MPLE: Maximum K-subtree Pseudo-likelihood Methods for Phylogeny Reconstruction

by

Weifeng Wang

(Under the Direction of Liang Liu)

Abstract

Phylogenetic trees are fundamental tools for understanding the evolution of species. Several computational approaches have been developed to estimate phylogenetic trees from molecular sequence data. Although maximum likelihood methods can accurately reconstruct phylogenetic trees, they face computational challenges when searching for optimal solutions in huge tree spaces. The computation time for maximum likelihood approaches increases exponentially as the size of molecular sequences increases. In this paper, I propose a method called Maximum K-subtree pseudo-likelihood estimate (**MPLE**) in which the K-subtree pseudo-likelihood function is used to replace the full likelihood function. The identifiability and consistency of **MPLE** method guarantee its accuracy. To learn the loss of efficiency caused by the misspecified assumption by **MPLE**, Fisher information of branch is compared between full likelihood function and pseudo-likelihood function. At last, we construct the `MPLE` algorithm and compare it with the most popular maximum-likelihood methods `RAxML`. Extensive experiments on simulated datasets are conducted to evaluate our approach's effectiveness in terms of time complexity and accuracy concerning the accuracy of topology and mean square error of branch estimate. The results show that our approach `MPLE` would obtain an accurate estimate and cost less time, compared with representative methods.

Index words: [Phylogenetic tree, K-subtree Pseudo-likelihood, Fisher information]

MPLE: Maximum K-subtree Pseudo-likelihood Methods for Phylogeny Reconstruction

by

Weifeng Wang

B.S., Henan University, China, 2014
M.S., University of Georgia, 2018

A Dissertation Submitted to the Graduate Faculty of the
University of Georgia in Partial Fulfillment of the Requirements for the Degree

Doctor of Philosophy

Athens, Georgia

2021

MPLE: Maximum K-subtree Pseudo-likelihood Methods for Phylogeny Reconstruction

by

Weifeng Wang

Major Professor: Liang Liu

Committee: Pengsheng Ji
Shuyang Bai
Liming Cai

Electronic Version Approved:

Ron Walcott
Dean of the Graduate School
The University of Georgia
December 2021

Acknowledgments

My Ph.D. is a long journey with many ups and downs. It was not possible without the help and encouragement of many amazing people around me.

I would first like to thank my major professor, Dr. Liang Liu, for his invaluable support and assistance during my study. He is not only my supervisor in research but also a teacher in my life. I learned a lot from him.

In addition, I would also like to extend my thanks to the members of my committee group, Dr. Ji, Dr. Bai, and Dr. Cai, for their time and service.

Finally, I must thank my family. Thank my parents who spent their whole life giving me the best. Thank my brother and my sister for their support during my life and study. And thank my wife, Xiaowan Zhong, and my son, Jasper Wang, for their love and encouragement in my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The study of phylogenetics has increased tremendously over the past decades. Phylogeny reconstruction aims to find the evolutionary history of a group of entities. It depends on the alignment of sequence data containing the characters of species, which could be nucleic acids (Jill Harrison and Langdale, 2006).

There are two types of nucleic acids, Deoxyribonucleic acid (DNA) and Ribonucleic acid (RNA), which carry the biological information. RNA strands are generated by treating DNA strands as a template which is called the *transcription* process. DNA has four nucleotides including adenine (A), cytosine (C), guanine (G), and thymine (T). Adenine (A) and cytosine (C) are *purines*. Guanine (G) and thymine (T) are *pyrimidines*.

DNA can be replicated/copied repeatedly over time. But there may be very few errors during copying, which caused mutations. There are different types of mutations, such as substitutions, deletions, insertions, re-combinations, and inversions. The occurrence of mutations is critical for evolution. Beneficial mutations can be passed from parents to offspring, and other non-beneficial mutations would be filtered out by natural selection. This would make the populations naturally vary and generate different species.

With the development of rapid DNA sequencing techniques, the amount of sequence data increased dramatically, which provides a fundamental resource for phylogenetic studies. Analyzing the similarities of sequence data among species can be used to infer their evolutionary history. The phylogenetic study is also useful to learn virus/disease transmission and build resistance development, such as COVID-19.

A *sequence alignment* can be seen as an $m \times n$ matrix which represents n sequences of molecular data (DNA, RNA, or protein) from m species/taxa, where each row corresponds to a sequence from one taxon and each column is one site. Sequence alignment could be used to recover the evolutionary history of species. Figure 1.1 shows the first 100 DNA sequences of 8 different species of yeast (Rokas et al., 2003).

In phylogenetics, a *Phylogenetic tree* is a branching diagram describing the evolutionary relationship among a set of species. The lines in a phylogenetic tree are known as *branches*. The *branch lengths* represent the evolutionary times that are proportional to the genetic distances. One common scale of the branch lengths for molecular data is the expected number of transitions per site. Branches start and end in *nodes*.

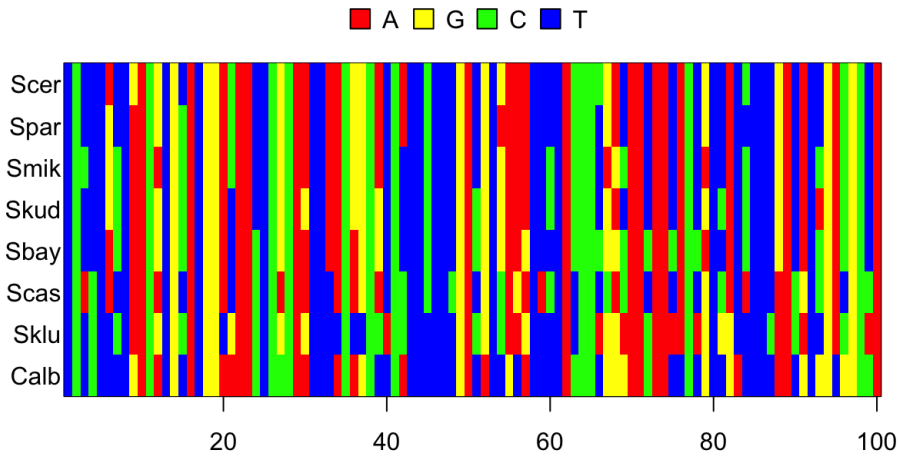

Figure 1.1: Alignment of the first 100 genes of 8 different species of yeast.

Nodes at the tip of the tree are *external nodes/taxa*, while nodes that have child nodes are *internal nodes*. The sequences data of the *external nodes* are observed, while that of the *internal nodes* are missing. The topmost node in a tree is called *root*.

Trees may be *rooted* or *unrooted*. Figure 1.2 shows a rooted tree with 4 taxa and an unrooted tree with the same number of taxa.

For a tree with m taxa/tips, there are m external nodes, $m-2$ internal nodes, and $2m-3$ branches. And the number of possible rooted tree topologies is

$$|\tau_{unrooted}| = \frac{(2m-5)!}{2^{m-3}(m-3)!},$$

and the number of all unrooted topologies is given by

$$|\tau_{rooted}| = \frac{(2m-3)!}{2^{m-3}(m-3)!}.$$

For example, Figure1.3 shows one possible unrooted tree topology of a 10-taxon tree. There are 8 internal nodes, 17 branches, and the total number of unrooted tree topologies is $\frac{15!}{2^7 \times 7!}$.

Phylogenetic reconstruction models are tree reconstruction models based on multiple sequence alignments of the external nodes. Under the time-reversible assumption, most phylogenetic methods estimate the unrooted trees which can not infer the ancestral root. One popular way to root an *unrooted tree* is the

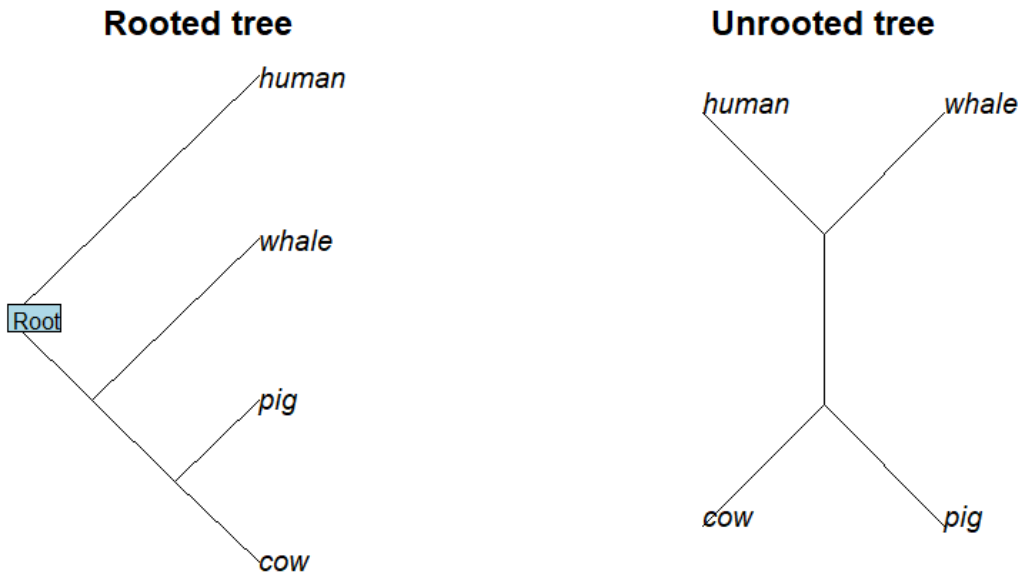

Figure 1.2: A rooted tree (left) and an unrooted tree (right) with 4 taxa

outgroup method. Add a taxon called the *outgroup* which is far away from the taxa of the interested group. Hence, the *outgroup* stems from the base of the tree, which can infer a sense of where the root of the tree is. For example, to find the relationship among humans, chimpanzees, and gorillas, adding an *outgroup* Dog. The root of the estimated unrooted tree should be on the branch joining dog and other species.

From 1963, Edwards and Cavalli-Sforza firstly tried to build the systematic foundation of phylogeny reconstruction models (L. Cavalli-Sforza and Edwards, 1963; L. L. Cavalli-Sforza and Edwards, 1967; L. L. Cavalli-Sforza et al., 1964). In their papers, they introduced 3 possible ways to reconstruct phylogenetic trees:

1. Maximum likelihood method (Original idea) uses the Gaussian random walk to fit the distances from ancestor forward time. This format of the likelihood method suffered from the singularity. It was the first time when researchers mentioned maximum likelihood and random walk, which was critical for the Markov process to be applied in the substation model.

2. Minimum-evolution **ME** method (L. Cavalli-Sforza and Edwards, 1963), in which the goal is to find the tree topology that has the minimum-evolution method minimum total number of evolution.

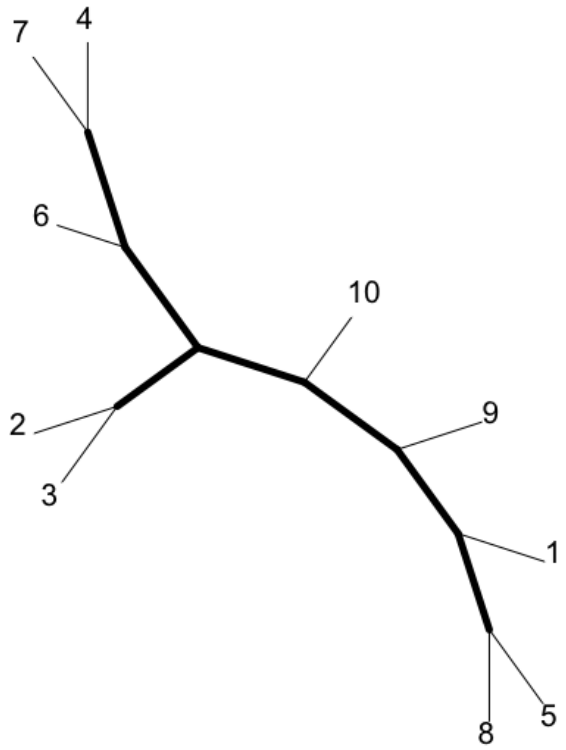

Figure 1.3: An unrooted phylogenetic tree with 10 taxa

3. Additive tree model (L. L. Cavalli-Sforza et al., 1964) is a kind of distance-based method which assumes independence of evolution among branches.

Compare 2 and 3, to change the tree topology, the additive tree method has a branch of negative length, which is unreasonable, while the **ME** method gives a zero-length branch. Another difference is that the longer branches in the additive method become likely shorter than the corresponding branches in the **ME** method L. L. Cavalli-Sforza and Edwards, 1967.

Note: Although, these ideas may not be optimal, they came up with the basic ideas of tree reconstructing methods based on sequential data, which built the foundation of graphical models and probabilistic models for phylogeny reconstruction.

Generally speaking, the Graphical models **GM's** for phylogeny reconstruction can be divided into two parts: Maximum parsimony (**MP**) method (Dayhoff, 1969; Fitch, 1977) and Distance-based methods, and the Probabilistic graphical models **PGM's** for phylogeny reconstruction come to the Maximum likelihood substitution models and Bayesian inferences.

1.2 Contributions

The main contributions of this paper are as follows,

- We propose a new phylogeny reconstruction method based on the Pseudo-likelihood function which is much easier to calculate than the full likelihood methods.
- The identifiability and consistency of **MPLE** guarantees the accuracy of the estimate.
- The efficiency of pseudo-likelihood function and full likelihood function is compared through Fisher information.
- Finally, extensive experiments on multiple simulated and real-world datasets show that our approach is promising.

1.3 Thesis Structure

This thesis is structured as follows: Chapter 2 provides preliminary work about phylogeny reconstruction methods. In Chapter 3.4, we define the Pseudo-likelihood function and construct the model Maximum pseudo-likelihood estimate (**MPLE**). And we show the model performance on simulated datasets and real-world datasets. Finally, we conclude and discuss future work in Chapter 7 and Chapter 8.

Chapter 2

Related Work

There are different models for reconstructing trees from molecular sequences, which can be classified into two groups distance-based methods and character-based methods.

2.1 Graphical models GM's

The distance-based methods include Unweighted pair group method with arithmetic mean (`UPGMA`) (Sokal, 1958), Fitch-Margoliash (`FM`) method (Fitch and Margoliash, 1967), Neighbor-joining (`NJ`) method (Saitou and Nei, 1987) and Minimum-evolution (**ME**) method (Rzhetsky and Nei, 1992). These methods are based on the distance matrix, in which consider the differences of sequences data as distances.

The `UPGMA` algorithm by Sokal and Michener (Sokal, 1958) assumes a constant substitution rate over time and branches and combines groups using average distances of two groups iteratively. The `FM` method does not assume a constant substitution rate, while it assumes the distances are additive.

The **ME** method assumes that the best tree has the smallest sum of branch lengths. The `NJ` method combines pairs of neighbors that are connected by an interior node successively. `NJ` is different from the **ME** method since its optimal function is minimizing the sum of branch lengths, which may not end up with the **ME** tree topology.

The **MP** method tries to minimize the branch lengths by minimizing the number of substitutions from the character matrix, which is different from the **ME** method that minimizes the branch lengths by minimizing the distance from the distance matrix. One common property is that both of them use a heuristic search strategy to search for the best tree.

A **Minimum spanning tree (MST)** of a connected, weighted, and undirected graph is a tree having the minimum total edge weight. **MST** based methods are distance-based, so it has the advantages of distance-based methods, such as lower time complexity and also the disadvantages of lack of statistical theory. In 2011, Choi et al. proposed another distance-based method called Chow-Liu grouping (`CLGrouping`) which is more accurate than **NJ** for trees have large diameter (Choi et al., 2011). In `CLGrouping` method, it considers the internal nodes which are unobserved as latent variables. First, it constructs a minimum spanning tree M based on the pairwise distances and then considers each in-

ternal node v_i and its neighbors V_i to construct subtrees T_i based on the distances. At last, it derives the reconstructed tree from T_i. One problem of the `CLGrouping` method is that the reconstructed tree may not be the true tree. To overcome this indeterminacy, Kalaghatgi and Lengauer introduced Vertex order-based minimum spanning trees (`VMSTs`) (Kalaghatgi and Lengauer, 2017) in 2017. To construct the desired **MST**, it modifies Kruskal's algorithm by inputting the sorted edges for edge weight and the new-defined vertex order.

2.1.1 GM algorithms

When we consider the **Tree of life** which may contain millions of species, it is too expensive to reconstruct the tree using **GM** algorithms. So speeding up the algorithms has become an important issue for phylogeny reconstruction since 2000. In general, there are two approaches to improve the **GM** algorithms:

1. One approach is changing the way to search the optimal function. Such as `QuickJoin` algorithm (Mailund and Pedersen, 2004) using `quad-tree`, `Clearcut` (Sheneman et al., 2006) applying relaxed neighbor joining (`RNJ`) to avoid searching all distances which is faster than original `NJ` and `QuickJoin`.

2. The most recent approach is modifying the selection criterion. Such as Fast neighbor-joining (`FNJ`) (Elias and Lagergren, 2009) proposed by Elias and Lagergren and `FastNJ` (Li, 2015), which only search the optimal sum of lengths in the stored set of all likely candidates.

3. Another way to improve `NJ` method is using parallel algorithms. For example, Rucci et al. (2013) presented a parallel algorithm for neighbor-joining based on the multicore cluster, and Al-Neama et al. (2014) implemented a parallel algorithm on OpenMP. In addition, Du and Feng (2006) proposed the `pNJTree` parallel method for neighbor-joining using a message passing interface (`MPI`) running on a workstation cluster, and `ERapidNJ` (Simonsen and Pedersen, 2011) using parallelization on both `CPUs` and `GPUs`.

At last, I'd like to introduce a popular modified **GM** method called `FastTree`. `FastTree1` (Price et al., 2009) uses sequence profiles of internal nodes to implement `NJ` and choose the candidate joins by heuristics. Then it applies Nearest neighbor interchanges `NNI` to reduce the length of the tree. Different from the `FastTree1`, `FastTree2` introduces Subtree pruning regrafting (`SPR`) and maximum-likelihood `NNIs` (Price et al., 2010). And `FastTree2` is more accurate than `FastTree` and other **GM** methods, since it applies the maximum-likelihood in the selection criterion.

2.2 Probabilistic Graphical Models

PGM's are more accurate and preferred than other methods, **MP** method and Distance-based methods, which ignore the underlying evolutionary mechanism and simply use the character matrix or the distance matrix which lose important phylogenetic information. Instead, **PGM's** are grounded in statistical theory and defined on explicit evolution models. In general, **PGM's** include maximum likelihood and Bayesian inference approaches.

2.2.1 Mutation Model

To reconstruct the tree topology which is a graph, we add probabilities on the tree which forms the Probabilistic graphical models (**PGM's**). Recall the Markov chains which have the property that the future state only depends on the present state. In **PGM's**, we can assume that the DNA of the descendants only depends on their ancestor, which is analog to the Markov property. In this way, Felsenstein firstly introduced Markov chains to Phylogenetic models (Felsenstein, 1981).

2.2.2 Continuous Time Markov Chain(CTMC)

Definition 1.1**:** $\{X(t), t \geq 0\}$ is a *continuous time Markov chain*(**CTMC**), if for $\forall s, t \geq 0, i, j \in S$, S is the state space, define the transition probabilities by

$$p_{ij}(s,t) = Pr(X_{t+s} = j | X_s = i, X_u = x(u), 0 \leq u \leq s) = Pr(X_{t+s} = j | X_s = i).$$

In general, the transition probability matrix is defined as

$$\mathbf{P}(t) = (p_{ij}(t)) = \begin{pmatrix} p_{1,1}(t) & p_{1,2}(t) & p_{1,3}(t) & \cdots \\ p_{2,1}(t) & p_{2,2}(t) & p_{2,3}(t) & \cdots \\ p_{3,1}(t) & p_{3,2}(t) & p_{3,3}(t) & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix},$$

which has the properties:

- $p_{ij}(t) \geq 0, i, j \in S$;
- $\Sigma_{j \in S} p_{ij}(t) = 1, \forall i \in S$.

A **CTMC** is also defined by the rate of change between pairs of states. The rates are usually defined as

- $\lim_{t \to 0} \frac{p_{ij}(t)}{t} = q_{ij}$
- $\lim_{t \to 0} \frac{1 - p_{ii}(t)}{t} = q_{ii}$

where q_{ij} is the rate of change from state i to state j. We often specify a **CTMC** using a rate matrix:

$$\mathbf{Q} = (q_{ij}) = \begin{pmatrix} q_{1,1} & q_{1,2} & q_{1,3} & \cdots \\ q_{2,1} & q_{2,2} & q_{2,3} & \cdots \\ q_{3,1} & q_{3,2} & q_{3,3} & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix},$$

1. **Time Homogenous**: A Markov chain$(X(t))$ is said to be *time homogeneous* if $p_{ij}(s,t) = Pr(X_{t+s} = j | X_s = i) = p_{ij}(t)$ is independent of s.

2. **Communicate**: States i and j are said to *communicate* if $p_{ij}^{(n)}(t) > 0$ for some n and $p_{ji}^{(m)}(t) > 0$ for some m.

3. **Irreducible**: A Markov chain is irreducible if all states communicate.

4. **Stationary**: A **CTMC** is stationary, if $\lim_{t\to\infty} p_{ij}(t) = \pi_j, where \sum_j \pi_j = 1$.

5. **Time reversible**: A **CTMC** is *time reversible* if $\pi_j q_{ji} = \pi_i q_{ij}$, which means the proportion of transitions occurring in the chain at stationarity from state i to state j equals the proportion of transitions from state j to state i. In other words, the distances of each pair of nodes are the same in the tree.

2.2.3 Mutation Model of a Single Nucleotide

Recall that a finite state continuous time Markov chain $X_t(t > 0)$ is a Markove process on the finite state space S. The state space for the nucleotide models is $\{A, C, G, T\}$, so state space of the Markov chain is of finite size. And the transition probability matrix is defined as

$$\mathbf{P}(t) = \begin{pmatrix} p_{A,A}(t) & p_{A,C}(t) & p_{A,G}(t) & p_{A,T}(t) \\ p_{C,A}(t) & p_{C,C}(t) & p_{C,G}(t) & p_{C,T}(t) \\ p_{G,A}(t) & p_{G,C}(t) & p_{G,G}(t) & p_{G,T}(t) \\ p_{T,A}(t) & p_{T,C}(t) & p_{T,G}(t) & p_{T,T}(t) \end{pmatrix}.$$

Because all transition rates are positive, all states *communicate*, i.e. the Markov chain will not be trapped in any state – it is irreducible. Let the *stationary probability* of being in state i be π_i, which can be found as the solution to the following set of equations.

$$\pi_i = \sum_j \pi_j p_{ji}, \sum_j \pi_j(t) = 1. \tag{2.1}$$

The transition probabilities can be found by solving a set of linear ordinary differential equations, which are know as *Kolmogorov Equations*. In this paper, we will use the particular versions known as the *Forward Kolmogorov Equations*:

$$\frac{\partial p_{ij}(t)}{\partial t} = \sum_k q_{kj} p_{ik}(t), t \geq 0. \tag{2.2}$$

An intuitive explanation of this equation goes as follows: if the current state of the chain is state $k, k \neq j$, the rate of change into state j is q_{kj}. If the current state is state k and $k = j$, the rate of change is away from state j is $\Sigma_{i\neq j} q_{ji}$. Starting from state i at time 0, the probability of being in state k at time t is $p_{ik}(t)$. The rate of change in the probability of being in state j at time t, $\frac{\partial p_{ij}(t)}{\partial t}$, is simply the sum over all $k, k \neq j$, of the probability of being in state k times the rate of change into state j from state k, minus the probability of being in state j times the rate of change away from state j.

In general, it is not so easy to find the transition probability matrix of a particular rate matrix,

$$\mathbf{Q} = \begin{pmatrix} - & a & b & c \\ a & - & d & e \\ b & d & - & f \\ c & e & f & - \end{pmatrix}.$$

However, by using matrix algebra we can usually find solutions to the differential equations. In general, we can get the *Komogorov's Forward Equations* as

$$\frac{\partial \mathbf{P}(t)}{\partial t} = \mathbf{P}(t)\mathbf{Q}, \tag{2.3}$$

where $\mathbf{P}(t) = \{p_{ij}(t)\}$ is the matrix of transition probabilities. From multivariable calculus we may have learned that the solution to this set of linear ordinary differential equations, with the initial condition $\mathbf{P}(0) = \mathbf{I}$ (the identity matrix), can be obtained by exponentiating $\mathbf{Q}$. The matrix of transition probabilities is obtained as

$$\mathbf{P}(t) = e^{\mathbf{Q}t} = \sum_{i=0}^{\infty} \frac{(\mathbf{Q}t)^i}{i!}. \tag{2.4}$$

2.2.4 Transition Matrix

According to the *Kolmogorov's Forward Equations*:

$$\frac{\partial \mathbf{P}(t)}{\partial t} = \mathbf{P}(t)\mathbf{Q}.$$

where $\mathbf{P}(t) = \{p_{ij}(t)\}$ is the matrix of transition probabilities with the initial condition $\mathbf{P}(0) = \mathbf{I}$. To obtain the matrix of transition probabilities, we can solve the following equation by Computing matrix exponentials.

$$\mathbf{P}(t) = e^{\mathbf{Q}t} = \sum_{i=0}^{\infty} \frac{(\mathbf{Q}t)^i}{i!}.$$

Suppose that Q is an $n * n$ matrix with n distinct eigenvectors. Then, letting Λ be a diagonal matrix consisting of the eigenvalues of Q, we can decompose Q into

$$Q = V\Lambda V^{-1},$$

where V consists of the eigenvectors of Q (ordered similarly to the order of the eigenvalues in Λ). In this case, we get the very nice identity

$$\mathbf{P}(t) = e^{\mathbf{Q}t} = Ve^{\Lambda t}V^{-1}, \tag{2.5}$$

where $e^{\Lambda t}$, because Λ is diagonal, is a diagonal matrix with diagonal elements $e^{\lambda_i t}$ where λ_i is the i^{th} eigenvalue.

Transition Probabilities for $K2P$ Model

First, we introduce two different substations in DNA. Transition is the substation that changes between purines or between pyrimidines. A transversion is a substation that changes between purine and pyrimidine. The idea of the Kimura 2 Parameter ($K2P$) model (Kimura, 1980) comes from that transition is less likely to result in substitution than transversion. So it assumes that transitions and transversions occur at different rates α and β separately.

$$\mathbf{Q} = \begin{pmatrix} -\alpha-2\beta & \beta & \beta & \alpha \\ \beta & -\alpha-2\beta & \alpha & \beta \\ \beta & \alpha & -\alpha-2\beta & \beta \\ \alpha & \beta & \beta & -\alpha-2\beta \end{pmatrix}.$$

In the following, we will derive the transition probabilities of $K2P$ model as an example. First, we calculate the eigenvalues and eigenvectors of matrix Q. Solving the equation $|Q-\lambda I|v=0$, we have

$$\lambda(\lambda+2\alpha+2\beta)^2(\lambda+4\beta)=0.$$

So the eigenvalues of matrix Q in $K2P$ model are $\lambda_1=\lambda_2=-2(\alpha+\beta), \lambda_3=-4\beta, \lambda_4=0$ and the corresponding eigenvectors are $v_1=v_2=(1,1,-1,-1), v_3=(1,-1,-1,1), v_4=(1,1,1,1)$. In other words, we have the following matrix:

$$\Lambda = \begin{pmatrix} -2\alpha-2\beta & 0 & 0 & 0 \\ 0 & -2\alpha-2\beta & 0 & 0 \\ 0 & 0 & -4\beta & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \text{ and } \mathbf{V} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ -1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 \end{pmatrix}.$$

Then applying them into Equation 2.5:

$$\mathbf{P}(t) = Ve^{\Lambda t}V^{-1} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ -1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 \end{pmatrix} \begin{pmatrix} e^{(-2\alpha-2\beta)t} & 0 & 0 & 0 \\ 0 & e^{(-2\alpha-2\beta)t} & 0 & 0 \\ 0 & 0 & e^{-4\beta t} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ -1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 \end{pmatrix}^{-1}.$$

The transition probabilities obtained by solving the *Kolmogorov's Forward Equations* for the $K2P$ model are

$$p_{ij}(t) = \begin{cases} \frac{1}{4}+\frac{1}{4}e^{-4\beta t}+\frac{1}{2}e^{-2(\alpha+\beta)t} & if i=j, \\ \frac{1}{4}+\frac{1}{4}e^{-4\beta t}-\frac{1}{2}e^{-2(\alpha+\beta)t} & if i\neq j (transition), \\ \frac{1}{4}-\frac{1}{4}e^{-4\beta t} & if i\neq j (transversion). \end{cases}$$

Transition Probabilities for JC Model

For JC mdoel, the probability that state i changes to state j after time t is given by

$$p_{ij}(t) = \begin{cases} \frac{1}{4} + \frac{3}{4}e^{-\frac{4t}{3}} & if i = j, \\ \frac{1}{4} - \frac{1}{4}e^{-\frac{4t}{3}} & if i \neq j. \end{cases}$$

Transition Probabilities for HKY85

$$\mathbf{P(t)} = \begin{pmatrix} h & i & p & q \\ j & k & p & q \\ r & s & l & m \\ r & s & n & o \end{pmatrix},$$

where

$$\begin{aligned}
h &= \pi_t + \frac{\pi_t(\pi_a + \pi_g)}{\pi_t + \pi_c}e^{-\beta t} + \frac{\pi_c}{\pi_t + \pi_c}e^{-(\alpha(\pi_t+\pi_c)+\beta(\pi_a+\pi_g))t} \\
i &= \pi_c + \frac{\pi_c(\pi_a + \pi_g)}{\pi_t + \pi_c}e^{-\beta t} - \frac{\pi_c}{\pi_t + \pi_c}e^{-(\alpha(\pi_t+\pi_c)+\beta(\pi_a+\pi_g))t} \\
j &= \pi_t + \frac{\pi_t(\pi_a + \pi_g)}{\pi_t + \pi_c}e^{-\beta t} - \frac{\pi_t}{\pi_t + \pi_c}e^{-(\alpha(\pi_t+\pi_c)+\beta(\pi_a+\pi_g))t} \\
k &= \pi_c + \frac{\pi_c(\pi_a + \pi_g)}{\pi_t + \pi_c}e^{-\beta t} + \frac{\pi_t}{\pi_t + \pi_c}e^{-(\alpha(\pi_t+\pi_c)+\beta(\pi_a+\pi_g))t} \\
l &= \pi_a + \frac{\pi_a(\pi_t + \pi_c)}{\pi_a + \pi_g}e^{-\beta t} + \frac{\pi_g}{\pi_a + \pi_g}e^{-(\alpha(\pi_a+\pi_g)+\beta(\pi_t+\pi_c))t} \\
m &= \pi_g + \frac{\pi_g(\pi_t + \pi_c)}{\pi_a + \pi_g}e^{-\beta t} - \frac{\pi_g}{\pi_a + \pi_g}e^{-(\alpha(\pi_a+\pi_g)+\beta(\pi_t+\pi_c))t} \\
n &= \pi_a + \frac{\pi_a(\pi_t + \pi_c)}{\pi_a + \pi_g}e^{-\beta t} - \frac{\pi_a}{\pi_a + \pi_g}e^{-(\alpha(\pi_a+\pi_g)+\beta(\pi_t+\pi_c))t} \\
o &= \pi_g + \frac{\pi_g(\pi_t + \pi_c)}{\pi_a + \pi_g}e^{-\beta t} + \frac{\pi_a}{\pi_a + \pi_g}e^{-(\alpha(\pi_a+\pi_g)+\beta(\pi_t+\pi_c))t} \\
p &= \pi_a(1 - e^{-\beta t}) \\
q &= \pi_g(1 - e^{-\beta t}) \\
r &= \pi_t(1 - e^{-\beta t}) \\
s &= \pi_c(1 - e^{-\beta t}).
\end{aligned}$$

2.2.5 Substitution Models

Usually, people assumed that the transition rate among sites is the same. Then, one can define the probabilistic models based on the Markov chains with homogeneous rate across sites. For the substitution model, it is assumed the substitutions follow Markov processes which can be specified by parameters in probabilistic models. The substitution models are different from how they specify the transition probabilities. Table shows a comparison of different substitution models.

Table 2.1: A comparison of different substitution models.

Model	Base frequencies	Rates	Free parameters	Comments
JC	$\pi_A = \pi_C = \pi_G = \pi_T$	a=b=c=d=e=f=1	0	simplest model
K2P	$\pi_A = \pi_C = \pi_G = \pi_T$	a=c=d=f=1,b=e	1	transition/transversion
F81	$\pi_A \neq \pi_C \neq \pi_G \neq \pi_T$	a=b=c=d=e=f=1	3	unequal base frequencies
HKY85	$\pi_A \neq \pi_C \neq \pi_G \neq \pi_T$	a=c=d=f=1,b=e	4	unequal base frequencies, and transition/transversion
GTR	$\pi_A \neq \pi_C \neq \pi_G \neq \pi_T$	a=b=c=d=e=f=1	8	all are different

For example, the **JC** model is the simplest model where all the changes among the 4 nucleotides are assumed to be with equal probability (Jukes, Cantor, et al., 1969). The **K2P** model allows different rates for transitions and transversions (Kimura, 1980). The **HKY85** model allows both different transitions and transversions rates and different frequencies (Hasegawa et al., 1985). And the General time reversible **GTR** model (Tavare 1986) requires 6 substitution rate parameters and 3 equilibrium frequency parameters, which is the most general case for all above models (Tavaré et al., 1986). Unlike **JC** model and **HKY85** model, there is no closed-form for probability transition matrix of **GTR** model. So, all calculations need to be done numerically by computer based on Equation 2.5.

2.2.6 Full Likelihood Function

To simplify, this paper considers 4 homologous sequences sampled from 4 different species with the length n, which means there are n sites in each sequence. There are $r = 4^4 = 256$ different patterns for 4 nucleotides. The *sequence alignment* Ω is often represented as a matrix, where columns are sites and rows are species.

$$\Omega_{4\times r} = \begin{pmatrix} A & A & A & A & A & A & A & \dots & C & \dots \\ A & A & A & A & A & A & A & \dots & C & \dots \\ A & A & A & A & G & C & T & \dots & C & \dots \\ A & G & C & T & A & A & A & \dots & C & \dots \end{pmatrix}_{4\times r}.$$

In the *sequence alignment* Ω, for $i = 1, 2, 3, \dots, r$, let column ω_i is one site pattern and p_i be the transition probability of site pattern ω_i, with $\sum_i^r p_i = 1$.

Let **D** denotes alignment of 4 species with the sequence length n.

$$\mathbf{D}_{4\times n} = \begin{pmatrix} A & A & C & T & \dots & C \\ A & A & C & A & \dots & G \\ A & A & G & A & \dots & G \\ A & T & C & A & \dots & G \end{pmatrix}_{4\times n} .$$

We want to find the probability of observing the alignment **D** given the phylogenetic tree and parameters in the substitution model. Let x_i be the count of pattern i in the alignment **D**. Let p_i be the probability of pattern i. Assuming that these n sites evolve independently, the random variables $(x_1, \dots, x_r)$ have a *Multinomial distribution* with parameters p_i for $i = 1, 2, 3, \dots, r$. We can find $\sum_{i=1}^{r} x_i = n$ and

$$Pr(\underline{\mathbf{x}}|p_1, p_2, \dots, p_r) = Pr(x_1, x_2, \dots, x_r|p_1, p_2, \dots, p_r) \propto \prod_{i=1}^{r} p_i^{x_i},$$

where x_i are the random variables, and p_i can be calculated based on the transition matrix defined by numerical parameter space Θ and the given tree defined by tree parameter $(\tau, \underline{t})$. For a given tree of 4

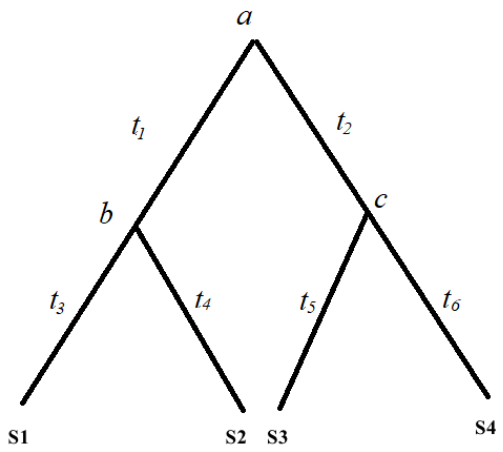

Figure 2.1: Phylogenetic tree example with 4 taxa

taxa shown as **Figure** 2.1 , one site with pattern $\omega_i = (S_{1i}, S_{2i}, S_{3i}, S_{4i})$. Since the site pattern may result from any combination of ancestral nodes, calculation of $p(\omega_i)$ needs to sum over the four nucleotides {A, C, G, T } for all ancestral nodes (a, b, c).

$$Pr(\omega_1|\tau, \underline{t}, \Theta) = \sum_{a,b,c\in\{A,G,C,T\}} \pi_a p_{b|a}(t_1) p_{c|a}(t_2) p_{S_{1i}|b}(t_3) p_{S_{2i}|c}(t_4) p_{S_{3i}|b}(t_5) p_{S_{4i}|c}(t_6),$$

where π_i is the stationary distribution at site i and $p_{ij}(t_k, \Theta)$ is the probability that state i mutates to state j given the tree parameters (τ, Θ).

Then the full likelihood function is

$$L(\tau, \underline{t}, \Theta|\mathbf{X}) \propto \prod_{i=1}^{r} (Pr(\omega_i|\tau, \underline{t}, \Theta))^{n_i},$$

where n_i is the number of pattern ω_i found in the observational pattern matrix $\mathbf{X}$.

2.2.7 Rates heterogeneous across sites (RHAS) models

The above models assume that the evolutionary rate is the same across sites, which is not realistic in practice. Because in the DNA sequence some parts of the sequence will decide some critical futures of the species, which are not likely to be changed. So the mutation rate of this section is much lower than that of other sections.

Thus, the more general work should be to model the rates heterogeneous across sites **RHAS** models. Felsenstein and Churchill firstly introduced the method of Hidden Markov Models to fit unequal and unknown evolutionary rates at different sites in molecular sequences (Felsenstein and Churchill, 1996). This method is not well used since too many parameters need to be estimated. More practical approximate methods including the discrete gamma model, fixed-rates model, and auto-discrete gamma model, are proposed by Yang in 1994 (Yang, 1994) and 1995 (Yang, 1995).

Recently, a model-selection method called `ModelFinder` (Kalyaanamoorthy et al., 2017) implements Yang's idea of auto-discrete gamma model for **RHAS** models. It assumes unequal weights of discrete gamma distribution and updates the weights using Expectation maximization **EM** algorithm. Simulated and real data sets show that `ModelFinder` gives higher accuracies than the previous model-selection methods without heterogeneous rates: `jModelTest` (Darriba et al., 2012) for DNA and `ProtTest3` (Darriba et al., 2011) for proteins.

Regardless of the branch-heterogeneity, the **ML** models for phylogenetic reconstruction can be divided into two parts: the substitution models and the Rate heterogeneity across sites (**RHAS**) models. The substitution models describe the evolution of a single nucleotide, and the **RHAS** models describe the transitions across sites. That's why we see the model like $GTR + I + \Gamma$.

2.2.8 Branch-heterogeneous models

Another heterogeneity occurs across branches in which case each branch may have a unique substitution process. The Branch-heterogeneous models are much more challenging because there are a large number of parameters to be estimated when considering the computational time and overparameterization problems. One common way is assuming a specific model for rate variation from branches to branches, such as lognormal distribution (Thorne et al., 1998) and compound Poisson process whose mean is controlled by a Gamma distribution (Huelsenbeck et al., 2000). In 2013, Groussin et al. introduced a new

branch-nonhomogeneous and nonstationary model of protein evolution that can cover more complicated sequence evolution. They applied the idea of Correspondence and likelihood analysis (**COaLA**) to reduce the number of parameters to be optimized through maximum likelihood (Groussin et al., 2013).

2.2.9 Algorithms based on PGM's

It is computationally expensive for **PGM** especially with thousands of alignments. `IQ-TREE` (Nguyen et al., 2015), `RAxML` (Stamatakis, 2014) and `PhyML` (Guindon et al., 2010) are the three most popular and effective **MLE** based algorithms for phylogenetic reconstruction problems. And the most recent algorithm `IQ-TREE` is faster and more accurate than the other two because of the implement of an efficient sampling method of local optima in the tree space. There are 3 possible directions to improve the **PGM's** algorithms.

Pruning algorithm

One way is to reorganize the full likelihood function and calculate the conditional likelihood, which can save time to calculate the likelihood values for each given tree topology. Note that for m interior nodes, it is expensive to sum over 4^{m-1} possible combinations of ancestral states. In the reformulation, calculating the full likelihood costs $O(m)$ time complexity. One popular technique is known as the pruning algorithm (Felsenstein, 1981), which is to identify the common factors and calculate them only one time. The pruning algorithm calculates the full likelihood function from the leaves towards the root, which is computationally efficient.

Take the tree in Figure 5.1 as an example,

$$Pr(\omega_1|\tau,\underline{t},\Theta) = \sum_a \pi_a p_{b|a|}(t_1)p_{c|a}(t_2)\sum_b p_{S_{1i}|b}(t_3)p_{S_{2i}|b}(t_4)\sum_c p_{S_{3i}|c}(t_5)p_{S_{4i}|c}(t_6).$$

It first calculates the conditional likelihood of the descendants (S_1, S_2) with ancestor b and the conditional likelihood of descendants (S_3, S_4) with ancestor c. Then it calculates the likelihood of descendants (b, c) with ancestor a, which gives the full likelihood.

Tree search strategy

For any tree searching method, exhaustive search, where all possible topologies are considered is unfeasible for even a small number of taxa. Most of **PGM's** algorithms focus on the searching strategy for the tree topologies to reduce computational time, such as Nearest-neighbor interchange (`NNI`) (Felsenstein and Felenstein, 2004) and Subtree Pruning and Re-grafting (`SPR`) algorithms (Allen and Steel, 2001).

A single pass of the `NNI` examines $O(m)$ trees at each pass, where m is the number of taxa in the tree. Figure 2.2 shows how `NNI` algorithm works for an unrooted 4-taxon tree.

In `SPR`, a subtree is selected and pruned. Then, it is regrafted onto another branch of the remaining tree to generate a new topology. This procedure is repeated for all regrafting positions. `SPR` algorithm examines $O(m^2)$ new tree topologies, This is because, for each subtree, there are $O(m)$ possible regraftings,

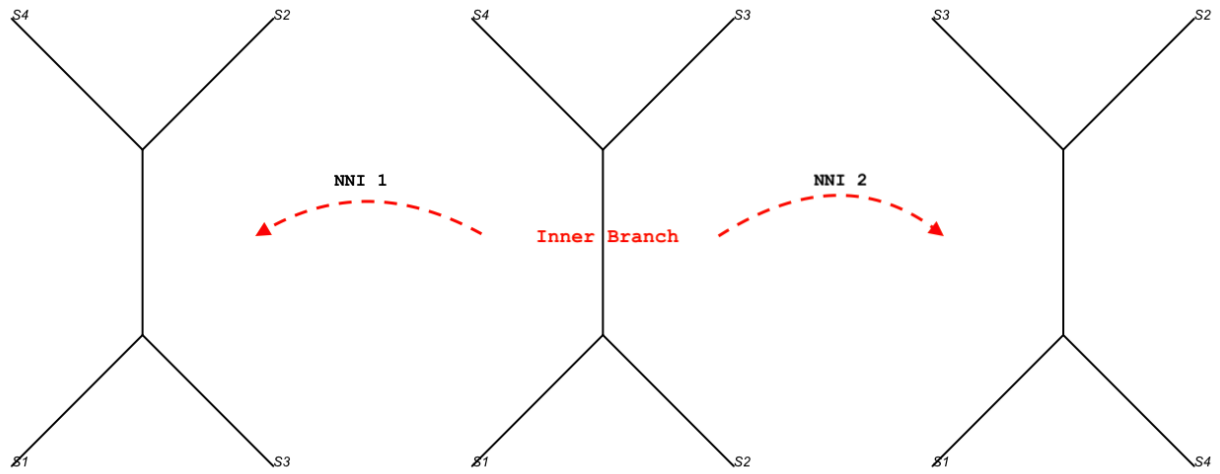

Figure 2.2: Tree rearrangement example with NNI on a 4-taxon tree

and there are $O(m)$ possible subtrees to consider. Figure 2.3 shows how SPR algorithm prunes the subtree S_5 and re-insert it with rearrangement radius of size 1.

Approximated likelihood method

There are other approaches to speed up the **PGM's** algorithms, in which one can sacrifice the accuracy using approximated likelihood methods.

Suppose that we have an intractable likelihood but the low-dimensional distributions are readily computed. To approximate the full likelihood function, we can use just a part of the data. For some weights $w_k \geq 0$, we can define a weighted product of the low-dimensional distributions called composite likelihood (Lindsay, 1988) as

$$CL(\theta; y) = \prod_{k=1}^{K} (L_k(\theta; y))^{w_k},$$

where $L_k(\theta; y) \propto f(y \in A_k; \theta)$ are component likelihoods associated with collection of marginal or conditional events $\{A_1, ..., A_K\}$.

The pseudo-likelihood (Besag, 1974) is the product of conditional densities. Quasi-likelihood (QL) is the product of marginal likelihoods for many small subsets. The pairwise likelihood is an example of a composite likelihood. Renard et al. show that inference based on the pairwise likelihood is quite efficient relative to that based on full likelihood (Renard et al., 2004).

The approximated likelihood method performs well when the length of the sequence is large and it is much faster than the full likelihood method when modeling tens of thousands of species. In compos-

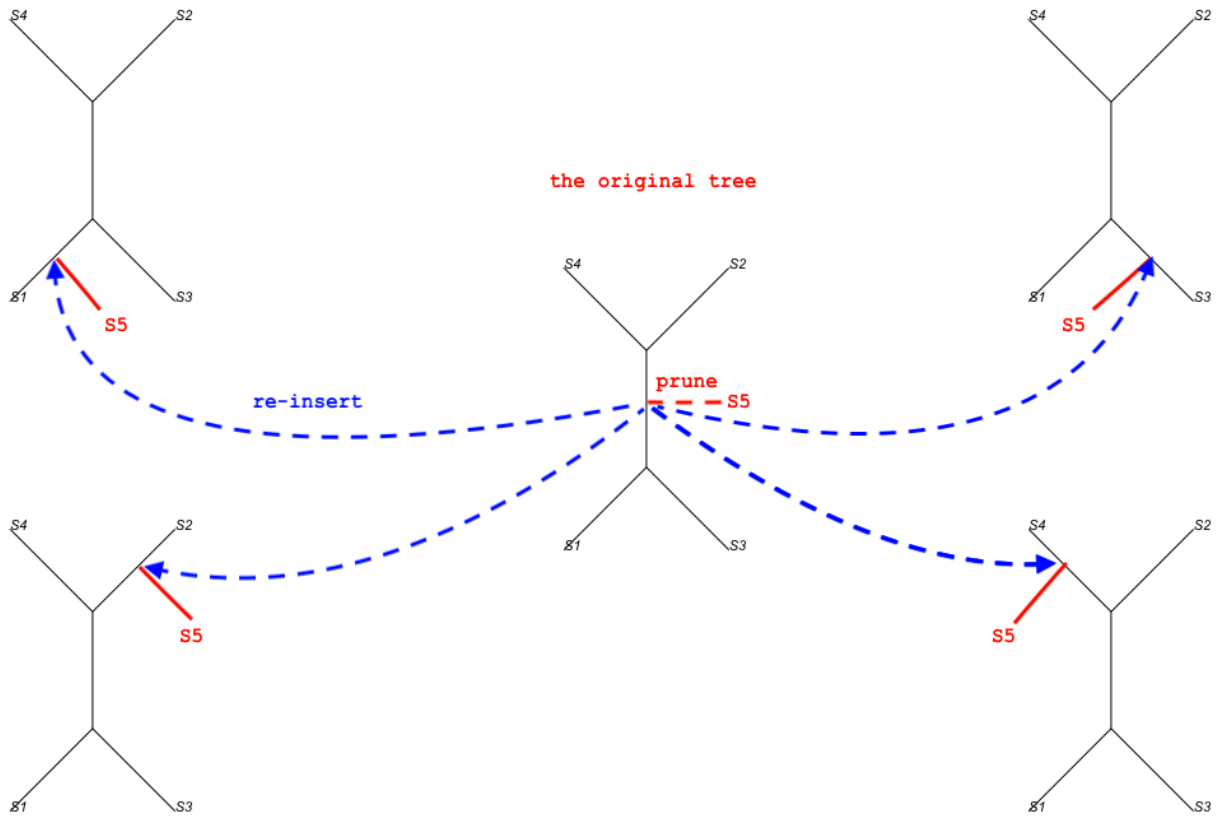

Figure 2.3: Tree rearrangement example by SPR on a 5-taxon tree

ite likelihood, each individual component is a conditional or marginal density, the resulting estimating equation obtained from the derivative of the composite log-likelihood is an unbiased estimating equation. Composite likelihood may be a misspecified likelihood because of the independence assumption (Varin et al., 2011). The composite likelihood is computational and able to avoid the need to model higher-order dependencies in the data when complex dependencies are involved (Larribe and Fearnhead, 2011).

2.3 Comparisons

2.3.1 Consistency

Consistency means when the length of the sequence goes to infinite, the estimated tree converges to the true tree. Non-consistency is serious since it means your model may have more than one reconstructed

tree when the data is from one true tree. The followings are the comparisons for consistency of these models:

1. Maximum likelihood methods are consistent.
2. Consistency of Distance-based methods:
   (a) **NJ** method is consistent (Saitou and Nei, 1987).
   (b) **ME** with ordinary least-squares criterion (**OLS**), while it is inconsistent with weighted least-squares criterion (**WLS**) (Rzhetsky and Nei, 1993).
3. Under many situations, **MP** method is not consistent.

2.3.2 Speed

Amount of paper jobs based on computer simulations talked about the efficiencies of Maximum parsimony and Distance-based methods (Hasegawa et al., 1991; Huelsenbeck, 1995a, 1995b; Saitou and Imanishi, 1989; Sourdis and Nei, 1988). In general, **ML** methods outperform other methods, especially when the substitution rate differs drastically. Most Distance-based methods are much faster than **ML** methods since distance-based methods reconstruct the best tree using "stepwise clustering" strategy, while **MP** and **ML** methods use "exhaustive search" which causes a lot of time-consuming during the tree topology searching. So Distance-based methods can handle large numbers of sequences.

ME and **NJ** methods would give similar results since both of them are distance based methods. However, **NJ** is faster than **ME**, since **ME** is "exhaustive search" which is more time-consuming than **NJ** who is "stepwise clustering".

The **MP** and **ME** are NP-complete (Bastkowski et al., 2016; Foulds and Graham, 1982), while **ML** is NP-hard (Chor and Tuller, 2006; Roch, 2006). The **MP** method is faster than the **ML** method since in **ML** method likelihood functions are calculated and optimal algorithms are recalled to search the optimal estimated parameters, which costs much computational time.

For an alignment with m sequences, the **NJ** and methods have a time complexity of $O(m^3)$ (Studier and Keppler, 1988). While the time complexity of the modified `NJ` methods, such as `FastNJ` (Li, 2015), can reach $O(m^2)$ at most of the time. Thus, the complexity of distance-based methods can be close to $O(m^2)$.

For "exhaustive search" based methods **ML** and **MP**, the total number of possible tree topologies is $(2m-1)!!$ which is a huge number when m is large. So the time complexities of these methods are much larger than distance based methods. For example, `FastTree2` is generally 100 times faster than `PhyML` (Guindon et al., 2010) or `RAxML` (Stamatakis, 2014) and has the similar accuracies.

2.3.3 Accuracy

For "stepwise clustering" based methods, such as `FastTree2`, may obtain the locally optimal tree instead of the true tree. In general, maximum likelihood-based algorithms shall archive higher accuracies than

MP or Distance-based methods in the simulation and cost acceptable computational time in practice. So if one cares more about the accuracy, **ML** methods are the best choice, even though it costs too much time.

2.3.4 Others

In recent years, some researchers try to apply neural networks(**NN**) into phylogenetic reconstructions. In 2016, Kasperskia and Kasperskab introduced a way to apply artificial neural networks (**ANN**) (Kasperski and Kasperska, 2016) to the automatic identification of organism evolution. In this paper, it shows that after a long time of training, **ANN** will reconstruct the evolutionary tree quickly. **ANN** method is totally different from **ML** or distance-based methods since the evolutionary relationships are trained or learned from 32 successive cytochrome *b* sequences in this paper.

Later on, Fioravanti et al. introduced a deep learning method called Phylogenetic Convolutional Neural Networks (`PhCNN`) (2017) in phylogenetic analysis. (Fioravanti et al., 2018) They use patristic distance (the sum of all branches connecting two external nodes) to measure the distance on the tree. It may be true that **NN** gives higher accuracy than other **GM's**, while one problem for **NN** based methods for phylogenetic reconstruction I doubt is how to do inferences which is a key point in statistical analyses.

For multilocus sequence data, accurate phylogenetic inference for the evolutionary history of species would be more difficult. It is shown that, for large phylogenomic data sets, model comparisons favor the coalescent model over the concatenation model (Jiang et al., 2020).

To end this summary, the main jobs in phylogenetics are focusing on the more realistic model-based methods and making the methods computationally efficient. Because the **PGM's** have a strong statistical foundation which ensures the accuracy and inferences of the models. So the most popular programs used in phylogenetic reconstruction are `RAxML` and `PhyML`.

Chapter 3

Pseudo-Likelihood Methods

3.1 Motivations

The **MLE**s of model parameters are obtained by maximizing the likelihood function with respect to the tree topology, branch lengths, and parameters in the substitution models. To find the **MLE** for the parameters $(\tau, \underline{t}, \Theta)$, first we can maximize the likelihood function for a given the tree topology, which is in a finite space. Then find the optimal tree topology that has the maximum likelihood.

Although ML methods have sound statistical foundations and perform well, the computing time quickly becomes unacceptable as m increases. For m species, the dimension for the alignment matrix Ω in Full likelihood methods is $m \times 4^m$. The computational time for all $p_i(\omega_i)$ is proportional to $O(4^m)$ ($m = 10, 4^m \approx 10^6; m = 100, 4^m \approx 10^{60}$). If we consider the K-subtree's marginal distributions instead, the time complexity decreases to constant $O(4^K) = O(1)$. Increasing K would improve accuracy but cost more time. It is interesting to explore the trade-off between speed and accuracy. When $K = 2$, it is the pairwise likelihood.

If we can work on the joint distribution of a subset of terminal nodes, it will be much easier to find the true tree. Chang and Hartigan (1991) find that, under mild conditions, the joint distributions of pairs of terminal nodes are sufficient to determine the tree topology (Chang, 1996).

The key idea of the Pseudo-likelihood method is considering the marginal distribution instead of the full joint distribution, which can save the computational time dramatically. In this chapter, we will state the Pseudo-likelihood function and its properties.

3.2 Pseudo-likelihood function

Recall the full likelihood function of a tree of m species, which can be written as:

$$l(\tau, \underline{t}, \underline{\theta}|\mathbf{X}) \propto \sum_{k=1}^{4^m} (Pr(\omega_k|\tau, \underline{t}, \underline{\theta}))^{x_k},$$

where $p(\omega_k)$ is the probability of pattern ω_k and x_k is the number of pattern ω_k found in the observational pattern matrix $\mathbf{X}$. It needs to calculate 4^m patterns of which the computing time increases dramatically. Instead, of full likelihood, the Pseudo-likelihood method considers the marginal distributions of K-subtree.

Now, let's state the K-subtree Pseudo-likelihood function at site k as:

$$P^*(\omega_k|\tau,\underline{t},\underline{\theta}) = \prod_{i_1=1}^{m-K+1} \dots \prod_{i_k=i_1+K-1}^{m} P(x^k_{i_1},...,x^k_{i_K}|\tau,\underline{t},\underline{\theta}), \tag{3.1}$$

where $x^k_{i_K}$ is the state of terminal node i_K at site k.

Then the log-likelihood function can be stated as:

$$l(\tau,\underline{t},\underline{\theta}|\mathbf{X}) = \sum_{k=1}^{n} \log(P^*(\omega_k|\tau,\underline{t},\underline{\theta})) = \sum_{i_1=1}^{m-K+1} \dots \sum_{i_k=i_1+K-1}^{m} \sum_{k=1}^{n} \log P(x^k_{i_1},...,x^k_{i_K}|\tau,\underline{t},\underline{\theta}). \tag{3.2}$$

Note that the K-subtree will hold only 4^K different state patterns instead of 4^m in the full likelihood function, which can save time to calculate the likelihood values. So, Equation 3.2 can be simplified as:

$$l(\tau,\underline{t},\underline{\theta}|\mathbf{X}) = \sum_{i_1=1}^{m-K+1} \dots \sum_{i_k=i_1+K-1}^{m} \sum_{k=1}^{4^K} y^k_{i_1,\dots,i_K} \log(P(x^k_{i_1},...,x^k_{i_K}|\tau,\underline{t},\underline{\theta})), \tag{3.3}$$

where $y^k_{i_1,\dots,i_K}$ is the observed number of pattern k of K-subtree containing nodes $(i_1,...,i_K)$.

3.2.1 Pairwise-likelihood function

Now, let's state the Pairwise log-likelihood function as:

$$l(\tau,\underline{t},\underline{\theta}|\mathbf{X}) = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \log[P(x_{ik},x_{jk}|t_{ij},\tau,\underline{\theta})],$$

where x_{ik} and x_{jk} are the states of terminal nodes i and j at site k, and t_{ij} is the sum of branches connecting node i and node j.

Note that there are only 16 different state patterns the pairwise nodes will hold instead of 4^m in the full likelihood function, which can save time to calculate the likelihood values. Then the Log function of Pairwise likelihood can be simplified as

$$l(\tau,\underline{t},\underline{\theta}|\mathbf{X}) = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sum_{k=1}^{16} y^{ij}_k \log p^{ij}_k(x_i,x_j|t_{ij},\tau,\underline{\theta}),$$

where p^{ij}_k is the probability of pattern k between sequence i and j, and y^{ij}_k is the frequency of pattern k between sequence i and sequence j.

3.2.2 Triple-likelihood function

Similar as pairwise-likelihood function, the Triple-likelihood function is defined as:

$$l(\tau, \underline{t}, \Theta|\mathbf{X}) = \sum_{h=1}^{m-2} \sum_{i=h+1}^{m-1} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \log[P(x_{hk}, x_{ik}, x_{jk}|t_{hij}, \tau, \underline{\theta})],$$

where x_i and x_j are the states of terminal nodes i and j, and t_{hij} is the branch length vector of the subtree containing nodes (h, i, j).

3.3 Maximum Pseudo-likelihood Estimate (MPLE)

Replacing the full likelihood function by Pseudo-likelihood function, the objective function of Maximum pseudo-likelihood Estimate could be stated as:

$$\hat{\Theta} = (\hat{\tau}, \hat{\underline{t}}, \hat{\underline{\theta}}) = \arg\max_{(\tau, \underline{t}, \underline{\theta})} l(\tau, \underline{t}, \underline{\theta}|\mathbf{X}).$$

To ensure the maximum estimator $\hat{\Theta}_n$ of random function l_n converges to the true parameter Θ_0 where the function l is maximized at, Identifiability and Convergence of the log-likelihood function are required.

3.4 Identifiability and Consistency of MPLE

In this section, first, we will show that the **MPLE** method is identifiable. Then, we will show that the **MPLE** method is consistent. Identifiability is required for consistency.

Under the assumptions that the evolution process is stationary and reversible, the distribution of the pairs of sites is enough to determine the full model including the tree and substitution model (Chang, 1996).

To simplify the Markov model, we state the assumptions as follows,
Assumption $\mathcal{H}$:

1. Identical Markov process across all sites and all sites are independent, which means site patterns X_k are independent and identically distributed;

2. Node distribution is stationary with π;

Note: Assumption $\mathcal{H}$ ensures the simplified class of the Markov model which is stationary and reversible with equilibrium measure π.

Chang showed that under assumption $\mathcal{H}$ the pairwise comparisons of the terminal nodes were enough to ensure the identifiability of the full model including the topology, branch lengths, and the transition matrices (Chang, 1996).

Theorem 3.4.1 *For stationary and time reversible substitution models defined by* $\Theta = (\tau, \underline{t}, \underline{\theta})$*, pairwise likelihood is identifiable.*

$$l_p(\Theta_i) \neq l_p(\Theta_j), \textit{for } \forall \Theta_i \neq \Theta_j.$$

The proof of theorem 3.4.1 can be derived from Chang's result: For the simplified class of Markov model that is stationary and reversible with equilibrium measure π, the full model is uniquely determined by the distribution of the pairs of sites.

Definition 3.4.1 *For a tree* T_0*, the full set of* K*-subtrees of* T_0 *is the set of all possible subtrees with* K *tips pruned from tree* T_0*.*

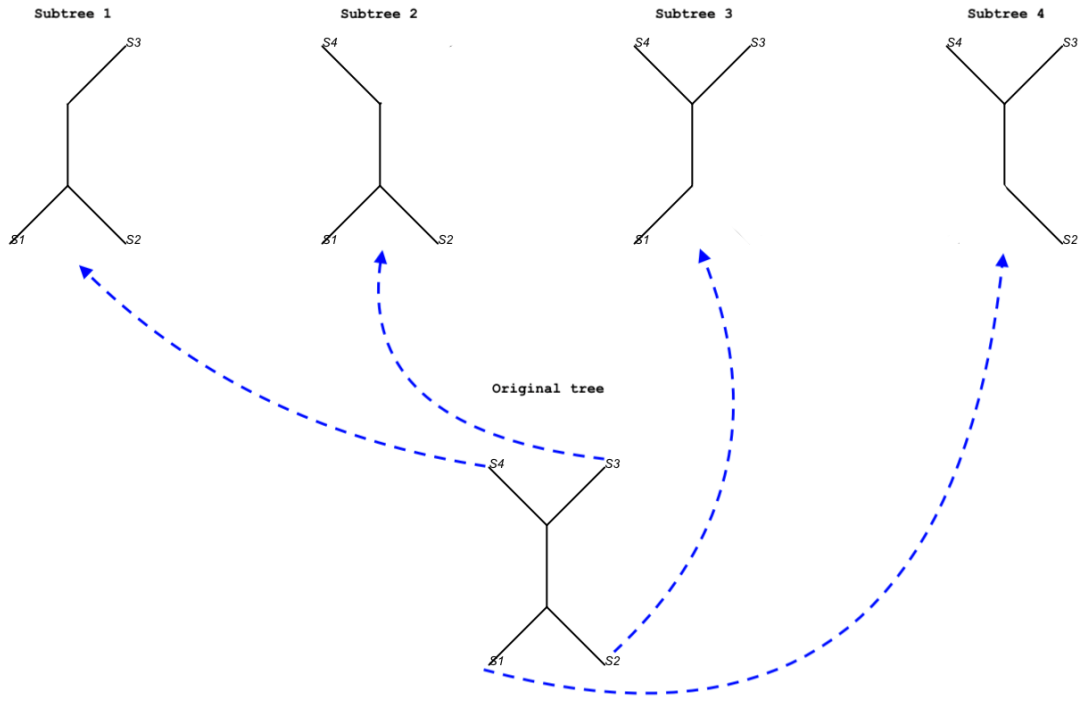

Figure 3.1: An example of full set of subtrees

Figure 3.1 shows the full set of triple-subtree of a 4-taxon tree. There are $\binom{4}{3} = 4$ triple-wise subtrees for a 4-taxon tree and subtree 1 contains tips S_1, S_2 and S_3.

Theorem 3.4.2 *There is a one-to-one mapping from the full set of subtrees to the true tree.*

Proof of Theorem 3.4.2: First, for a tree T, there is only one full set of subtrees. To construct the full set of subtrees, delete other taxon and branches not connecting taxa of interest. On the other hand, one

needs to show that the full set of subtrees would reconstruct only one tree containing all taxon. Assuming that the full set of subtrees are pruned from a tree T^*. So, there is at least one tree that can be reconstructed from the full set of subtrees. Define the subtree as $T^{(1,...,K)}$ that contains taxa $1, ..., K$ with $T^{(1,...,K)} \in T^*$. All the taxa and branches of $T^{(1,...,K)}$ are nested in tree T^*. Subtree $T^{(1,...,K)}$ which can be seen as a full tree with K taxa is identifiability. In addition, for each subtree $T^{(1,...,K)}$, there is only one subtree the same as it in the original full tree, which means the tree can be uniquely determined by the full set of subtrees. Thus, the map between the full set of subtrees and the original tree is one-to-one.

Then we will use Theorem 3.4.2 to prove Theorem 3.4.1.

Proof of Theorem 3.4.1: For a tree containing m taxa, assuming the full set of $\binom{m}{2}$ pairwise subtrees as $\{T^{(i,j)}\}_{1\leq i<j\leq m}$, where $T^{(i,j)}$ is the subtree of tree T containing only taxa S_i and taxa S_j. Let's assume $P((X_i, X_j)|T^{(i,j)}) = P((X_i, X_j)|T)$ be the marginal distribution of taxa S_i and S_j, where X_i, X_j are the random variable indicating the DNA states for taxon S_i and taxon S_j. Theorem 3.4.2 shows that the full set of pairwise subtrees $\{T^{(i,j)}\}_{1\leq i<j\leq m}$ can be uniquely mapped to a tree which is the same of the original tree T. There is an one-to-one map from set $\{P((X_i, X_j)|T^{(i,j)})\}_{1\leq i<j\leq m}$ to $P(X_1, ..., X_m|T)$. And the full likelihood function $P(X_1, ..., X_m|T)$ is identifiable. T contains the parameters of tree topology τ, branch length $\underline{t}$ and the substitution parameters $\underline{\theta}$. So, the pairwise likelihood is identifiable.

3.4.1 Consistency of Pairwise-likelihood method

Theorem 3.4.3 *For stationary and time reversible substitution models, Maximum pairwise likelihood estimate* $\hat{\Theta}_n = (\hat{\tau_n}, \hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ *of the true model parameter* $\Theta_0 = (\tau_0, \underline{t}_0, \underline{\theta}_0)$ *is consistent. That is, as the sequence length* n *goes to infinity,* $\hat{\tau_n} = \tau_0$ *and* $(\hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ *converges to* $(\underline{t}_0, \underline{\theta}_0)$*, with probability* 1.

Holder and Steel showed that the maximum Pairwise-likelihood estimates $\hat{T}_k$ and $\hat{d}_k$ of the true tree T_0 and edge lengths d_0 are consistent (Holder and Steel, 2011), that is, as k goes to infinity, $\hat{T}_k = T_0$ and $\hat{d}_k$ converges to d_0, with probability 1. The proof of Theorem 3.4.3 is similar to Steel's proof of the consistency of the tree topology. However, we used a different and easy approach to prove the consistency of the maximum pairwise likelihood estimates.

The proof of Theorem 3.4.3: Let's assume $Y_{ij}^s = (X_i^s, X_j^s)$ is the pairwise site pattern for taxon i and taxon j at site s where $i < j$ and $i, j \in \mathcal{L} = (1, ..., m)$. We have $Y_{ij}^s = (X_i^s, X_j^s)$ is identically and independently distributed, because (X_i^s, X_j^s) is independent with $(X_i^{s^*}, X_j^{s^*})$ where site s is independent with s^*. Define the marginal pairwise probability distribution as,

$$P_\Theta(Y = y) = \prod_{(i,j)\in\mathcal{L}^{(2)}} P((X_i, X_j) = y_{ij}|\Theta_{ij})),$$

where $\Theta_{ij} = (\tau, \underline{t}_{ij}, \underline{\theta})$ and $\underline{t}_{ij}$ is the path from tip i to tip j in tree τ.

The pairwise log-likelihood function can be written as,

$$l_p(\Theta) = l(\tau, \underline{t}, \underline{\theta}|\mathbf{X}) = \sum_{s=1}^{n} \log[P_\Theta(Y_s = y)],$$

where s is the site index from 1 to n. Theorem3.4.1 states the identifiability of $l_p(\Theta)$. We can consider Θ_{ij} as the parameters defined on subtree $T^{(ij)}$. To maximize $l_p(\Theta)$, we first maximize the likelihood of each pairwise subtree which gives an estimate $\{\hat{\Theta}^{(ij)}\}_{1\leq i<j\leq m}$ for the full set of subtrees $\{\hat{T}^{(ij)}\}_{1\leq i<j\leq m}$.

Because the consistency of MLE $\hat{\Theta}_n^{(ij)}$, we have, as n goes to infinity, for $1 \leq i < j \leq m$ the following limit holds with probability 1: $|\hat{\Theta}_n^{(ij)} - \Theta^{(ij)}| \to 0$. And let ϵ be the smallest value that makes $|\hat{\Theta}_n^{(ij)} - \Theta^{(ij)}| \to 0$ for $1 \leq i < j \leq m$. which makes $|\hat{\Theta}_n - \Theta| < \epsilon$. Using the full set of subtrees, one can construct an unique full tree $\hat{T}_n$ defined by $\hat{\Theta}_n$ which contains m taxa the same as taxa in full set of subtrees $\{\hat{T}^{(ij)}\}_{1\leq i<j\leq m}$.

Thus, we have as n goes to infinity, $\hat{\tau_n} = \tau_0$ and $(\hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ converges to $(\underline{t}_0, \underline{\theta}_0)$, with probability 1.

3.4.2 More General case

Chang showed that the pairwise comparisons of the terminals were not enough to ensure the identifiability of the transition matrices (Chang, 1996), which means it can not reconstruct the full model only based on the pairwise distributions of the terminals. And He also showed that the full model could be reconstructed from the triples of the sites under the assumption H^*. And Baake showed that pairwise distributions of the terminal nodes can not guarantee the identifiability of the topology when a substitution rate factor varies across sites (Baake, 1998).

Thus, we will explore how to ensure the identifiability of the full model based on triples of the terminals. For the general Markov model, Chang showed that the distribution of triples of sites is required to ensure the identifiability of the full model.

Triple-likelihood function

$$l(T, \underline{t}) = \sum_{h=1}^{m-2} \sum_{i=h+1}^{m-1} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \log[P(x_{hk}, x_{ik}, x_{jk}|t_{hij}, \tau, \underline{\theta})],$$

where x_hk, x_ik and x_jk are the states of terminal nodes h, i and j at site k, and t_{hij} is the vector of edge lengths among nodes h, i and j.

Allman and Rhodes established the first proof of identifiability of a phylogenetic model with a continuous distribution of rates and so methods such as maximum likelihood will be statistically consistent (Allman et al., 2008). The result shows that just 3–way sequence comparisons are sufficient to identify the shape parameter. This suggests that it may be possible to develop statistically consistent but fast modifications of distance-based tree reconstruction methods (such as neighbor-joining) that some allow triple-wise calculations. The k-state $GTR + \Gamma$ model is identifiable from the joint distributions of triples of taxa for generic parameters on any tree with three or more taxa. Moreover, when $k = 4$, the model is identifiable for all parameters.

Theorem 3.4.4 *For stationary and time reversible substitution models defined by $\Theta = (\tau, \underline{t}, \underline{\theta})$, Triple-likelihood is identifiable for the full model.*

$$l_p(\Theta_i) \neq l_p(\Theta_j), \textit{for} \, \forall \Theta_i \neq \Theta_j.$$

The proof of Theorem 3.4.4 is similar as that for pairwise likelihood. Instead of considering pairwise subtrees, triple-wise subtrees are used. For a tree containing m taxa, assuming the full set of $\binom{m}{3}$ subtrees as $\{T^{(h,i,j)}\}_{1\leq h<i<j\leq m}$, where $T^{(h,i,j)}$ is the subtree of tree T containing only taxa S_h, S_i and taxa S_j. Let's assume $P((X_h, X_i, X_j)|T^{(h,i,j)}) = P((X_h, X_i, X_j)|T)$ be the marginal distribution of taxa S_h, S_i and S_j, where X_h, X_i, X_j are the random variable indicating the (DNA) states for taxon S_h, S_i and taxon S_j. Theorem 3.4.2 shows that the full set of subtrees can be uniquely mapped to the original tree T. There is an one-to-one map from set $\{P((X_h, X_i, X_j)|T^{(h,i,j)})\}_{1\leq h<i<j\leq m}$ to $P(X_1, ..., X_m|T)$. Thus, the triple-wise likelihood is identifiable.

Theorem 3.4.5 *For stationary and time reversible substitution models, Maximum triple-likelihood estimate $\hat{\Theta}_n = (\hat{\tau_n}, \hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ is consistent. That is, as sequence length n goes to infinity, $\hat{\tau_n} = \tau_0$ and $(\hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ converges to $(\underline{t}_0, \underline{\theta}_0)$, with probability* 1.

The proof of Theorem 3.4.5: Let's assume $Y^s_{hij} = (X^s_h, X^s_i, X^s_j)$ is the triple-wise site pattern for taxa h, i and j at site s, where $h < i < j$ and $h, i, j \in \mathcal{L} = (1, ..., m)$. We have $Y^s_{hij} = (X^s_h, X^s_i, X^s_j)$ is identically and independently distributed. Define the triple-marginal distribution as,

$$P_\Theta(Y = y) = \prod_{(h,i,j)\in\mathcal{L}^{(3)}} P((X_h, X_i, X_j) = y_{hij}|\Theta_{ij})),$$

where $\Theta_{hij} = (\tau, \underline{t}_{hij}, \underline{\theta})$ and $\underline{t}_{hij}$ is the subset of branch $\underline{t}$ that connects tips h, i and j.

The triple-wise log-likelihood function is,

$$l_3(\Theta) = \sum_{s=1}^{n} \log[P_\Theta(Y_s = y)],$$

where s is the site index from 1 to n. Theorem3.4.4 states the identifiability of triple-wise log-likelihood function $l_3(\Theta)$. We can consider Θ_{hij} as the parameters defined on subtree $T^{(hij)}$. To maximize $l_3(\Theta)$, we first maximize the likelihoods of each triple-wise subtree which gives the estimate of the full set of subtrees $\{\hat{T}^{(hij)}\}_{1\leq h<i<j\leq m}$. Using the full set of subtrees, we can construct one and only one full tree $\hat{T}_n = \hat{\Theta}_n$.

Because of the consistency of the estimate $\hat{\Theta}^{(hij)}_n$ for subtrees $\{\hat{T}^{(hij)}\}_{1\leq h<i<j\leq m}$, as n goes to infinity, for $1 \leq h < i < j \leq m$ the following limit holds with probability 1: $|\hat{\Theta}^{(hij)}_n - \Theta^{(hij)}| \to 0$. And let ϵ be the smallest value that makes $|\hat{\Theta}^{(hij)}_n - \Theta^{(hij)}| \to 0$ for $1 \leq h < i < j \leq m$. which makes $|\hat{\Theta}_n - \Theta| < \epsilon$. Thus, we have as n goes to infinity, $\hat{\tau_n} = \tau_0$ and $(\hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ converges to $(\underline{t}_0, \underline{\theta}_0)$, with probability 1.

The identifiability and consistency can be extended to K-subtree as follows,

Theorem 3.4.6 *For stationary and time reversible substitution models defined by $\Theta = (\tau, \underline{t}, \underline{\theta})$, K-subtree likelihood is identifiable for the full model.*

$$l_p(\Theta_i) \neq l_p(\Theta_j), for\, \forall \Theta_i \neq \Theta_j.$$

The proof of Theorem 3.4.6: For a tree containing m taxa, assuming the full set of $M = \binom{m}{K}$ subtrees which contains K ordered taxa. Let's assume $P(\underline{X}_i^K | T_i^{(K)}) = P(\underline{X_i^K} | T)$ be the marginal distribution of K taxa, where $\underline{X}_i^K$ is a vector of the K random variables indicating the states for the K taxa of the i^{th} subtree $i = 1, ..., M$. Theorem 3.4.2 shows that the full set of subtrees can be uniquely mapped to the original tree T. There is an one-to-one map from set $\{P(\underline{X}_i^K | T_i^{(K)})\}_{i=1,...,M}$ to $P(X_1, ..., X_m | T)$. Thus, the K-subtree likelihood is identifiable.

Theorem 3.4.7 *For stationary and time reversible substitution models, Maximum K-subtree pseudo-likelihood estimate $\hat{\Theta}_n = (\hat{\tau_n}, \hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ is consistent. That is, as sequence length n goes to infinity, $\hat{\tau_n} = \tau_0$ and $(\hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ converges to $(\underline{t}_0, \underline{\theta}_0)$, with probability* 1.

The proof of Theorem 3.4.7: Let's assume $Y^s_{i_1,...i_K} = (X^s_{i_1}, ..., X^s_{i_K})$ is the site pattern for K-subtree containing taxa $X_{i_1}, ..., X_{i_K}$ at site s, where $i_1 < ... < i_K$ and $i_1, ..., i_K \in \mathcal{L} = (1, ..., m)$. We have $Y^s_{i_1,...i_K} = (X^s_{i_1}, ..., X^s_{i_K})$ is identically and independently distributed. Define the K-subtree marginal distribution as,

$$P_\Theta(Y = y) = \prod_{(i_1,...,i_K) \in \mathcal{L}^{(K)}} P((X_{i_1}, ..., X_{i_K}) = y_{i_1,...i_K} | \Theta_{i_1,...,i_K})),$$

where $\Theta_{i_1,...,i_K} = (\tau, \underline{t}_{i_1,...,i_K}, \underline{\theta})$ and $\underline{t}_{i_1,...,i_K}$ is the subset of branch $\underline{t}$ that connects tips $(i_1, ..., i_K) \in \mathcal{L}^{(3)}$.

The pseudo K-subtree log-likelihood function is defined as,

$$l_K(\Theta) = \sum_{s=1}^{n} \log[P_\Theta(Y_s = y)],$$

where site index $s = 1, ..., n$. Theorem3.4.6 states the identifiability of log-likelihood function $l_K(\Theta)$. $\Theta_{i_1,...,i_K}$ includes the parameters defined on the K-subtree $T^{(i_1,...,i_K)}$. To maximize $l_K(\Theta)$, let's first maximize the likelihood of each K-subtree, which gives an estimate $\{\hat{\Theta}^{(i_1,...,i_K)}\}_{1 \leq i_1 < ... < i_K \leq m}$ of the corresponding $\{\Theta^{(i_1,...,i_K)}\}_{1 \leq i_1 < ... < i_K \leq m}$ from the K-subtree $T^{(i_1,...,i_K)}$. And each K-subtree can be treated as a full tree which ensures the identifiability and consistency of MLE $\{\hat{\Theta}_n^{(i_1,...,i_K)}\}_{1 \leq i_1 < ... < i_K \leq m}$. For $1 \leq i_1 < ... < i_K \leq m$, we have:

$$\lim_{n \to \infty} P(|\hat{\Theta}_n^{(i_1,...,i_K)} - \Theta^{(i_1,...,i_K)}| < \epsilon_{i_1,...,i_K}) = 1.$$

An estimate of the full set of subtrees can be constructed from all $\binom{m}{K}$ MLE's of all K-subtrees. Using the full set of subtree estimate, we can construct a full tree $\hat{T}_n$ which is defined by $\hat{\Theta}_n$. From , the full set of K-subtrees and the full tree T is one-to-one mapping, which means there is one and only one tree that

can give the same set of MLE's. And let $\epsilon = min\{\{\epsilon_{i_1,\ldots,i_K}\}_{1\le i_1<\ldots<i_K\le m}\}$, we have

$$\lim_{n\to\infty} P(|\hat{\Theta}_n - \Theta| < \epsilon) = 1.$$

That is, as n goes to infinity, $\hat{\tau_n} = \tau_0$ and $(\hat{\underline{t}}_n, \hat{\underline{\theta}}_n)$ converges to $(\underline{t}_0, \underline{\theta}_0)$, with probability 1.

3.4.3 Simple example for identifiability of pairwise likelihood function

We can derive the Pseudo-likelihood function as

$$PL(\tau, \underline{t}, \underline{\theta}|\mathbf{X}) \propto \prod_{i=1}^{r} (P^*(\omega_i|\tau, \underline{t}, \underline{\theta}))^{x_i},$$

where x_i is the number of pattern ω_i found in the observational pattern matrix $\mathbf{X}$,

$$P^*(\omega_1|\tau, \underline{t}, \underline{\theta}) = p_{AA}(d_{12})p_{AA}(d_{13})p_{AA}(d_{14})p_{AA}(d_{23})p_{AA}(d_{24})p_{AA}(d_{34})$$

$$\ldots$$

$$P^*(\omega_r|\tau, \underline{t}, \underline{\theta}) = p_{TT}(d_{12})p_{TT}(d_{13})p_{TT}(d_{14})p_{TT}(d_{23})p_{TT}(d_{24})p_{TT}(d_{34}).$$

Important facts:

$$\underline{d} = \begin{bmatrix} d_{12} \\ d_{13} \\ d_{14} \\ d_{23} \\ d_{24} \\ d_{34} \end{bmatrix} = \begin{bmatrix} t_1 + t_2 \\ t_0 + t_1 + t_3 \\ t_0 + t_1 + t_4 \\ t_0 + t_2 + t_3 \\ t_0 + t_2 + t_4 \\ t_3 + t_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} * \begin{bmatrix} t_0 \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{bmatrix} = M * \underline{t}$$

For general cases, the mapping matrix M is full rank, which means it is one-to-one mapping from $\underline{d}$ to $\underline{t}$ and pairwise distance branch vector $\underline{d}$ will define the same tree as $\underline{t}$ does. Thus the pairwise distances of the terminal nodes can uniquely determine the tree topology.

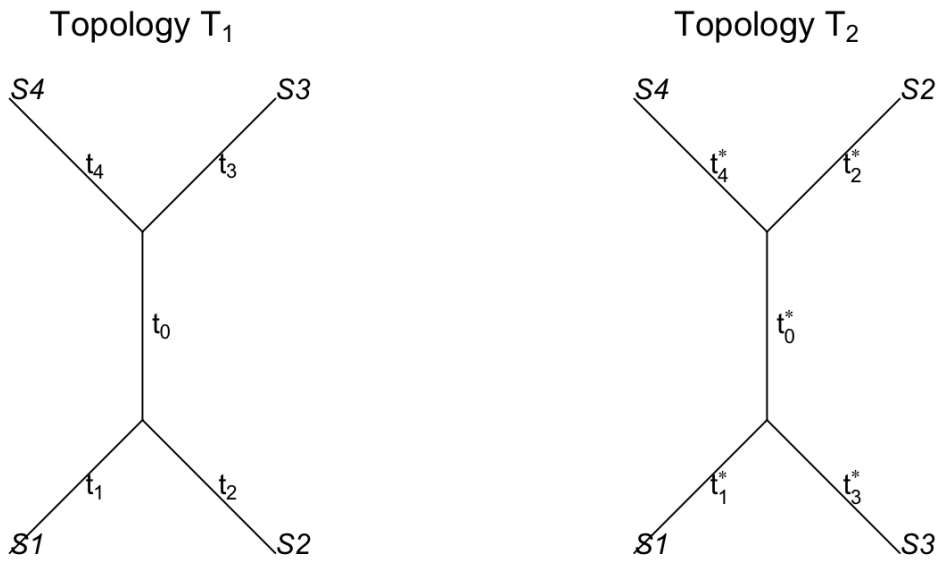

Figure 3.2: Two topologies of a 4-taxon unrooted tree.

Another point of view to see the identifiability of the tree topology is assuming there is another tree T_2 which is different from T_1 gives the same pairwise-likelihood function. Let's switch the terminal nodes S_2 and S_3 as T_2, and assume T_1 and T_2 have the same likelihood function which gives the following equations:

$$\begin{bmatrix} t_1 + t_2 \\ t_0 + t_1 + t_3 \\ t_0 + t_1 + t_4 \\ t_0 + t_2 + t_3 \\ t_0 + t_2 + t_4 \\ t_3 + t_4 \end{bmatrix} = \begin{bmatrix} t_0^* + t_1^* + t_2^* \\ t_1^* + t_3^* \\ t_0^* + t_1^* + t_4^* \\ t_0^* + t_2^* + t_3^* \\ t_2^* + t_4^* \\ t_0^* + t_3^* + t_4^* \end{bmatrix}$$

The only way to fit all these equations is $t_0^* = 0$ and $t_0 = 0$, which also makes T_1 and T_2 the same topology. In addition, if we assumed there was no branch with 0 length, these equations would not fit, which means no topology T_2 giving the same pairwise likelihood function. Now we have the conclusion that the pairwise distributions of the terminal nodes can uniquely determine the tree topology and branch lengths.

Chapter 4

Package mple

This chapter describes the R package called `mple` which will implement the **MPLE** method. It includes functions to calculate the maximum pseudo-likelihood function of a given tree and the topology search in tree space. This chapter states the details of the algorithm **MPLE**.

4.1 Data Processing

Since **MPLE** approach calculates the pairwise likelihood values of terminal nodes i and j for $1 \leq j < i \leq m$, it is helpful to construct the input sequence data as a matrix $Y_{16\times\binom{m}{2}} = \{y_{ij}\}$ where $y_{ij} = [y_{ij}^{AA}, \ldots, y_{ij}^{TT}]^T$ counts the observed number of all possible patterns for nodes i and j. For DNA alignment, there are only 16 possible patterns for two species.

$$Y_{16\times\binom{m}{2}} = \begin{pmatrix} y_{12}^{AA} & y_{13}^{AA} & \cdots & y_{(m-1)m}^{AA} \\ y_{12}^{AC} & y_{13}^{AC} & \cdots & y_{(m-1)m}^{AC} \\ \vdots & \vdots & \vdots & \vdots \\ y_{12}^{TT} & y_{13}^{TT} & \cdots & y_{(m-1)m}^{TT} \end{pmatrix}.$$

When pairwise likelihood function is applied on DNA data, function `store_seq()` in `mple` would transform an object of class *phyDat* to a $16 \times \binom{m}{2}$ matrix.

4.2 Calculating Pseudo-likelihood Value

Given the tree topology τ and the initialized values of $\underline{t}$, we can calculate the pairwise distance matrix $T = \{t_{ij}\}$ where $t_{ij} = t_{ji}$ is the sum of the branch lengths connecting node i and node j.

$$T_{1\times\binom{m}{2}} = [t_{12}, t_{13}, \ldots, t_{(m-1)m}]$$

Given the initialized values of the parameters Θ in substitution model, we can use Equation 2.5 to get the transition matrix $P(t)$.

Given the initialized values of the parameters Θ in substitution model and pairwise distance matrix T, we can get the transition probabilities $P_{16\times\binom{m}{2}} = \{p_{ij}\}$ where with $p_{ij} = [p^{AA}(t_{ij}), \ldots, p^{TT}(t_{ij})]^T$.

$$P_{16\times\binom{m}{2}} = \begin{pmatrix} p^{AA}(t_{12}) & p^{AA}(t_{13}) & \ldots & p^{AA}(t_{(m-1)m}) \\ p^{AC}(t_{12}) & p^{AC}(t_{13}) & \ldots & p^{AC}(t_{(m-1)m}) \\ \vdots & \vdots & \vdots & \vdots \\ p^{TT}(t_{12}) & p^{TT}(t_{13}) & \ldots & p^{TT}(t_{(m-1)m}) \end{pmatrix}.$$

The value of the target function which is the pairwise likelihood function is calculated by

$$l_p(\Theta, \underline{t}|\tau) = \mathrm{tr}(Y^T \times log(P)).$$

In package `mple`, functions `pml_jc()` and `pml_gtr()` are used to calculate the pairwise likelihood value of input alignments given a tree topology, branch length $\underline{t}$ and substitution models JC and GTR respectively.

4.3 Optimizing Likelihood Function Given Tree Topology

Given the tree topology τ, we can optimize the the branch length $\underline{t}$ and the parameters Θ in substitution model by maximizing $l_p(\Theta, \underline{t}|\tau)$. This will give us the optimal solution for this tree topology, then one can change the topology and repeat the process.

4.4 Searching in Tree Space

The objective of `MPLE` is to find the tree topology that maximizes the pseudo-likelihood score, given the alignment data and a substitution model of evolution. A simple strategy for searching the tree space is the exhaustive search, in which all possible unrooted tree topologies are enumerated. And, for each tree topology, the branch lengths and substitution parameters are optimized to maximize the pseudo-likelihood. Finally, the tree topology with the highest pseudo-likelihood value is the optimal topology and the corresponding branch lengths and substitution parameters are the optimal branch lengths and substitution parameters. However, It is computationally not feasible for trees with a large number of taxa, because the number of possible topologies increases exponentially with the number of taxa m. It was shown that finding the optimal tree topology is NP-hard (Roch, 2006).

Almost all of the current algorithms used to estimate phylogenetic trees are based on a heuristic search on tree space. The idea is generating a starting tree, changing the topology using some Tree-arrangement algorithms, and updating the estimated tree based on likelihood values. There are mainly three ways to construct a comprehensive starting tree: random topology, neighbor-joining (NJ), and parsimony.

Neighbor-joining (NJ) and tree-arrangement algorithm *SPR* are applied in the algorithm `MPLE`. To avoid getting stuck on local optimum, random staring trees are also added at the beginning.

In phylogenetics, finding the correct tree topology is much more important than optimizing the branch lengths and substitution parameters. So, the branch length and substitution parameters will not be optimized for every searched tree topology. They will be updated with a probability that is generated from a binomial distribution with parameter α.

Finally, the Algorithm `MPLE` can be addressed as follows,

	Algorithm `MPLE`
Step 1	Store input data as $Y_{16\times\binom{m}{2}} = \{y_{ij}\}$.
Step 2	Starting from NJ tree, randomly select $start\_n$ `SPR` trees of it.
Step 3	For each starting tree: **3.1**: Maximize $l_p(\underline{t}, \underline{\theta}\|\tau^{(k)}) = \mathrm{tr}(Y^T log(P))$ with respect to $(\underline{t}, \underline{\theta})$. **3.2**: Change topology to $\tau^{(k+1)}$ using `SPR`, and optimize $(\hat{\underline{t}}, \hat{\underline{\theta}})$ with probability α. Update $(\hat{\tau}, \hat{\underline{t}}, \hat{\underline{\theta}})$ and $l_p^{(k+1)}$, if $l_p^{(k+1)} > l_p^{(k)}$. **3.3**: Repeat *3.2* until reaching maximum number of topologies.
Step 4	Return the optimal estimate $(\hat{\tau}, \hat{\underline{t}}, \hat{\underline{\theta}})$.

Chapter 5

Comparison of **MLE** and **PMLE**

This chapter compares some basic properties of **MLE** and **PMLE** including time complexity and efficiency, which shows that **PMLE** is promising.

5.1 Time Complexity

Finding the optimal tree is exceptionally difficult because search of the tree space and evaluation of each tree requires a considerable amount of calculations (Bryant et al., 2007). For example, there are more than 10^{21} possible tree topologies for a set of 20 taxa and there are over 10^{13} different patterns needed to calculate the full likelihood for each tree topology.

Table 5.1: Time Complexity for **MLE** and **MPLE**

Methods	Topology	Likelihood
MLE	$(2m-3)!!$	$O(m4^m)$
MPLE	$(2m-3)!!$	$O(m^2)$

In comparison, the calculation complexity of pseudo-likelihood functions is substantially reduced, especially when the sequences have a large number of nucleotides. For methods based on the full likelihood, the total number of patterns in the alignment $X_{m\times n}$ is at most $max\{4^m, n\}$, where m is the number of taxa and n is the length of the alignment (i.e., the number of sites in the alignment). Table 5.1 shows that the time complexity for calculating the full likelihood function given the tree topology is $O(m4^m)$. However, for the pairwise likelihood function, the time complicity is $O(m^2)$, which is independent of the sequence length n and much smaller than $O(4^m)$ when m is large.

Table 5.2 shows the time complexity for an exhaustive search for the optimal tree topology and calculating likelihood functions when the number of taxa (m) increases from 10 to 100. The size of the tree space increases dramatically as the number of taxa (m) increases. Calculating the pairwise likelihood function is much simpler than calculating the full likelihood function. For example, when $m = 100$ the full likelihood function requires 10^58 times more calculations than the pairwise likelihood function does.

Table 5.2: Time complexity for computing likelihood function

m	Topology	Full likelihood	Pairwise likelihood
10	3.45×10^{7}	1.05×10^{7}	1.00×10^{2}
20	8.20×10^{21}	2.20×10^{13}	4.00×10^{2}
50	2.75×10^{76}	6.34×10^{31}	2.50×10^{3}
100	3.35×10^{184}	1.61×10^{62}	1.00×10^{4}

In addition, it is straightforward to parallelize the calculation of the pairwise likelihood function, which can further reduce the computational time to a reasonable range.

5.1.1 Simulation Results for Time Complexity

Theoretically, calculating the pseudo-likelihood function is much faster than calculating the full likelihood function given the phylogenetic tree $(\tau, \underline{t})$ where τ is the tree topology and $\underline{t}$ is the vector of branch lengths. In practice, the speed of calculating the full likelihood function depends on the observed number of patterns, which is $\leq max\{n, 4^m\}$. In contrast, the speed of pseudo-likelihood methods only depends on the size (K) of the sub-tree, which is much smaller than 4^m and is free from sequence length n. So, the maximum pseudo-likelihood estimate (MPLE) method has computational advantages over the maximum full-likelihood estimate (MLE) method when n and m are large.

In Table 5.3, the internal branches were set to $t_0 = 0.01$ and external branches were set to $t_1 = 0.05$. By increasing the sequence length n or the branch length $\underline{t}$, more substitutions and more patterns would be generated. Thus, in Table 5.4 we set the internal branches to $t_0 = 0.1$ and external branches to $t_1 = 0.5$.

When $n = 500$, the number of observed patterns generated from tree with $t_0 = 0.1, t_1 = 0.5$ is 4.97×10^2 which is more close to $n = 500$ than 2.35×10^2 that generated from tree with $t_0 = 0.01, t_1 = 0.05$.

Table 5.3 and Table 5.4 show that the speed of calculating pairwise likelihood function and full likelihood function given the phylogenetic tree $(\tau, \underline{t})$ under the JC substitution model.

Calculating the pseudo-likelihood function is much faster than calculating the full likelihood function especially when the observed number of patterns is close to $max\{n, 4^m\}$. For example, in Table 5.4 when $m = 30, n = 10^5$, the observed number of patterns equals to n and the time needed to calculate the full likelihood function is 1.38×10^{-1} which is about 20 times slower than that for the pairwise likelihood function.

5.1.2 Empirical data

The GenBank is the NIH genetic sequence database (Benson et al., 2010) which is designed to provide the most up-to-date and comprehensive DNA sequence data for research. The real-world datasets used in this paper are shown in Table 5.5. The *primates* dataset has 4 sequences (Human, Chimp, Gorilla, and Orang),

Table 5.3: Simulation for computing likelihood function given tree with $t_0 = 0.01, t_1 = 0.05$

Species	Length	Pattern_max	Pattern_obs	Model	Time_AVG
10	500	1.05×10^6	1.58×10^2	MLE	1.01×10^{-3}
10	500	1.05×10^6	1.58×10^2	MPLE	7.01×10^{-4}
20	500	1.10×10^{12}	3.03×10^2	MLE	1.14×10^{-3}
20	500	1.10×10^{12}	3.03×10^2	MPLE	5.38×10^{-2}
30	500	1.15×10^{18}	3.92×10^2	MLE	1.31×10^{-3}
30	500	1.15×10^{18}	3.92×10^2	MPLE	5.49×10^{-3}
10	1×10^3	1.05×10^6	2.48×10^2	MLE	1.05×10^{-3}
10	1×10^3	1.05×10^6	2.48×10^2	MPLE	7.05×10^{-4}
20	1×10^3	1.10×10^{12}	5.36×10^2	MLE	1.20×10^{-3}
20	1×10^3	1.10×10^{12}	5.36×10^2	MPLE	2.47×10^{-3}
30	1×10^3	1.15×10^{18}	7.46×10^2	MLE	1.69×10^{-3}
30	1×10^3	1.15×10^{18}	7.46×10^2	MPLE	5.58×10^{-3}
10	1×10^4	1.05×10^6	1.07×10^3	MLE	1.31×10^{-3}
10	1×10^4	1.05×10^6	1.07×10^3	MPLE	7.13×10^{-4}
20	1×10^4	1.10×10^{12}	3.35×10^3	MLE	2.94×10^{-3}
20	1×10^4	1.10×10^{12}	3.35×10^3	MPLE	2.50×10^{-3}
30	1×10^4	1.15×10^{18}	5.61×10^3	MLE	6.60×10^{-3}
30	1×10^4	1.15×10^{18}	5.61×10^3	MPLE	5.96×10^{-3}
10	1×10^5	1.05×10^6	4.45×10^3	MLE	2.26×10^{-3}
10	1×10^5	1.05×10^6	4.45×10^3	MPLE	7.32×10^{-4}
20	1×10^5	1.10×10^{12}	2.10×10^4	MLE	1.45×10^{-2}
20	1×10^5	1.10×10^{12}	2.10×10^4	MPLE	2.68×10^{-3}
30	1×10^5	1.15×10^{18}	4.27×10^4	MLE	4.39×10^{-2}
30	1×10^5	1.15×10^{18}	4.27×10^4	MPLE	5.92×10^{-3}

which contains $25,785$ characters and 213 different site patterns (Shaw et al., 2013). *yeast* is 8-species data set with DNA sequences of $127,026$ base pairs collected by some researchers (Rokas et al., 2003).

Table 5.5: Real-world DNA alignment data sets.

Name	Source	Taxa	Length	Patterns	Reference
primates	STRAW	4	$25,785$	213	species of Human, Chimp, Gorilla, and Orang from STRAW (Shaw et al., 2013)
yeast	`phangorn`	8	$127,026$	$8,899$	species of yeast (Scer, Spar, Smik, Skud, Sbay, Scas, Sklu, Calb) (Rokas et al., 2003)

Table 5.4: Simulation for computing likelihood function given tree with $t_0 = 0.1, t_1 = 0.5$

Species	Length	Pattern_max	Pattern_obs	Model	Time_AVG
10	500	1.05×10^6	4.97×10^2	MLE	1.27×10^{-3}
10	500	1.05×10^6	4.97×10^2	MPLE	7.81×10^{-4}
20	500	1.10×10^{12}	5.00×10^2	MLE	1.37×10^{-3}
20	500	1.10×10^{12}	5.00×10^2	MPLE	2.53×10^{-3}
30	500	1.15×10^{18}	5.00×10^2	MLE	1.55×10^{-3}
30	500	1.15×10^{18}	5.00×10^2	MPLE	5.85×10^{-3}
10	1×10^3	1.05×10^6	9.90×10^2	MLE	1.27×10^{-3}
10	1×10^3	1.05×10^6	9.90×10^2	MPLE	7.32×10^{-4}
20	1×10^3	1.10×10^{12}	1.00×10^3	MLE	1.66×10^{-3}
20	1×10^3	1.10×10^{12}	1.00×10^3	MPLE	2.61×10^{-3}
30	1×10^3	1.15×10^{18}	1.00×10^3	MLE	1.98×10^{-3}
30	1×10^3	1.15×10^{18}	1.00×10^3	MPLE	5.71×10^{-3}
10	1×10^4	1.05×10^6	9.30×10^3	MLE	4.06×10^{-3}
10	1×10^4	1.05×10^6	9.30×10^3	MPLE	7.24×10^{-4}
20	1×10^4	1.10×10^{12}	1.00×10^4	MLE	7.42×10^{-3}
20	1×10^4	1.10×10^{12}	1.00×10^4	MPLE	2.48×10^{-3}
30	1×10^4	1.15×10^{18}	1.00×10^4	MLE	1.07×10^{-2}
30	1×10^4	1.15×10^{18}	1.00×10^4	MPLE	5.68×10^{-3}
10	1×10^5	1.05×10^6	7.47×10^4	MLE	2.82×10^{-2}
10	1×10^5	1.05×10^6	7.47×10^4	MPLE	7.44×10^{-4}
20	1×10^5	1.10×10^{12}	1.00×10^5	MLE	8.00×10^{-2}
20	1×10^5	1.10×10^{12}	1.00×10^5	MPLE	2.94×10^{-3}
30	1×10^5	1.15×10^{18}	1.00×10^5	MLE	1.38×10^{-1}
30	1×10^5	1.15×10^{18}	1.00×10^5	MPLE	7.18×10^{-3}

5.2 Efficiency

Section 3.4 has shown that **MPLE** is statistically consistent in estimating phylogenetic trees and parameters of the substitution model as the sequence length goes to infinity. However, multiplication of pairwise likelihoods in the **MPLE** method implicitly assumes that pairs of species evolve independently, which ignores the dependent structure of the species evolution in the phylogenetic tree. Since the MPLE method does not take full advantage of the phylogenetic information in the sequence data, it may not be as efficient as the **MLE** methods based on the full likelihood function. To compare the performance of these two methods on finite sequence alignment, we calculate the Fisher information as a measurement of the efficiency of a statistical estimation method. A detailed discussion on the Fisher information is given in the next section.

The mean squared error is another popular measurement for comparing the performance of two estimation methods. The result that follows in the next section helps relate the variance of an estimator to the Fisher information. The message here is that larger Fisher information corresponds to smaller mean squared errors.

5.2.1 Fisher information

Fisher information plays a critical role throughout statistical modeling or probabilistic models (Van der Vaart, 2000). It can be used to measure the amount of information that data $\mathbf{X}$ provide about unknown parameters Θ. It also has applications in finding the variance of an estimator, as well as in the asymptotic distribution of maximum likelihood estimates.

Let $\mathbf{X} = (X_1, \ldots, X_n)$ be the random sample, and $f(\mathbf{X}|\Theta)$ be the probability density function for some model of the data, which has parameter vector $\Theta = (\theta_1, \ldots, \theta_k)$. Then the Fisher information matrix of sample size n is given by the $k \times k$ symmetric matrix whose ij^{th} element is given by the expected values of the second partial derivatives of the log-likelihood function,

$$\mathcal{I}_n(\Theta)_{ij} = E[-\frac{\partial^2 \ln f(\mathbf{X}|\Theta)}{\partial\theta_i \partial\theta_j}].$$

Fisher information is useful to construct the confidence intervals of maximum likelihood estimates (MLE's). For iid samples $(X_1, ..., X_n)$, the distribution of the difference between the true Θ and the MLE $\hat{\Theta}$ which is a function of $(X_1, ..., X_n)$ converges to a Normal distribution as n goes to infinity, which is given by, (Van der Vaart, 2000)

$$\sqrt{n}(\hat{\Theta} - \Theta) \xrightarrow{D} \mathcal{N}(0, \mathcal{I}_n^{-1}(\Theta)), n \to \infty.$$

This can be used to calculate the variance of MLE's or generate potential estimates around the true Θ with a standard error given by $\mathcal{I}_n^{-1}(\Theta)$ at the true value of Θ when n is large enough.

In our case, the parameters $\Theta = (\tau, \underline{t}, \underline{\theta})$ includes the tree topology τ, tree branch lengths $\underline{t}$ and substitution model parameters $\underline{\theta}$.

To simplify, we consider the Fisher information matrix for the branch lengths $\underline{t}$ while the tree topology and substitution model parameters are fixed, which is given by

$$\mathcal{I}_n(\underline{t}|\tau)_{ij} = E[-\frac{\partial^2 \ln f(\mathbf{X}|\tau, \underline{t}, \underline{\theta})}{\partial t_i \partial t_j}]. \tag{5.1}$$

5.2.2 Fisher information for Full Likelihood

We consider m homologous sequences sampled from m different species with the length n, which means there are n sites in each sequence. There are $r = 4^m$ different patterns for 4 nucleotides (A, C, G, T). The sequence alignment Ω is often represented as a matrix, where columns are sites and rows are species. Let column ω_k is one site pattern and p_k be the probability of observing site pattern ω_k, with $\sum_k^r p_k = 1$.

p_k can be calculated based on the transition matrix defined by numerical parameter $\underline{\theta}$ and the given tree defined by tree parameter $(\tau, \underline{t})$. Assuming that these n sites evolve independently, the random variables $X = (X_1, \ldots, X_r)$ have a *Multinomial distribution* with parameters p_k for $k = 1, 2, 3, \ldots, r$, and $\sum_{k=1}^{r} X_k = n$, and $E[X_k] = np_k$.

Then the Log function of full likelihood is defined as

$$l(\tau, \underline{t}, \underline{\theta}|\mathbf{X}) \propto \sum_{k=1}^{4^m} x_k \ln p_k,$$

where x_k is the number of pattern ω_k found in the observational pattern matrix $\mathbf{X}$.

Given other branches as constant, Fisher information of t_h is

$$I(t_h) = E[-\frac{\partial^2 \ln P(\mathbf{X}|\tau, \underline{t}, \underline{\theta})}{\partial t_h^2}] = n \sum_{k=1}^{4^m} [\frac{(\frac{\partial p_k}{\partial t_h})^2}{p_k} - \frac{\partial^2 p_k}{\partial t_h^2}]. \tag{5.2}$$

5.2.3 Fisher information for Pairwise Likelihood

Recall the Log function of Pairwise likelihood function:

$$l_p(\tau, \underline{t}, \underline{\theta}|\mathbf{X}) = \ln f_p(\mathbf{X}|\tau, \underline{t}, \underline{\theta}) = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sum_{k=1}^{16} y_k^{ij} \ln p_k^{ij}(x_i, x_j|t_{ij}),$$

where p_k^{ij} is the probability of pattern k between sequence i and j, and y_k^{ij} is the frequency of pattern k between sequence i and sequence j with $E[y_k^{ij}] = np_k^{ij}$.

Given other branches as constant, Fisher information of t_h is

$$I_p(t_h) = E[-\frac{\partial^2 l_p(\tau, \underline{t}, \underline{\theta}|\mathbf{X})}{\partial t_h^2}] = n \sum_{(i,j) \in C^h} \sum_{k=1}^{16} [\frac{(\frac{\partial p_k^{ij}}{\partial t_h})^2}{p_k^{ij}} - \frac{\partial^2 p_k^{ij}}{\partial t_h^2}], \tag{5.3}$$

where C^h is the set of index (i, j) satisfying the path between node i and node j pass through branch t_h in the tree.

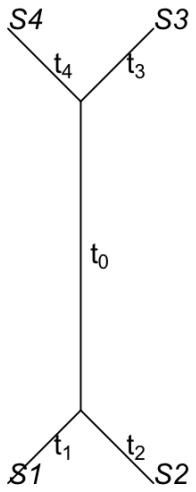

Figure 5.1: An unrooted tree with 4-taxon

For example, given the unrooted tree in Figure 5.1, Fisher information of t_0 is

$$I_p(t_0) = n \sum_{(i,j)\in\{(S_1,S_3),(S_1,S_4),(S_2,S_3),(S_2,S_4)\}} \sum_{k=1}^{16} [\frac{(\frac{\partial p_k^{ij}}{\partial t_h})^2}{p_k^{ij}} - \frac{\partial^2 p_k^{ij}}{\partial^2 t_h}],$$

and Fisher information of t_1 is

$$I_p(t_1) = n \sum_{(i,j)\in\{(S_1,S_2),(S_1,S_3),(S_1,S_4)\}} \sum_{k=1}^{16} [\frac{(\frac{\partial p_k^{ij}}{\partial t_h})^2}{p_k^{ij}} - \frac{\partial^2 p_k^{ij}}{\partial^2 t_h}],$$

where $p_k^{ij} = p_k^{ij}(x_i, x_j|t_{ij})$ with

$$\begin{aligned} t_{12} &= t_1 + t_2 \\ t_{13} &= t_0 + t_1 + t_3 \\ t_{14} &= t_0 + t_1 + t_4 \\ t_{23} &= t_0 + t_2 + t_3 \\ t_{24} &= t_0 + t_2 + t_4 \end{aligned}$$

5.3 Simulation for Fisher information

To simplify the calculation, a 4-taxon tree in Figure 5.1 is used to demonstrate how to calculate Fisher information from sequence data. We keep the terminal branches t_2, t_3 and t_4 constant and calculate the Fisher information for branches t_0 and t_1, which gives us a 2×2 Fisher information matrix $\mathcal{I}(\underline{t})$. The two diagonal values $\mathcal{I}_{ii}^{-1}(\underline{t})$ of the inverse of the Fisher information matrix correspond to the variance of the estimates of t_0 and t_1.

5.3.1 Settings for Efficiency

In this section, we calculate the Fisher information for the full likelihood function and pairwise likelihood function. To simplify the calculation, we start from a 4-taxon unrooted tree shown in Figure 5.1. The Fisher information of two likelihood functions is only calculated for the internal branch t_0 and an external branch t_1, while the other 3 external branches are assumed to be given. We considered the following cases:

- Case 1: Given external branches $t_1 = t_2 = t_3 = t_4 = 0.02$, and setting internal branch $t_0 = 0.005, 0.05, 0.1, 1.0$, increase sequence length n from 500 to 10000.
- Case 2: Given sequence length $n = 1000$ and external branches $t_1 = t_2 = t_3 = t_4 = 0.02$, change internal branch t_0 from 0.005 to 0.1.
- Case 3: Given sequence length $n = 1000$, internal branch $t_0 = 0.02$ and external branches $t_2 = t_3 = t_4 = 0.02$, change external branch t_1 from 0.005 to 0.1.

5.3.2 Case 1: Given t_0, t_1, t_2, t_3, t_4 and Change n

Figure 5.2, Figure 5.4, 5.6 and Figure 5.8 show the Fisher information $I(\underline{t})_{ii}$ when $t_0 = 0.005$, $t_0 = 0.05$, $t_0 = 0.1$ and $t_0 = 1.0$ respectively as n increases from 500 to 100000. Higher value of Fisher information means more efficient.

Figure 5.3, Figure 5.5, 5.7 and Figure 5.9 show the inverse of Fisher information $I^{-1}(\underline{t})_{ii}$ when $t_0 = 0.005$, $t_0 = 0.05$, $t_0 = 0.1$ and $t_0 = 1.0$ respectively as n increases.

As we can see, the inverse of Fisher information decreases as n increases for all scenarios, which indicates the variance of the maximum likelihood estimate converge to 0 and the Maximum likelihood estimates converge to the true values as n goes to infinity. And the difference of the information from two functions becomes close to 0 as n increases, which means two methods would obtain similar results when n is large. Because both methods are consistent, which indicates their estimates converge to the same and true values as n goes to infinity.

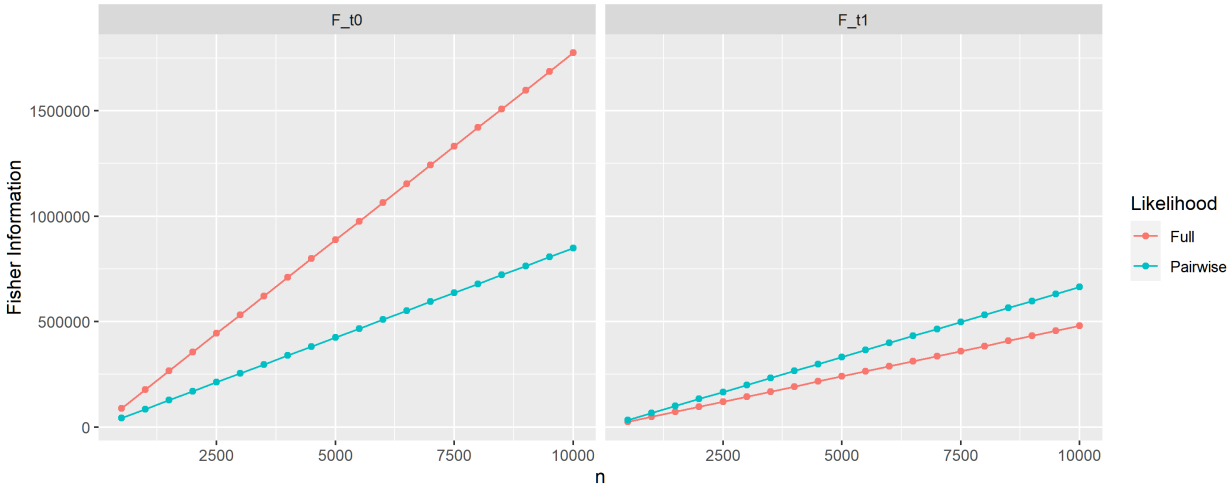

Figure 5.2: Fisher information for Case 1: $t_0 = 0.005$

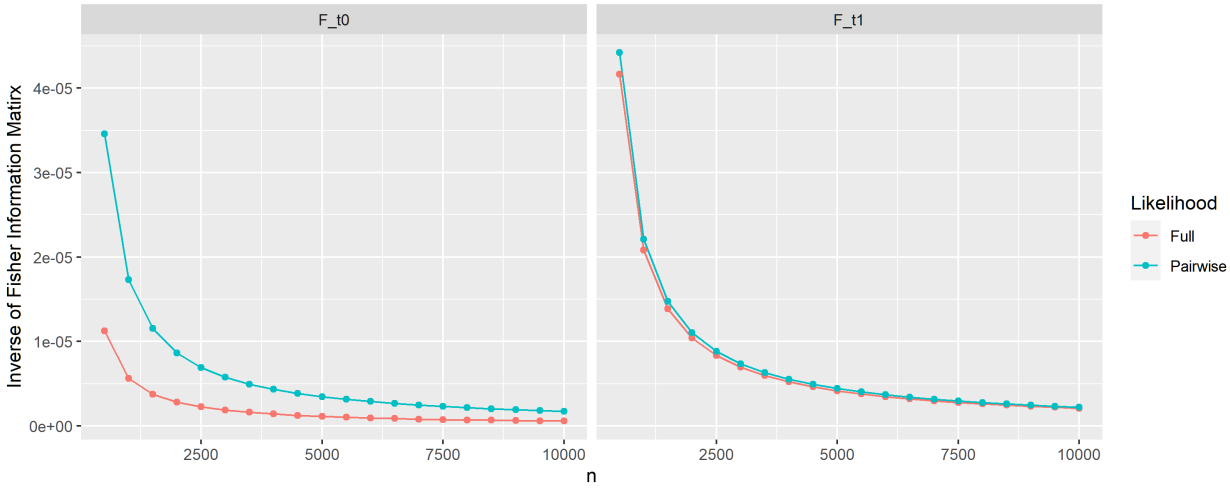

Figure 5.3: Inverse of Fisher information Matrix for Case 1: $t_0 = 0.005$

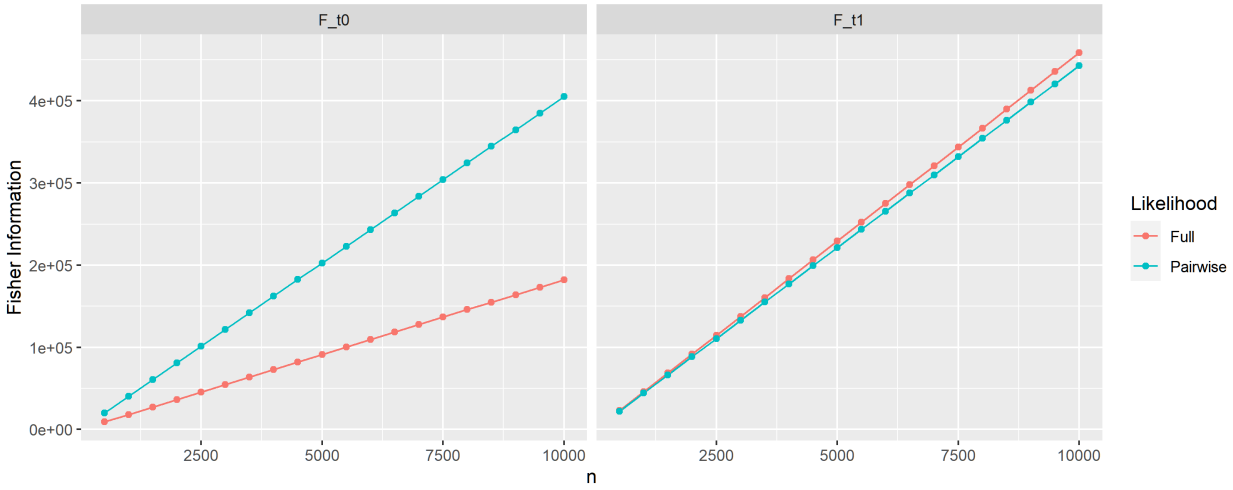

Figure 5.4: Fisher information for Case 1: $t_0 = 0.05$

In Figure 5.3, when $t_0 = 0.005$ which is smaller than t_1, both $\mathcal{I}^{-1}(t_0)$ and $\mathcal{I}^{-1}(t_1)$ from pairwise likelihood function is higher than that from full likelihood function, which means full likelihood function obtains more information and is more efficient than pairwise likelihood function.

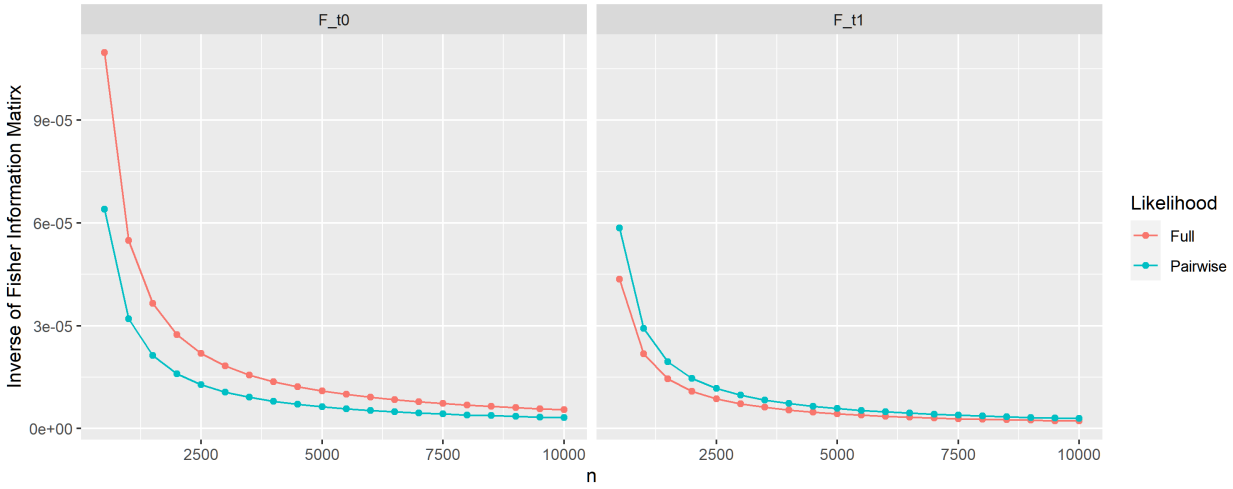

Figure 5.5: Inverse of Fisher information Matrix for Case 1: $t_0 = 0.05$

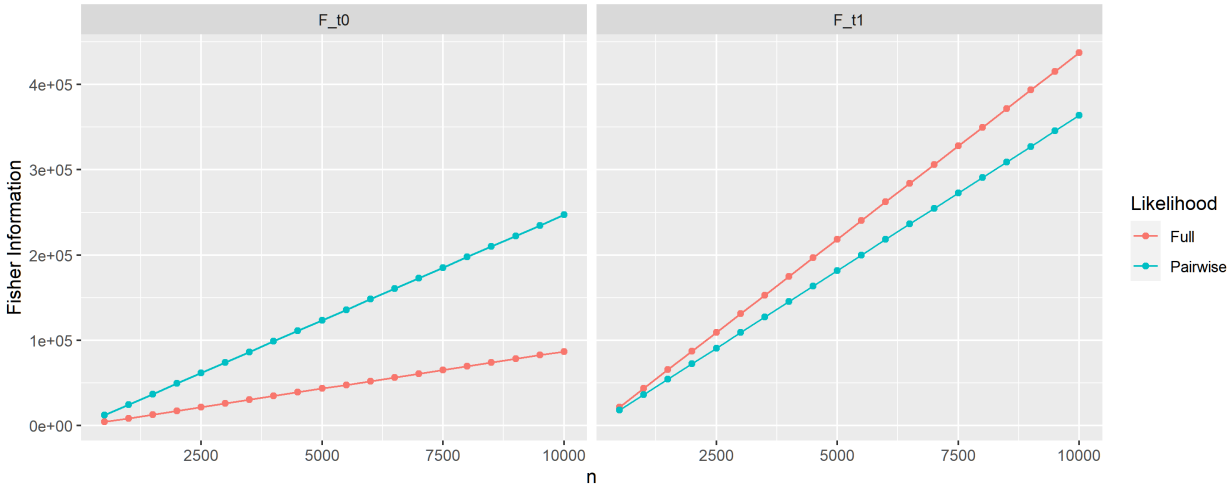

Figure 5.6: Fisher information Matrix for Case 1: $t_0 = 0.1$

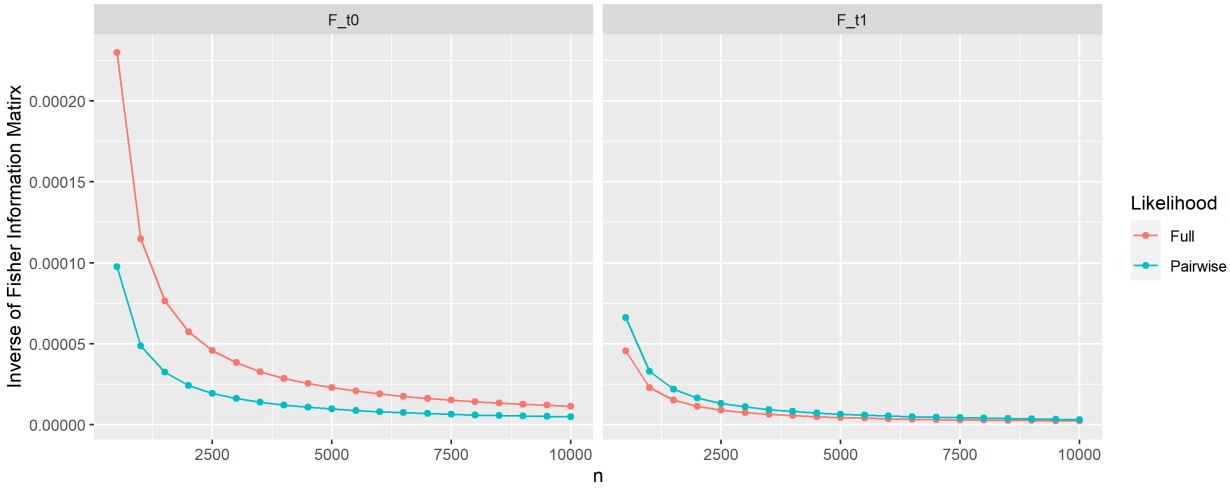

Figure 5.7: Inverse of Fisher information Matrix for Case 1: $t_0 = 0.1$

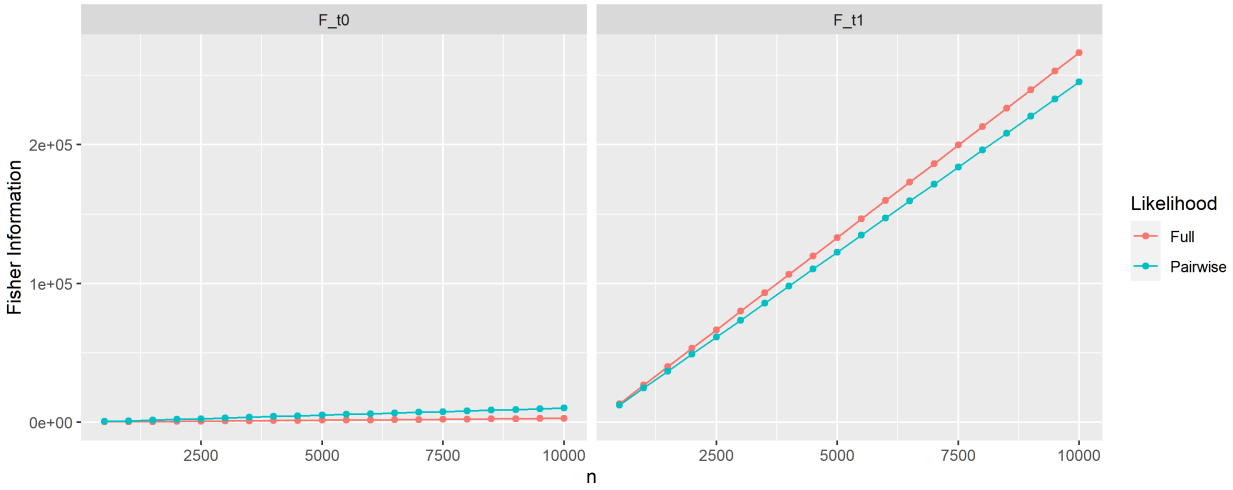

Figure 5.8: Fisher information Matrix for Case 1: $t_0 = 1$

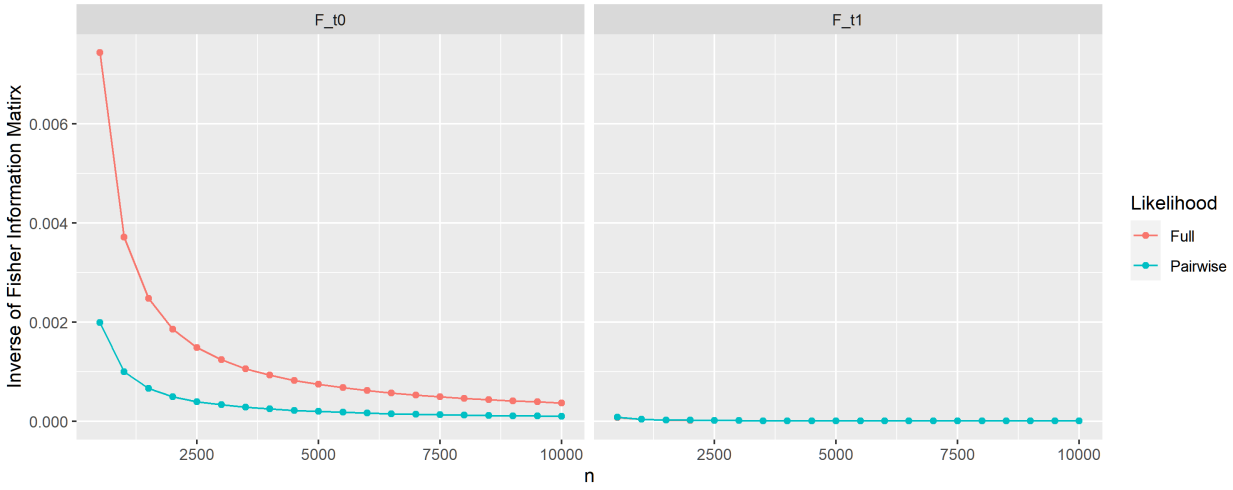

Figure 5.9: Inverse of Fisher information Matrix for Case 1: $t_0 = 1$

In Figure 5.5, Figure 5.7 and Figure 5.9 when t_0 is larger than t_1, $I^{-1}(t_1)$ from pairwise likelihood function is higher than that from full likelihood function, which means full likelihood function is more efficient on t_1. However, $I^{-1}(t_0)$ from pairwise likelihood function is is lower than that from full likelihood function when t_0 is larger than t_1, which means pairwise likelihood method might obtain more accuracy estimate on internal branch t_0 when $t_0 > t_1$.

5.3.3 Case 2: Given t_1, t_2, t_3, t_4 and Changing t_0

Figure 5.10 and Figure 5.11 show the of Fisher information $I_{ii}(t)$ and the inverse of Fisher information $I_{ii}^{-1}(t)$ when t_0 increases. First, $I_{ii}^{-1}(t)$ increases as t_0 increases for both methods because a longer branch would generate a higher variance in the estimate. Full likelihood function is more efficient on t_1 and less efficient on t_0 than pairwise likelihood function when increase t_0. The pairwise likelihood function is more efficient when the internal branch t_0 is relatively smaller than the external branches.

What's more, $I^{-1}(t_0)$ from full likelihood function increases more than that from pairwise likelihood function, which means increasing t_0 has a stronger effect on the estimate of t_0 for the full likelihood function.

However, $I^{-1}(t_1)$ from pairwise likelihood function increases more than that from full likelihood function, which means increasing t_0 has a stronger effect on the estimate of t_1 for the pairwise likelihood function. I think it is because the dependence of t_0 and t_1 is stronger in the pairwise likelihood function. So, the change of t_0 would be diluted by t_1 in the pairwise likelihood function.

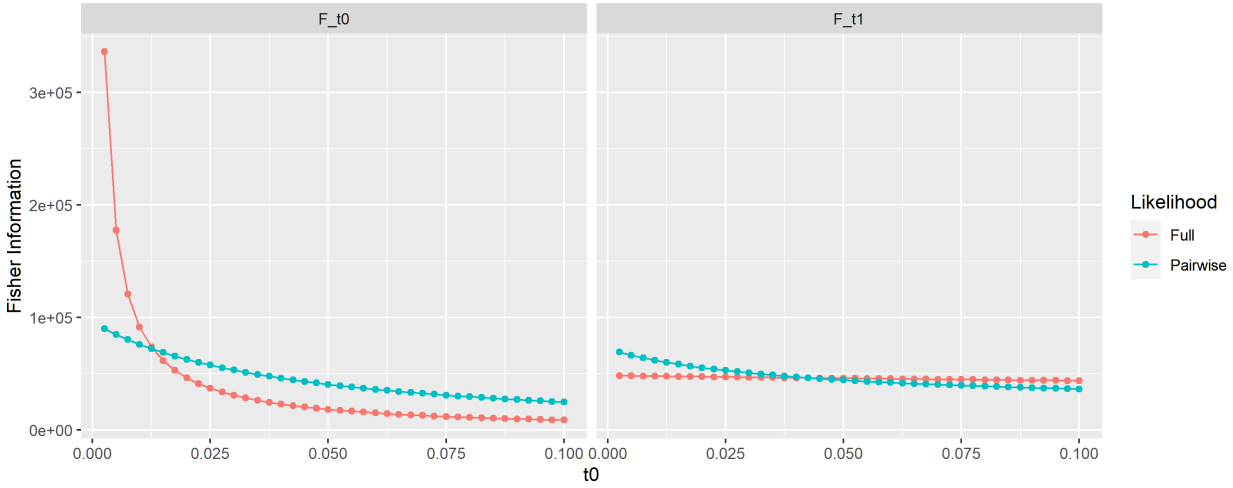

Figure 5.10: Fisher information Matrix for Case 2:t1=0.02

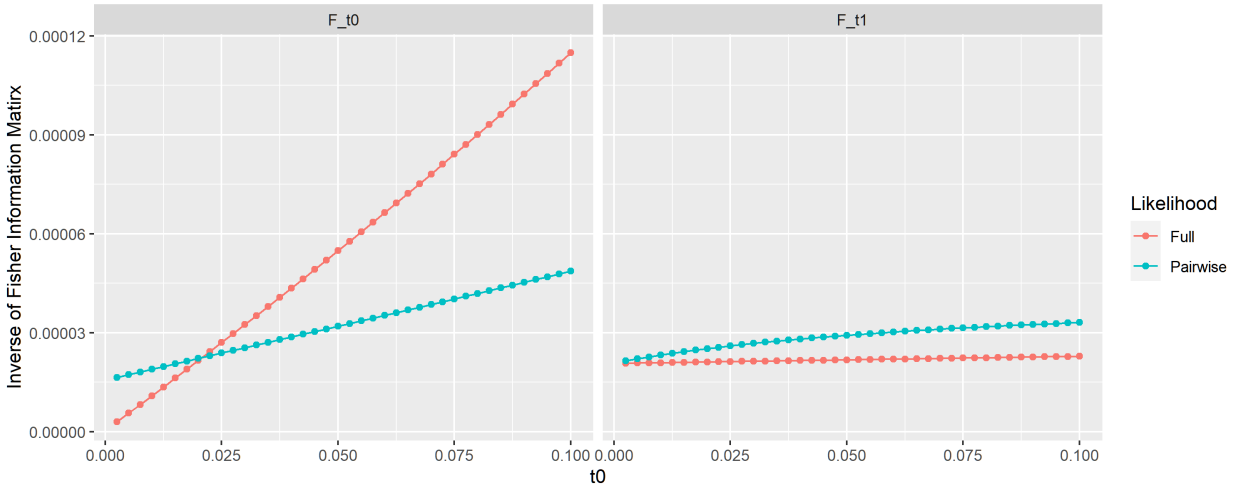

Figure 5.11: Inverse of Fisher information Matrix for Case 2:t1=0.02

5.3.4 Case 3: Given t_0, t_2, t_3, t_4 and Changing t_1

Figure 5.12 and Figure 5.13 show the Fisher information $I_{ii}(t)$ and the inverse of Fisher information $I_{ii}^{-1}(t)$ when t_1 increases.

First, $I_{ii}^{-1}(t)$ increases as t_1 increase for both methods, because longer branch would generate higher variance in the estimate. Full likelihood function is more efficient on t_0 and less efficient on t_1 than pairwise likelihood function when increase t_1. Pairwise likelihood function is more efficient when internal branch t_1 is relatively smaller than internal branch.
What's more, $I^{-1}(t_1)$ from full likelihood function increases more than that from pairwise likelihood function, which means increasing t_1 has stronger effect on estimate of t_1 for full likelihood function. However, $I^{-1}(t_0)$ from pairwise likelihood function increases more than that from full likelihood function, which means increasing t_1 has stronger effect on estimate of t_0 for pairwise likelihood function. This states the similar conclusion as that in Case 2, the dependence of t_0 and t_1 is stronger in pairwise likelihood function. So, the change of t_0 would be diluted by t_1 in pairwise likelihood function.

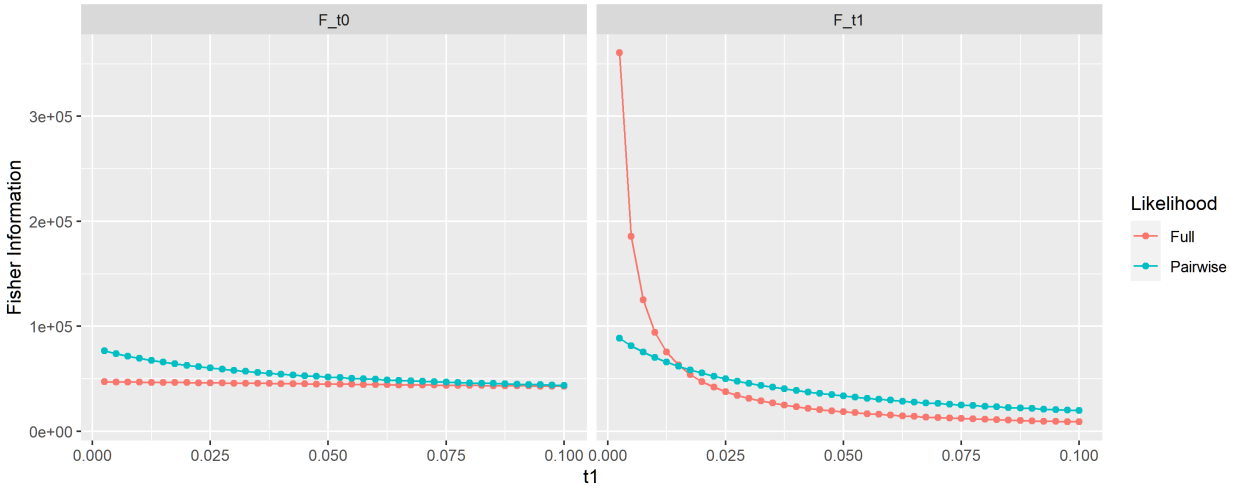

Figure 5.12: Fisher information Matrix for Case 3:to=0.02

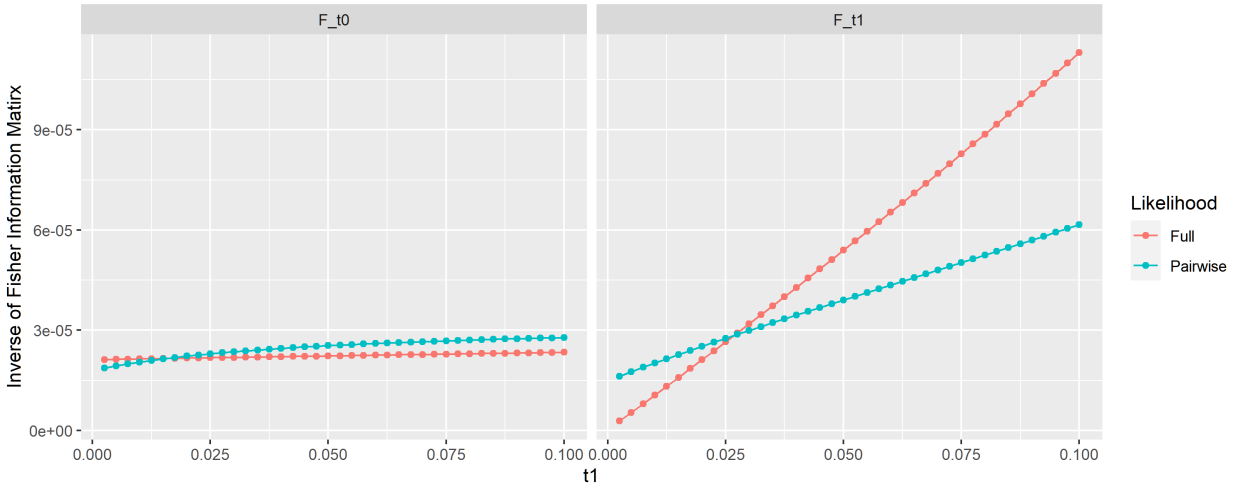

Figure 5.13: Inverse of Fisher information Matrix for Case 3:to=0.02

5.4 Discussions

This chapter compares the **MPLE** method and **MLE** method based on time complexity and efficiency. The calculating time of the full likelihood function depends on the number of taxa m and the sequence length n with respect to $m \times \max\{n, 4^m\}$. However, **MPLE** method considers the K sub-tree likelihoods (pairwise likelihoods when $K = 2$) instead of the full likelihoods, the time complexity for calculating the pseudo-likelihood function decreases to $O(m^K)$ which is free from n and much less than 4^m. Theoretically, the

consistency of **MLE/MPLE** requires sequence length n to go to ∞, which would cause a huge number of observed patterns. In addition, when the number of taxa increases, which would also cause a huge number of possible patterns (4^m). In both cases, our proposed method **MPLE** would outperforms traditional **MLE** methods on speed, because our method is free from n and costs $O(m^2)$ time.

Analysis of Fisher information shows that the full likelihood method is not always better than pseudo-likelihood methods, which can be seen in the simulation results in Chapter 6. For instances, When t_0 is relatively smaller than t_1, full likelihood function obtains lower variance on estimation of t_0 and higher variance on estimation of t_1 than pairwise likelihood function does, which is shown in Figure 5.3. When t_0 is relatively larger than t_1, pairwise likelihood function archives lower variance on estimation of t_0 and higher variance on estimation of t_1, which is shown in Figure 5.5 and Figure 5.7.

Above all, the proposed method **MPLE** cost less computational source, and large sequences would guarantee its efficiency.

Chapter 6

Experiments

Chapter 3.4 theoretically proves that the Pseudo-likelihood method is consistent which means **MPLE** converges to the true tree as sequence length n goes to infinity. In this chapter, I compared the performance of our approach (**MPLE**) and baseline (**MLE** which is implemented in `RAxML`) on the simulated datasets from some 4-taxon/5-taxon unrooted trees. Theoretically, for trees with longer internal branches, it is easier to estimate the tree topology, but it is harder to estimate the branch length.

6.1 Data Simulation and Settings

To test the model performance on 4-taxon trees, three unrooted trees are used to generate DNA alignments including (**Tree 1:** $(((S_1 : 0.02, S_2 : 0.02) : 0.005, S_3 : 0.02), S_4 : 0.02)$ on which internal branch t_0 is shorter than external branches, **Tree 2:** $(((S_1 : 0.02, S_2 : 0.02) : 0.05, S_3 : 0.02), S_4 : 0.02)$ and **Tree 3:** $(((S_1 : 0.02, S_2 : 0.02) : 0.1, S_3 : 0.02), S_4 : 0.02)$ on which internal branch t_0 is longer than external branches) shown in Figure 5.1.

To test the model performance on 5-taxon trees, **Tree 4**: $((((S_1 : 0.01, S_2 : 0.01) : 0.005, S_3 : 0.01), S_4 : 0.01) : 0.005, S_5 : 0.01)$ and **Tree 5**: $((((S_1 : 0.01, S_2 : 0.01) : 0.05, S_3 : 0.01), S_4 : 0.01) : 0.05, S_5 : 0.01)$ were used. The results will show whether we have similar conclusions when the internal branch changes with respect to the external branches.

By setting the transition parameters in **GTR** model as $\theta = (s_{ac} = 1, s_{ag} = 1, s_{at} = 1, s_{cg} = 1, s_{ct} = 1, s_{gt} = 1, \pi_a = 1/4, \pi_c = 1/4, \pi_g = 1/4)$, **JC** model was used as the transition model.

To explore how **MPLE** approach performs when the sequence length increases, the sequence length was changed from 200 to 2000. On each tree, 200 DNA alignments with different sequence lengths (n from 200 to 2000) were generated. Then `MPLE` and `RAxML` were implemented to estimate the tree including the tree topology and branch lengths from the simulated datasets.

To evaluate the model performance, I employ the following measurements:

- **Accuracy of topology**: Proportion of simulations recovering topology.

- **Accuracy of branch t_i**:

$$MSE(\hat{t_i}) = \frac{1}{N}\sum_{i=1}^{N}(\hat{t_i} - t_i)^2,$$

where $\hat{t_i}$ is the estimate of i^{th} branch t_i and N is the number of simulations recovering the true topology.

- **Accuracy of branch $\underline{t}$**:

$$MSE(\hat{\underline{t}}) = \frac{1}{N}\sum_{i=1}^{N}(\hat{\underline{t_i}} - \underline{t_i})^2.$$

- MSE of internal/external branches: MSE(t_{IN})/MSE(t_{EX})

6.1.1 Others

The simulations and model implementation are done in `R` by using the some build-in functions in `R` packages `phangorn` and `ape` (Paradis and Schliep, 2019; Schliep et al., 2017).

- `simSeq(tree0,l,type="DNA")` was used to generate DNA alignments from **JC** model.
- `allTrees()` in `R` package `phangorn` was used to generate all possible topology of an unrooted tree.
- `rtree()` in `R` package `ape` was used to generate random starting trees.
- `distTips(tree, "patristic")` in `R` package `adephylo` was used to calculate the pairwise distance matrix.
- `pml()` and `optim.pml()` were used to estimate branches based on full likelihood method.
- `all.equal(tree,tree0,use.edge.length = FALSE)` was used to check the accuracy of estimated topology.
- `comparePhylo()` in `R` package `ape` was used to compare estimated trees and print out trees.
- `optim()` in `R` package `stats` was used to optimize the parameters such as branch length in `MPLE`. The lower bound of branch length is set as 10^{-8}.
- `rSPR()` in `R` package `phangorn` was used to implement the tree-arrangement algorithm `SPR`.
- `nni()` in `R` package `phangorn` was used to implement the tree-arrangement algorithm `NNI`.

6.1.2 4-taxon Tree

Case 1: Tree 1 with $t_0 < t_1$

Firstly, **Tree 1:** $(((S_1 : 0.02, S_2 : 0.02) : 0.005, S_3 : 0.02), S_4 : 0.02)$ was used as the true tree.

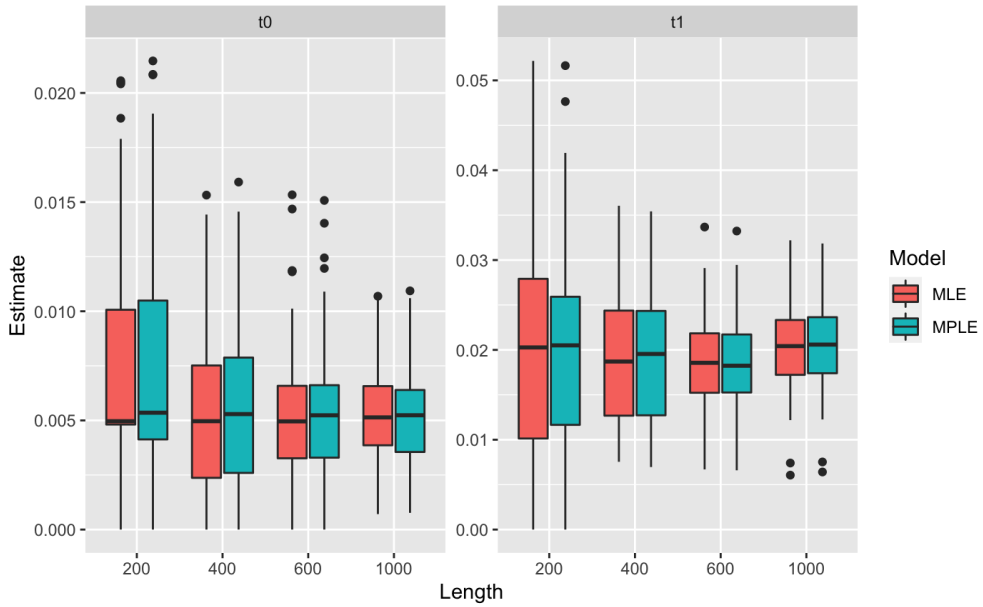

Figure 6.1: Branch Estimate from **MLE** and **MPLE** for **Tree 1**

Figure 6.1 shows the branch length estimation from **MLE** and **MPLE** on **Tree 1**. The mean of the branch length estimates were close to the correct/original branches ($t_0 = 0.005, t_1 = 0.02, t_2 = 0.02, t_3 = 0.02, t_4 = 0.02$). For all branches, the estimate of branch gathered closer to the true branch as sequence length n increases.

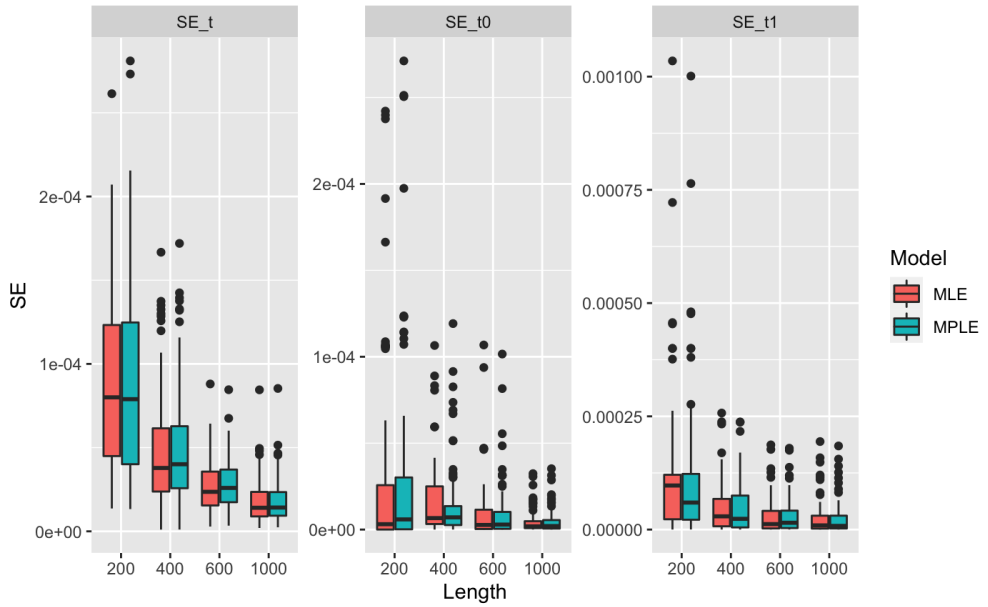

Figure 6.2: SE of Branch Estimate from **MLE** and **MPLE** for **Tree 1**

Figure 6.2 shows square error of branch length estimation from **MLE** and **MPLE** on **Tree 1**. We can see the square error converges to 0 as sequence length n increases, which also indicates that the estimate of branch gathered closer to the true branch.

Table 6.1 shows the estimate results from these two approaches. We can see, more DNA bases improve the accuracy of topology estimates, because of more information provided. It is interesting to find that our approach `MPLE` obtained better accuracy of topology than approach **MLE** implemented in `RAxML` did in some scenarios. I guess that `RAxML` set negative branches as 0 which might change the topology, while `MPLE` does not allow branches with 0 length by setting the minimum of branch length as 10^{-8}. Another possible reason is randomization in the sequence generation process. It also shows that both methods could correctly detect the true tree topology $(((S_1, S_2), S_3), S_4)$ with an accuracy of 100% with over 1000 bases.

Comparing the accuracy of topology with respect to the same length and model, Table 6.1, Table 6.2 and Table 6.3 show that it is easier to estimate the topology when internal branch is longer. It also shows that the **MPLE** method estimated branches more precisely with square error range in $[10^{-6}, 10^{-5}]$ as sequence length n increased which means the estimation of tree converges to the true tree as n increases.

Table 6.1: Simulated Results for 4-taxon **Tree 1** with $t_0 = 0.005$.

Length	Model	Topology	MSE($\hat{\underline{t}}$)	MSE($\hat{t}_0$)	MSE($\hat{t}_1$)
200	MLE	0.73	8.529×10^{-5}	3.437×10^{-5}	1.237×10^{-4}
200	MPLE	0.71	8.600×10^{-5}	3.438×10^{-5}	1.228×10^{-4}
400	MLE	0.85	4.971×10^{-5}	1.493×10^{-5}	5.086×10^{-5}
400	MPLE	0.86	5.038×10^{-5}	1.480×10^{-5}	5.103×10^{-5}
600	MLE	0.95	2.704×10^{-5}	8.799×10^{-6}	2.984×10^{-5}
600	MPLE	0.94	2.724×10^{-5}	8.709×10^{-6}	2.994×10^{-5}
800	MLE	0.975	1.128×10^{-5}	5.082×10^{-6}	1.280×10^{-5}
800	MPLE	0.985	1.129×10^{-5}	5.220×10^{-6}	1.287×10^{-5}
1000	MLE	0.99	1.806×10^{-5}	4.353×10^{-6}	2.443×10^{-5}
1000	MPLE	0.99	1.811×10^{-5}	4.658×10^{-6}	2.389×10^{-5}
2000	MLE	1	4.646×10^{-6}	2.539×10^{-6}	5.286×10^{-6}
2000	MPLE	1	4.677×10^{-6}	2.566×10^{-6}	5.324×10^{-6}

However, it shows that **MPLE** method obtains smaller values of MSE($\hat{\underline{t}}$), MSE($\hat{t}_0$) and MSE($\hat{t}_1$) than that from **MLE**, which means **MPLE** estimated branches more precisely. This is NOT consistent with what we excepted. **MPLE** should be not better than **MLE**, because **MPLE** assumes the nodes are pairwisely independent which is not appropriate. So, I change the internal branch in Case 2 and Case 3, and construct simulations with different values of t_0.

Case 2: Tree 2 with $t_0 > t_1$

By increasing the internal branch t_0 from 0.005 to 0.05, **Tree 2:** $(((S_1 : 0.02, S_2 : 0.02) : 0.05, S_3 : 0.02), S_4 : 0.02)$ was used to simulate sequence data.

Figure 6.3 shows the branch length estimation from **MLE** and **MPLE** on **Tree 2**. The mean of the branch length estimates were close to the true branches ($t_0 = 0.05, t_1 = t_2 = t_3 = t_4 = 0.02$). For all branches, the estimate of branch gathered closer to the true branch as sequence length n increases.

Figure 6.4 shows square error of branch length estimation from **MLE** and **MPLE** on **Tree 2**. We can see the square error converges to 0 as sequence length n increases, which also indicates that the estimate of branch gathered closer to the true branch.

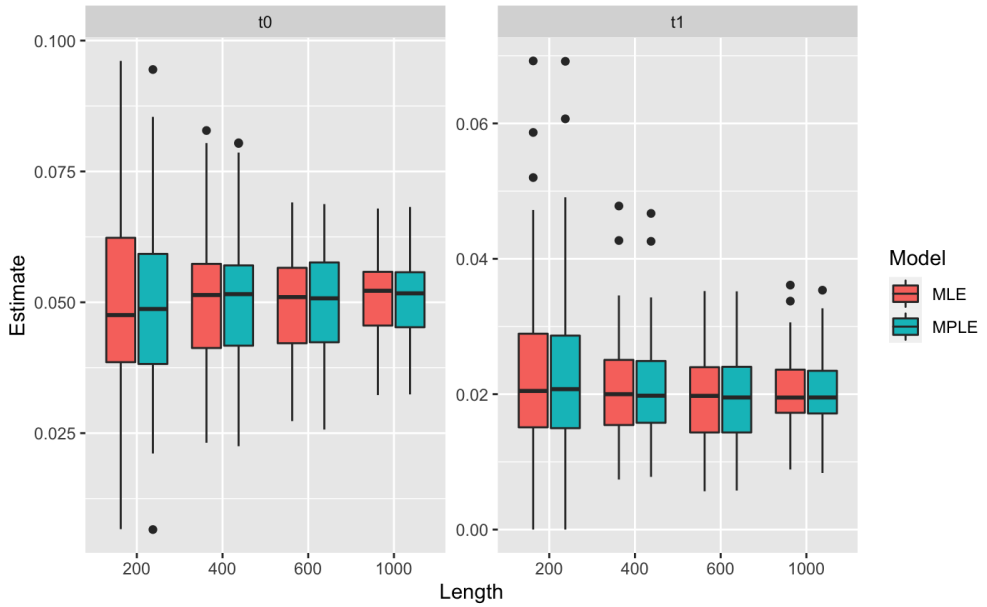

Figure 6.3: Branch Estimate from **MLE** and **MPLE** for **Tree 2**

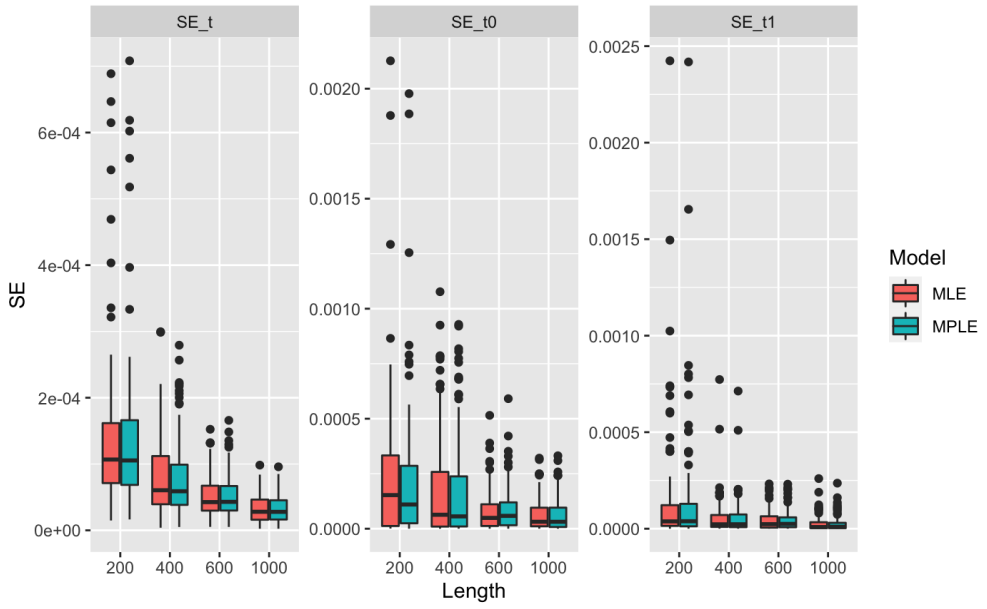

Figure 6.4: SE of Branch Estimate from **MLE** and **MPLE** for **Tree 2**

Table 6.2 shows the estimate results when $t_0 = 0.05$. Similar conclusions can be made as Case 1. However, `RAxML` has better accuracy of topology than `MPLE` does in general. In addition, it shows that **MLE** method obtains smaller values of MSE($\hat{\underline{t}}$), MSE($\hat{t}_0$) and MSE($\hat{t}_1$) than that from **MPLE**, which means **MLE** estimated branches more precisely. This is consistent with what we except, **MPLE** should

be not better than **MLE**, because **MPLE** assumes the nodes are pair-wisely independent which is not appropriate.

Table 6.2: Simulated Results for 4-taxon **Tree 2** with $t_0 = 0.05$.

Length	Model	Topology	MSE($\hat{\underline{t}}$)	MSE($\hat{t}_0$)	MSE($\hat{t}_1$)
200	MLE	1	1.412×10^{-4}	2.197×10^{-4}	1.580×10^{-4}
200	MPLE	1	1.405×10^{-4}	2.156×10^{-4}	1.573×10^{-4}
400	MLE	1	8.016×10^{-5}	1.820×10^{-4}	5.872×10^{-5}
400	MPLE	1	7.915×10^{-5}	1.799×10^{-4}	5.761×10^{-5}
600	MLE	1	5.088×10^{-5}	8.565×10^{-5}	4.500×10^{-5}
600	MPLE	1	5.155×10^{-5}	8.838×10^{-5}	4.490×10^{-5}
800	MLE	1	2.504×10^{-5}	7.546×10^{-5}	1.084×10^{-5}
800	MPLE	0.98	2.451×10^{-5}	7.233×10^{-5}	1.082×10^{-5}
1000	MLE	1	3.280×10^{-5}	6.574×10^{-5}	2.580×10^{-5}
1000	MPLE	1	3.229×10^{-5}	6.383×10^{-5}	2.538×10^{-5}
2000	MLE	1	1.085×10^{-5}	3.294×10^{-5}	5.770×10^{-6}
2000	MPLE	1	1.074×10^{-5}	3.252×10^{-5}	5.754×10^{-6}

Case 3: Tree 3 with $t_0 > t_1$

At last, increasing the internal branch t_0 to 0.1, **Tree 3:** $(((S_1 : 0.02, S_2 : 0.02) : 0.1, S_3 : 0.02), S_4 : 0.02)$ was used to simulate sequence data. Figure 6.5 and Figure 6.6 shows the branch length estimation and square error of branch length estimation from **MLE** and **MPLE**. Similar results can be concluded as on **Tree 1** and **Tree 2**.

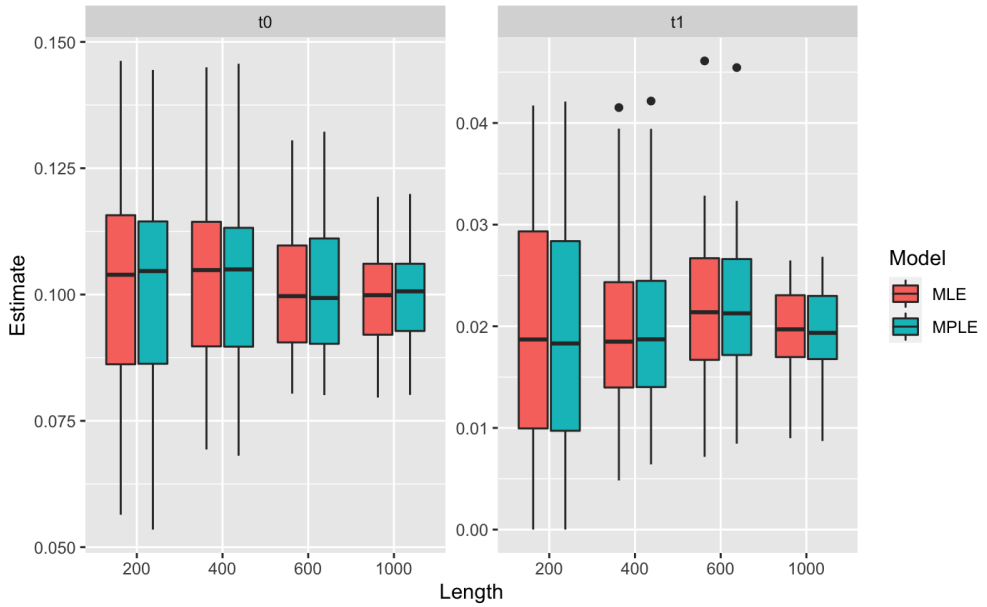

Figure 6.5: Branch Estimate from **MLE** and **MPLE** for **Tree 3**

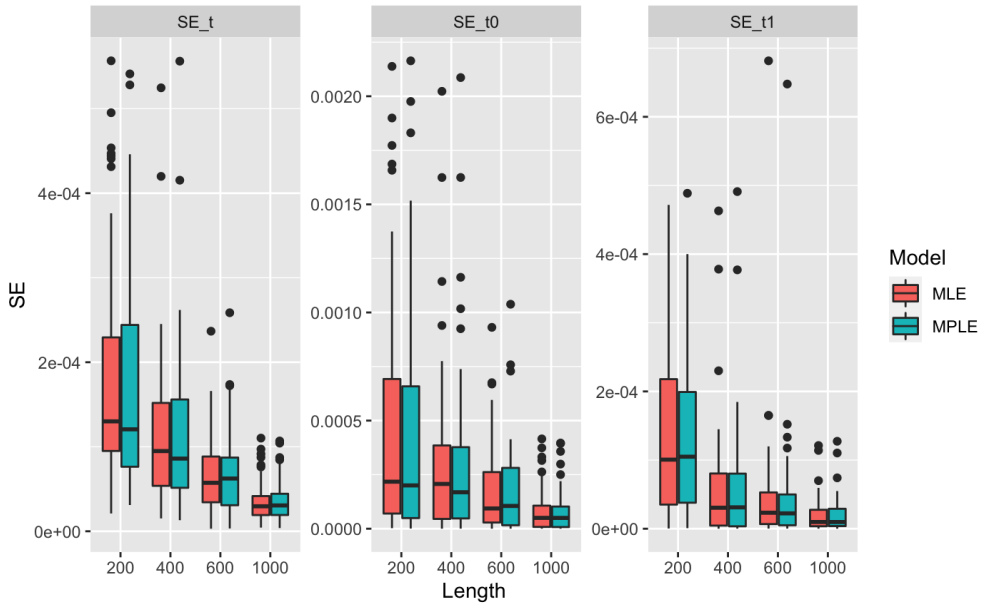

Figure 6.6: SE of Branch Estimate from **MLE** and **MPLE** for **Tree 3**

Table 6.3 shows the estimate results when $t_0 = 0.1$. It shows that **MLE** method obtains smaller values of MSE than that from **MPLE**, which is consistent with Case 2. We can see only 800 bases are enough to 100% correctly detect the true tree topology for both methods, which is smaller than the size needed in Case 2 and Case 1. That is, it is easier to detect the topology from a tree with longer internal branches.

Table 6.3: Simulated Results for 4-taxon **Tree 3** with $t_0 = 0.1$.

Length	Model	Topology	MSE($\hat{\underline{t}}$)	MSE($\hat{t}_0$)	MSE($\hat{t}_1$)
200	MLE	1	1.821×10^{-4}	4.701×10^{-4}	1.342×10^{-4}
200	MPLE	1	1.790×10^{-4}	4.611×10^{-4}	1.273×10^{-4}
400	MLE	1	1.147×10^{-4}	3.217×10^{-4}	5.982×10^{-5}
400	MPLE	1	1.148×10^{-4}	3.253×10^{-4}	5.830×10^{-5}
600	MLE	1	6.740×10^{-5}	1.777×10^{-4}	4.965×10^{-5}
600	MPLE	1	6.756×10^{-5}	1.797×10^{-4}	4.706×10^{-5}
800	MLE	1	3.732×10^{-5}	1.300×10^{-4}	1.430×10^{-5}
800	MPLE	1	3.679×10^{-5}	1.275×10^{-4}	1.422×10^{-5}
1000	MLE	1	3.518×10^{-5}	8.725×10^{-5}	1.997×10^{-5}
1000	MPLE	1	3.547×10^{-5}	8.642×10^{-5}	2.059×10^{-5}
2000	MLE	1	1.517×10^{-5}	5.482×10^{-5}	5.230×10^{-6}
2000	MPLE	1	1.518×10^{-5}	5.494×10^{-5}	5.186×10^{-6}

6.1.3 5-taxon Tree

To extend the work to more general tree, I also generated DNA alignments on 5-taxon unrooted trees including **Tree 4**: $((((S_1 : 0.01, S_2 : 0.01) : 0.005, S_3 : 0.01), S_4 : 0.01) : 0.005, S_5 : 0.01)$ and **Tree 5**: $((((S_1 : 0.01, S_2 : 0.01) : 0.05, S_3 : 0.01), S_4 : 0.01) : 0.05, S_5 : 0.01)$.

Table 6.4: Simulated Results for 5-taxon **Tree 4** with $t_0 = 0.005$.

Length	Model	Topology	MSE($\hat{\underline{t}}$)	MSE($\hat{t}_{IN}$)	MSE($\hat{t}_{EX}$)
200	MLE	0.535	5.795×10^{-4}	1.904×10^{-3}	4.959×10^{-5}
200	MPLE	0.49	5.692×10^{-4}	1.871×10^{-3}	4.863×10^{-5}
500	MLE	0.875	5.850×10^{-4}	1.998×10^{-3}	1.969×10^{-5}
500	MPLE	0.84	5.814×10^{-4}	1.986×10^{-3}	1.970×10^{-5}
1000	MLE	0.99	5.899×10^{-4}	2.040×10^{-3}	9.943×10^{-6}
1000	MPLE	0.985	5.894×10^{-4}	2.038×10^{-3}	9.950×10^{-6}
2000	MLE	1	5.832×10^{-4}	2.028×10^{-3}	5.157×10^{-6}
2000	MPLE	1	5.835×10^{-4}	2.029×10^{-3}	5.188×10^{-6}

Table 6.5: Simulated Results for 5-taxon **Tree 5** with $t_0 = 0.05$.

Length	Model	Topology	MSE($\underline{\hat{t}}$)	MSE($\hat{t}_{IN}$)	MSE($\hat{t}_{EX}$)
200	MLE	1	1.109×10^{-4}	2.616×10^{-4}	5.061×10^{-5}
200	MPLE	1	1.134×10^{-4}	2.697×10^{-4}	5.091×10^{-5}
500	MLE	1	4.635×10^{-5}	1.027×10^{-4}	2.383×10^{-5}
500	MPLE	1	4.686×10^{-5}	1.036×10^{-4}	2.417×10^{-5}
1000	MLE	1	2.187×10^{-5}	4.902×10^{-5}	1.101×10^{-5}
1000	MPLE	1	2.238×10^{-5}	5.042×10^{-5}	1.116×10^{-5}
2000	MLE	1	1.129×10^{-5}	2.580×10^{-5}	5.483×10^{-6}
2000	MPLE	1	1.143×10^{-5}	2.595×10^{-5}	5.620×10^{-6}

Table 6.4 and Table 6.5 show the results from `RAxML` and `MPLE`. We can see `MPLE` detects the tree topology with accuracy almost 100% when 1000 DNA bases are provided, which means `MPLE` could easily detect the tree topology when there are 5 species. And `MPLE` obtains competitive values of Mean square error as `RAxML` does.

6.1.4 8-taxon Tree

Since there are 10395 candidate tree topologies when $m = 8$, we compared the efficiency of two methods given the true topology which is shown in Figure 6.7.

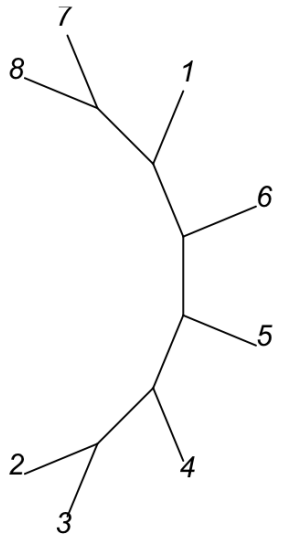

Figure 6.7: An Unrooted Tree with 8-taxon for Simulation.

There are 5 internal branches and 8 external branches. We also consider the Accuracy of internal branches $t_{IN} = (t_{01}, t_{02}, t_{03}, t_{04}, t_{05})$: $MSE(\hat{t}_{IN}) = \frac{1}{5N}\sum_{i=1}^{N}(\hat{t}_{IN}^{(i)} - t_{IN})^2$ where $\hat{t}_{IN}^{(i)}$ is the i^{th} estimate of internal branch vector t_{IN}, and Accuracy of external branches $t_{EX} = (t_1, \ldots, t_8)$: $MSE(\hat{t}_{EX}) = \frac{1}{8N}\sum_{i=1}^{N}(\hat{t}_{EX}^{(i)} - t_{EX})^2$ where $\hat{t}_{EX}^{(i)}$ is the i^{th} estimate of internal branch vector t_{EX}.

Table 6.6 and Table 6.7 show the estimation for 8-taxon tree with internal branch length $t_0 = 0.005$ and $t_0 = 0.05$ given topology respectively. MPLE obtains competitive values of Mean square error as RAxML does.

Table 6.6: Simulated Results for 8-taxon Tree with $t_0 = 0.005$.

Length	Model	MSE($\hat{\underline{t}}$)	MSE($\hat{t}_{IN}$)	MSE($\hat{t}_{EX}$)
500	MLE	1.61×10^{-5}	4.443×10^{-5}	1.652×10^{-4}
500	MPLE	1.63×10^{-5}	4.469×10^{-5}	1.671×10^{-4}
1000	MLE	8.38×10^{-6}	2.697×10^{-5}	8.196×10^{-5}
1000	MPLE	8.49×10^{-6}	2.833×10^{-5}	8.205×10^{-5}
5000	MLE	1.65×10^{-6}	5.321×10^{-6}	1.613×10^{-5}
5000	MPLE	1.67×10^{-6}	5.536×10^{-6}	1.623×10^{-5}

Table 6.7: Simulated Results for 8-taxon Tree with $t_0 = 0.05$.

Length	Model	MSE($\underline{t}$)	MSE(t_{IN})	MSE(t_{EX})
500	MLE	5.68×10^{-5}	5.603×10^{-4}	1.775×10^{-4}
500	MPLE	5.89×10^{-5}	5.738×10^{-4}	1.921×10^{-4}
1000	MLE	2.79×10^{-5}	2.711×10^{-4}	9.151×10^{-5}
1000	MPLE	2.94×10^{-5}	2.835×10^{-4}	9.890×10^{-5}
5000	MLE	5.69×10^{-6}	5.521×10^{-5}	1.871×10^{-5}
5000	MPLE	6.04×10^{-6}	5.806×10^{-5}	2.044×10^{-5}

6.1.5 Trees with more taxon

To explore the efficiency of MPLE on more complicated, I implemented MPLE and MLE on trees with $m = 10, 15, 20, 25, 30$ and $t_{IN} = 0.01, t_{EX} = 0.05$. JC model was used to simulate sequence data on tress and the sequence length n is from 500 to 10^4. To check the robustness of the algorithm, the process was repeated $N = 20$ times.

Table 6.8 shows the simulated results for m-taxon trees with $m = 10, 15, 20, 25, 30$ and $t_0 = 0.01, t_1 = 0.05$. For the accuracy of Topology: as length n increases, both MLE and MPLE estimate the topology more accurately, and when $n > 1000$ MPLE could obtain $> 95\%$ of accuracy even for large species size $m = 30$. So, $n = 1000$ base pairs are large enough for MPLE to achieve good topology estimate compared with RAxML, which was promising. For the branch length estimation which was measured by mean square error of branch (MSE.t), mean square error of internal branches (MSE.t_{IN}) and mean square error of external branches (MSE.t_{EX}). It shows that MPLE obtained competitive efficiency as RAxML did. Over all, MPLE would obtain as the most efficient algorithm RAxML did both on topology and branch length.

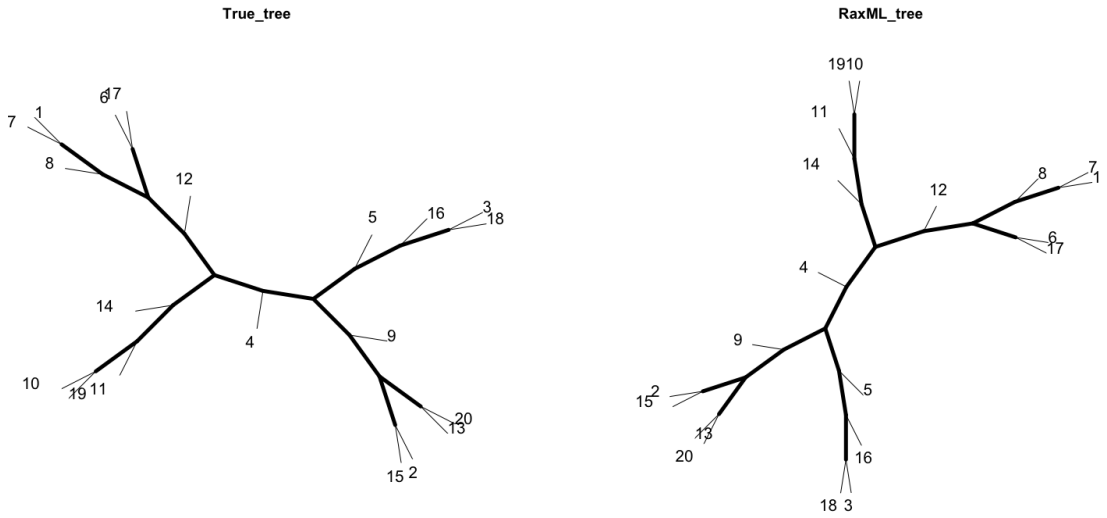

Figure 6.8: Results from **MLE** for a 20-taxon Tree with $n = 10^5, t_0 = 0.01, t_1 = 0.05$

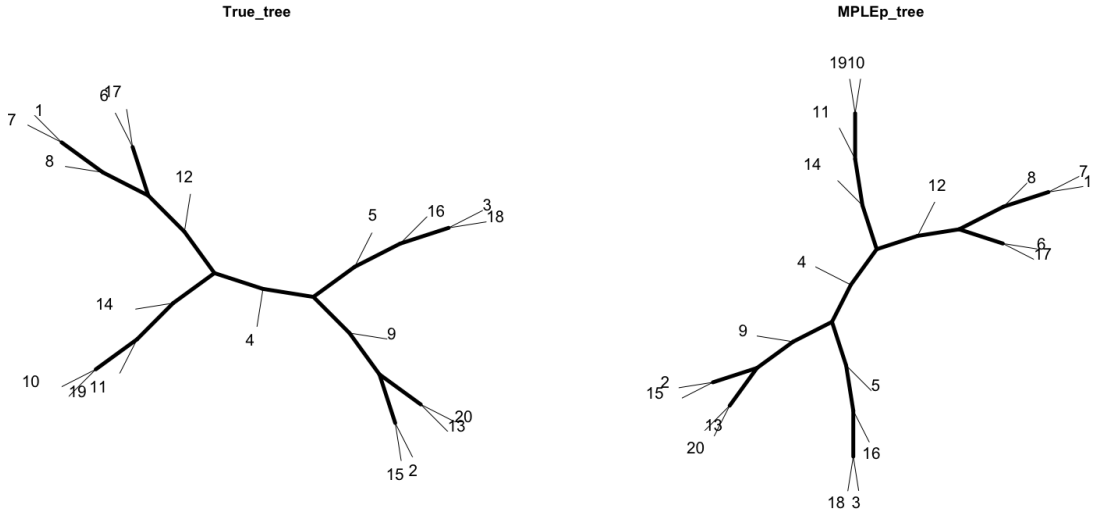

Figure 6.9: Results from **MPLE** for a 20-taxon Tree with $n = 10^5, t_0 = 0.01, t_1 = 0.05$

Figure 6.8 shows the estimate tree from `MLE` and Figure 6.9 shows the estimate tree from `MPLE` when the true tree is a 20-taxon unrooted tree with internal branches $t_0 = 0.01$ and external branches $t_1 = 0.05$.

The tree topology is correctly detected and the branches are close to the true values, which also proves that the algorithm MPLE performs well.

6.2 Real-world Data

Last, we applied the proposed algorithm MPLE on some real-world dataests in Table 5.5 and compared the estimate results with that from MLE.

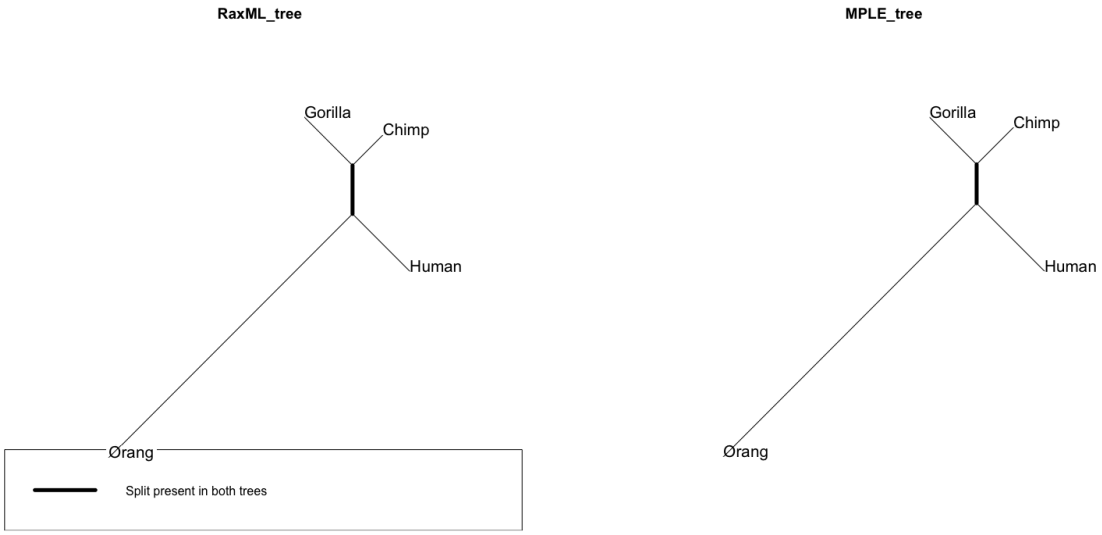

Figure 6.10: Estimate Result for Dataset Primates

The *primates* dataset has 4 sequences (Human, Chimp, Gorilla, and Orang), which contains $25,785$ characters and 213 different site patterns. Figure 6.10 shows the reconstructed trees from MLE and MPLE methods. Two methods get the same topology and similar branch lengths. And the mean square error of the branch estimates from MLE and MPLE methods is 1.851752×10^{-6}, which also shows two methods have close branch estimates.

yeast is a DNA alignment data set of 8 yeast (including Scer, Spar, Smik, Skud, Sbay, Scas, Sklu, and Calb) with $127,026$ base pairs. Figure 6.11 shows the estimate trees for data set *yeast* from MLE and MPLE methods using JC substitution model. It shows that two methods share the same estimated topology.

6.3 Results

The simulation results show that the performance (Accuracy of topology and Accuracy of branch estimate) of algorithm MPLE would be very close to that from RAxML especially when n is large (> 1000).

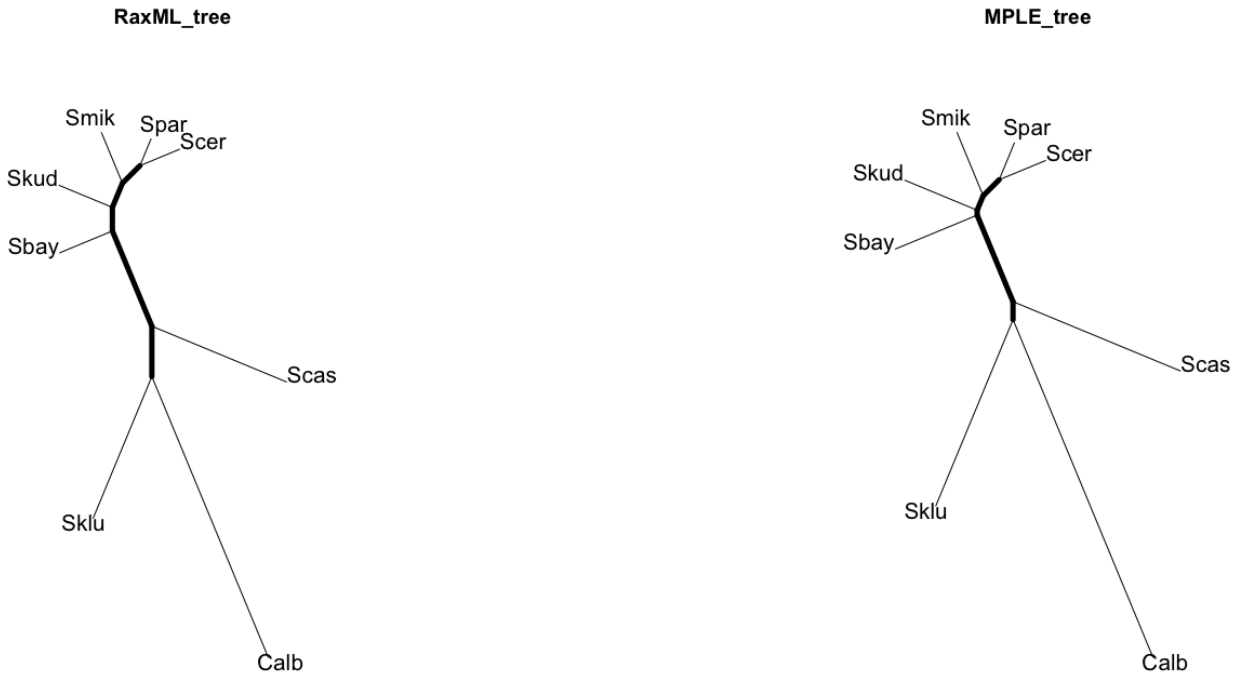

Figure 6.11: Estimate Result for Dataset Yeast

Table 6.8: Simulated Results for m-taxon Tree with $t_0 = 0.01, t_1 = 0.05$ ($N = 20$).

m	Length	Pattern	Model	Topology	MSE.t	MSE.t_{IN}	MSE.t_{EX}
10	500	159	MLE	0.85	1.208×10^{-4}	1.922×10^{-5}	1.016×10^{-4}
10	500	156	MPLE	0.85	1.223×10^{-4}	1.780×10^{-5}	1.045×10^{-4}
15	500	236	MLE	0.75	1.178×10^{-4}	2.114×10^{-5}	9.669×10^{-5}
15	500	236	MPLE	0.65	1.162×10^{-4}	2.454×10^{-5}	9.168×10^{-5}
20	500	305	MLE	0.65	1.396×10^{-4}	2.364×10^{-5}	1.160×10^{-4}
20	500	306	MPLE	0.60	1.484×10^{-4}	2.811×10^{-5}	1.203×10^{-4}
25	500	357	MLE	0.65	1.456×10^{-4}	2.456×10^{-5}	1.210×10^{-4}
25	500	356	MPLE	0.60	1.354×10^{-4}	2.799×10^{-5}	1.074×10^{-4}
30	500	389	MLE	0.45	1.115×10^{-4}	1.853×10^{-5}	9.297×10^{-5}
30	500	391	MPLE	0.40	1.195×10^{-4}	2.144×10^{-5}	9.811×10^{-5}
10	1000	247	MLE	1.00	6.025×10^{-5}	1.225×10^{-5}	4.799×10^{-5}
10	1000	247	MPLE	1.00	6.405×10^{-5}	1.347×10^{-5}	5.058×10^{-5}
15	1000	403	MLE	0.95	7.017×10^{-5}	1.206×10^{-5}	5.810×10^{-5}
15	1000	400	MPLE	0.75	7.244×10^{-5}	1.396×10^{-5}	5.848×10^{-5}
20	1000	538	MLE	1.00	6.625×10^{-5}	1.098×10^{-5}	5.527×10^{-5}
20	1000	538	MPLE	1.00	6.897×10^{-5}	1.245×10^{-5}	5.652×10^{-5}
25	1000	654	MLE	1.00	6.796×10^{-5}	1.141×10^{-5}	5.655×10^{-5}
25	1000	655	MPLE	0.95	6.949×10^{-5}	1.258×10^{-5}	5.691×10^{-5}
30	1000	750	MLE	0.95	5.851×10^{-5}	1.068×10^{-5}	4.783×10^{-5}
30	1000	747	MPLE	0.80	5.979×10^{-5}	1.260×10^{-5}	4.720×10^{-5}
10	100000	4427	MLE	1.00	6.648×10^{-7}	1.436×10^{-7}	5.212×10^{-7}
10	100000	4427	MPLE	1.00	7.053×10^{-7}	1.588×10^{-7}	5.465×10^{-7}
15	100000	11511	MLE	1.00	5.799×10^{-7}	1.031×10^{-7}	4.769×10^{-7}
15	100000	11511	MPLE	1.00	5.928×10^{-7}	1.126×10^{-7}	4.802×10^{-7}
20	100000	20976	MLE	1.00	6.767×10^{-7}	1.108×10^{-7}	5.660×10^{-7}
20	100000	20976	MPLE	1.00	7.019×10^{-7}	1.302×10^{-7}	5.717×10^{-7}
25	100000	31721	MLE	1.00	5.955×10^{-7}	1.140×10^{-7}	4.815×10^{-7}
25	100000	31721	MPLE	1.00	6.410×10^{-7}	1.467×10^{-7}	4.943×10^{-7}
30	100000	42683	MLE	1.00	6.487×10^{-7}	1.173×10^{-7}	5.314×10^{-7}
30	100000	42683	MPLE	1.00	6.806×10^{-7}	1.378×10^{-7}	5.428×10^{-7}

Chapter 7

Conclusion

Phylogeny reconstruction which aims to find the evolutionary history of a group of entities is a popular but challenging topic. The evolutionary history is very useful to learn virus/disease transmission and build resistance development, such as COVID-19.

However, finding the optimal tree is exceptionally difficult not only because of searching the tree space but also because the evaluation of each tree requires a considerable amount of calculations. For a set of 20 taxa, there are at most 10^{13} possible patterns needed to be considered. In practice, there would be much more taxa especially when we work on the **Tree of life** which may contain millions of species.

It is shown that statistical models are more accurate and preferred than other methods including **MP** method and Distance-based methods because they are grounded in statistical theory and defined on explicit evolution models.

It is too expensive to reconstruct the tree using algorithms based on the full likelihood function. So speeding up the algorithms has become an important issue for phylogeny reconstruction either on searching tree typologies or calculating likelihoods. This paper focuses on approximating the full likelihood function by considering the marginal distributions.

We construct the `MPLE` algorithm and compare it with the most popular maximum-likelihood methods `RAxML`. Extensive experiments on simulated datasets are conducted to evaluate our approach's effectiveness in terms of time complexity and accuracy concerning the accuracy of topology and mean square error of branch estimate. The results show that our approach `MPLE` would obtain an accurate estimate and cost less time, compared with representative methods.

Likelihood-based algorithms such as `RAxML` are time-consuming to calculate full likelihood values. While, it is much easier to calculate pseudo-likelihood values instead, especially for large size of species. In this paper, we propose a method called Maximum K-subtree pseudo-likelihood estimate (**MPLE**) in which the K-subtree pseudo-likelihood function is used to evaluate the estimates.

This paper analyzes the time complexity of **MPLE** and compares it with that of the **MLE** method. Because the consistency of **MLE**/**MPLE** requires sequence length n to go to ∞, which would cause a huge number of observed patterns. In addition, when the number of taxa increases, which would also cause

a huge number of possible patterns (4^m). In both cases, our proposed method **MPLE** would definitely outperforms traditional **MLE** methods, because our method is free from n and costs $O(m^2)$ time.

For the efficiency and accuracy of the algorithm, both the full likelihood method and pseudo-likelihood method are theoretically consistent, which means both of them to converge to the true tree as sequence length n goes to infinity. So, it is reasonable to use **MPLE** to estimate the tree. **MPLE** method misspecified the marginal independence of k-subtrees, which causes loss of information when sequence length n is finite. Fisher information is used to compare the efficiency of the full likelihood function and pseudo-likelihood function. Based on our knowledge, this paper is the first one that analyzes the Fisher information of full likelihood function and compares the Fisher information with pairwise likelihood function in phylogenetics. And we found that the full likelihood function was not always better than the pairwise likelihood function for an unrooted tree with 4-taxon, which means pairwise likelihood function might obtain more accurate branch estimates than the full likelihood function would do.

A `R` package called `mple` was provided in this paper. It contains the functions to calculate pairwise likelihood values of a DNA alignment given the tree and substitution model and to search the tree space to find the maximum likelihood estimation of tree and substitution parameters.

The simulation results in Chapter 6 show that the performance of our algorithm `MPLE` would be very close to that from `RAxML` especially when n is large (> 1000). And n is usually large enough in practice. For instance, n would be 2000 for mammals on average. And `MPLE` obtained competing performance as `RAxML` did even there are only $n = 400$ bases. And the analysis of efficiency for two methods in Section **??** shows **MPLE** would obtain similar variance of estimate as **MLE** does as n increases. Thus, our approach `MPLE` is promising, because it is much faster and would not lose too much accuracy.

Chapter 8

Feature Work

Theoretically, increasing the number of taxa in K-subtree pseudo-likelihood method would improve the accuracy. It would be interesting to compare the improvement from pairwise likelihood to triple-wise likelihood method.

In this paper, we assumed that the evolutionary rate was the same across sites, which is not realistic in practice. The more general work should be the rates heterogeneous across sites (**RHAS**) models. The pseudo-likelihood method for **RHAS** models might be more complected. For example, pairwise distributions of the terminal nodes can not guarantee the identifiability of the topology when a rate factor varies across sites (Baake, 1998).

Bibliography

Allen, B. L., & Steel, M. (2001). Subtree transfer operations and their induced metrics on evolutionary trees. *Annals of combinatorics*, *5*(1), 1–15.

Allman, E. S., Ané, C., & Rhodes, J. A. (2008). Identifiability of a markovian model of molecular evolution with gamma-distributed rates. *Advances in Applied Probability*, *40*(1), 229–249.

Baake, E. (1998). What can and what cannot be inferred from pairwise sequence comparisons? *Mathematical biosciences*, *154*(1), 1–21.

Bastkowski, S., Moulton, V., Spillner, A., & Wu, T. (2016). The minimum evolution problem is hard: A link between tree inference and graph clustering problems. *Bioinformatics*, *32*(4), 518–522.

Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., & Sayers, E. W. (2010). Genbank. *Nucleic acids research*, *39*(suppl_1), D32–D37.

Besag, J. (1974). Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, *36*(2), 192–225.

Bryant, D., Galtier, N., & Poursat, M.-A. (2007). Likelihood calculation in molecular phylogenetics.

Cavalli-Sforza, L., & Edwards, A. (1963). The reconstruction of evolution. *Ann. Hum. Genet*, *27*, 105–106.

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

Cavalli-Sforza, L. L., Barrai, I., & Edwards, A. W. (1964). Analysis of human evolution under random genetic drift. *Cold Spring Harbor symposia on quantitative biology*, *29*, 9–20.

Chang, J. T. (1996). Full reconstruction of markov models on evolutionary trees: Identifiability and consistency. *Mathematical biosciences*, *137*(1), 51–73.

Choi, M. J., Tan, V. Y., Anandkumar, A., & Willsky, A. S. (2011). Learning latent tree graphical models. *Journal of Machine Learning Research*, *12*, 1771–1812.

Chor, B., & Tuller, T. (2006). Finding a maximum likelihood tree is hard. *Journal of the ACM (JACM)*, *53*(5), 722–744.

Darriba, D., Taboada, G. L., Doallo, R., & Posada, D. (2011). Prottest 3: Fast selection of best-fit models of protein evolution. *Bioinformatics*, *27*(8), 1164–1165.

Darriba, D., Taboada, G. L., Doallo, R., & Posada, D. (2012). Jmodeltest 2: More models, new heuristics and parallel computing. *Nature methods*, *9*(8), 772–772.

Dayhoff, M. O. (1969). *Atlas of protein sequence and structure* (Vol. 4). National Biomedical Research Foundation.

Elias, I., & Lagergren, J. (2009). Fast neighbor joining. *Theoretical computer science*, *410*(21-23), 1993–2000.

Felsenstein, J. (1981). Evolutionary trees from dna sequences: A maximum likelihood approach. *Journal of molecular evolution*, *17*(6), 368–376.

Felsenstein, J., & Churchill, G. A. (1996). A hidden markov model approach to variation among sites in rate of evolution. *Molecular biology and evolution*, *13*(1), 93–104.

Felsenstein, J., & Felenstein, J. (2004). *Inferring phylogenies* (Vol. 2). Sinauer associates Sunderland, MA.

Fioravanti, D., Giarratano, Y., Maggio, V., Agostinelli, C., Chierici, M., Jurman, G., & Furlanello, C. (2018). Phylogenetic convolutional neural networks in metagenomics. *BMC bioinformatics*, *19*(2), 1–13.

Fitch, W. M. (1977). On the problem of discovering the most parsimonious tree. *The American Naturalist*, *111*(978), 223–257.

Fitch, W. M., & Margoliash, E. (1967). Construction of phylogenetic trees. *Science*, *155*(3760), 279–284.

Foulds, L. R., & Graham, R. L. (1982). The steiner problem in phylogeny is np-complete. *Advances in Applied mathematics*, *3*(1), 43–49.

Groussin, M., Boussau, B., & Gouy, M. (2013). A branch-heterogeneous model of protein evolution for efficient inference of ancestral sequences. *Systematic biology*, *62*(4), 523–538.

Guindon, S., Dufayard, J.-F., Lefort, V., Anisimova, M., Hordijk, W., & Gascuel, O. (2010). New algorithms and methods to estimate maximum-likelihood phylogenies: Assessing the performance of phyml 3.0. *Systematic biology*, *59*(3), 307–321.

Hasegawa, M., Kishino, H., & Saitou, N. (1991). On the maximum likelihood method in molecular phylogenetics. *Journal of molecular evolution*, *32*(5), 443–445.

Hasegawa, M., Kishino, H., & Yano, T.-a. (1985). Dating of the human-ape splitting by a molecular clock of mitochondrial dna. *Journal of molecular evolution*, *22*(2), 160–174.

Holder, M. T., & Steel, M. (2011). Estimating phylogenetic trees from pairwise likelihoods and posterior probabilities of substitution counts. *Journal of theoretical biology*, *280*(1), 159–166.

Huelsenbeck, J. P. (1995a). Performance of phylogenetic methods in simulation. *Systematic biology*, *44*(1), 17–48.

Huelsenbeck, J. P. (1995b). The robustness of two phylogenetic methods: Four-taxon simulations reveal a slight superiority of maximum likelihood over neighbor joining. *Molecular biology and evolution*, *12*(5), 843–849.

Huelsenbeck, J. P., Larget, B., & Swofford, D. (2000). A compound poisson process for relaxing the molecular clock. *Genetics*, *154*(4), 1879–1892.

Jiang, X., Edwards, S. V., & Liu, L. (2020). The multispecies coalescent model outperforms concatenation across diverse phylogenomic data sets. *Systematic biology*, *69*(4), 795–812.

Jill Harrison, C., & Langdale, J. A. (2006). A step by step guide to phylogeny reconstruction. *The Plant Journal*, *45*(4), 561–572.

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

Kalaghatgi, P., & Lengauer, T. (2017). Computing phylogenetic trees using topologically related minimum spanning trees. *Journal of Graph Algorithms and Applications*, *21*(6), 1003–1025.

Kalyaanamoorthy, S., Minh, B. Q., Wong, T. K., Von Haeseler, A., & Jermiin, L. S. (2017). Modelfinder: Fast model selection for accurate phylogenetic estimates. *Nature methods*, *14*(6), 587–589.

Kasperski, A., & Kasperska, R. (2016). A new approach to the automatic identification of organism evolution using neural networks. *BioSystems*, *142*, 32–42.

Kimura, M. (1980). A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of molecular evolution*, *16*(2), 111–120.

Larribe, F., & Fearnhead, P. (2011). On composite likelihoods in statistical genetics. *Statistica Sinica*, 43–69.

Li, J. (2015). A fast neighbor joining method. *Genetics and molecular research: GMR*, *14*(3), 8733–8743.

Lindsay, B. G. (1988). Composite likelihood methods. *Contemporary mathematics*, *80*(1), 221–239.

Mailund, T., & Pedersen, C. N. (2004). Quickjoin—fast neighbour-joining tree reconstruction. *Bioinformatics*, *20*(17), 3261–3262.

Nguyen, L.-T., Schmidt, H. A., Von Haeseler, A., & Minh, B. Q. (2015). Iq-tree: A fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. *Molecular biology and evolution*, *32*(1), 268–274.

Paradis, E., & Schliep, K. (2019). Ape 5.0: An environment for modern phylogenetics and evolutionary analyses in r. *Bioinformatics*, *35*(3), 526–528.

Price, M. N., Dehal, P. S., & Arkin, A. P. (2009). Fasttree: Computing large minimum evolution trees with profiles instead of a distance matrix. *Molecular biology and evolution*, *26*(7), 1641–1650.

Price, M. N., Dehal, P. S., & Arkin, A. P. (2010). Fasttree 2–approximately maximum-likelihood trees for large alignments. *PloS one*, *5*(3), e9490.

Renard, D., Molenberghs, G., & Geys, H. (2004). A pairwise likelihood approach to estimation in multilevel probit models. *Computational Statistics & Data Analysis*, *44*(4), 649–667.

Roch, S. (2006). A short proof that phylogenetic tree reconstruction by maximum likelihood is hard. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *3*(1), 92–94.

Rokas, A., Williams, B. L., King, N., & Carroll, S. B. (2003). Genome-scale approaches to resolving incongruence in molecular phylogenies. *Nature*, *425*(6960), 798–804.

Rzhetsky, A., & Nei, M. (1993). Theoretical foundation of the minimum-evolution method of phylogenetic inference. *Molecular biology and evolution*, *10*(5), 1073–1095.

Rzhetsky, A., & Nei, M. (1992). A simple method for estimating and testing minimum-evolution trees.

Saitou, N., & Imanishi, T. (1989). Relative efficiencies of the fitch-margoliash, maximum-parsimony, maximum-likelihood, minimum-evolution, and neighbor-joining methods of phylogenetic tree construction in obtaining the correct tree.

Saitou, N., & Nei, M. (1987). The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, *4*(4), 406–425.

Schliep, K., Potts, A. J., Morrison, D. A., & Grimm, G. W. (2017). Intertwining phylogenetic trees and networks. *Methods in ecology and evolution*, *8*(10), 1212–1220.

Shaw, T. I., Ruan, Z., Glenn, T. C., & Liu, L. (2013). Straw: Species tree analysis web server. *Nucleic acids research*, *41*(W1), W238–W241.

Sheneman, L., Evans, J., & Foster, J. A. (2006). Clearcut: A fast implementation of relaxed neighbor joining. *Bioinformatics*, *22*(22), 2823–2824.

Simonsen, M., & Pedersen, C. N. (2011). Rapid computation of distance estimators from nucleotide and amino acid alignments. *Proceedings of the 2011 ACM Symposium on Applied Computing*, 89–93.

Sokal, R. R. (1958). A statistical method for evaluating systematic relationships. *Univ. Kansas, Sci. Bull.*, *38*, 1409–1438.

Sourdis, J., & Nei, M. (1988). Relative efficiencies of the maximum parsimony and distance-matrix methods in obtaining the correct phylogenetic tree. *Molecular biology and evolution*, *5*(3), 298–311.

Stamatakis, A. (2014). Raxml version 8: A tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, *30*(9), 1312–1313.

Studier, J. A., & Keppler, K. J. (1988). A note on the neighbor-joining algorithm of saitou and nei. *Molecular biology and evolution*, *5*(6), 729–731.

Tavaré, S. et al. (1986). Some probabilistic and statistical problems in the analysis of dna sequences. *Lectures on mathematics in the life sciences*, *17*(2), 57–86.

Thorne, J. L., Kishino, H., & Painter, I. S. (1998). Estimating the rate of evolution of the rate of molecular evolution. *Molecular biology and evolution*, *15*(12), 1647–1657.

Van der Vaart, A. W. (2000). *Asymptotic statistics* (Vol. 3). Cambridge university press.

Varin, C., Reid, N., & Firth, D. (2011). An overview of composite likelihood methods. *Statistica Sinica*, 5–42.

Yang, Z. (1994). Maximum likelihood phylogenetic estimation from dna sequences with variable rates over sites: Approximate methods. *Journal of Molecular evolution*, *39*(3), 306–314.

Yang, Z. (1995). A space-time process model for the evolution of dna sequences. *Genetics*, *139*(2), 993–1005.