Multidimensional Scaling for Phylogenetics

by

Jiarui (Erin) Zhang

B.Sc., Simon Fraser University, 2017

Project Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
Department of Statistics and Actuarial Science
Faculty of Science

SIMON FRASER UNIVERSITY
Spring 2019

Approval

Name:	**Jiarui (Erin) Zhang**
Degree:	**Master of Science (Statistics)**
Title:	**Multidimensional Scaling for Phylogenetics**
Examining Committee:	**Chair:** Jinko Graham Professor Department of Statistics and Actuarial Science
	Liangliang Wang Senior Supervisor Assistant Professor Department of Statistics and Actuarial Science
	Jiguo Cao Supervisor Associate Professor Department of Statistics and Actuarial Science
	Caroline Colijn Internal Examiner Professor Department of Mathematics
Date Defended:	**April 11, 2019**

Abstract

We study a novel approach to determine the phylogenetic tree based on multidimensional scaling and Euclidean Steiner minimum tree. Pairwise sequence alignment method is implemented to align the objects such as DNA sequences and then some evolutionary models are applied to get the estimated distance matrix. Given the distance matrix, multidimensional scaling is widely used to reconstruct the map which has coordinates of the data points in a lower-dimensional space while preserves the distance. We employ both Classical multidimensional scaling and Bayesian multidimensional scaling on the distance matrix to obtain the coordinates of the objects. Based on the coordinates, the Euclidean Steiner minimum tree could be constructed and served as a candidate for the phylogenetic tree. The result from the simulation study indicates that the use of the Euclidean Steiner minimum tree as a phylogenetic tree is feasible.

Keywords: Classical multidimensional scaling; Bayesian multidimensional scaling; sequential Monte Carlo; particle Markov Chain Monte Carlo; Steiner tree; phylogenetic tree

Acknowledgements

First, I am very grateful to my senior supervisor Liangliang Wang for her advice and patience. I would like to thank for her motivation, support and encouragement throughout my master program. She introduced me to an exciting area, phylogenetics, and shared many insights with me. Without her kind help, I cannot achieve what I have done in my thesis. Whenever I got stuck in the research, she always encouraged me and offered assistance to keep me going forward.

I would also like to thank the rest of my committee. It is my honour to have all of you as my committee and thank you for all the valuable comments and suggestions. All the comments are constructive for my further study.

I would like to express my most profound appreciation to my family for your love, understand and support throughout my study. Thank you, my fiance, for accompanying me and bringing a lot of happiness. Lastly, I would like to say a big thank you to my friends and fellow students for always being there for me.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

According to biologists' estimations, Planet Earth is home to about 5 to 100 million species of organisms today. Evidence from gene sequence data and other areas indicates that all the organisms on Earth are genetically related. We have knowledge about the organisms that are alive today, and we would like to explore their ancestors that lived thousands or millions of years ago by tracing their history back. In the field of modern biology, the most critical problems involve understanding how species are related to each other and how they are evolved through time.

Such genealogical relationships among genes, among species, or even among individuals can be represented by phylogenetic trees. In mathematics, a graph is defined as a data structure containing a set of vertices and edges, and a tree is defined as a connected graph without loops [24]. In general, for a phylogenetic tree, the vertices represent the sequence that can identify the species, for instance, DNA sequences or protein sequences. The edges represent the evolutionary changes including substitutions, insertions and deletions. The leaves or terminal nodes represent the present-day species, while the internal nodes represent their ancestors. The root of the tree represents a common ancestor of all the species. Figure 1.1 shows an example of a rooted phylogenetic tree.

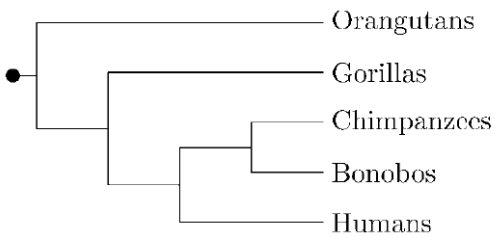

Figure 1.1: An example of a rooted phylogenetic tree. The large black dot represents a hypothetical common ancestor for all of the species. [3]

The major difficulty in constructing a phylogenetic tree is that we have minimal information about our ancestors. In fact, the ancestors represented by the internal nodes and root are usually extinct, and knowledge of sequences is unavailable. Therefore, the goal is to infer the internal nodes and study the evolutionary relationships among biological species.

There are several well-known methods for determining the phylogenetic tree. Most of the methods can be classified into two categories: character-based methods or distance-based methods. Character-based methods include maximum parsimony methods and maximum likelihood methods. Distance-based methods include the unweighted pair group method with arithmetic mean (UPGMA) method proposed by Sokal and Michener in 1958 [9] and the neighbour joining (NJ) method introduced by Saitou and Nei in 1987 [17]. In this project, we will focus on distance-based methods. The main idea of the distance-based method is to integrate the evolutionary models in calculating the pairwise distances. Then the phylogenetic tree is constructed based on the pairwise distances with some progressive clustering methods.

These distance-based methods have an advantage in speed and provide polynomial-time algorithms. Thus, the UPGMA and NJ methods have the capacity for dealing with huge data sets which contain hundreds or thousands of sequences. Despite the strength, the distance-based methods still have some limitations. One significant limitation of the distance-based methods is that it generates only a single tree, while some character-based methods compute several trees as candidates and select the optimal one. Furthermore, the distance-based methods produce different trees depending on the order of the input sequences.

In this project, we would like to propose a novel approach to construct the phylogenetic trees. The goal is to find the phylogenetic trees with high accuracy as the distance-based methods. In the pre-processing step, we use the pairwise sequence alignment to get the aligned sequences. Then we compute the pairwise dissimilarities between sequences and form the symmetric dissimilarity matrix. Once the dissimilarity matrix has been obtained, we implement both Classical multidimensional scaling and Bayesian multidimensional scaling to represent the objects by points in a low-dimensional space. We seek to find a low-dimensional configuration such that the Euclidean distances match the given pairwise dissimilarities as closely as possible. In the Bayesian multidimensional scaling, the Sequential Monte Carlo algorithm is employed to reach a Bayesian solution for the multidimensional scaling. The Particle Markov Chain Monte Carlo method is applied to estimate the model parameters.

Once the coordinates of the objects have been determined in a low-dimensional space using multidimensional scaling methods, we find a good approximation of the tree spanning the points. We construct the Euclidean Steiner minimum tree which is defined as a tree of short-

est Euclidean length that interconnects all terminals and additional Steiner points. In other words, we aim to approximate the shortest tree spanning the points in a low-dimensional space. In the end, some post-processing steps are performed to use the Euclidean Steiner minimum tree as a phylogenetic tree. These post-processing steps assure that all terminal nodes have a degree of 1 and all Steiner points (internal nodes) have a degree of 3.

The thesis is organized as follows. Some background related to phylogenetics is presented in Chapter 2. A detailed description of the methodologies which include the Classical multidimensional scaling, the Bayesian multidimensional scaling and the Steiner minimum tree problem is discussed in Chapter 3. A simulation study with some computational results is given in Chapter 4. Some concluding remarks and discussion for future research are provided in Chapter 5.

Chapter 2

Some Backgrounds in Inferring Phylogenies

2.1 Pairwise Sequence Alignment

In bioinformatics, sequence alignment is the process of arranging the sequences to identify the degree of similarity. After sequence alignment, the sequence comparison is vital in discovering the functional, evolutionary and structural relationships in biological sequences [9]. Biological sequences evolved by evolution, which involves substitutions, insertions and deletions within genes. Sequence alignment consists in inferring where these changes may have occurred within genes. Figure 2.1 shows an example of gene mutations, which includes substitution, insertion and deletion.

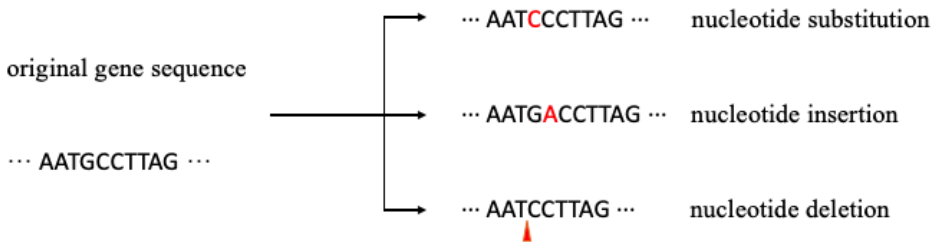

Figure 2.1: An example of gene mutations.

The alignment between the two sequences is called a pairwise alignment. The pairwise alignment is the fundamental operation in bioinformatics. It is necessary first to align the sequences, and then calculate the dissimilarities between sequences. In this project, we consider the global pairwise alignment which is an alignment over the whole length of the two DNA sequences.

Given two DNA sequences as inputs, the question is to find out how many mutations are required to transform one sequence into the other. Different scores are assigned to differ-

ent type of mutations according to the frequencies of occurrence. For instance, a positive score can be assigned to a match, and a negative score can be assigned to a mismatch. For sequences of different length, we need to take into account the gap penalty. A more substantial gap opening penalty is assigned to the first position in a gap, and a smaller gap extension penalty is assigned to every subsequent position in the same gap [4]. The goal is to find the minimum scores of mutations, which transform one sequence into the other. The Needleman-Wunsch algorithm [18] is implemented to find the optimal alignment solution given a particular scoring system.

Suppose we want to find the optimal global alignment between two sequences "GAATTC" and "GATTA". For the scoring system, we assign a score of $+2$ to a match, a score of -1 to a mismatch, a score of -2 to a gap opening and a score of -8 to a gap extension. The optimal global alignments are "GAATTC" and "GA-TTA" using the Needleman-Wunsch algorithm. The alignment includes four matches, one mismatch, and one gap of length 1. The score of the optimal solution is $(4 * 2) + (1 * -1) + [1 * (-2 - 8)] = -3$.

2.2 Pairwise Distance Estimation

The dissimilarities between DNA sequences can be calculated after we get the aligned DNA sequences. Calculation of the distance between two DNA sequences is important since obtaining the pairwise distances is the first step in phylogenetic tree construction using distance matrix methods. The pairwise distance between two sequences is defined as the expected number of nucleotide substitutions divided by the length of the sequences [24].

One simple measure of pairwise distance is called the raw distance, which is defined as the proportion of sites that are different between each pair of sequences. The idea of raw distance is straightforward, but it produces an underestimate of the number of nucleotide substitutions that have occurred. In some cases, substitutions such as multiple substitutions, parallel substitution and back substitution may happen at the same site. We may observe the same nucleotide in the two sequences because some changes are hidden in the process [24]. Under such circumstances, raw distance is not a reasonable choice to measure pairwise distance.

Some probabilistic models are proposed to describe the changes of nucleotides over the evolution. The continuous time Markov Chain is one commonly used model to explain the evolution and estimate the number of substitutions. In the continuous time Markov Chain, we assume that the nucleotide sites are independent of each other in a sequence. A continuous time Markov Chain is defined by the states of the Markov Chain which are the set

of four nucleotides {A, C, G, T}, and the transition rate matrix which summarizing the substitution rate from state i to j, with $i, j = A, C, G$, or T. Furthermore, the transition rate matrix with different constraints can lead to different molecular evolutionary models. We will use the simplest model, JC 69 model, as a demonstration to explain the process of the pairwise distance estimation.

JC69, the Jukes and Cantor 1969 model [13], assumes all nucleotides have the same base frequencies ($\pi_A = \pi_G = \pi_C = \pi_T = \lambda$). The substitution rate in JC69 model is assumed to be equal for every nucleotide. Let λ denote the instantaneous rate of one nucleotide changing into other possible nucleotide. Then, the substitution rate matrix Q is defined as follows.

$$Q = \begin{bmatrix} & A & G & C & T \\ A & -3\lambda & \lambda & \lambda & \lambda \\ G & \lambda & -3\lambda & \lambda & \lambda \\ C & \lambda & \lambda & -3\lambda & \lambda \\ T & \lambda & \lambda & \lambda & -3\lambda \end{bmatrix}$$

The diagonals of the rate matrix Q are determined by the property of the rate matrix that each row should have a sum of 0. The substitution rate matrix provides information about the substitution rate from one nucleotide to another. To consider the sequence data, we also need to calculate the probability of moving from one nucleotide to another in the one time interval t. The probability matrix can be derived from the rate matrix as follows.

$$P(t) = e^{Qt} = \begin{bmatrix} & A & G & C & T \\ A & P_1(t) & P_2(t) & P_2(t) & P_2(t) \\ G & P_2(t) & P_1(t) & P_2(t) & P_2(t) \\ C & P_2(t) & P_2(t) & P_1(t) & P_2(t) \\ T & P_2(t) & P_2(t) & P_2(t) & P_1(t) \end{bmatrix}, \text{with} \begin{cases} P_1(t) = \frac{1}{4} + \frac{3}{4}e^{-4\lambda t} \\ P_2(t) = \frac{1}{4} - \frac{1}{4}e^{-4\lambda t} \end{cases}$$

Consider two sequences of the same length n which are separated by time t, the observed proportion of different sites can be denoted as

$$\hat{p} = \frac{x}{n}, \tag{2.1}$$

where x is the number of different sites between the two sequences.

From the substitution rate matrix Q, the total substitution rate is 3λ for any nucleotide, The distance d between the two sequences separated by time t is $3\lambda t$. The two sequences can be treated as one descendant sequence and one ancestral sequence. From the probability

matrix $P(t)$, the probability that the nucleotide in one descendant sequence is different from the nucleotide in one ancestral sequence is

$$P(d) = 3P_2t = \frac{3}{4} - \frac{3}{4}e^{-4\lambda t} = \frac{3}{4} - \frac{3}{4}e^{-\frac{4}{3}d} \tag{2.2}$$

By equating the observed proportion of different sites in (2.1) and the expected probability of different sites using the JC69 model in (2.2), we can obtain an estimation of the distance between the two sequences as

$$\hat{d} = -\frac{3}{4}\log(1 - \frac{4}{3} \times \frac{x}{n}) \tag{2.3}$$

The estimated distance in equation (2.3) is calculated using the JC69 model. Other commonly used models of nucleotide substitution can also be applied to obtain the distance estimation. The process will be more complicated than the JC69 model.

2.3 Phylogenetic Tree Construction

There exist several methods for phylogeny reconstruction. This section provides some brief background of some distance-based methods. The distance-based method first calculates the pairwise distance between each pair of sequences using algorithms that integrate evolutionary models. Then, the technique reconstructs the phylogenetic tree based on the pairwise distance. From the pairwise distance matrix, the phylogenetic tree is rebuilt with a progressive clustering algorithm. Two clustering methods will be described in the following sections.

2.3.1 Unweighted pair group method with arithmetic mean (UPGMA)

UPGMA is the simplest distance-based method. UPGMA is a hierarchical clustering algorithm proposed by Sokal and Michener in 1958 [9]. The distance between two clusters is defined to be the mean distance between all elements in each cluster. In each step of the UPGMA algorithm, the smallest element in the distance matrix is found, and new clusters are created.

UPGMA can be used if the molecular clock assumption is satisfied. Specifically, if one can assume that the rate of evolution is constant, then UPGMA can be suitable to infer phylogeny. In general, the molecular clock assumption is often violated. Therefore, the method is rarely used in the application. In some situations, the technique UPGMA can be used to produce a quick initial phylogeny for some sophisticated phylogeny reconstruction methods.

2.3.2 Neighbour Joining method (NJ)

Neighbour Joining (NJ) method is another divisive clustering algorithm introduced by Saitou and Nei in 1987 [17]. The NJ method does not require the molecular clock assumption and has a speed advantage. The underlying idea of the NJ method is to join pairs of items, recompute the pairwise distances between the joined item and the remaining item and repeat the process.

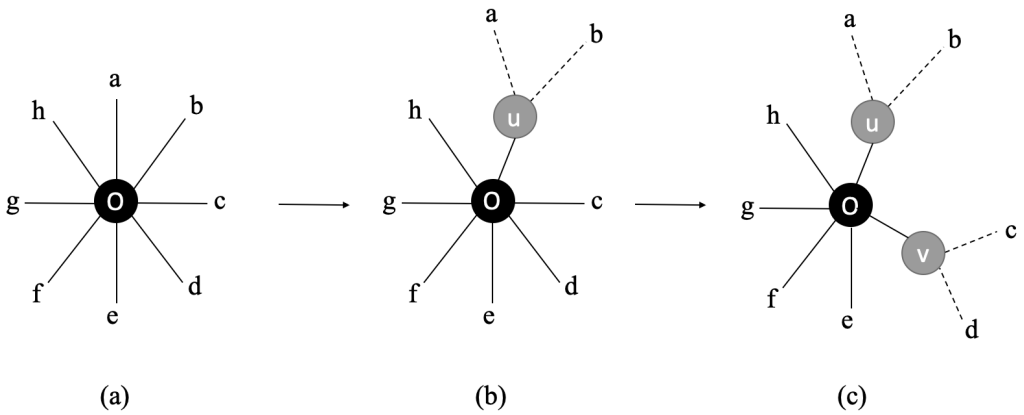

Figure 2.2: An example of Neighbor Joining method.

Figure 2.2 gives an example of the first several steps in the NJ method. NJ method constructs a phylogenetic tree from the distance matrix. The process starts with a star-like tree as seen in (a). The star tree has eight nodes connected to the root O. The pair that minimizes some criteria based on the distance matrix is chosen in (b) to join. In the example shown in Figure 2.2, node a and node b are joined and replaced by a new node u. Next, node c and node d are chosen to be joined and replaced by a new node v. In each step, the dimension of the distance matrix is reduced by one. There are several ways to assign the distance from the remaining nodes to the new node. A more detailed description of the NJ algorithm can be found in [9]. The NJ algorithm is iterated until the tree is fully resolved.

2.4 Tree Distance Metric

We measure the distance between trees using two types of distance metrics: the symmetric difference and the SPR difference. Both distance metrics focus on comparing the topology of the phylogenetic tree. The details of the distance metrics are present as follows.

Symmetric Difference (Robinson-Foulds Distance)

The Symmetric Difference or Robinson-Foulds Distance [16] is a widely used method for calculating the difference between trees. The symmetric difference is defined as the number of partitions which are not shared by the two trees. Let T_1 and T_2 be two trees with the same set of leaves. Following the notation from Steel and Penny in 1993 [20], the symmetric difference is defined as:

$$d(T_1, T_2) = i(T_1) + i(T_2) - 2 * V_s(T_1, T_2),$$

where $i(T_1)$ is the number of internal edges of tree T_1, $i(T_2)$ is the number of internal edges of tree T_2, and $V_s(T_1, T_2)$ is the number of shared internal splits.

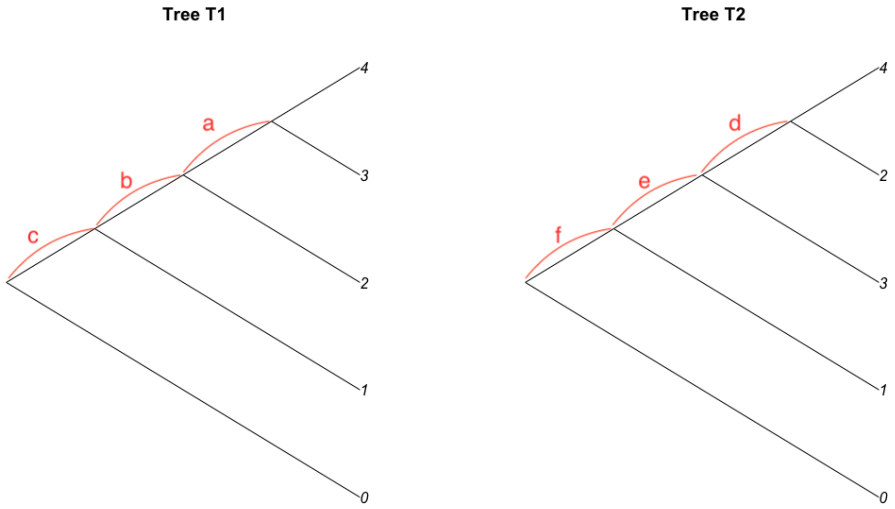

Figure 2.3: A simple example of symmetric difference.

Figure 2.3 shows an example of two trees with five leaves. Internal edges are labelled as a, b, c in tree T_1 and d, e, f in tree T_2. In total, there are three internal edges in each tree. By deleting the internal edge a and d, different splits of the leaves induce in tree T_1 and T_2. Therefore, the number of pairs of shared internal splits is 2. The symmetric difference between tree T_1 and T_2 in Figure 2.3 is $3 + 3 - 2 * 2 = 2$.

The symmetric difference only uses the tree topologies to get the difference between trees. The branch lengths of the trees are not considered in the symmetric difference metric. Overall, the symmetric difference among two trees simply counts twice the number of partitions which exist in one tree but not in the other tree. Also, the symmetric difference of a pair of

trees with n leaves ranges from 0 (if the two trees are identical) to $2n - 6$ (if the two trees do not share any partitions).

SPR Difference

Subtree pruning and regrafting (SPR) is a rearrangement operation that converts one tree to another [9]. Figure 2.4 shows an SPR move. SPR rearrangement consists of subtree prune and regraft operations. The prune operation removes an edge from the tree and forms a subtree which is disconnected to the remaining tree. The regraft operation reinserts the subtree by the same cut edge into any possible places in the remaining tree.

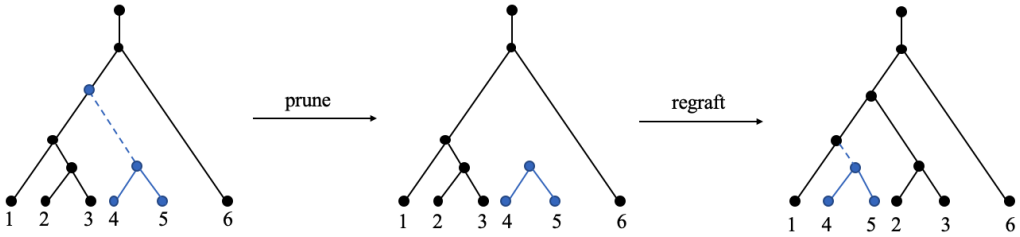

Figure 2.4: A simple SPR move.

SPR distance [2] is defined as the minimum number of SPR operations needed to transform one tree into another tree. SPR distance is proved to be an NP-hard problem by Hein et al [12]. De Oliveira Martins et al. in 2008 and 2016 [5, 6] computed the approximate SPR distance through some Bayesian approaches.

Chapter 3

Methodology

3.1 Classical Multidimensional Scaling

The general goal of multidimensional scaling (MDS) is to represent the objects by points in a multidimensional metric space using a given collection of pairwise dissimilarities between objects. Consider a dissimilarity matrix $\mathbf{D} = d_{i,j}$, MDS aims at finding $x_1, \ldots, x_n \in \mathbb{R}^p$ so that

$$d_{i,j} \approx \| x_i - x_j \|_p \text{ as closed as possible.}$$

MDS seeks to find an optimal low-dimensional configuration, usually $p = 2$ or 3. In our approach, we mainly consider representing the objects by points in a two-dimensional Euclidean space. However, we do not need to restrict our attention to low-dimensional space since the Steiner tree problem can be solved in a higher dimensional space. Thus, we also explore the performance of the methods in a higher dimensional space. We wish to match the given pairwise dissimilarities and the pairwise distance between the points in the resulting p-dimensional Euclidean space as close as possible. In this manner, the given dissimilarities are well-reproduced by the resulting spatial configuration.

Classical MDS is a commonly used method developed by Torgerson [21]. Classical MDS does not require the original data points $x_{1,\text{org}}, \ldots, x_{n,\text{org}} \in \mathbb{R}^k$. The input of the Classical MDS is the Euclidean distance matrix giving pairwise dissimilarities between objects. The output of the Classical MDS is the coordinates of points $x_1, \ldots, x_n \in \mathbb{R}^p (p < k)$ up to locations, rotations and reflections.

The steps of a Classical MDS algorithm are summarized as follows:

1. Suppose $\mathbf{D}$ is the $n * n$ symmetric distance matrix of n objects. Set up the squared distance matrix $\mathbf{D}^{(2)}$ by squaring each term in the matrix.
2. Let $\mathbf{J} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T$ be the centering matrix (where $\mathbf{1}$ is a vector of n 1's). Use double centering to get the matrix $\mathbf{B} = -\frac{1}{2}\mathbf{J}\mathbf{D}^{(2)}\mathbf{J}$.

3. Apply the spectral decomposition on the matrix $\mathbf{B} : \mathbf{B} = \mathbf{E}\mathbf{\Lambda}\mathbf{E}^T$, where $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues $\{\lambda_i\}$ and $\mathbf{E}$ is the matrix of the corresponding eigenvectors $\{e_i\}$.

4. Let p denote a chosen dimension of the solution. Determine the p largest eigenvalues $\lambda_1, \ldots, \lambda_p$ in $\mathbf{\Lambda}$ and the corresponding eigenvectors $e_1, \ldots, e_p$.

5. The coordinate matrix of points $x_1, \ldots, x_n$ using the Classical MDS is $\mathbf{X} = \mathbf{E}_p\mathbf{\Lambda}_p^{1/2}$, where $\mathbf{\Lambda}_p$ is the diagonal matrix of p eigenvalues of $\mathbf{B}$ and $\mathbf{E}_p$ is the matrix of corresponding p eigenvectors.

3.2 Bayesian Multidimensional Scaling

Classical MDS has a good performance when the given pairwise dissimilarities are exactly equal to the Euclidean distances and when the optimal low-dimensional configuration p is correctly specified [15]. In certain situations, it is reasonable to assume that there exist some measurement errors in the observed pairwise dissimilarities. Thus, we also implement a more explicit modelling framework, Bayesian MDS (BMDS), which incorporates the measurement error into the MDS [15]. In Bayesian MDS, we assume a Gaussian measurement error in the observed pairwise dissimilarities. The Sequential Monte Carlo (SMC) algorithm is employed to reach a Bayesian solution for the problem of object configuration.

3.2.1 Model and Prior Distribution

Let $\mathbf{x}_i$ denote the original unobserved vector $(x_{i,1}, \ldots, x_{i,p})$ representing the values of p significant attributes in object i. The number of significant attributes in object i is unknown in most cases. Let $d_{i,j}$ denote the observed dissimilarity measure between objects i and j. We assume that $d_{i,j}$ is observed with some Gaussian error:

$$d_{i,j} = \delta_{i,j} + \epsilon_{i,j}, \tag{3.1}$$

where $\delta_{i,j}$ denotes the true, unobserved dissimilarity measure and $\epsilon_{i,j} \sim N(0, \sigma^2)$, $i, j = 1, \ldots, n$ and $i \neq j$.

We restrict the observed dissimilarity measure $d_{i,j}$ to be always positive, and assume $d_{i,j}$ to follow the truncated normal distribution:

$$d_{i,j} \sim N(\delta_{i,j}, \sigma^2)I(d_{i,j} > 0), \qquad i \neq j,\ i, j = 1, \ldots, n, \tag{3.2}$$

where $\delta_{i,j} = \sqrt{\sum_{k=1}^{p}(x_{i,k} - x_{j,k})^2}$.

Let $\mathbf{D}_n = (d_{1,n+1},\ d_{2,n+1}, \ldots,\ d_{n,n+1})^T$ denote the pairwise dissimilarities between the $(n+1)^{th}$ $\mathbf{x}_i$ and all previous $\mathbf{x}_i$'s. At time $t = 1$, we consider one object $\mathbf{x}_1$, and there exists no pairwise dissimilarity since we only have one object. At time $t = 2$, we obtain the second object $\mathbf{x}_2$ and the pairwise dissimilarity is denoted as $d_{1,2}$. Thus, $\mathbf{D}_1$ consists of the pairwise dissimilarities between object 1 and object 2, and $\mathbf{D}_1 = (d_{1,2})^T$. A demonstration of the process for generating D_n for $n = 1, \ldots, 4$ is shown in Figure 3.1.

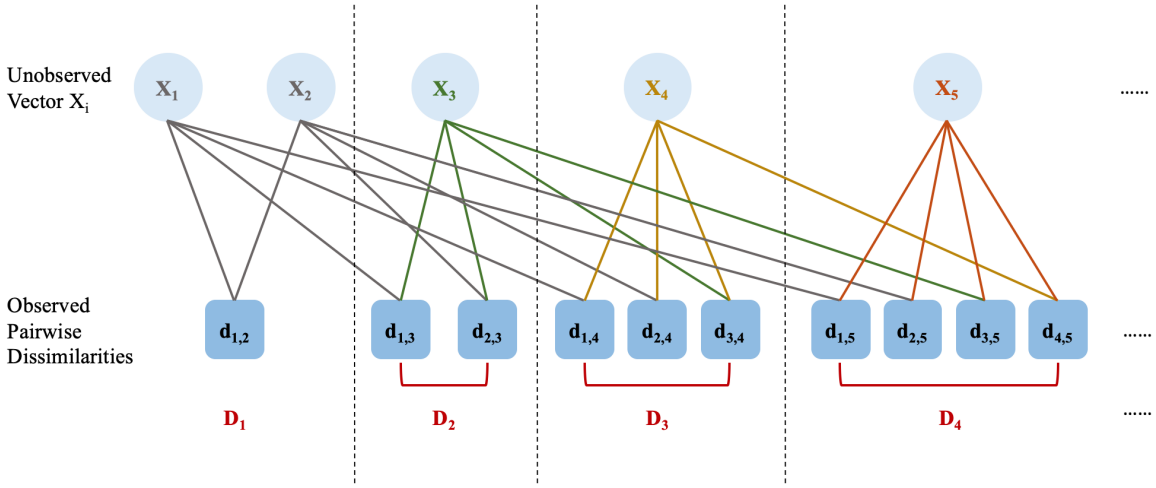

Figure 3.1: An illustration of $\mathbf{D}_n$.

The likelihood function of the unknown parameters $(\mathbf{X} = \{\mathbf{x}_{1:n}\},\ \sigma^2)$, given $\mathbf{D}_{1:n}$, is:

$$
\begin{aligned}
l(\{\mathbf{x}_{1:n}\},\ \sigma^2 \mid \mathbf{D}_{1:n}) &= \prod_{i<j} \frac{1}{\sqrt{2\pi\sigma^2}} \times \exp\{-\frac{(d_{i,j} - \delta_{i,j})^2}{2\sigma^2}\} \times \{1 - \Phi(-\frac{\delta_{i,j}}{\sigma})\}^{-1} \\
&= (2\pi\sigma^2)^{-\frac{m}{2}} \times \exp\{-\frac{1}{2\sigma^2}\mathrm{SSR} - \sum_{i<j} \log \Phi(\frac{\delta_{i,j}}{\sigma})\}, \qquad (3.3)
\end{aligned}
$$

where $\mathrm{SSR} = \sum_{i<j}(d_{i,j} - \delta_{i,j})^2$ is the sum of squared residuals, $\Phi(\cdot)$ is the standard normal cdf, and $m = n(n-1)/2$ is the total number of dissimilarities for n objects. The second term in the exponent of equation 3.3 is the modification to the normalizing constant from the truncated normal distribution.

Under the Bayesian framework, we need to specify the prior distribution for the unknown parameters $\mathbf{x}_i$ and σ^2. For $\mathbf{x}_i$, we assume a multivariate normal distribution with mean $\mathbf{0}$ and a diagonal covariance matrix Λ as the prior distribution. In other words, $\mathbf{x}_i \sim N(\mathbf{0},\ \Lambda)$, independently for $i = 1,\ \ldots,\ n$. For the elements of the diagonal covariance matrix $\Lambda = \mathrm{diag}(\lambda_1,\ \ldots, \lambda_p)$, we use an inverse Gamma distribution for the hyperprior distribution,

i.e., $\lambda_k \sim IG(\alpha,\ \beta_k)$, independently for $k = 1,\ \ldots,\ p$. For σ^2, we assume an inverse Gamma distribution for the prior distribution, i.e., $\sigma^2 \sim IG(a,\ b)$. The joint density of $(\mathbf{X} = \{\mathbf{x}_{1:n}\},\ \sigma^2,\ \Lambda)$, given $\mathbf{D}_{1:n}$, is:

$$\begin{aligned}
\pi(\{\mathbf{x}_{1:n}\},\ \sigma^2,\ \Lambda \mid \mathbf{D}_{1:n}) &= l(\{\mathbf{x}_{1:n}\}, \sigma^2 \mid \mathbf{D}_{1:n}) \times \pi(\{\mathbf{x}_{1:n}\}, \sigma^2, \Lambda) \\
&= l(\{\mathbf{x}_{1:n}\}, \sigma^2 \mid \mathbf{D}_{1:n}) \times \pi(\{\mathbf{x}_{1:n}\} \mid \Lambda) \times \pi(\sigma^2) \times \pi(\Lambda) \\
&= (2\pi\sigma^2)^{-\frac{m}{2}} \times \exp\{-\frac{1}{2\sigma^2}\mathrm{SSR} - \sum_{i<j} \log \Phi(\frac{\delta_{i,j}}{\sigma})\} \\
&\quad \times \prod_{i=1}^{n} [\det(2\pi\Lambda)^{-\frac{1}{2}} \times \exp(-\frac{1}{2}\mathbf{x}_i' \Lambda^{-1} \mathbf{x}_i)] \\
&\quad \times \frac{b^a}{\Gamma(a)} \times (\sigma^2)^{-a-1} \times \exp(-\frac{b}{\sigma^2}) \times \prod_{k=1}^{p} [\frac{\beta_k^\alpha}{\Gamma(\alpha)} \times (\lambda_k)^{-\alpha-1} \times \exp(-\frac{\beta_k}{\lambda_k})].
\end{aligned} \tag{3.4}$$

3.2.2 Sequential Monte Carlo Method (SMC)

Sequential Monte Carlo (SMC) Method is used to compute the Bayesian estimates of the unknown quantities from some given observations. In some circumstance, the observations arrive in time in a sequential way, and it is, therefore, reasonable to perform inference online [19]. Besides, the computational cost can be alleviated in the form of no need to store all the data, which is another motivating factor for solving sequential Bayesian inference problems.

The SMC method uses a set of random samples, called particles, to approximate the sequence of probability distributions of interest [7]. These particles are propagated through time using sequential importance sampling with resampling mechanisms. The major drawback of the sequential importance sampling method is that the variance of the resulting estimates tends to increase over time [8]. Thus, after a few time steps, a small number of particles' weights will be relatively large, and most particles' weights will be negligible. The SMC method implements the resampling step which associates with each particle different number of offspring proportional to its weight. Therefore, the particles with low weights have higher probabilities of being removed in the resampling step. The intuition behind this idea is that low weight particles are not carried forward and the computational efforts can be focused on regions of high probability mass [8].

We can write down the intermediate distributions at time t and $t+1$ by plugging in $n = t$ and $n = t+1$ in equation (3.4):

$$\begin{aligned}\pi_t(\{\mathbf{x}_{1:t}\},\ \sigma^2,\ \Lambda \mid \mathbf{D}_{1:t}) = (2\pi\sigma^2)^{-\frac{m_t}{2}} \times \exp\{-\frac{1}{2\sigma^2}\mathrm{SSR}_t - \sum_{\substack{i<j \\ i=1,\ \ldots,\ t-1 \\ j=1,\ \ldots,\ t}} \log\Phi(\frac{\delta_{i,j}}{\sigma})\} \\ \times \prod_{i=1}^{t}[\det(2\pi\Lambda)^{-\frac{1}{2}} \times \exp(-\frac{1}{2}\mathbf{x}_i'\Lambda^{-1}\mathbf{x}_i)] \\ \times \frac{b^a}{\Gamma(a)} \times (\sigma^2)^{-a-1} \times \exp(-\frac{b}{\sigma^2}) \times \prod_{k=1}^{p}[\frac{\beta_k^\alpha}{\Gamma(\alpha)} \times (\lambda_k)^{-\alpha-1} \times \exp(-\frac{\beta_k}{\lambda_k})],\end{aligned} \tag{3.5}$$

$$\begin{aligned}\pi_{t+1}(\{\mathbf{x}_{1:t+1}\},\ \sigma^2,\ \Lambda \mid \mathbf{D}_{1:t+1}) = (2\pi\sigma^2)^{-\frac{m_{t+1}}{2}} \times \exp\{-\frac{1}{2\sigma^2}\mathrm{SSR}_{t+1} - \sum_{\substack{i<j \\ i=1,\ \ldots,\ t \\ j=1,\ \ldots,\ t+1}} \log\Phi(\frac{\delta_{i,j}}{\sigma})\} \\ \times \prod_{i=1}^{t+1}[\det(2\pi\Lambda)^{-\frac{1}{2}} \times \exp(-\frac{1}{2}\mathbf{x}_i'\Lambda^{-1}\mathbf{x}_i)] \\ \times \frac{b^a}{\Gamma(a)} \times (\sigma^2)^{-a-1} \times \exp(-\frac{b}{\sigma^2}) \times \prod_{k=1}^{p}[\frac{\beta_k^\alpha}{\Gamma(\alpha)} \times (\lambda_k)^{-\alpha-1} \times \exp(-\frac{\beta_k}{\lambda_k})].\end{aligned} \tag{3.6}$$

We assume the normal distribution for sampling each $\mathbf{x}_i$. The initial distribution is denoted as $h_1(\mathbf{x}_1)$. The transition probability $h(\mathbf{x}_{1:\,t} \to \mathbf{x}_{1:\,t+1})$ of moving from $\mathbf{x}_t$ to $\mathbf{x}_{t+1}$ is given as:

$$h(\mathbf{x}_{1:\,t} \to \mathbf{x}_{1:\,t+1}) = (2\pi)^{-\frac{p}{2}} \prod_{k=1}^{p} \lambda_k^{-\frac{1}{2}} \times \exp(-\frac{1}{2}\mathbf{x}'_{t+1}\Lambda^{-1}\mathbf{x}_{t+1}). \tag{3.7}$$

At each time point, we need to update the weight function which considers the discrepancy between the two objects. Doucet and Holenstein provided a detailed derivation of the formula for the weight functions in [1]. At time $t = 2$, we use the importance sampling weight, $w_2(\mathbf{x}_{1:\,2})$. The unnormalized weight function $w_2(\mathbf{x}_{1:\,2})$ is defined as:

$$\begin{aligned} w_2(\mathbf{x}_{1:\,2}) &= \frac{\pi_2(\{\mathbf{x}_{1:\,2}\},\ \sigma^2,\ \Lambda \mid \mathbf{D}_{1:\,2})}{h(\mathbf{x}_1 \to \mathbf{x}_{1:\,2})} \\ &= (2\pi\sigma^2)^{-\frac{1}{2}} \times \exp\{-\frac{1}{2\sigma^2}\mathrm{SSR}_2 - \log\Phi(\frac{\delta_{1,2}}{\sigma})\} \\ &\quad \times \frac{b^a}{\Gamma(a)} \times (\sigma^2)^{-a-1} \times \exp(-\frac{b}{\sigma^2}) \times \prod_{k=1}^{p}\left[\frac{\beta_k^\alpha}{\Gamma(\alpha)} \times (\lambda_k)^{-\alpha-1} \times \exp(-\frac{\beta_k}{\lambda_k})\right]. \end{aligned} \tag{3.8}$$

After t = 2, we can update the incremental unnormalized weight function in each iteration t as follows:

$$\begin{aligned} w_t^{\text{incremental}}(\mathbf{x}_{1:\,t}) &= \frac{\pi_t(\{\mathbf{x}_{1:\,t}\},\ \sigma^2,\ \Lambda \mid \mathbf{D}_{1:\,t})}{\pi_{t-1}(\mathbf{x}_{1:\,t-1},\ \sigma^2,\ \Lambda \mid \mathbf{D}_{1:\,t-1}) \times h(\mathbf{x}_{1:\,t-1} \to \mathbf{x}_{1:\,t})} \\ &= (2\pi\sigma^2)^{-\frac{m_t - m_{t-1}}{2}} \times \exp\{-\frac{1}{2\sigma^2}(\mathrm{SSR}_t - \mathrm{SSR}_{t-1}) - \sum_{\substack{i=1,\,\ldots,\,t-1 \\ j=t}} \log\Phi(\frac{\delta_{i,j}}{\sigma})\}, \end{aligned} \tag{3.9}$$

and the new unnormalized weight function is defined as

$$w_t(\mathbf{x}_{1:\,t}) = W_{t-1}(\mathbf{x}_{1:\,t-1}) \times W_t^{\text{incremental}}(\mathbf{x}_{1:\,t}). \tag{3.10}$$

The resampling step is performed only when many particles with low weights indeed exist in the current iteration. In each iteration, we monitor the degeneracy of the particle representation with the use of the effective sampling size (ESS) [14]:

$$\mathrm{ESS} = \frac{1}{\sum_{i=1}^{N}(W_t^i(\mathbf{x}_{1:\,t}))^2}, \tag{3.11}$$

where N is the total number of particles used at each iteration in the SMC, and W_t^i is the normalized weight for particle i at iteration t. The ESS is maximized to N if all particles have the same weight, and minimized to 1 if all of the normalized weights converge to 0 except for one particle. The strategy is to set a given threshold, say N/2, and resample when the ESS is below the threshold.

The complete algorithm for our SMC method is summarized in Algorithm 1.

Algorithm 1 Standard Sequential Monte Carlo Algorithm

- At time $t = 1, 2$
  For $i = 1, \ldots, N$:

  i. Sample $\mathbf{X}_1^i \sim h_1(\mathbf{x}_1)$, and $\mathbf{X}_2^i \sim h(\mathbf{x}_1 \to \mathbf{x}_{1:\,2})$ base on equation (3.7).

  ii. Compute the unnormalized weights base on equation (3.8):
  $w_2(\mathbf{x}_{1:\,2}^i) = \frac{\pi_2(\{\mathbf{x}_{1:\,2}^i\},\, \sigma^2,\, \Lambda|\mathbf{D}_{1:\,2})}{h(\mathbf{x}_1 \to \mathbf{x}_{1:\,2})}$.

  iii. Compute the normalized weights: $W_2^i = \frac{w_2(\mathbf{x}_{1:\,2}^i)}{\sum_{j=1}^N w_2(\mathbf{x}_{1:\,2}^j)}$.

- At time $t = 3, \ldots, n$
  For $i = 1, \ldots, N$:

  i. Sample $\mathbf{X}_t^i \sim h(\mathbf{x}_{1:\,t-1} \to \mathbf{x}_{1:\,t})$ base on equation (3.7).

  ii. Compute the unnormalized weights: $w_t(\mathbf{x}_{1:\,t}^i) = w_{t-1}(\mathbf{x}_{1:\,t-1}^i) \times w_t^{\text{incremental}}(\mathbf{x}_{1:\,t}^i)$, where $w_t^{\text{incremental}}(\mathbf{x}_{1:\,t}^i)$ is calculated base on equation (3.9).

  iii. Compute the normalized weights: $W_t^i = \frac{w_t(\mathbf{x}_{1:\,t}^i)}{\sum_{j=1}^N w_t(\mathbf{x}_{1:\,t}^j)}$.

Note: At each time step t, compute the effective sample size: ESS $= \frac{1}{\sum_{i=1}^N (W_t^i(\mathbf{x}_{1:\,t}))^2}$.

- If ESS $< \frac{N}{2}$, resample $\{\mathbf{X}_t^{1:N}\}$ according to the normalized weights $\{W_t^{1:N}\}$. After resampling, all particles are assigned the same weight, i.e. $W_t^{1:N} = N^{-1}$.
- If ESS $\geq \frac{N}{2}$, do not perform resampling.

The standard SMC method we discuss above can provide an approximation for the unknown quantities of interest, i.e. $\{\mathbf{x}_{1:n}\}$. The estimation of the model parameters in the prior distribution and hyperprior distribution are not covered in this section. The Particle Markov

Chain Monte Carlo (PMCMC) method will be used to estimate the model parameters. The details of the PMCMC method will be explained in the next section.

3.2.3 Particle Markov Chain Monte Carlo (PMCMC)

Particle Markov Chain Monte Carlo (PMCMC) method is a combination of the Markov chain Monte Carlo (MCMC) and sequential Monte Carlo (SMC) methods. PMCMC method is essentially SMC iterations inside the MCMC algorithm. Both Metropolis-Hasting samplers and Gibbs samplers can be used in the MCMC algorithm. In our project, we focus on the particle marginal Metropolis-Hastings sampler (PMMH) in which SMC method acts as a Metropolis-Hasting proposal [1]. PMCMC employs the strength of its two components, and can jointly update the estimation for the model parameters and the posterior distribution, $\pi(\{\mathbf{x}_{1:n}\},\ \sigma^2,\ \Lambda \mid \mathbf{D}_{1:n})$.

Recall that the prior distribution for $\mathbf{x}_i$ is the multivariate normal distribution with mean $\mathbf{0}$ and a diagonal covariance matrix Λ. For the elements of the diagonal covariance matrix $\Lambda = \text{diag}(\lambda_1,\ \ldots, \lambda_p)$, we use an inverse Gamma distribution for the hyperprior distribution, i.e., $\lambda_k \sim IG(\alpha,\ \beta_k)$, independently for $k = 1,\ \ldots,\ p$. In addition, the prior distribution for σ^2 is also the inverse Gamma distribution, i.e., $\sigma^2 \sim IG(a,\ b)$.

Let θ denote the set which contains all model parameters to be estimated using PMMH, $\theta = \{\lambda_1,\ \ldots, \lambda_p,\ \sigma^2\}$. Let m denote the total number of MCMC iterations and $\theta^{(i)}$ denote the parameter values at the i^{th} iteration ($i = 1,\ \ldots, m$). The initial values for the model parameters can be chosen arbitrarily. In the following iterations, the new values for the model parameters, θ^*, can be proposed from the proposal distribution, denoted as $q(\theta^* \mid \theta^{(i-1)})$. We also denote the prior distribution for the model parameters as $p(\theta)$, which represents the beliefs about the parameters before taking into account the set of observations.

The model parameters, $\theta = \{\lambda_1,\ \ldots, \lambda_p,\ \sigma^2\}$, are updated using a random walk with the lognormal distribution as the proposal distribution. The details of the proposal distributions are given as follows:

$$\begin{aligned} &\log(\lambda_k^*) \sim N(\log(\lambda_k^{(i-1)}),\ \tau_{\lambda_k}^2),\ k = 1,\ \ldots,\ p, \\ &\log(\sigma^{2*}) \sim N(\log(\sigma^{2(i-1)}),\ \tau_{\sigma^2}^2). \end{aligned} \tag{3.12}$$

For the proposal distributions, several values of the variances in the lognormal distribution will be explored. The one which makes the Markov chains perform better will be selected. If the variances are too small, then it is easy to get accepted, but it explores the parameter space slowly. On the other hand, if the variances are too significant, we will encounter a

high rejection rate due to some big moves in the MCMC iterations. The ideal proposal distribution can lead to a chain that wanders thoroughly and unpredictably around the parameter space.

Assume that sampling from the conditional density $p_\theta(\mathbf{x}_{1:n} \mid \mathbf{D}_{1:n})$ is feasible for any $\theta \in \Theta$. The following form of proposal density for an Metropolis-Hastings update is suggested:

$$q\{(\theta^*, \mathbf{x}^*_{1:n}) \mid (\theta, \mathbf{x}_{1:n})\} = q(\theta^* \mid \theta) p_{\theta^*}(\mathbf{x}^*_{1:n} \mid \mathbf{D}_{1:n}). \tag{3.13}$$

The resulting Metropolis-Hastings acceptance ratio, denoted by r, is given as follows:

$$\begin{aligned} r &= \frac{p(\theta^*, \mathbf{x}^*_{1:n} \mid \mathbf{D}_{1:n})}{p(\theta^{(i-1)}, \mathbf{x}_{1:n} \mid \mathbf{D}_{1:n})} \frac{q\{(\theta^{(i-1)}, \mathbf{x}_{1:n}) \mid (\theta^*, \mathbf{x}^*_{1:n})\}}{q\{(\theta^*, \mathbf{x}^*_{1:n}) \mid (\theta^{(i-1)}, \mathbf{x}_{1:n})\}} \\ &= \frac{p_{\theta^*}(\mathbf{D}_{1:n}) p(\theta^*)}{p_{\theta^{(i-1)}}(\mathbf{D}_{1:n}) p(\theta^{(i-1)})} \frac{q(\theta^{(i-1)} \mid \theta^*)}{q(\theta^* \mid \theta^{(i-1)})}. \end{aligned} \tag{3.14}$$

At each iteration, a decision must be made to either stay on the current values $\theta^{(i-1)}$ or update to the new values θ^* generated by the proposal density. Let u denote a value which is generated from a Uniform(0, 1) distribution. The process of updating the parameter values are as follows:

- If $u \leq r$, then accept θ^* and set $\theta^{(i)} = \theta^*$.

- If $u > r$, then reject θ^* and set $\theta^{(i)} = \theta^{(i-1)}$.

An alternative to the Metropolis-Hastings Ratio is the log Metropolis-Hastings Ratio. The log Metropolis-Hastings Ratio could be used for the sake of simplicity in calculations. The log Metropolis-Hastings Ratio is:

$$\log(r) = \log\left[\frac{p_{\theta^*}(\mathbf{D}_{1:n})}{p_{\theta^{(i-1)}}(\mathbf{D}_{1:n})}\right] + \log\left[\frac{p(\theta^*)}{p(\theta^{(i-1)})}\right] + \log\left[\frac{q(\theta^{(i-1)} \mid \theta^*)}{q(\theta^* \mid \theta^{(i-1)})}\right]. \tag{3.15}$$

In equation (3.15), the first term is the log marginal likelihood ratio, the second term is the log prior ratio, and the third term is the log proposal ratio. At each iteration, a decision whether stay on the current values $\theta^{(i-1)}$ or update to the new values θ^* can be made by comparing $\log(r)$ to $\log(u)$. The remaining steps in the PMCMC algorithm stay the same.

The complete algorithm for our PMCMC method is summarized in Algorithm 2.

Algorithm 2 Particle Markov Chain Monte Carlo Algorithm

- Initialization, $i = 0$

  (a) Set initial parameter values $\theta^{(0)}$ arbitrarily.

  (b) Run an SMC algorithm with $\theta^{(0)}$, the marginal likelihood estimate is:

$$\hat{p}_{\theta^{(0)}}(\mathbf{D}_{1:n}) = \prod_{t=1}^{n} \left[\frac{1}{N} \sum_{i=1}^{N} w_t(\mathbf{x}_{1:\,t}^{i(0)}) \right].$$

- For iteration $i \geq 1$:

  (a) Propose new parameter values, $\theta^* \sim q(\cdot \mid \theta^{(i-1)})$

  (b) Run an SMC algorithm with θ^*, the marginal likelihood estimate is:

$$\hat{p}_{\theta^*}(\mathbf{D}_{1:n}) = \prod_{t=1}^{n} \left[\frac{1}{N} \sum_{i=1}^{N} w_t(\mathbf{x}_{1:\,t}^{i*}) \right].$$

  (c) Compute the Metropolis-Hastings Ratio:

$$r = \frac{\hat{p}_{\theta^*}(\mathbf{D}_{1:n}) p(\theta^*)}{\hat{p}_{\theta^{(i-1)}}(\mathbf{D}_{1:n}) p(\theta^{(i-1)})} \frac{q(\theta^{(i-1)} \mid \theta^*)}{q(\theta^* \mid \theta^{(i-1)})}.$$

  (d) Generate $u \sim \text{Uniform}(0, 1)$ distribution.

  - If $u \leq r$, then accept θ^*. Set $\theta^{(i)} = \theta^*$ and $\hat{p}_{\theta^{(i)}}(\mathbf{D}_{1:n}) = \hat{p}_{\theta^*}(\mathbf{D}_{1:n})$.
  - If $u > r$, then reject θ^*. Set $\theta^{(i)} = \theta^{(i-1)}$ and $\hat{p}_{\theta^{(i)}}(\mathbf{D}_{1:n}) = \hat{p}_{\theta^{(i-1)}}(\mathbf{D}_{1:n})$.

3.3 Spanning Tree

Once the coordinates of the objects have been determined in a p-dimensional space using multidimensional scaling, we need to find a good approximation of the tree spanning the given set of points using the matrix of pairwise distance metrics.

Given an undirected graph with edge lengths and a finite set of vertices, also referred to as terminals, the minimum spanning tree is a tree that connects all the terminals with minimum possible total edge lengths. In the minimum spanning tree, only direct connections between the terminals are allowed [3]. The Steiner minimum tree is defined as a tree of shortest length that interconnects all terminals [22]. Contrary to the minimum spanning tree problem, connections in the Steiner minimum tree are not restricted to be between the terminals. The Steiner minimum tree may also introduce some additional junctions, so-called Steiner points. Thus, the shortest Steiner tree may include the terminals as well as some Steiner points. The minimum spanning tree problem is solvable in polynomial time whereas the Steiner minimum tree problem is NP-hard which means there is currently no polynomial-time algorithm known for solving it.

However, we choose to consider the Steiner minimum tree in this project rather than the minimum spanning tree. There are two main reasons for our decision. The first reason is that the Steiner minimum tree usually has a substantially shorter total edge lengths than the minimum spanning tree. The second reason is that the topology of the Steiner minimum tree naturally produces a candidate for a phylogenetic tree [3]. Only some simple post-processing steps are needed to use the Steiner minimum tree as a phylogenetic tree of interest. These post-processing steps will be discussed later.

In this project, the Euclidean Steiner minimum tree is constructed. In other words, the pairwise Euclidean distances are calculated from the given set of terminals. The input of the Steiner tree problem is a finite set of terminals. The output provides the Euclidean Steiner minimum tree, which is defined as a tree of shortest Euclidean length that interconnects all terminals and additional Steiner points.

Some properties and definitions are introduced first to make it easier to understand the Euclidean Steiner minimum tree problem [22]. Some basic features of the Steiner minimum trees are listed as follows:

- Angle Condition

  - Steiner points have a degree of 3. The three incident edges are co-planar, and each pair makes an angle of 120°.

- Degrees of Vertices

  - All Steiner points have a degree of 3, and all terminals have a degree of less than or equal to 3.

- Number of Steiner points

  - A Steiner minimum tree with n terminals has at most $n-2$ Steiner points.

A full Steiner tree is defined as the tree with the maximum number of Steiner points, which is $n-2$. A full Steiner tree has all terminals as the leaves of the tree. In other words, all terminals in a full Steiner tree have a degree of 1. Full topologies are defined as all possible tree topologies which interconnect n terminals and $n-2$ Steiner points. Given a full topology, the unique full Steiner tree can be found in polynomial time if it exists. Figure 3.2 shows three full topologies for 4 terminals.

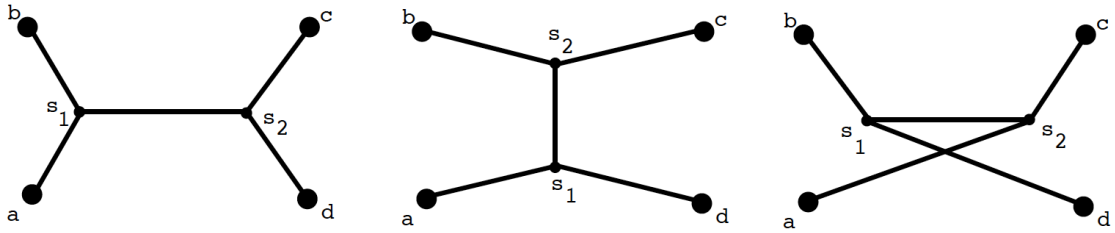

Figure 3.2: Different full topologies for 4 terminals [22].

The Steiner minimum trees are unions of full Steiner trees [22]. In this case, a possible approach to get the Steiner minimum tree involves enumerating all possible full Steiner trees, pruning some non-optimal ones, and concatenate the surviving full Steiner tree. Even though the idea is straightforward, the problem of finding all possible full Steiner tree is extremely time-consuming. For n terminals, the total number of full topologies is given as

$$f(n) = \sum_{k=2}^{n} \begin{pmatrix} n \\ k \end{pmatrix} f(k), \tag{3.16}$$

where $f(k) = \frac{(2k-4)!}{2^{k-2}(k-2)!}$ is a super-exponential function in k [22]. This means that $f(k)$ increases even faster than an exponential function. The total number of full topologies is a huge number even for a small set of terminals. Therefore, the approach of attempting to enumerate all possible full Steiner trees is almost impossible.

In this project, we used the GeoSteiner package to solve the Euclidean Steiner Tree Problem in 2-dim case. GeoSteiner package stands for the computational state of the art for the Euclidean Steiner tree problem. In brief, the algorithm employed by GeoSteiner package first

enumerates equilateral points which will be used repeatedly during the process of enumerating full topologies. Then for each surviving equilateral points after some pruning tests, a full Steiner tree can be identified. The final step is to concatenate the full Steiner trees to find the shortest possible union of the full Steiner trees interconnecting all terminals. A more detailed description of the algorithm is presented in [23].

For some higher dimensional cases, we implemented a simple Steiner tree heuristic method as described in [3]. There are two phases in this simple Steiner tree heuristic method. In the first phase, a Euclidean minimum spanning tree for all terminals is constructed, and some local improvements could be performed to decrease the tree length. In the second phase, some short-cutting steps are performed to decrease the tree length further.

As noted above, some post-processing steps are needed to use the Euclidean Steiner minimum tree as a phylogenetic tree of interest. These post-processing steps ensure that all Steiner points have a degree of 3 and all terminals have a degree of 1. For a terminal of degree higher than 1, we can insert a new Steiner point with the same coordinates and replace the terminal with the new Steiner point. Then the terminal can be connected to the new Steiner point by a zero-length edge (or an edge with very short length).

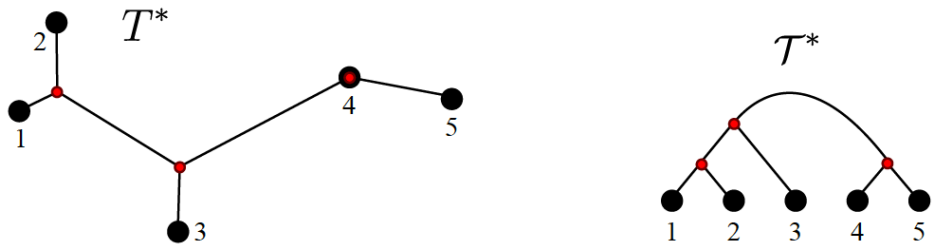

Figure 3.3: An example of the Euclidean Steiner minimum tree and the corresponding tree topology [10].

The left panel in Figure 3.3 is the Euclidean Steiner minimum tree in $\mathbb{R}^2$. Five terminals are shown as filled black circles, and Steiner points are shown as filled red circles. Terminal 4 has a degree of 2, so the post-processing step is performed. A new Steiner point with the same coordinates as Terminal 4 is inserted, and terminal 4 is connected to the new Steiner point with a zero-length edge. The right panel in Figure 3.3 is the corresponding topology of the Euclidean Steiner minimum tree.

Chapter 4

Simulation Studies

4.1 Simulation Design

We used the HKY85 model of DNA evolution in the process of simulating DNA sequences. HKY85, the Hasegawa, Kishino and Yano 1985 model [11], allows nucleotides to occur at different frequencies ($\pi_A \neq \pi_G \neq \pi_C \neq \pi_T$) and accounts for the difference between the rate of transitions ($A \leftrightarrow G, C \leftrightarrow T$) and transversions with the parameter κ. The substitution rate matrix Q is defined as follows.

$$Q = \begin{bmatrix} & A & G & C & T \\ A & * & \kappa\pi_G & \pi_C & \pi_T \\ G & \kappa\pi_A & * & \pi_C & \pi_T \\ C & \pi_A & \pi_G & * & \kappa\pi_T \\ T & \pi_A & \pi_G & \kappa\pi_C & * \end{bmatrix}$$

In the simulation setting, we assumed that the base frequencies are 0.2, 0.2, 0.3 and 0.3 for the four states A, G, C and T and $\kappa = 1.1$.

There are two parts in the simulation study. The first part aims to validate the methodology of the Classical MDS and Steiner tree and compare the performance of the methods with some existing methods. In the second part, we propose to measure the performance of the Bayesian MDS by comparing the estimated distance matrix with the actual distance matrix.

The first part contains three experiments which mainly focus on the Classical MDS and Steiner tree. In the first experiment, we want to check the performance of the Classical MDS and Steiner tree on a small example to establish that the approach can work. We generated a small tree with 5 leaves and used parameters π_A, π_G, π_C, π_T, κ to evolve the ancestral sequences down the tree. We generated a nucleotide sequence of length 2000 for each leaf. Then a matrix of pairwise dissimilarities from DNA sequences using raw distance was computed. The raw distance was calculated as the proportion of sites that are different

between each pair of DNA sequences.

In the second experiment, we would like to examine the relationship between the number of dimensions selected in the Classical MDS and the symmetric tree difference. We generated a random tree with 10 leaves and the nucleotide sequences with the same parameters as in the first experiment. For each leaf, a nucleotide sequence of length 2000 was generated. We calculated the raw distance to get the pairwise distance matrix from DNA sequences.

In the third experiment, the goal is to compare the phylogenetic tree generated from our method in a given dimension with some existing methods. In this experiment, we created the random tree and the nucleotide sequences with the same parameters as in the first experiment. The difference was that the random tree with 20 leaves was generated. The matrix of pairwise dissimilarities was computed using raw distance, JC69 model, K80 model and F81 model.

Once we obtained the pairwise dissimilarity matrices, the Classical MDS was directly applied to get the coordinates of the leaves. Then we searched the Euclidean Steiner minimum trees and applied some post-processing steps to acquire the phylogenetic tree. To evaluate the performance of the methods, we used both the Symmetric difference and the SPR difference. For the Symmetric difference, a normalized version was applied so that the maximum Symmetric difference was 1.

We examined the performance of the Bayesian MDS in the second part. In the Bayesian MDS, the SMC and PMCMC algorithms were implemented to get the coordinates of the leaves. The estimated distance matrix was calculated to check how well Bayesian MDS retains the information required to find good phylogenetic trees.

We generated the true values of $\mathbf{x}_i$ from the multivariate normal distribution with mean $\mathbf{0}$ and a diagonal covariance matrix $\Lambda = \begin{bmatrix} 1 & 0 \\ 0 & 0.64 \end{bmatrix}$. The true value for σ^2 was set to 0.01. The true pairwise distances were calculated by adding some Gaussian errors to the pairwise distance between $\mathbf{x}_i$'s.

The PMCMC algorithm was run using $N = 2000$ particles and $m = 1500$ MCMC iterations. For p-dim, the initial parameter values were set to $a = 1$, $b = 1$, $\alpha = 1$, $\beta = \begin{bmatrix} 1 & \dots & 1 \end{bmatrix}_{1\times p}$, $\sigma^2 \sim IG(a,\ b)$ and $\lambda_k \sim IG(\alpha,\ \beta_k)$, independently for $k = 1,\ \dots,\ p$. In the proposal distribution, the values of the standard deviations in the lognormal distribution were set to $\tau^2_{\lambda_k} = 0.1$ for $k = 1,\ \dots,\ p$ and $\tau^2_{\sigma^2} = 0.1$.

4.2 Simulation Results

4.2.1 Classical MDS with Steiner Tree

For the first experiment, we simulated 20 generated data instances with 5 sequences each. The success rate of giving the correct phylogenetic tree for the Classical MDS with Steiner tree is 85%. The possible sources of error include the multidimensional scaling, the heuristic that obtains the Steiner tree, and the fact that the Steiner tree may differ from the correct phylogenetic tree. Overall, the Classical MDS can provide a good representation of the distances, and the Steiner tree based on the Classical MDS has a relatively high success rate.

For the second experiment, the results of applying Euclidean Steiner minimum tree to the pairwise distance matrix obtained by the multidimensional scaling with an increasing number of dimensions from 2-dim to 5-dim are the primary interests. Figure 4.1 shows the relationship between the symmetric tree difference and number of dimensions. The results are the averages of applying Classical MDS and Euclidean Steiner minimum tree method to 10 generated data instances with 10 sequences each. We conclude that an increasing number of dimensions can yield better results. The results are reasonable since points in a higher dimension contain more information which can lead to a better approximation.

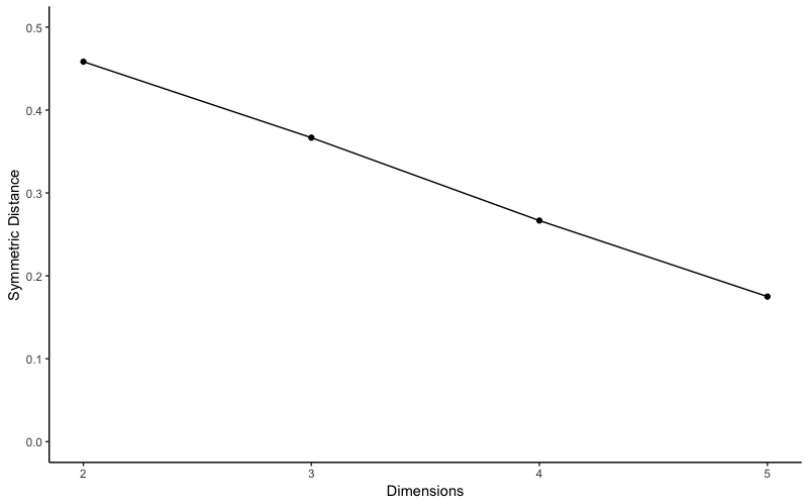

Figure 4.1: Results of applying Euclidean Steiner minimum tree method to the distance matrices obtained by Classical MDS with dimensions from 2 to 5.

Figure 4.2 displays the true phylogenetic tree from a random simulation. Figure 4.3 shows the reconstructed phylogenetic tree using the Classical MDS with dimensions from 2 to 5. In this example, the reconstructed phylogenetic tree using the Classical MDS with dimension 5 yields the correct phylogeny in terms of the tree topology.

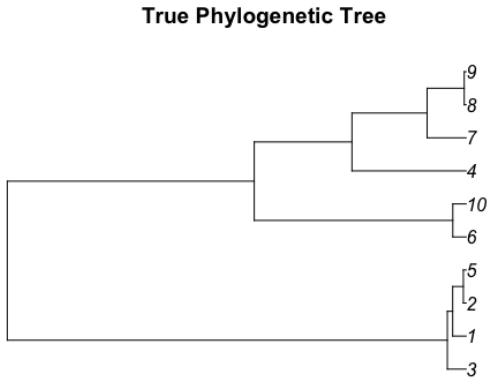

Figure 4.2: The topology of the true phylogenetic tree in a random simulation.

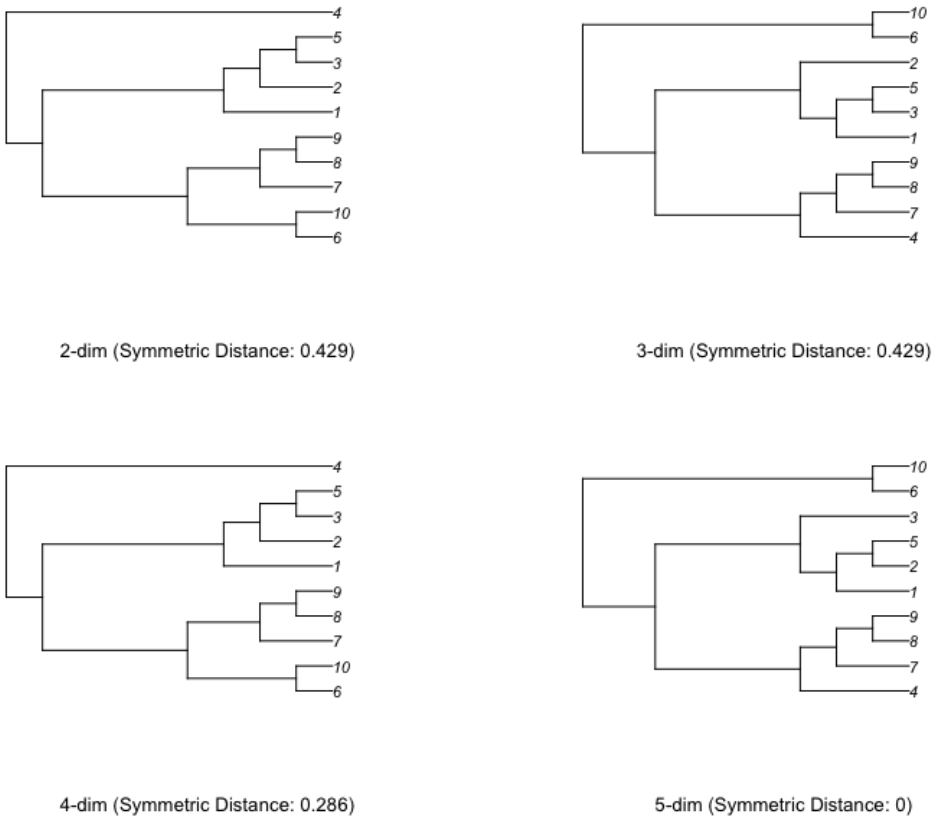

Figure 4.3: The topology of the reconstructed phylogenetic tree by applying Euclidean Steiner minimum tree method to the distance matrices obtained by Classical MDS with dimensions from 2 to 5.

For the third experiment, the phylogenetic trees generated from our methods were compared with some existing methods in 2-dim. Table 4.1 and Table 4.2 display the results of the tree distances based on the pairwise dissimilarities computed using raw distance, JC69 model, K80 model and F81 model. The results are the averages of applying multidimensional scaling and Euclidean Steiner minimum tree method to 10 generated data instances with 20 sequences each.

Model	Classical MDS, Steiner Tree	UPGMA	NJ
raw	0.529	0.1	0.0941
JC69	0.573	0.565	0.588
K80	0.559	0.697	0.665
F81	0.574	0.635	0.694

Table 4.1: Symmetric Difference between the original tree and the resulting tree for 2-dim.

Model	Classical MDS, Steiner Tree	UPGMA	NJ
raw	6.0	1.6	1.4
JC69	5.8	4.9	5.1
K80	5.5	5.5	5.4
F81	5.7	5.1	5.9

Table 4.2: SPR Difference between the original tree and the resulting tree for 2-dim.

From Table 4.1 and Table 4.2, Classical MDS with Steiner tree method is comparable with the UPGMA and NJ methods in most cases. Only for the distance matrix estimated using raw distance, both UPGMA and NJ have excellent performance in finding the correct tree.

4.2.2 Bayesian MDS

In the Bayesian multidimensional scaling, we need to keep track of the estimated parameters for each PMCMC iteration. Thus, we created some plots to trace the iterations and check the performance of the PMCMC algorithm.

Figure 4.4 shows the trace plots of the parameters resulting from each PMCMC iteration. In this case, we used a burn-in of about 500 iterations to ensure the convergence of the Markov chain.

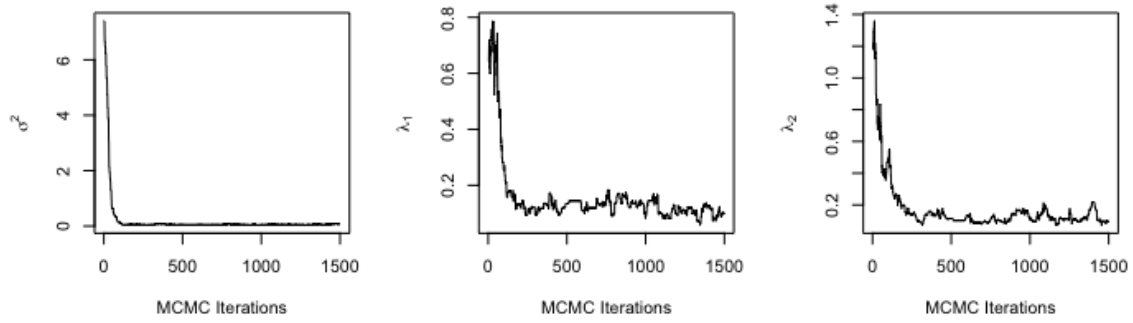

Figure 4.4: Trace plots of σ^2, λ_1, λ_2 from PMCMC iterations for 2-dim case.

Figure 4.5 displays the histograms of the parameters σ^2, λ_1, λ_2 draws after the burn-in period.

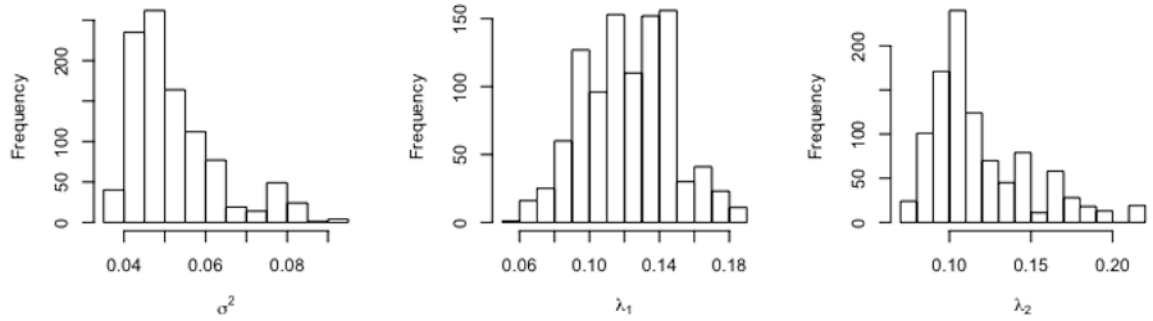

Figure 4.5: Histograms of σ^2, λ_1, λ_2 from PMCMC iterations for 2-dim case.

In the PMCMC algorithm, the acceptance rate is 0.2827. After the burn-in period, the posterior means of σ^2, λ_1 and λ_2 were calculated and served as the estimated model parameters in the SMC algorithm.

We implemented the Bayesian MDS with dimension 2 to obtain the distance matrix. Table 4.3 shows the estimated distances which are the posterior means with the 95% credible intervals for every pair of nodes. The results in Table 4.3 is computed for 5 nodes. For all pairs, the Bayesian MDS provides reasonably estimated distances which means that the Bayesian MDS can retain the information and give a low-dimensional configuration. The analysis was also completed for the Bayesian MDS with nodes numbers 10, 15, and 20. The results showed that the estimated pairwise distances in the 2-dim space were close to

the given pairwise distances. Thus, the combination of the Bayesian MDS and Steiner tree could be an alternative way to produce good phylogenetic trees.

Pair	Estimated Distance	95% Credible Interval	True Distance
(1, 2)	2.232	(0.076, 12.464)	1.558
(1, 3)	1.706	(0.057, 9.010)	0.846
(1, 4)	1.426	(0.047, 7.502)	1.248
(1, 5)	2.461	(0.120, 11.541)	0.696
(2, 3)	1.767	(0.064, 9.606)	1.479
(2, 4)	1.580	(0.049, 8.357)	1.671
(2, 5)	2.573	(0.085, 11.857)	2.146
(3, 4)	1.111	(0.050, 4.684)	0.407
(3, 5)	1.704	(0.097, 6.810)	1.163
(4, 5)	2.666	(0.046, 4.933)	1.468

Table 4.3: Summary of the posterior means with 95% credible intervals from the SMC algorithm in the Bayesian MDS.

Chapter 5

Conclusion

We have explored some methods for constructing phylogenetic trees using multidimensional scaling and Steiner tree. In the multidimensional scaling step, we applied both Classical MDS and Bayesian MDS to the distance matrix. The goal of multidimensional scaling is to project the data points in a higher-dimensional space onto a lower-dimensional subspace while preserving the distances between points as much as possible. Classical MDS is the most straightforward and most fundamental approach to accomplish this task. Furthermore, there are various techniques which can also provide reasonable solutions to the dimension reduction problem. Besides the Classical MDS, we also applied the Bayesian approach to object configuration in multidimensional scaling. More specifically, Bayesian MDS incorporates the measurement error into the Classical MDS. A cautionary note is that the assumptions of the types of distributions in the Bayesian MDS are difficult to verify. The SMC algorithm which involves resampling in each sequential importance sampling iteration is employed to reach a Bayesian solution. In the SMC algorithm, resampling is a way to prune particles with low weights while preserving consistency. The PMCMC algorithm is used to update the estimation for the model parameters.

For the 2-dim case, the performance of our methods with Classical MDS is comparable with the UPGMA method and the NJ method in terms of the Symmetric difference and the SPR difference between the reconstructed tree and the original tree. Moreover, as we showed in the second experiment in the simulation study, the increasing number of dimensions can yield better results. Thus, using more dimensions in Classical MDS may deliver a higher quality tree. For the Bayesian MDS, instead of using the SMC algorithm, we could also explore implementing the MCMC algorithm for object configuration. The concept of applying multidimensional scaling and Steiner tree is proved as a viable approach for finding the phylogenetic trees. The resulting phylogenetic tree could be used as the initial tree in the Bayesian MCMC exploration of the phylogenetic tree space.

Although the results of building phylogenetic trees using multidimensional scaling and Steiner tree are promising, the NJ method still has virtues of simplicity and efficiency which make it practical to analyze massive data sets which contain hundreds or thousands of sequences. Thus, some extensions to our approach could be suggested. First, we could explore some more advanced improvements in the multidimensional scaling which could, in turn, improve the results of the Steiner tree problem. Second, we could develop some tuning process to discover the optimal number of dimensions in the solution and the optimal values of parameters for the proposal distribution in the PMCMC. These improvements could further boost the performance of the SMC algorithm. Third, we could make use of the coordinates information in the solution of Steiner tree problem to construct the tree with branch length. In that case, we could also consider the correctness in estimates of the branch lengths.

Bibliography

[1] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

[2] Robert G Beiko and Nicholas Hamilton. Phylogenetic identification of lateral genetic transfer events. *BMC evolutionary biology*, 6(1):15, 2006.

[3] Marcus Brazil, Doreen A Thomas, Benny K Nielsen, Pawel Winter, Christian Wulff-Nilsen, and Martin Zachariasen. A novel approach to phylogenetic trees: d-dimensional geometric Steiner trees. *Networks: An International Journal*, 53(2):104–111, 2009.

[4] Avril Coghlan. Little book of R for bioinformatics, 2011.

[5] Leonardo De Oliveira Martins, Elcio Leal, and Hirohisa Kishino. Phylogenetic detection of recombination with a Bayesian prior on the distance between trees. *PLoS One*, 3(7):e2651, 2008.

[6] Leonardo De Oliveira Martins, Diego Mallo, and David Posada. A Bayesian supertree model for genome-wide species tree reconstruction. *Systematic biology*, 65(3):397–416, 2014.

[7] Arnaud Doucet, Mark Briers, and Stéphane Sénécal. Efficient block sampling strategies for sequential Monte Carlo methods. *Journal of Computational and Graphical Statistics*, 15(3):693–711, 2006.

[8] Arnaud Doucet and Adam M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

[9] Joseph Felsenstein and Joseph Felenstein. *Inferring phylogenies*, volume 2. Sinauer associates Sunderland, MA, 2004.

[10] Rasmus Fonseca, Marcus Brazil, Pawel Winter, and Martin Zachariasen. Faster exact algorithms for computing Steiner trees in higher dimensional Euclidean spaces. *11th DIMACS Implementation Challenge. Brown University*, 2014.

[11] Masami Hasegawa, Hirohisa Kishino, and Taka-aki Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *Journal of molecular evolution*, 22(2):160–174, 1985.

[12] Jotun Hein, Tao Jiang, Lusheng Wang, and Kaizhong Zhang. On the complexity of comparing evolutionary trees. *Discrete Applied Mathematics*, 71(1-3):153–169, 1996.

[13] Thomas H Jukes, Charles R Cantor, et al. Evolution of protein molecules. *Mammalian protein metabolism*, 3(21):132, 1969.

[14] Jun S Liu. *Monte Carlo strategies in scientific computing*. Springer Science & Business Media, 2008.

[15] Man-Suk Oh and Adrian E Raftery. Bayesian multidimensional scaling and choice of dimension. *Journal of the American Statistical Association*, 96(455):1031–1044, 2001.

[16] David F Robinson and Leslie R Foulds. Comparison of phylogenetic trees. *Mathematical biosciences*, 53(1-2):131–147, 1981.

[17] Naruya Saitou and Masatoshi Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4):406–425, 1987.

[18] Manfred J Sippl. Biological sequence analysis. Probabilistic models of proteins and nucleic acids. *Protein Science*, 8(3):695–695, 1999.

[19] Adrian Smith. *Sequential Monte Carlo methods in practice*. Springer Science & Business Media, 2013.

[20] Mike A Steel and David Penny. Distributions of tree comparison metrics - some new results. *Systematic biology*, 42(2):126–141, 1993.

[21] Warren S Torgerson. Multidimensional scaling: I. Theory and method. *Psychometrika*, 17(4):401–419, 1952.

[22] Pawel Winter and Martin Zachariasen. *Large Euclidean Steiner minimum trees in an hour*. Citeseer, 1996.

[23] Pawel Winter and Martin Zachariasen. Euclidean Steiner minimum trees: An improved exact algorithm. *Networks: An International Journal*, 30(3):149–166, 1997.

[24] Ziheng Yang. *Molecular evolution: a statistical approach*. Oxford University Press, 2014.