional	-96915.535	10	2
AFKS–canberra	-96916.465	14	2
ksnp3	-96917.306	20	2
co-phylog	-96917.306	12	2
skmer	-96917.306	12	2
AFKS–k_divergence	-96917.999	12	2
phylonium	-96919.240	12	3
original topology	-96923.982	0	0
andi	-96938.589	16	4
ASW(1,3,EUC)	-98951.448	44	17
random topology	-101048.468	44	—

Concluding Remarks

From the results we obtained from all the experiments, we can come to the conclusion that our model does impose a higher likelihood to tree topologies that are "good" or closer to the original tree than tree topologies that are further away and thus can act as a good model for phylogeny estimation. In case of our afproject benchmark, even though model did not pick the "True" tree (or the reference) tree for the as the best topology for any of the datasets in the afproject, it did consistently place a higher likelihood on high quality topologies over ones that are further away from the reference tree.

In the future, once a more performant implementation of the model is available, we may do a more thorough benchmark with complete topology search.

References

Bailey, N. T. 1991. The elements of stochastic processes with applications to the natural sciences, volume 25. John Wiley & Sons.

Brent, R. P. 1973. Algorithms for minimization without derivatives. Courier Corporation.

Byrd, R. 1987. Robust trust region methods for constrained optimization. In Third SIAM Conference on Optimization, Houston, Texas.

Byrd, R. H., Hribar, M. E., and Nocedal, J. 1999. An interior point algorithm for large-scale nonlinear programming. SIAM Journal on Optimization, 9(4): 877–900.

Cauchy, A. et al. 1847. Méthode générale pour la résolution des systemes déquations simultanées. Comp. Rend. Sci. Paris, 25(1847): 536–538.

Cavalli-Sforza, L. L. and Edwards, A. W. 1967. Phylogenetic analysis. models and estimation procedures. American journal of human genetics, 19(3 Pt 1): 233.

Conn, A. R., Gould, N. I., and Toint, P. L. 2000. Trust region methods. SIAM.

Dekker, T. J. 1969. Finding a zero by means of successive linear interpolation. Constructive aspects of the fundamental theorem of algebra, 1.

Dencker, T., Leimeister, C.-A., Gerth, M., Bleidorn, C., Snir, S., and Morgenstern, B. 2018. Multi-spam: a maximum-likelihood approach to phylogeny reconstruction based on multiple spaced-word matches. arXiv preprint arXiv:1803.09222.

Dencker, T., Leimeister, C.-A., Gerth, M., Bleidorn, C., Snir, S., and Morgenstern, B. 2020. multi-spam: a maximum-likelihood approach to phylogeny reconstruction using multiple spaced-word matches and quartet trees. NAR Genomics and Bioinformatics, 2(1): lqz013.

Edwards, A. 1963. Cavalli-sforza. 1963. the reconstruction of evolution. Annals of Human Genetics, 27: 105–196.

Edwards, A. and Cavalli, L. 1964. Sforza, reconstruction of evolutionary trees. Systematic association Publication, 6.

Felsenstein, J. 1973. Maximum likelihood and minimum-steps methods for estimating evolutionary trees from data on discrete characters. Systematic Biology, 22(3): 240–249.

Felsenstein, J. 1978. Cases in which parsimony or compatibility methods will be positively misleading. Systematic zoology, 27(4): 401–410.

Felsenstein, J. 1981. Evolutionary trees from dna sequences: a maximum likelihood approach. Journal of molecular evolution, 17: 368–376.

Fischer, C., Koblmüller, S., Gülly, C., Schlötterer, C., Sturmbauer, C., and Thallinger, G. G. 2013. Complete mitochondrial dna sequences of the threadfin cichlid (petrochromis trewavasae) and the blunthead cichlid (tropheus moorii) and patterns of mitochondrial genome evolution in cichlid fishes. PLoS One, 8(6): e67048.

Fisher, R. 1921. On the'probable error'of a coefficient of correlation deduced from a small sample." metron 1, 4: 3-32.-(1915). Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. Biometrika, 10: 507–521.

Fisher, R. A. 1922. On the mathematical foundations of theoretical statistics. Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character, 222(594-604): 309–368.

Fitch, W. M. 1971. Toward defining the course of evolution: Minimum change for a specific tree topology. Systematic Zoology, 20(4): 406–416.

Fitch, W. M. and Margoliash, E. 1967. Construction of phylogenetic trees: a method based on mutation distances as estimated from cytochrome c sequences is of general applicability. Science, 155(3760): 279–284.

Gardner, S. N. and Hall, B. G. 2013. When whole-genome alignments just won't work: ksnp v2 software for alignment-free snp discovery and phylogenetics of hundreds of microbial genomes. PloS one, 8(12): e81760.

Gardner, S. N. and Slezak, T. 2010. Scalable snp analyses of 100+ bacterial or viral genomes. J Forensic Res, 1(03): 1–5.

Gardner, S. N., Slezak, T., and Hall, B. G. 2015. ksnp3. 0: Snp detection and phylogenetic analysis of genomes without genome alignment or reference genome. Bioinformatics, 31(17): 2877–2878.

Gaut, B. S. and Lewis, P. O. 1995. Success of maximum likelihood phylogeny inference in the four-taxon case.

Molecular Biology and Evolution, 12(1): 152–162.

Hasegawa, M., Kishino, H., and Yano, T.-a. 1985. Dating of the human-ape splitting by a molecular clock of mitochondrial dna. Journal of molecular evolution, 22: 160–174.

Hatje, K. and Kollmar, M. 2012. A phylogenetic analysis of the brassicales clade based on an alignment-free sequence comparison method. Frontiers in plant science, 3: 192.

Haubold, B. 2014. Alignment-free phylogenetics and population genetics. Briefings in bioinformatics, 15(3): 407–418.

Haubold, B., Klötzl, F., and Pfaffelhuber, P. 2015. andi: Fast and accurate estimation of evolutionary distances between closely related genomes. Bioinformatics, 31(8): 1169–1175.

Höhl, M. and Ragan, M. A. 2007. Is multiple-sequence alignment required for accurate inference of phylogeny? Systematic biology, 56(2): 206–221.

Jukes, T. H., Cantor, C. R., et al. 1969. Evolution of protein molecules. Mammalian protein metabolism, 3: 21–132.

Kemena, C. and Notredame, C. 2009. Upcoming challenges for multiple sequence alignment methods in the high-throughput era. Bioinformatics, 25(19): 2455–2465.

Kidd, K. K. and Sgaramella-Zonta, L. A. 1971. Phylogenetic analysis: concepts and methods. American journal of human genetics, 23(3): 235.

Kiefer, J. 1953. Sequential minimax search for a maximum. Proceedings of the American mathematical society, 4(3): 502–506.

Kimura, M. 1981. Estimation of evolutionary distances between homologous nucleotide sequences. Proceedings of the National Academy of Sciences, 78(1): 454–458.

Kimura, M., Dijk, J., and Heiland, I. 1980. The primary structure of protein bl17 isolated from the large subunit of the bacillus stearothermophilus ribosome. FEBS Letters, 121(2): 323–326.

Kuhner, M. K. and Felsenstein, J. 1994. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. Molecular biology and evolution, 11(3): 459–468.

Lalee, M., Nocedal, J., and Plantenga, T. 1998. On the implementation of an algorithm for large-scale equality constrained optimization. SIAM Journal on Optimization, 8(3): 682–706.

Marais, G. and Kingsford, C. 2011. A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. Bioinformatics, 27(6): 764–770.

Omojokun, E. O. 1989. Trust region algorithms for optimization with nonlinear equality and inequality constraints. University of Colorado at Boulder.

Ondov, B. D., Treangen, T. J., Melsted, P., Mallonee, A. B., Bergman, N. H., Koren, S., and Phillippy, A. M. 2016. Mash: fast genome and metagenome distance estimation using minhash. Genome biology, 17(1): 1–14.

Polyak, B. T. 1987. Introduction to optimization.

Ross, S. M. 2014. Introduction to probability models. Academic press.

scipy.org 2023a. scipy.minimize v1.10.1 trust-constr documentation.

scipy.org 2023b. scipy.minimize v1.10.1 trust-constr reference.

scipy.org 2023c. scipy.minimize_scalar v1.10.1 bounded method documentation.

Shannon, C. E. 1948. A mathematical theory of communication. The Bell System Technical Journal, 27(3): 379–423.

Sherwin, W. B. 2010. Entropy and information approaches to genetic diversity and its expression: Genomic geography. Entropy, 12(7): 1765–1798.

Tavaré, S. 1986. Some probabilistic and statistical problems in the analysis of dna sequences. Lect Math Life Sci (Am Math Soc), 17: 57–86.

Thompson, J. D., Plewniak, F., and Poch, O. 1999. A comprehensive comparison of multiple sequence alignment programs. Nucleic acids research, 27(13): 2682–2690.

Thompson, J. D., Linard, B., Lecompte, O., and Poch, O. 2011. A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives. PloS one, 6(3): e18093.

Wang, L. and Jiang, T. 1994. On the complexity of multiple sequence alignment. Journal of computational biology, 1(4): 337–348.

Wright, S. J. 2015. Coordinate descent algorithms. Mathematical programming, 151(1): 3–34.

Yang, Z. 1994. Statistical Properties of the Maximum Likelihood Method of Phylogenetic Estimation and Comparison With Distance Matrix Methods. Systematic Biology, 43(3): 329–342.

Yi, H. and Jin, L. 2013. Co-phylog: an assembly-free phylogenomic approach for closely related organisms. Nucleic acids research, 41(7): e75–e75.

Zahin, T., Abrar, M. H., Rahman, M., Tasnim, T., Bayzid, M. S., and Rahman, A. 2019. An alignment-free method for phylogeny estimation using maximum likelihood. bioRxiv, pages 2019–12.

Zielezinski, A., Girgis, H. Z., Bernard, G., Leimeister, C.-A., Tang, K., Dencker, T., Lau, A. K., Röhling, S., Choi, J. J., Waterman, M. S., et al. 2019. Benchmarking of alignment-free sequence comparison methods. Genome biology, 20(1): 1–18.