Maximum Likelihood Star Alignment of Multiple Molecular Sequences

MAXIMUM LIKELIHOOD STAR ALIGNMENT OF MULTIPLE MOLECULAR SEQUENCES

By

ZHIGEN JIANG, Ph. D. (Mathematics)

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Master of Engineering

McMaster University

June 1996

MASTER OF ENGINEERING (1996) MCMASTER UNIVERSITY

(Electrical and Computer Engineering) Hamilton, Ontario

TITLE: **Maximum Likelihood Star Alignment of Multiple Molecular Sequences**

AUTHOR: Zhigen Jiang

B. Sc. (Mathematics, Nanjing University, China)

Ph. D. (Mathematics, Chinese Academy of Sciences)

SUPERVISOR: Dr. Tao Jiang

Associate Professor

Department of Computer Science and Systems

B.Sc. (Univ. of Sci. and Tech. of China, Hefei)

Ph.D. (Minnesota)

NUMBER OF PAGES: ix, 71

Abstract

In the study of pairwise sequence alignment, a clear relationship between the scoring system and assumptions about the occurrence of evolutionary events has been established in [BT86], [TKF91], [TKF92] and [TC95] by proposing an evolutionary model. To align two given sequences, one need estimate some evolutionary parameters through maximum likelihood method, and find an alignment with the maximum probability using the estimated parameters.

In this thesis, we extend the above model and the maximum likelihood method to star alignment of three molecular sequences along the same line. We overcome the duplications of star alignments by defining canonical star alignments. Two star alignment algorithms, i.e. sum approach and direct alignment approach, are proposed in this thesis based on two different likelihood functions. A software system, called MLSAS (Maximum Likelihood Star Alignment System), is developed to implement the two algorithms with a friendly graphical user interface. Simulation studies show their behaviors are satisfactory for closely related sequences. A few real examples are also provided.

Acknowledgments

I would like to express my sincere gratitude to my supervisor Dr. Tao Jiang for his expert guidance, continued assistance, and supervision throughout the course of this work.

Thanks also go to Dr. Brian Golding for his useful suggestions and supplying real DNA sequences and Mr. Xiangdong Chen for his guidance on MOTIF programming.

Much appreciation goes to Mr. Dan Trottier for his help on computer systems and related software.

Thanks to my wife, my mother and my father for their love, patience, constant support and encouragement.

This project was supported in part by Canadian Genome Analysis and Technology Grant GO-12278 and NSERC Research Grant OGP0046613.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

With the advent of modern molecular biology, the ability to collect biological sequence data has outpaced the ability to adequately analyze this data. One tool for reducing this surfeit of inadequately treated data is sequence alignment. A sequence alignment is a hypothesis about the evolutionary correspondence between the bases in a pair of sequences. A common representation of an alignment (Figure 1.1) is to exhibit the bases of one sequence on a line above those of the other sequence. Corresponding bases appear stacked one above the other. When two bases in a column are the same type, the alignment position is termed a match. When corresponding bases are different, the alignment position is termed a mismatch and at least one substitution event must have occurred. In this "stacked" alignment representation, a base that has no corresponding base in the other sequence is said to be opposite a space. If we consider sequence A to be ancestral to sequence B, spaces in sequence A are the result of insertion events and spaces in sequence B are the result of deletion events. In a pairwise alignment, insertions cannot be distinguished from deletions. Therefore, the term indel is used to describe an evolutionary event that may be either an insertion or a deletion.

```
A T A G A G - T T T G T A C G
- T A G C G G T T C G T T C G
```

Figure 1.1: A common "stacked" representation of a sequence alignment

Because a single-base indel leads to a single-base space in the alignment and because a nucleotide mismatch in the alignment is caused by one or more nucleotide substitutions, the alignment in Figure 1.1 implies that at least three substitutions and two single-base indels took place.

It is possible and, among some researchers, popular to align sequences by eyeballs. The eyeball technique is time-consuming, tedious, and irreproducible. In 1970, Needleman and Wunsch [NW70] presented a dynamic programming algorithm for the alignment of two biological sequences by computers. Computer-aided sequence alignment does not possess these drawbacks of the eyeball technique. In a basic dynamic programming algorithm, a weight for each indel and a weight for each mismatch are defined. Then the weight of alignment is defined as the sum of the total weight of all the spaces and mismatches in the alignment. For example, if the weight of a mismatch is 1 and the weight of an indel is 5, the weight associated with the alignment shown in Figure 1.1 is 13 (=1+1+1+5+5). After the weights for indels and mismatchs are defined, the dynamic programming algorithm chooses the best alignment by finding the alignment with the minimum associated weight. This is assumed to be the best of all alignments between the two sequences in question. A complete explanation of the dynamic programming algorithm can be found in [SK83].

The weakness of the basic dynamic programming method and its subsequent modification is the lack of an objective procedure to choose the relative weights of spaces and mismatches. The results of this weakness is that researchers are forced to use either of two

flawed approaches to obtain an alignment between two sequences. One approach is to arbitrarily choose these weights and then obtain an alignment. If this alignment is aesthetically pleasing to the researcher, the process stops. Otherwise, the researcher continues to adjust the weights until an aesthetically pleasing alignment is obtained. Obviously, the subjective nature of this approach is not ideal. Another approach is to use the same set of weights for every pairwise alignment. This approach is less subjective than the former approach – only the initial choice of weights is subjective.

A few objective alignment techniques [RW73] have been proposed. Allison and Yee [AY90] have applied minimum message length (MML)-encoding techniques to solve the optimization problem. The MML method does allow a statistical comparison of alternative scoring schemes [AWY92a] and is promising for the inference of evolutionary trees from non-aligned sequences [AWY92b]. Fitch and Smith [FS83] introduced a Monte Carlo method that allows the scores of spaces, matches, and mismatches to be chosen objectively.

In the above objective alignment techniques, the relationship between the scoring system and the assumptions about the occurrence of evolutionary events is not clear. Bishop and Thompson [BT86] were the first to consider pairwise sequence alignment in a likelihood framework. They proposed a Markovian model with explicit assumptions about how evolutionary events occur. Estimates of parameters in their model could be obtained and subsequently used to find an optimal alignment. The Bishop and Thompson method has some limitations. An exact treatment of the likelihood is not feasible and the approximations used become less accurate for comparison of more distantly related sequences. Also their model is restricted to allow only single base insertions and deletions. Thorne, Kishino, and Felsenstein [TKF91] introduced a model to overcome some of these limitations. They

believe that because evolution is the force that promotes divergence between biological sequences, it is desirable to view biological sequence alignment algorithms in the context of evolution. The weights of evolutionary events should be a function of evolutionary rates and divergence times. Under this interpretation, the basic dynamic programming procedure assumes that the types of evolutionary events that can change a biological sequence fall into three categories. For a DNA sequence, these three possible types of events are insertion of exactly one base, deletion of exactly one base, and substitution of one base for another. The basic dynamic programming procedure assigns an evolutionary weight to each type of evolutionary event. The evolutionary weight should be proportional to the negative logarithm of the probability of the evolutionary event [Fel81]. Thus the most basic alignment algorithm requires one evolutionary weight for a substitution and another evolutionary weight for a single base indel. It is incorrect to use the same set of weights for every pairwise alignment because the probabilities of evolutionary events depend on the particular pair of sequences to be aligned. In this model, its mathematical tractability eliminates the need for approximations. This model forms the basis of the work presented in this thesis.

To further overcome the limitation of only single base insertions and deletions, Thorne, Kishino, and Felsenstein [TKF92] introduced a model allowing multiple-base insertion and deletion events as well as regional heterogeneity in the substitution process. For a data set of two sequences, the likelihood is a function of the alignment and the probabilities assigned to the different types of evolutionary events. In [Fel81] and [TKF92], a numerical maximization routine, the simplex method [NM65] is employed to maximize the likelihood.

In 1995, Throne and Churchill [TC95] introduced the EM [DLR77] algorithm to maximize the likelihood function for a pair of molecular sequences. In their approach, they can calculate and visually display the arc probabilities, which measure the reliability of

alignment positions by summing up the probabilities of all alignments that contain the arc.

In all of the above work, only pairs molecular sequences are considered. Throne and Churchill [TC95] expected that some progress will be made to consider more than two sequences simultaneously along the line of the EM algorithm. In this paper, we develop algorithms to generate a star alignment with the maximum likelihood based on the evolutionary model in [TKF91]. Given 3 observed molecular sequences, P, Q and R, the **star alignment** problem is to find the unknown ancestral sequence X which evolved into P, Q and R. So we should not only find the evolutionary parameters, i.e. the insertion, deletion, substitution rates for each descedant sequence P, Q or R, but also the unknown ancestral sequence for the set of three sequences. This is the first work in the light of the maximum likelihood method to estimate a sequence as one of the parameters to maximize the likelihood function.

We will devise two star alignment algorithms,the sum approach and direct alignment approach, based on two different likelihood functions to be maximized. In the sum approach, we define the likelihood function as the sum of the likelihoods of all alignments with each possible unknown ancestral sequence. In the direct alignment approach, we define the likelihood function as the the likelihood of a single alignment with a well-chosen ancestral sequence which has the largest likelihood among all the likelihoods of the alignments with other possible ancestors.

A software system, called MLSAS (Maximum Likelihood Star Alignment System), is developed to implement the above two algorithms. MLSAS is programmed in C and MOTIF. It includes a friendly graphical user interface, an editor to edit input and a file browser to view the output. Moreover, it supports multi-tasking so that the users can run the program on different sets of inputs simultaneously.

In this thesis, we mainly deal with DNA sequences. RNA sequences are also considered in Chapter 5. Without much difficulty, the MLSAS could be extended to the alignment of other molecular sequences such as proteins.

The remainder of this paper is organized as follows. Chapter 2 will introduce the statistical model for molecular sequence evolution consisting of the substitution process and the insertion and deletion process. The auxiliary functions to derive the likelihood of pairwise alignment are also given. Chapter 3 will define the concept of **canonical star alignments** in order to avoid duplicate alignments and then define two likelihood functions for star alignments, which are used in the sum approach and the direct alignment approach respectively.

Chapter 4 will illustrate how the software, MLSAS, is developed, how the code is organized, and how to use the MLSAS. Chapter 5 will describe simulation results and provide some real life examples. A brief discussion of the future study relevant to this work is included in Chapter 6.

Chapter 2

The Statistical Model of DNA Sequence Evolution

Suppose we have observed three DNA sequences P, Q, R, which independently evolved from an unknown common ancestor X.

The probability of observing P, Q and R can be expressed as

$$\begin{aligned} \Pr(P,Q,R) &= \sum_X \Pr(X)\cdot\Pr(P,Q,R \mid X) \\ &= \sum_X \Pr(X)\Pr(P \mid X)\cdot\Pr(Q \mid X)\cdot\Pr(R \mid X), \end{aligned}$$

where $\Pr(X)$ is the equilibrium probability of sequence X and the summation runs through all possible ancestral sequences X.

The model for evolution of one sequence from another sequence is composed of two independent processes, a substitution process and an insertion-deletion process. This model is a Markov process: the probability of a transition from the current state of a sequence is independent of previous states of the sequence. Hence our model allows only substitutions,

single-base insertions and single-base deletions.

Section 2.1 will illustrate what the substitution process is. Section 2.2 will explain what the insertion-deletion process is in detail. The description of these two processes can also be found in [TC95, TKF91, TKF92].

Section 2.3 will elaborate on how to calculate the likelihood of two DNA sequences under our model. The calculation will then be extended to calculate the likelihood of multiple DNA sequences in Chapter 3.

2.1 The Substitution Process

The substitution process of the evolutionary model was defined by Churchill [TC95] and Thorne, Kishino and Felsenstein [TKF91].

Let $f_{ij}(t)$, where i and j take on values 0, 1, 2 and 3 corresponding to the four nucleotides A, C, G, and T, denote the probability that a lineage which is initially in state i will be in state j after t units of time have elapsed. Now let us compute $f_{ij}(t)$. We assume that in a small interval of time of length dt, there is a probability of $s \cdot dt$ with which the current nucleotide at a site is replaced. The quantity s is the rate of nucleotide substitution per unit time. When a nucleotide is replaced, its substitute is A, C, G, or T with probabilities π_A, π_G, π_C, or π_T (or π_0, π_1, π_2, or π_3). Note that this means that a nucleotide could be replaced by the same nucleotide, so that not all substitutions are observable even in principle.

Let δ_{ij} be 0 if $i \neq j$ and 1 if $i = j$ (the Kronecker delta function), then we have

$$f_{ij}(dt) = (1 - s(dt))\delta_{ij} + s(dt)\pi_j.$$

By solving the differential equation, we get

$$f_{ij}(t) = e^{-st}\delta_{ij} + \pi_j(1 - e^{-st}).$$

That is:

$$f_{ij}(t) = \begin{cases} e^{-st} + \pi_j(1 - e^{-st}) & \text{if } i = j \\ \pi_j(1 - e^{-st}) & \text{otherwise} \end{cases}$$

2.2 The Insertion–Deletion Process

The insertion-deletion process is, for the sake of clarity, presented not in terms of nucleotides but in terms of imaginary links that separate the DNA nucleotides of a sequence. In our model, there are N normal links and one immortal link in a sequence of N nucleotides. Specifically, there is a normal link to the right of each nucleotide. In addition, the leftmost nucleotide in the sequence can be considered to have an immortal link to its left.

For example, if $\odot$ represents a normal link and $\bullet$ represents the immortal link then the DNA sequence AGCTATATATAT can be depicted as

$$\bullet \mathrm{A} \odot \mathrm{G} \odot \mathrm{C} \odot \mathrm{T} \odot \mathrm{A} \odot \mathrm{T} \odot \mathrm{A} \odot \mathrm{T} \odot \mathrm{A} \odot \mathrm{T} \odot \mathrm{A} \odot \mathrm{T} \odot,$$

or, if the presence of nucleotides is considered without regard to the actual type of nucleotide then the same DNA sequence could be depicted as

$$\bullet \odot \odot \odot \odot \odot \odot \odot \odot \odot \odot \odot \odot$$

The insertion-deletion process is framed in terms of a birth-death process of these links. Each link evolves independently from all other links; a birth or death of one link does not affect the probability of a birth or death of any other link. Both types of links can be

associated with births. The birth rate per normal link (denoted as λ) is equal to the birth rate per immortal link. A newborn link is always a normal link. We adopt the convention that a new born link appears immediately to the right of its parent link. Accompanying the birth of a normal link is the birth of a DNA nucleotide immediately to the left of the newborn link. The probabilities that a newborn DNA nucleotide will be A, G, C, or T are π_A, π_G, π_C, or π_T, respectively. Normal links are subject to death (μ is the death rate per normal link) but immortal links are not.

Because of the chance of more than one birth or death taking place on a sequence at the same instant is small enough to be neglected, a sequence will either increase its length by a single nucleotide, decrease its length by a single nucleotide, or stay in the same length at a given instant. A sequence of n nucleotides will increase its length to $n+1$ nucleotides at rate $(n+1)\lambda$ because it has $n+1$ links. A sequence of n nucleotides will decrease its length to $n-1$ nucleotides (assuming $n > 0$) at rate $n\mu$ because it has n normal links and only normal links can die. This birth-death process is related to the more general linear birth-death process [Fell68]. The relationship between these two birth-death processes can also be seen by examining the form of the transition probabilities associated with each process.

The presence of immortal links in this model is necessary for the existence of a realistic equilibrium distribution of sequence lengths. Without immortal links, sequences would tend over time to have length either 0 or toward infinity. With immortal links and a death rate per normal link that exceeds the birth rate per link, a realistic equilibrium distribution of sequence lengths can exist. If γ_n is the equilibrium probability of sequences n nucleotides in length, then the distribution of γ_n obtained under the birth-death model is the geometric

distribution

$$\gamma_n = (1 - \frac{\lambda}{\mu})(\frac{\lambda}{\mu})^n$$

where $0 < \lambda < \mu$.

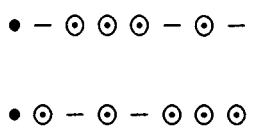
2.3 The Likelihood of a Pair of DNA Sequences

The likelihood of three descendant DNA sequences evolving from a common ancestral sequence is the product of the likelihood of each descendant DNA sequence evolving from the ancestral sequence. So the calculation of likelihood of a pair of DNA sequences becomes the basic step to calculate the likelihood of multiple DNA sequences. In this section, we will explain in principle how to compute the likelihood of a pair of DNA sequences, which was proposed in [TKF91].

Let us consider two DNA sequences. The first, sequence A, is TGTC. The second, sequence B, is GCACA. Various paths are possible for a transition from sequence A to sequence B. The transition probability from A to B is the sum of probabilities of all possible paths connecting the two sequences. The particular path of a transition from A to B can be expressed well by an alignment. As an example, consider the following artificial alignment which will be denoted as α :

```
- T G T - C -
G - C - A C A
```

The information on the presence and absence of nucleotides in alignment α will be denoted as α'. In terms of links, α' can be represented as:



The probability of the specific transition path represented by alignment α denoted as $P(\alpha \mid \theta)$ where θ is the collection of parameters μt, λt, st, π_A, π_G, π_C, π_T, can be decomposed into two components, $P(\alpha' \mid \theta)$ (the transition probability of insertion–deletion) and $P(\alpha \mid \alpha', \theta)$. This decomposition is possible because α contains all of the information of α'. In other words,

$$P(\alpha \mid \theta) = P(\alpha, \alpha' \mid \theta) = P(\alpha \mid \alpha', \theta) P(\alpha' \mid \theta).$$

$P(\alpha \mid \alpha', \theta)$ can be expressed easily. In the above example, we have

$$P(\alpha \mid \alpha', \theta) = \pi_G f_{GC}(t) \pi_A f_{CC}(t) \pi_A$$

$P(\alpha' \mid \theta)$ will be the product of $n + 2$ terms in general when n is the number of nucleotides in the ancestral sequence. The first term of it is the equilibrium probability of the ancestral sequence. The second term is a transition probability for the immortal link. The remainder of the terms in $P(\alpha' \mid \theta)$ are transition probabilities for normal links. The specific transition probability for each link depends on its type, i.e. whether the link has survived and the number of descendant links. The number of descendant links for a particular ancestral link is easily determined by depicting the information on the presence and absence of nucleotides in terms of links. The number of descendant links of a particular survived ancestral link is one plus the number of descendant links to the right of the particular ancestral link and to the left of this ancestral link's right neighbor.

There are three categories of transition probabilities in the insertion–deletion process: $p_n(t)$ is the probability that after a time of duration t, n links have descended from a normal

link and one of them is the original link; $p'_n(t)$ is the probability that after a time of duration t, n links have descended from a normal link and the original link has died; and $p''_n(t)$ is the probability that after a time of duration t, the immortal link has n descendants including itself. In the above example

$$P(\alpha' \mid \theta) = \gamma_4 p''_2(t) p'_0(t) p_1(t) p'_1(t) p_2(t).$$

By definition, $p_0(t) = p''_0(t) = 0$. The remainder of the transition probabilities can be obtained by solving the differential equations governing this birth-death process. These differential equations can be formally expressed as:

$$\begin{aligned}
\frac{dp_n(t)}{dt} &= \lambda(n-1)p_{n-1}(t) - (\lambda+\mu)np_n(t) + \mu np_{n+1}(t), \quad n > 0; \\
\frac{dp'_n(t)}{dt} &= \lambda(n-1)p'_{n-1}(t) - (\lambda+\mu)np'_n(t) + \mu(n+1)p'_{n+1}(t) + \mu p_{n+1}(t), \quad n > 0; \\
\frac{dp'_0(t)}{dt} &= \mu p'_1(t) + \mu p_1(t); \\
\frac{dp''_n(t)}{dt} &= \lambda(n-1)p''_{n-1}(t) - [\lambda n + \mu(n-1)]p''_n(t) + \mu n p''_{n+1}(t), \quad n > 0;
\end{aligned}$$

where the initial conditions are

$$\begin{aligned}
p_1(0) &= p''_1(0) = 1; \\
p_n(0) &= p''_n(0) = 0, \qquad n = 2, 3, \cdots; \\
p'_n(0) &= 0, \qquad n = 0, 1, \cdots.
\end{aligned}$$

By solving the differential equations, we get

$$\begin{aligned}
p_n(t) &= e^{-\mu t}[1 - \lambda\beta(t)][\lambda\beta(t)]^{n-1}, \quad n > 0; \\
p'_n(t) &= [1 - e^{-\mu t} - \mu\beta(t)][1 - \lambda\beta(t)][\lambda\beta(t)]^{n-1}, \quad n > 0;
\end{aligned}$$

$$p_0'(t) = \mu\beta(t);$$

$$p_n''(t) = [1 - \lambda\beta(t)][\lambda\beta(t)]^{n-1}, \quad n > 0;$$

where

$$\beta(t) = \frac{1 - e^{(\lambda-\mu)t}}{\mu - \lambda e^{(\lambda-\mu)t}}.$$

Note that for $k > 1$ we have

$$p_k(t) = p_1(t)[\lambda\beta(t)]^{k-1};$$

$$p_k'(t) = p_1'(t)[\lambda\beta(t)]^{k-1};$$

$$p_k''(t) = p_1''(t)[\lambda\beta(t)]^{k-1}.$$

Let s_A be the length of sequence A, and s_B the length of sequence B. The likelihood of the pair of A and B is

$$L_\theta(A, B) = \pi_A^{r_A}\pi_G^{r_G}\pi_C^{r_C}\pi_T^{r_T}\gamma_{s_A}P_t(B \mid A, \theta)$$

where r_A, r_G, r_C, r_T are the number of occurrences of each type of nucleotide in sequence A.

A pairwise alignment can be represented as a matrix of two rows with columns recording the process of evolution. For example,

```
Sequence A:   G - C - A C A - T G T - C -

Sequence B:   - T G T - C - G - C - A C A
```

A space (-) in sequence A means that an insertion took place, while a space in sequence B means that a deletion took place. In other cases, a substitution took place.

Because of the convention that newborn links are inserted to the right of their parental link, the following two alignments represent two distinct evolutionary histories

(I)

```
A - C
A G -
```

(II)

```
A C -
A - G
```

In alignment (I), the A link is the parent of the G link whereas in alignment (II) the C link is the parent of the G link. Alignment (II) implies that the G link was inserted before the C link was deleted. Alignment (I) does not specify any chronological ordering of the two events. The difference in the interpretation of the two alignments results in the alignments having distinct probabilities. It is convenient for computational reasons to consider each case of a deletion immediately to the left of an insertion as a "special substitution". A substitution column that indicates a match or a mismatch will be termed a "normal substitution".

Therefore, in an alignment of a pair of DNA sequences, there are four types of columns, the deletion columns, the insertion columns, the special substitution columns and the normal substitution columns.

At this point, we can calculate the likelihood of a pair of DNA sequences by induction on the lengths of sequences and the types of columns. We will not show these inductive steps here since we will give the inductive steps and likelihood expressions in detail for multiple sequences in Chapter 3. The reader could refer to [TC95, TKF91] for the likelihood calculation in the case of a pair of DNA sequences.

Chapter 3

The Likelihood of Three DNA Sequences

In the previous chapter, we defined the statistical model for DNA sequence evolution and derived various auxiliary functions to compute the likelihood of a DNA sequence pair. In this chapter, we will show how to calculate the likelihood of three observed DNA sequences P, Q, R, evolving from an unknown ancestral sequence X.

We know that the likelihood of observing P, Q and R is

$$\Pr(P, Q, R) = \sum_X \Pr(X)\Pr(P \mid X) \cdot \Pr(Q \mid X) \cdot \Pr(R \mid X)$$

where $\Pr(X)$ is the equilibrium probability of sequence X and the summation runs through all possible ancestral sequences.

The calculation of $\Pr(P, Q, R)$ is obviously an intractable problem because the number of possible X's is infinite. So throughout the paper we assume that the length of X is no more than the sum of lengths of P, Q, R. That is, we do not allow a base in X to be deleted

in all the three evolutionary processes to P, Q and R. Even though we limit the length of the unknown sequence, the calculation is still very time–consuming because we should add up all the likelihoods of each alignment with every possible unknown sequence X using four layers of loops.

In this chapter, we define two kinds of the objective likelihood functions each of which simplifies the complexity of computation and meanwhile maintains a reasonable accuracy.

In section 3.1, we define canonical star alignments and 33 types of columns (or evolution processes) for a nucleotide in the ancestral DNA sequence to evolve into another nucleotide or to be deleted (or to be inserted) in the resulting sequence, either P or Q or R. The 33 types of columns will be frequently used in the calculation of the likelihood and the alignment of the sequences in the whole system.

In section 3.2, we define the likelihood function as the summation of all alignments with each possible unknown ancestral sequence.

In section 3.3, we define the likelihood function as the likelihood of a single alignment with a well-chosen ancestral sequence which has the largest likelihood among the likelihoods of all possible alignments.

The two likelihood functions will be maximized by the simplex method to find the evolutionary parameters. The simplex method was well-studied in [NM65] and has been coded in many programming languages. The C code for this optimization routine can be found in [PFTV88].

3.1 The Canonical Star Alignment with Column Representation

In this section, we will define canonical star alignments with column representation, the 33 types of columns in a star alignment, and illustrate how to avoid non-canonical star alignments during the enumeration of all star alignments.

Suppose that we observed three DNA sequences P, Q, R which are believed to have evolved from a common unknown ancestral DNA sequence X under the evolutionary model introduced in Chapter 2. A set of evolutionary parameters is denoted as $\theta = (\lambda_1 t, \mu_1 t, s_1 t, \lambda_2 t, \mu_2 t, s_2 t, \lambda_3 t, \mu_3 t, s_3 t)$, where λ_1 is the birth rate per normal or immortal link, μ_1 is the death rate per normal link and s_1 is the substitution rate in the evolution from X to P. λ_2, μ_2, s_2 and λ_3, μ_3, s_3 are defined similarly for the evolutionary processes from X to $Q$and R; t is the time that the evolutions have taken.

We assume that the evolutionary parameters θ is estimated by the maximum likelihood method based on some kind of likelihood function. Based on these parameters, the star alignment is to find the the ancestral DNA sequence X and to align it with P, Q and R with the maximum probability.

As in the case of pairwise DNA sequence alignment, a star alignment can also be represented as a matrix of four rows with the columns recording the history of evolution. The first row is the unknown common ancestral DNA sequence X, while the second, the third and the fourth rows are the observed sequences P, Q, R, respectively. Figure 3.1 is an example of an arbitary star alignment

However, after carefully checking the first five columns in the Figure 3.1, we know the

```
Sequence X:   - - G - - C A - T C T A C -
Sequence P:   G - G T A C - G T C - A C A
Sequence Q:   G - G - A C A G - C - A C A
Sequence R:   - G G T - C - G - C T A C A
```

Figure 3.1: An example of an arbitary star alignment

```
Sequence X:   G - - - C A - T C T A C -
Sequence P:   G G T A C - G T C - A C A
Sequence Q:   G G A - C A G - C - A C A
Sequence R:   G G T - C - G - C T A C A
```

Figure 3.2: The cannonical star alignment of Figure 3.1

alignment in Figure 3.2 describes the exactly same evolutionary process under the birth-death model.

This is because the substitution "space $\longrightarrow$ space" means nothing has happened. So we can move "space $\longrightarrow$ space" forward or backward without changing the history of evolution. We should eliminate the duplicated alignment because we cannot take the same evolution event into consideration more than once. The duplication can be avoided by defining **canonical star alignments.**

A star alignment is **canonical** in a column representation if each "spacep $\longrightarrow$ space" (if any) from X to P (or Q, or R) cannot be moved to the right without changing the history of evolution under the model we defined in Chapter 2.

As we discussed in chapter 2, in a pairwise DNA sequence alignment, there are four types of columns, which are insertion, deletion, normal substitution and special substitution. In the case of star alignment, there are 33 possible types of columns listed below if we assume that no nucleotide in X is deleted in all descendant sequences P, Q, R.

The 33 types of columns recording the evolution history for each sequence will be used intensively in the calculation of likelihoods and the alignment of sequences. So we single them out and define them as follows. The first 7 types are related to insertions. The last 26 types are related to deletions and substitution.

1. Column type 0: Insertion occurs in P. Denote it as "ins–P".
2. Column type 1: Insertion occurs in Q. Denote it as "ins–Q".
3. Column type 2: Insertion occurs in R. Denote it as "ins–R".
4. Column type 3: Insertion occur in Q, R. Denote it as "ins–P; ins–Q".
5. Column type 4: Insertions occur in P, R. Denote it as "ins–P; ins–R".
6. Column type 5: Insertions occur in P, Q. Denote it as "ins–P; ins–Q".
7. Column type 6: Insertions occur in P, Q, R. Denote it as "ins–P; ins–Q; ins–R".
8. Column type 7: Deletions occur in P, Q and a normal substitution occurs in R. Denote it as "del–P; del–Q; nor–R".
9. Column type 8: Deletions occur in P, Q and a special substitution occurs in R. Denote it as "del–P; del–Q; spe–R".
10. Column type 9: Deletions occur in P, R and a normal substitution occurs in Q. Denote it as "del–P; nor–Q; del–R; ".
11. Column type 10: Deletions occur in P, R and a special substitution occurs in Q. Denote it as "del–P; spe–Q; del–R".
12. Column type 11: Deletions occur in Q, R and a normal substitution occurs in P. Denote it as "nor–P; del–Q; del–R".

13. Column type 12: Deletions occur in Q, R and a special substitution occurs in P. Denote it as "spe–P; del–Q; del–R".

14. Column type 13: A deletion occurs in P, normal substitutions occur in Q and R. Denote it as "del–P; nor–Q; nor–R".

15. Column type 14: A deletion occurs in P, a normal substitution occurs in Q and a special substitution occurs in R. Denote it as "del–P; nor–Q; spe–R".

16. Column type 15: A deletion occurs in P, a special substitution occurs in Q and a normal substitution occurs in R. Denote it as "del–P; spe–Q; nor–R".

17. Column type 16: A deletion occurs in P, special substitutions occur in Q and R. Denote it as "del–P; spe–Q; spe–R".

18. Column type 17: Normal substitutions occur in P and R, a deletion occurs in Q. Denote it as "nor–P; del–Q; nor–R".

19. Column type 18: A normal substitution occurs in P, a deletion occurs in Q and a special substitution occurs in R. Denote it as "nor–P; del–Q; spe–R".

20. Column type 19: A special substitution occurs in P, a deletion occurs in Q and a normal substitution occurs in R. Denote it as "spe–P; del–Q; nor–R".

21. Column type 20: Special substitutions occur in P and R, a deletion occurs in Q. Denote it as "spe–P; del–Q; spe–R".

22. Column type 21: Normal substitutions occur in P and Q, a deletion occurs in R. Denote it as "nor–P; nor–Q; del–R".

23. Column type 22: A normal substitution occurs in P, a special substitution occurs in Q and a deletion occurs in R. Denote it as "nor–P; spe–Q; del–R".

24. Column type 23: A special substitution occurs in P, a normal substitution occurs in Q and a deletion occurs in R. Denote it as "spe–P; nor–Q; del–R".

25. Column type 24: Special substitutions occur in P and Q, a deletion occurs in R. Denote it as "spe–P; spe–Q; del–R".

26. Column type 25: Normal substitutions occur in P, Q and R. Denote it as "nor–P; nor–Q; nor–R".

27. Column type 26: Normal substitutions occur in P and Q, and a special substitution occurs in R. Denote it as "nor–P; nor–Q; spe–R".

28. Column type 27: Normal substitutions occur in P and R, and a special substitution occurs in Q. Denote it as "nor–P; spe–Q; nor–R".

29. Column type 28: Normal substitutions occur in Q and R, and a special substitution occurs in P. Denote it as "spe–P; nor–Q; nor–R".

30. Column type 29: Normal substitution occurs in P, and special substitutions occur in Q and R. Denote it as "nor–P; spe–Q; spe–R".

31. Column type 30: Normal substitution occurs in Q, and special substitutions occur in P and R. Denote it as "spe–P; nor–Q; spe–R".

32. Column type 31: Special substitutions occur in P and Q, and normal substitution occurs in R. Denote it as "spe–P; spe–Q; nor–R".

33. Column type 32: Special substitutions occur in P, Q and R. Denote it as "spe–P; spe–Q; spe–R".

Next, we will discuss how to avoid non-canonical alignments when enumerating all star alignments by induction.

Table 3.1: Right–Offensive Columns

j	0	1	2	3	4	5
i	1,2,3,4,5,6	0,2,3,4,5,6	0,1,3,4,5,6	0,4,5,6	1,3,5,6	2,3,4,6

Table 3.2: Left–Offensive Columns

i	0	1	2	3	4	5	6
j	1,2,3	0,2,4	0,1,5	0,1,2,4,5	0,1,2,3,5	0,1,2,3,4	0,1,2,3,4,5

Suppose we have a canonical star alignment of length n with type j as the type of the last column. We will add a column of type i to the right of this alignment to form a canonical star alignment of length $n+1$. The question is: what kind of column, i.e. the value of i, cannot be selected in order to have a canonical star alignment?

First, we note that if the nucleotide of X (the ancestral sequence) in the last column is not a space, i.e. $7 \leq j \leq 32$, or an insertion took place in each descendant sequence, i.e. $j = 6$, then it is safe to add any kind of column to the right of the alignment.

Second, if $0 \leq j \leq 5$, by checking each value of j, we find the types of columns (the values of i) that cannot be added to the right of the alignment because there is at least one pair of gaps that can be shifted to the right without changing the history of evolution. Such a column with type i is called a **right–offensive column** of i. For each j, $0 \leq j \leq 5$, its right–offensive column type i's are shown in Table 3.1.

Conversely, if we know the type of the newly added column is i, then we can easily find the value (more than one) of j with which type the column cannot be to the left of the newly added column. Such a column with type j is called a **left–offensive column** of i. For each i, $0 \leq i \leq 6$, the left–offensive column type, j's are shown in Table 3.2.

In a word, in a canonical alignment, a column of type j cannot be the left neighbor of a column of type i and conversely, a column of type i cannot be the right neighbor of a column of type j, where i and j are the values in Table 3.1 or Table 3.2.

3.2 The Sum Approach

In this section, we will define the likelihood function, $L_\theta^S(|P|, |Q|, |R|)$, as the summation of the likelihood of each possible canonical star alignment with each possible ancestral DNA sequence by induction on the lengths of P, Q and R, where $|P|$, $|Q|$, $|R|$ are the lengths of P, Q and R respectively and θ is defined as in Section 3.1.

Let us define some auxiliary functions for each sequence. In the evolutionary process from X to P, define

$$f_{1,ij}(t) = \begin{cases} e^{-s_1 t} + \pi_j(1 - e^{-s_1 t}) & \text{if } i = j \\ \pi_j(1 - e^{-s_1 t}) & \text{otherwise} \end{cases}$$

where i, j are nucleotides, and

$$\begin{aligned} p_{1,n}(t) &= e^{-\mu_1 t}[1 - \lambda_1\beta_1(t)][\lambda_1\beta_1(t)]^{n-1}, \quad n > 0, \\ p'_{1,n}(t) &= [1 - e^{-\mu_1 t} - \mu_1\beta_1(t)][1 - \lambda_1\beta_1(t)][\lambda_1\beta_1(t)]^{n-1}, \quad n > 0, \\ p'_{1,0}(t) &= \mu_1\beta_1(t), \\ p''_{1,n}(t) &= [1 - \lambda_1\beta_1(t)][\lambda_1\beta_1(t)]^{n-1}, \quad n > 0, \end{aligned}$$

where

$$\beta_1(t) = \frac{1 - e^{(\lambda_1 - \mu_1)t}}{\mu_1 - \lambda_1 e^{(\lambda_1 - \mu_1)t}}.$$

Similarly, we can define $f_{2,ij}(t)$, $p_{2,n}(t)$, $p'_{2,n}(t)$, $p'_{2,0}(t)$, and $p''_{2,n}(t)$ for sequence Q and $f_{3,ij}(t)$, $p_{3,n}(t)$, $p'_{3,n}(t)$, $p'_{3,0}(t)$ and $p''_{3,n}(t)$ for sequence R.

The notations not defined in this section can be found in Chapter 2.

Now we define $L_\theta^S(|P|,|Q|,|R|)$ by induction on the lengths of P, Q and R. In order to avoid adding the likelihood of non-canonical alignment into the sum, during the induction, for each l, m, n, where $0 \le l \le |P|$, $0 \le m \le |Q|$, $0 \le n \le |R|$, we have to define another six likelihood functions: $L_\theta^S(i,l,m,n)$, $0 \le i \le 6$, which is the same as $L_\theta^S(l,m,n)$ except that every canonical star alignment considered in the sum should have a column of type i as its rightmost column.

Initially, i.e. $l = m = n = 0$, we have one immortal link for each sequence and the number of links in each sequence is one. So we get

$$L_\theta^S(0,0,0) = p''_{1,1}(t)(1-\tfrac{\lambda_1}{\mu_1}) \cdot p''_{2,1}(t)(1-\tfrac{\lambda_2}{\mu_2}) \cdot p''_{3,1}(t)(1-\tfrac{\lambda_3}{\mu_3}),$$

$$L_\theta^S(0,0,0,0) = 0,\ L_\theta^S(1,0,0,0) = 0,\ L_\theta^S(2,0,0,0) = 0,$$

$$L_\theta^S(3,0,0,0) = 0,\ L_\theta^S(4,0,0,0) = 0,\ L_\theta^S(5,0,0,0) = 0.$$

Define $L_\theta^S(l,m,n) = 0$, $L_\theta^S(i,l,m,n) = 0$, $0 \le i \le 5$, for $l < 0$, or $m < 0$, or $n < 0$. For $0 \le l \le |P|$, $0 \le m \le |Q|$, $0 \le n \le |R|$, define

$$L_\theta^S(0,0,m,n) = L_\theta^S(1,l,0,n) = L_\theta^S(2,l,m,0) = 0,$$

and

$$L_\theta^S(3,l,m,0) = L_\theta^S(3,l,0,n) = 0,$$

$$L_\theta^S(4,0,m,n) = L_\theta^S(4,l,m,0) = 0,$$

$$L_\theta^S(5,0,m,n) = L_\theta^S(5,l,0,n) = 0.$$

Let P_l denote the l^{th} nucleotide in sequence P. Similarly Q_m denotes m^{th} nucleotide in sequence Q, and R_n denotes n^{th} nucleotide in sequence R. So sequence P can also be expressed as $P_1 \cdots P_{|P|}$.

Let i, $7 \leq i \leq 32$, be a column type. $L_\theta^S(i, A, l, m, n)$ denotes the summation of the likelihoods of all possible canonical star alignments of sequences $P_1 \cdots P_l$, $Q_1 \cdots Q_m$, $R_1 \cdots R_n$ with the last column having type i and the nucleotide of the ancestral sequence in the last column being A. $L_\theta^S(i, G, l, m, n)$, $L_\theta^S(i, C, l, m, n)$ and $L_\theta^S(i, T, l, m, n)$ are similarly defined.

By induction, suppose that we have computed the values of $L_\theta^S(p, q, r)$, and $L_\theta^S(0, p, q, r)$, $L_\theta^S(1, p, q, r)$, $L_\theta^S(2, p, q, r)$, $L_\theta^S(3, p, q, r)$, $L_\theta^S(4, p, q, r)$, $L_\theta^S(5, p, q, r)$ where $p \leq l, q \leq m, r \leq n$ and $(p, q, r) \neq (l, m, n)$. In the rest of this section, we will compute $L_\theta^S(i, l, m, n), 0 \leq i \leq 6$, and $L_\theta^S(i, Y, l, m, n)$ where $7 \leq i \leq 32$ and $Y \in \{A, G, C, T\}$. Thus $L_\theta^S(l, m, n)$ will be the summation of all these likelihoods.

Now let us first consider the insertion cases, which need extra attention because the last column in the current alignment could be a left offensive column of the new insertion column, thus causing duplicate alignments.

We will define first the values of TMP_i for each i, $0 \leq i \leq 6$. TMP_i will then be used to define $L_\theta(i, l, m, n)$ for each i, $0 \leq i \leq 6$.

If $i = 0$, the left offensive column types of i are 1, 2, 3. Define

$$\begin{aligned} TMP_0 &= L_\theta^S(l-1, m, n) - L_\theta^S(1, l-1, m, n) - \\ &\quad L_\theta^S(2, l-1, m, n) - L_\theta^S(3, l-1, m, n). \end{aligned}$$

If $i = 1$, the left offensive column types of i are 0,2,4. Define

$$\begin{aligned} TMP_1 &= L_\theta^S(l, m-1, n) - L_\theta^S(0, l, m-1, n) - \\ &\quad L_\theta^S(2, l, m-1, n) - L_\theta^S(4, l, m-1, n). \end{aligned}$$

If $i = 2$, the left offensive column types of i are 0,1,5. Define

$$\begin{aligned} TMP_2 &= L_\theta^S(l, m, n-1) - L_\theta^S(0, l, m, n-1) - \\ &\quad L_\theta^S(1, l, m, n-1) - L_\theta^S(5, l, m, n-1). \end{aligned}$$

If $i = 3$, the left offensive column types of i are 0, 1, 2, 4, 5. Define

$$\begin{aligned} TMP_3 &= L_\theta^S(l, m-1, n-1) - L_\theta^S(0, l, m-1, n-1) - \\ &\quad L_\theta^S(1, l, m-1, n-1) - L_\theta^S(2, l, m-1, n-1) - \\ &\quad L_\theta^S(4, l, m-1, n-1) - L_\theta^S(5, l, m-1, n-1). \end{aligned}$$

If $i = 4$, the left offensive column types of i are 0, 1, 2, 3, 5. Define

$$\begin{aligned} TMP_4 &= L_\theta^S(l-1, m, n-1) - L_\theta^S(0, l-1, m, n-1) - \\ &\quad L_\theta^S(1, l-1, m, n-1) - L_\theta^S(2, l-1, m, n-1) - \\ &\quad L_\theta^S(3, l-1, m, n-1) - L_\theta^S(5, l-1, m, n-1). \end{aligned}$$

If $i = 5$, the left offensive column types of i are 0, 1, 2, 3, 4. Define

$$\begin{aligned} TMP_5 &= L_\theta^S(l-1, m-1, n) - L_\theta^S(0, l-1, m-1, n) - \\ &\quad L_\theta^S(1, l-1, m-1, n) - L_\theta^S(2, l-1, m-1, n) - \\ &\quad L_\theta^S(3, l-1, m-1, n) - L_\theta^S(4, l-1, m-1, n). \end{aligned}$$

If $i = 6$, the left offensive column types of i are 0, 1, 2, 3, 4, 5. Define

$$\begin{aligned} TMP_6 &= L_\theta^S(l-1, m-1, n-1) - L_\theta^S(0, l-1, m-1, n-1) - \\ &\quad L_\theta^S(1, l-1, m-1, n-1) - L_\theta^S(2, l-1, m-1, n-1) - \\ &\quad L_\theta^S(3, l-1, m-1, n-1) - L_\theta^S(4, l-1, m-1, n-1) - \\ &\quad L_\theta^S(5, l-1, m-1, n-1) \end{aligned}$$

Then $L_\theta(i, l, m, n)$ for each i, $0 \leq i \leq 6$ can be easily computed as follows.

$i = 0$: "ins–P"

$$L_\theta^S(0, l, m, n) = \pi_{P_l}\lambda_1\beta_1(t) \cdot TMP_0.$$

$i = 1$: "ins–Q"

$$L_\theta^S(1, l, m, n) = \pi_{Q_m}\lambda_2\beta_2(t) TMP_1.$$

$i = 2$: "ins–R"

$$L_\theta^S(2, l, m, n) = \pi_{R_n}\lambda_3\beta_3(t) \cdot TMP_2.$$

$i = 3$: "ins–Q, ins–R"

$$L_\theta^S(3, l, m, n) = \pi_{Q_m}\lambda_2\beta_2(t) \cdot \pi_{R_n}\lambda_3\beta_3(t) \cdot TMP_3.$$

$i = 4$: "ins–P, ins–R"

$$L_\theta^S(4, l, m, n) = \pi_{P_l}\lambda_1\beta_1(t) \cdot \pi_{R_n}\lambda_3\beta_3(t) \cdot TMP_4.$$

$i = 5$: "ins–P, ins–Q"

$$L_\theta^S(5, l, m, n) = \pi_{P_l}\lambda_1\beta_1(t) \cdot \pi_{Q_m}\lambda_2\beta_2(t) \cdot TMP_5.$$

$i = 6$: "ins–P, ins–Q, ins-R"

$$L_\theta^S(6, l, m, n) = \pi_{P_l}\beta_1(t) \cdot \pi_{Q_m}\lambda_2\beta_2(t) \cdot \pi_{R_n}\lambda_3\beta_3(t) \cdot TMP_6.$$

Now, let us deal with the deletion-substitution columns, which are much easier because these columns cannot generate non-canonical alignments. $L_\theta^S(i, A, l, m, n)$, $7 \leq i \leq 33$, can be calculated by the usual recurrence equations below.

$i = 7$: "del–P; del–Q; nor–R"

$$L_\theta^S(7,A,l,m,n) = \frac{\lambda_1}{\mu_1}\pi_A p'_{1,0}(t)\cdot\frac{\lambda_2}{\mu_2}\pi_A p'_{2,0}(t)\cdot\frac{\lambda_3}{\mu_3}\pi_A f_{3AR_n}(t)p_{3,1}(t)\cdot L_\theta^S(l,m,n-1),$$

$i = 8$: "del–P; del–Q; spe–R"

$$L_\theta^S(8,A,l,m,n) = \frac{\lambda_1}{\mu_1}\pi_A p'_{1,0}(t)\cdot\frac{\lambda_2}{\mu_2}\pi_A p'_{2,0}(t)\cdot\frac{\lambda_3}{\mu_3}\pi_A\pi_{R_n} p'_{3,1}(t)\cdot L_\theta^S(l,m,n-1),$$

$i = 9$: "del–P; nor–Q; del–R; "

$$L_\theta^S(9,A,l,m,n) = \frac{\lambda_1}{\mu_1}\pi_A p'_{1,0}(t)\cdot\frac{\lambda_3}{\mu_3}\pi_A p'_{3,0}(t)\cdot\frac{\lambda_2}{\mu_2}\pi_A f_{2AQ_m}(t)p_{2,1}(t)\cdot L_\theta^S(l,m-1,n),$$

$i = 10$: "del–P; spe–Q; del–R"

$$L_\theta^S(10,A,l,m,n) = \frac{\lambda_1}{\mu_1}\pi_A p'_{1,0}(t)\cdot\frac{\lambda_3}{\mu_3}\pi_A p'_{3,0}(t)\cdot\frac{\lambda_2}{\mu_2}\pi_A\pi_{Q_m} p'_{2,1}(t)\cdot L_\theta^S(l,m-1,n),$$

$i = 11$: "nor–P; del–Q; del–R"

$$L_\theta^S(11,A,l,m,n) = \frac{\lambda_2}{\mu_2}\pi_A p'_{2,0}(t)\cdot\frac{\lambda_3}{\mu_3}\pi_A p'_{3,0}(t)\cdot\frac{\lambda_1}{\mu_1}\pi_A f_{1AP_l}(t)p_{1,1}(t)\cdot L_\theta^S(l-1,m,n),$$

$i = 12$: "spe–P; del–Q; del–R"

$$L_\theta^S(12,A,l,m,n) = \frac{\lambda_2}{\mu_2}\pi_A p'_{2,0}(t)\cdot\frac{\lambda_3}{\mu_3}\pi_A p'_{3,0}(t)\cdot\frac{\lambda_1}{\mu_1}\pi_A\pi_{P_l} p'_{1,1}(t)\cdot L_\theta^S(l-1,m,n),$$

$i = 13$: "del–P; nor–Q; nor–R"

$$\begin{aligned} L_\theta^S(13,A,l,m,n) \;=\; & \frac{\lambda_1}{\mu_1}\pi_A p'_{1,0}(t)\cdot\frac{\lambda_2}{\mu_2}\pi_A f_{2AQ_m}(t)p_{2,1}(t)\cdot \\ & \frac{\lambda_3}{\mu_3}\pi_A f_{3AR_n}(t)p_{3,1}(t)L_\theta^S(l,m-1,n-1). \end{aligned}$$

$i = 14$: "del–P; nor–Q; spe–R"

$$L_\theta^S(14,A,l,m,n) = \frac{\lambda_1}{\mu_1}\pi_A p'_{1,0}(t)\cdot\frac{\lambda_2}{\mu_2}\pi_A f_{2AQ_m}(t)p_{2,1}(t)\cdot\frac{\lambda_3}{\mu_3}\pi_A\pi_{R_n} p'_{3,1}(t)L_\theta^S(l,m-1,n-1),$$

$i = 15$: "del–P; spe–Q; nor–R"

$$L_\theta^S(15, A, l, m, n) = \frac{\lambda_1}{\mu_1}\pi_A p'_{1,0}(t) \cdot \frac{\lambda_2}{\mu_2}\pi_A \pi_{Q_m} p'_{2,1}(t) \cdot \frac{\lambda_3}{\mu_3}\pi_A f_{3AR_n}(t) p_{3,1}(t) \cdot L_\theta^S(l, m-1, n-1),$$

$i = 16$: "del–P; spe–Q; spe–R"

$$L_\theta^S(16, A, l, m, n) = \frac{\lambda_1}{\mu_1}\pi_A p'_{1,0}(t) \cdot \frac{\lambda_2}{\mu_2}\pi_A \pi_{Q_m}(t) p'_{2,1}(t) \cdot \frac{\lambda_3}{\mu_3}\pi_A \pi_{R_n} p'_{3,1}(t) \cdot L_\theta^S(l, m-1, n-1),$$

$i = 17$: "nor–P; del–Q; nor–R"

$$L_\theta^S(17, A, l, m, n) = \frac{\lambda_2}{\mu_2}\pi_A p'_{2,0}(t) \cdot \frac{\lambda_1}{\mu_1}\pi_A f_{1AP_l}(t) p_{1,1}(t) \cdot \frac{\lambda_3}{\mu_3}\pi_A f_{3AR_n}(t) p_{3,1}(t) \cdot L_\theta^S(l-1, m, n-1),$$

$i = 18$: "nor–P; del–Q; spe–R"

$$L_\theta^S(11, A, l, m, n) = \pi_A^3 \frac{\lambda_2}{\mu_2} p'_{2,0}(t) \cdot \frac{\lambda_1}{\mu_1} f_{1AP_l}(t) p_{1,1}(t) \cdot \frac{\lambda_3}{\mu_3}\pi_{R_n} p'_{3,1}(t) \cdot L_\theta^S(l-1, m, n-1),$$

$i = 19$: "spe–P; del–Q; nor–R"

$$L_\theta^S(19, A, l, m, n) = \pi_A^3 \frac{\lambda_2}{\mu_2} p'_{2,0}(t) \cdot \frac{\lambda_1}{\mu_1}\pi_{P_l} p'_{1,1}(t) \cdot \frac{\lambda_3}{\mu_3} f_{3AR_n}(t) p_{3,1}(t) \cdot L_\theta^S(l-1, m, n-1),$$

$i = 20$: "spe–P; del–Q; spe–R"

$$L_\theta^S(20, A, l, m, n) = \pi_A^3 \frac{\lambda_2}{\mu_2} p'_{2,0}(t) \cdot \frac{\lambda_1}{\mu_1}\pi_{P_l} p'_{1,1}(t) \cdot \frac{\lambda_3}{\mu_3}\pi_{R_n} p'_{3,1}(t) \cdot L_\theta^S(l-1, m, n-1),$$

$i = 21$: "nor–P; nor–Q; del–R"

$$L_\theta^S(22, A, l, m, n) = \pi_A^3 \frac{\lambda_3}{\mu_3} p'_{3,0}(t) \cdot \frac{\lambda_1}{\mu_1} f_{1AP_l}(t) p_{1,1}(t) \cdot \frac{\lambda_2}{\mu_2} f_{2AQ_m}(t) p_{2,1}(t) \cdot L_\theta^S(l-1, m-1, n),$$

$i = 22$: "nor–P; spe–Q; del–R"

$$L_\theta^S(22, A, l, m, n) = \pi_A^3 \frac{\lambda_3}{\mu_3} p'_{3,0}(t) \cdot \frac{\lambda_1}{\mu_1} f_{1AP_l}(t) p_{1,1}(t) \cdot \frac{\lambda_2}{\mu_2}\pi_{Q_m} p'_{2,1}(t) \cdot L_\theta^S(l-1, m-1, n),$$

$i = 23$: "spe–P; nor–Q; del–R"

$$L_\theta^S(23, A, l, m, n) = \pi_A^3 \frac{\lambda_3}{\mu_3} p'_{3,0}(t) \cdot \frac{\lambda_1}{\mu_1} \pi_{P_l} p'_{1,1}(t) \cdot \frac{\lambda_2}{\mu_2} f_{2AQ_m}(t) p_{2,1}(t) \cdot L_\theta^S(l-1, m-1, n),$$

$i = 24$: "spe–P; spe–Q; del–R"

$$L_\theta^S(24, A, l, m, n) = \pi_A^3 \frac{\lambda_3}{\mu_3} p'_{3,0}(t) \cdot \frac{\lambda_1}{\mu_1} \pi_{P_l} p'_{1,1}(t) \cdot \frac{\lambda_2}{\mu_2} \pi_{Q_m} p'_{2,1}(t) \cdot L_\theta^S(l-1, m-1, n),$$

$i = 25$: "nor–P; nor–Q; nor–R"

$$\begin{aligned} L_\theta^S(25, A, l, m, n) &= \pi_A^3 \frac{\lambda_1}{\mu_1} p_{1,1}(t) f_{1AP_l}(t) \cdot \frac{\lambda_2}{\mu_2} p_{2,1}(t) f_{2AQ_m}(t) \cdot \\ &\quad \frac{\lambda_3}{\mu_3} p_{3,1}(t) f_{3AR_n}(t) \cdot L_\theta^S(l-1, m-1, n-1). \end{aligned}$$

$i = 26$: "nor–P; nor–Q; spe–R"

$$\begin{aligned} L_\theta^S(26, A, l, m, n) &= \pi_A^3 \frac{\lambda_1}{\mu_1} p_{1,1}(t) f_{1AP_l}(t) \cdot \frac{\lambda_2}{\mu_2} p_{2,1}(t) f_{2AQ_m}(t) \cdot \\ &\quad \frac{\lambda_3}{\mu_3} p'_{3,1}(t) \pi_{R_n} \cdot L_\theta^S(l-1, m-1, n-1) \end{aligned}$$

$i = 27$: "nor–P; spe–Q; nor–R"

$$\begin{aligned} L_\theta^S(27, A, l, m, n) &= \pi_A^3 \frac{\lambda_1}{\mu_1} p_{1,1}(t) f_{1AP_l}(t) \cdot \frac{\lambda_2}{\mu_2} p'_{2,1}(t) \pi_{Q_m} \cdot \\ &\quad \frac{\lambda_3}{\mu_3} p_{3,1}(t) f_{3AR_n}(t) \cdot L_\theta^S(l-1, m-1, n-1). \end{aligned}$$

$i = 28$: "spe–P; nor–Q; nor–R"

$$\begin{aligned} L_\theta^S(28, A, l, m, n) &= \pi_A^3 \frac{\lambda_1}{\mu_1} p'_{1,1}(t) \pi_{P_l} \cdot \frac{\lambda_2}{\mu_2} p_{2,1}(t) f_{2AQ_m}(t) \cdot \\ &\quad \frac{\lambda_3}{\mu_3} p_{3,1}(t) f_{3AR_n}(t) \cdot L_\theta^S(l-1, m-1, n-1). \end{aligned}$$

$i = 29$: "nor–P; spe–Q; spe–R"

$$L_\theta^S(29,A,l,m,n) = \pi_A^3\frac{\lambda_1}{\mu_1}p_{1,1}(t)f_{1AP_l}(t)\cdot\frac{\lambda_2}{\mu_2}p'_{2,1}(t)\pi_{Q_m}\cdot \frac{\lambda_3}{\mu_3}p'_{3,1}(t)\pi_{R_n}\cdot L_\theta^S(l-1,m-1,n-1).$$

$i = 30$: "spe–P; nor–Q; spe–R"

$$L_\theta^S(30,A,l,m,n) = \pi_A^3\frac{\lambda_1}{\mu_1}p'_{1,1}(t)\pi_{P_l}\cdot\frac{\lambda_2}{\mu_2}p_{2,1}(t)f_{2AQ_m}(t)\cdot \frac{\lambda_3}{\mu_3}p'_{3,1}(t)\pi_{R_n}\cdot L_\theta^S(l-1,m-1,n-1),$$

$i = 31$: "spe–P; spe–Q; nor–R"

$$L_\theta^S(31,A,l,m,n) = \pi_A^3\frac{\lambda_1}{\mu_1}p'_{1,1}(t)\pi_{P_l}\cdot\frac{\lambda_2}{\mu_2}p'_{2,1}(t)\pi_{Q_m}\cdot \frac{\lambda_3}{\mu_3}p_{3,1}(t)f_{3AR_n}(t)\cdot L_\theta^S(l-1,m-1,n-1),$$

$i = 32$: "spe–P; spe–Q; spe–R"

$$L_\theta^S(32,A,l,m,n) = \pi_A^3\frac{\lambda_1}{\mu_1}p'_{1,1}(t)\pi_{P_l}\cdot\frac{\lambda_2}{\mu_2}p'_{2,1}(t)\pi_{Q_m}\cdot \frac{\lambda_3}{\mu_3}p'_{3,1}(t)\pi_{R_n}\cdot L_\theta^S(l-1,m-1,n-1),$$

Then, we consider G, C, T as the ancestral nucleotide in X, respectively and get $L^S(i,G,l,m,n)$, $L^S(i,C,l,m,n)$, $L^S(i,T,l,m,n)$, where $7 \le i \le 32$. Now we define

$$L_\theta^S(l,m,n) = \sum_{0\le i\le 6} L_\theta(i,l,m,n) + \sum_{7\le i\le 32}\sum_{Y\in\{A,G,C,T\}} L_\theta^S(i,Y,l,m,n)$$

Thus $L_\theta^S(|P|,|Q|,|R|)$ is the summation of all likelihoods of all possible canonical star alignments of the three observed sequences P, Q, R with all possible ancestral DNA sequences (see [Wat84]).

The sum approach will estimate the evolutionary parameter $\theta = (\lambda_1 t, \mu_1 t, s_1 t, \lambda_2 t, \mu_2 t, s_2 t, \lambda_3 t, \mu_3 t, s_3 t)$ by maximizing the likelihood function $L^S_\theta(|P|, |Q|, |R|)$ using the simplex method. The simplex method will not be described here. The C code for this maximization routine was published in [PFTV88]. The simplex method is often used to estimate the maximum (or minimum) value of a function when the maximum (or minimum) cannot be found exactly. In our case, the simplex method searches the surface of the likelihood function $L^S_\theta(|P|, |Q|, |R|)$ for the value of θ that maximizes the likelihood. At the beginning, the simplex method requires 10 initial values of θ. Then it will climb the likelihood function surface toward the maximum value of the likelihood untill a predefined tolerance is reached. A single evaluation of $L^S_\theta(|P|, |Q|, |R|)$ is called an *iteration.* If I is the number of iterations required by the simplex method, then the computational complexity for the sum approach is $O(I \cdot |P| \cdot |Q| \cdot |R|)$.

At last, the estimated evolutionary parameter θ is used to find a star alignment with the maximum likelihood by the usual dynamic programming method. The time and space complexity of the dynamic programming is $O(|P| \cdot |Q| \cdot |R|)$.

3.3 The Direct Alignment Approach

In this section, we will define the likelihood function, $L^D_\theta(|P|, |Q|, |R|)$, to be maximized in the direct alignment approach.

Given a set of evolutionary parameters θ, we define $L^D_\theta(|P|, |Q|, |R|)$ to be the likelihood of a star alignment of four DNA sequences, X, P, Q, R, with the maximum likelihood among all other possible alignments with all possible X's. As in the sum approach, $L^D_\theta(|P|, |Q|, |R|)$ can also be defined inductively.

Initially, $L_\theta^D(0,0,0) = p''_{1,1}(t)(1-\frac{\lambda_1}{\mu_1}) \cdot p''_{2,1}(t)(1-\frac{\lambda_2}{\mu_2}) \cdot p''_{3,1}(t)(1-\frac{\lambda_3}{\mu_3})$.

Define $L_\theta^D(l,m,n) = 0$ for $l < 0$, or $m < 0$, or $n < 0$.

Let i, $0 \leq i \leq 6$, be a column type. $L_\theta^D(i,l,m,n)$ denotes the maximum likelihood of any canonical star alignment with the last column having type i of the sequences, $P_1 \cdots P_l, Q_1 \cdots Q_m, R_1 \cdots R_n$. $L_\theta^D(i,A,l,m,n)$ denotes the maximum likelihood of all possible canonical star alignments of sequences $P_1 \cdots P_l$, $Q_1 \cdots Q_m$, $R_1 \cdots R_n$ with the last column having type i, and the nucleotide of the ancestral sequence in the last column being A. $L_\theta^S(i,G,l,m,n)$, $L_\theta^D(i,C,l,m,n)$ and $L_\theta^D(i,T,l,m,n)$ can be similarly defined.

Then $L_\theta^D(l,m,n)$ is defined to be the maximum value of $L_\theta^D(i,l,m,n)$ and $L_\theta^D(j,Y,l,m,n)$ where $0 \leq i \leq 7$, $7 \leq j \leq 32$ and $Y \in \{A,G,C,T\}$. Moreover, specially for the direct alignment approach, define $T(l,m,n)$ to be the i such that either $L_\theta^D(i,l,m,n) = L_\theta^D(l,m,n)$ or $L_\theta^D(i,Y,l,m,n) = L_\theta^D(l,m,n)$ where $Y \in \{A,G,C,T\}$. Initially set $T(l,m,n) = -1$ for all l,m,n.

It is easy to see that $L_\theta^D(|P|,|Q|,|R|)$ is the maximum likelihood among the likelihoods of all possible canonical star alignments of sequences P, Q, R.

Suppose that we we have computed the values of $L_\theta^D(p,q,r)$ and $T(p,q,r)$ for $p \leq l, q \leq m, r \leq n$ and $(p,q,r) \neq (l,m,n)$. Then we will inductively compute $L_\theta^D(l,m,n)$ and $T(l,m,n)$ by computing $L_\theta^D(i,l,m,n)$ where $0 \leq i \leq 6$ and $L_\theta^D(j,Y,l,m,n)$ where $7 \leq j \leq 32$ and $Y \in \{A,G,C,T\}$.

First, let's consider $L_\theta^D(i,l,m,n)$ where $0 \leq i \leq 6$. The important idea is that if the next to last column is a left-offensive column of i, the likelihood, $L_\theta^D(i,l,m,n)$, is zero because the alignment is a duplicate.

$$L_\theta^D(0,l,m,n) = \begin{cases} 0 & \text{if } T(l-1,m,n) \in \{1,2,3\}, \\ \pi_{P_l}\lambda_1\beta_1(t)\cdot L_\theta^D(l-1,m,n) & \text{otherwise.} \end{cases}$$

$$L_\theta^D(1,l,m,n) = \begin{cases} 0 & \text{if } T(l,m-1,n) \in \{0,2,4\}, \\ \pi_{Q_m}\lambda_2\beta_2(t)\cdot L_\theta^D(l,m-1,n) & \text{otherwise.} \end{cases}$$

$$L_\theta^D(2,l,m,n) = \begin{cases} 0 & \text{if } T(l,m,n-1) \in \{0,1,5\}, \\ \pi_{R_n}\lambda_3\beta_3(t)\cdot L_\theta^D(l,m,n-1) & \text{otherwise.} \end{cases}$$

$$L_\theta^D(3,l,m,n) = \begin{cases} 0 & \text{if } T(l,m-1,n-1) \in \{0,1,2,4,5\}, \\ \pi_{Q_m}\lambda_2\beta_2(t)\cdot\pi_{R_n}\lambda_3\beta_3(t)\cdot L_\theta^D(l,m-1,n-1) & \text{otherwise.} \end{cases}$$

$$L_\theta^D(4,l,m,n) = \begin{cases} 0 & \text{if } T(l-1,m,n-1) \in \{0,1,2,3,5\}, \\ \pi_{P_l}\lambda_1\beta_1(t)\cdot\pi_{R_n}\lambda_3\beta_3(t)\cdot L_\theta^D(l-1,m,n-1) & \text{otherwise.} \end{cases}$$

$$L_\theta^D(5,l,m,n) = \begin{cases} 0 & \text{if } T(l-1,m-1,n) \in \{0,1,2,3,4\}, \\ \pi_{P_l}\lambda_1\beta_1(t)\cdot\pi_{Q_m}\lambda_2\beta_2(t)\cdot L_\theta^D(l-1,m-1,n) & \text{otherwise.} \end{cases}$$

$$L_\theta^D(6,l,m,n) = \begin{cases} 0 & \text{if } T(l-1,m-1,n-1) \in \{0,1,2,3,4,5\}, \\ \pi_{P_l}\lambda_1\beta_1(t)\cdot\pi_{Q_m}\lambda_2\beta_2(t)\cdot\pi_{R_n}\lambda_3\beta_3(t)\cdot L_\theta^D(l-1,m-1,n-1) & \text{o.w.} \end{cases}$$

Secondly, we consider the deletion-substitution columns. As we mentioned in Section 3.1, the deletion-substitution columns will not give rise to non-canonical star alignment. Therefore $L_\theta^D(i,A,l,m,n)$, $L^D(i,G,l,m,n)$, $L^D(i,C,l,m,n)$, $L^D(i,T,l,m,n)$, where

$7 \leq i \leq 32$, can be calculated in the same way as computing $L^S_\theta(i, A, l, m, n)$ except that we should replace $L^S_\theta(i, A, l, m, n)$ with $L^D_\theta(i, A, l, m, n)$ (see Section 3.2).

Finally, define $L^D_\theta(l, m, n)$ to be the maximum likelihood among all the likelihoods defined above. That is,

$$L^D_\theta(l, m, n) = \max\{\max_{0 \leq i \leq 6} L^D_\theta(i, l, m, n), \max_{7 \leq j \leq 32} \max_{Y \in \{A,G,C,T\}} L^D_\theta(j, Y, l, m, n)\};$$

and define

$$I = \{i \mid L^D_\theta(i, l, m, n) = L^D_\theta(l, m, n) \vee \exists_{Y \in \{A,G,C,T\}}(L^D_\theta(i, Y, l, m, n) = L^D_\theta(l, m, n))\}$$

and

$$T(l, m, n) = \min\{i \mid i \in I\}.$$

Thus, we obtain $L^D_\theta(|P|, |Q|, |R|)$ which is the objective function to be maximized in the direct alignment approach.

Simplex method is again used here to estimate the evolutionary parameter θ which maximumizes the value of $L^D_\theta(|P|, |Q|, |R|)$ in the same way as in the sum approach. Then a star alignment with maximum likelihood is found by dynamic programming using the estimated parameter θ.

Chapter 4

The MLSAS System Development

The programs in MLSAS are divided into two parts. The first part deals with input from a file containing three observed molecular sequences, calculates the likelihood functions, estimates the evolutionary parameters by maximizing the likelihood functions, and aligns the three sequence with their ancestral sequences. This part is written in C. The second part of MLSAS generates a graphical user interface featuring multi-tasking. It makes MLSAS easy to use. The second part is written in C and MOTIF.

The first part is independent of the second part. So MLSAS can also be used on computers without graphical facilities as a command-driven system.

In this chapter, we will briefly explain how the C files are organized in MLSAS. Hopefully, it will be helpful for interested readers to further develop the system. For the details of the programs, please contact the author for the code which is well-commented.

4.1 The Code Organization of the Algorithms

The first part of the programs in MLSAS consists of 9 files.

"*global_var.h*" declares the global variables in the whole system. They are four pointers to and the lengths of the four sequences involved in the star alignment and the four equilibrium base frequencies.

"*arrayutil.c*" prepares some useful routines to allocate and deallocate one-dimensional, two-dimensional, three-dimensional and four dimensional arrays. These routines are frequently used in the system to allocate and deallocate memory.

"*Get_input.c*" allocates memory for the three input sequences, converts the characters A, G, C, T or A, G, C, U into 0, 1, 2, 3 respectively. It calculates the equilibrium base frequencies by dividing the number of each nucleotide appearing in the three input sequences by the total length of the sequences. It defines 10 groups of initial values of the 9 evolutionary parameters. The choice of the initial values is sometimes important for the optimization results when the iterations of the optimization routines are limited due to the computer speed.

"*Sum.c*" calculates the likelihood for the sum approach. It has 6 extra 4-dimensional arrays to hold the likelihoods related to insertions in order to guarantee that all alignment taken into account are canonical. It then maximizes the likelihood by calling the maximization routine "`double simplex_max(double [][], double, double (*obj_func)(double []), int *)`" (defined in "*simplex_max.c* ") and outputs the evolutionary parameters reaching the maximum point. Finally it finds the canonical star alignment with the maximum likelihood for these evolutionary parameters. The routine for constructing a star alignment "`star_align(double *)`" is defined in the file "*Align.c*"

"*Direct.c*" calculates the $-\log$ likelihood[1] for the direct approach. It then minimizes the likelihood by calling minimization routine "`double simplex_min(double [][], double, double (*obj_func)(double []), int *)`" (defined in "*simplex_min.c*") and outputs the evolutionary parameters reaching the minimum point. Finally it finds the canonical star alignment with the maximum likelihood using the evolutionary parameters. It uses the star alignment routine "`star_align(double *)`"

"*Align.c*" finds a canonical star alignment with the maximum likelihood. The unknown ancestral sequence is inferred during the alignment. It also converts 0, 1, 2, 3, 4 into A, G, C, T, space (–) or A, G, C, U, space (–), respectively, and prints out the alignment.

"*simplex_max.c*" and "*simplex_min.c*" implement the simplex method to maximize and minimize a non-differential function. The maximum number of iterations for these optimization routine is set to bes 100 at the moment. The user could change it to a larger number to get better results if the computer speed permits.

"*main.c*" is the main program for the two algorithms. From it, two executable files "*sum*" and "*direct*" can be generated to fully implement the sum approach and the direct alignment approach.

4.2 The Graphical User Interface

The second part dealing with the graphical user interface consists of 10 files.

"*mlsas.c*" generates the main window of the user interface. The main window belongs

[1] Taking $-\log$ of the likelihood for the direct alignment approach is possible since $L_\theta^D(|P|,|Q|,|R|)$ has the form of product of $L_\gamma^D(l,m,n)$ for $0 \le l \le |P|, 0 \le m \le |Q|, 0 \le n \le |R|$. If P, Q, R become longer, the value of $L_\theta^D(|P|,|Q|,|R|)$ becomes smaller and may be smaller than the smallest floating number the computer can store. This problem can be avoided by taking $-\log$ of $L_\theta^D(|P|,|Q|,|R|)$.

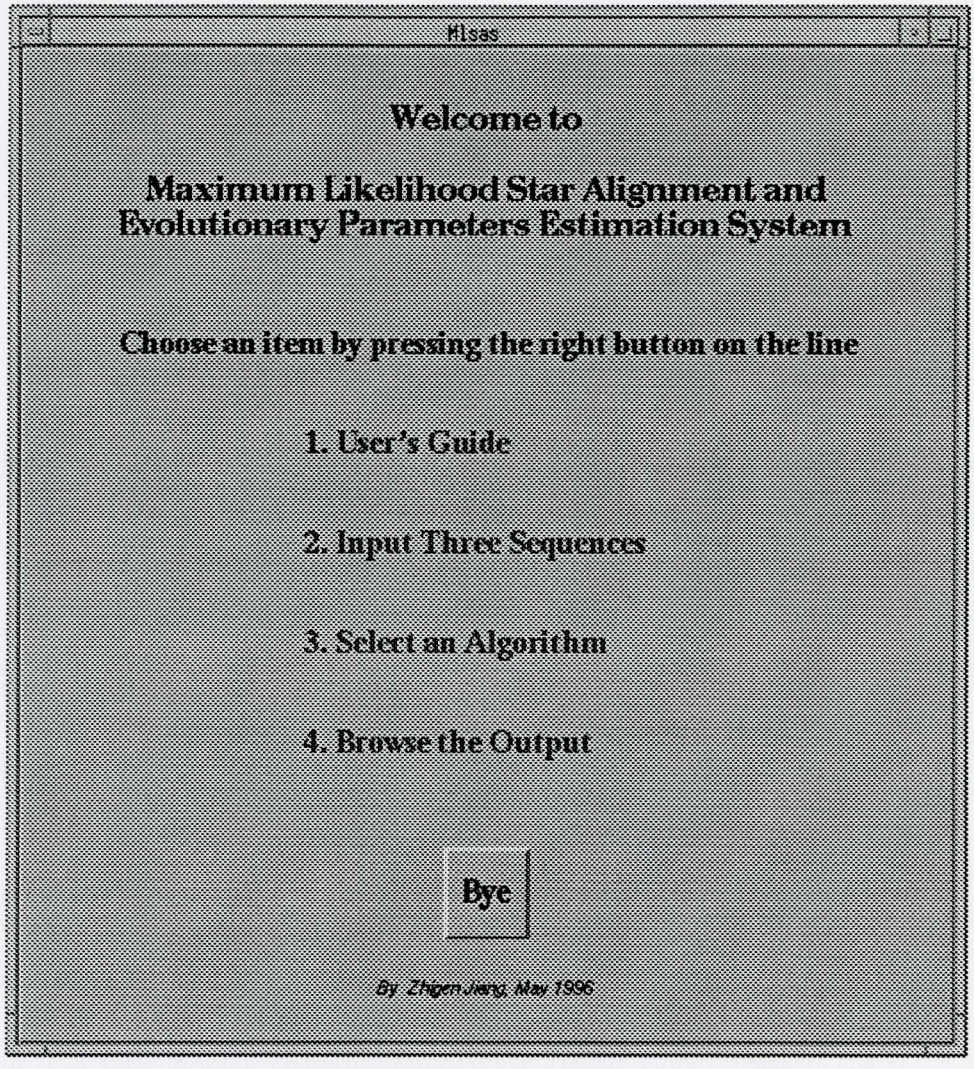


Figure 4.1: The main window of MLSAS

to "Form Class". It has 9 children eight of which are labels of either text or bitmap and one of which is "Dialog Class" letting the user exit from MLSAS after she/he is sure to quit. The layout, background and foreground colors, and the types of fonts of its children are specified in the file "./app-defaults/Mlsas". Figure 4.1 shows what the main window looks like after the MLSAS is started.

When the left button of the mouse is pressed on one of the 4 numbered lines, a popup menu is popped up and the user can choose relevant item to either get help or edit a input file or start a process or browse the output.

"*GuidePanel.c*" creates a popup menu on the line of "User's Guide" containing four items, which are general information about MLSAS, the format of the input file (delimiting each molecular sequence with a pair of "$\star$"'s), a brief description about the two approaches on the maximum likelihood star alignment, and how to read the running results within MLSAS. Figure 4.2 shows how the interface looks like when the left button of the mouse is pressed. Figure 4.3 shows the interface when the item "General Information" in the popup menu is chosen.

"*InputPanel.c*" creates a popup menu on the line of "Input Three Sequences", which consists of two items for the user to choose concerning an input method. The user can either type the name of the input file containing the three sequences in the dialog box (see Figure 4.4) or edit the input file within MLSAS.

"*AlgSelection.c*" creates a popup menu on the line of "Select an Algorithm". From this menu, the user could choose either the sum approach or the direct alignment approach to run on the current input (see Figure 4.5).

Once an algorithm is selected, a confirmation dialog box is popped up. If the user

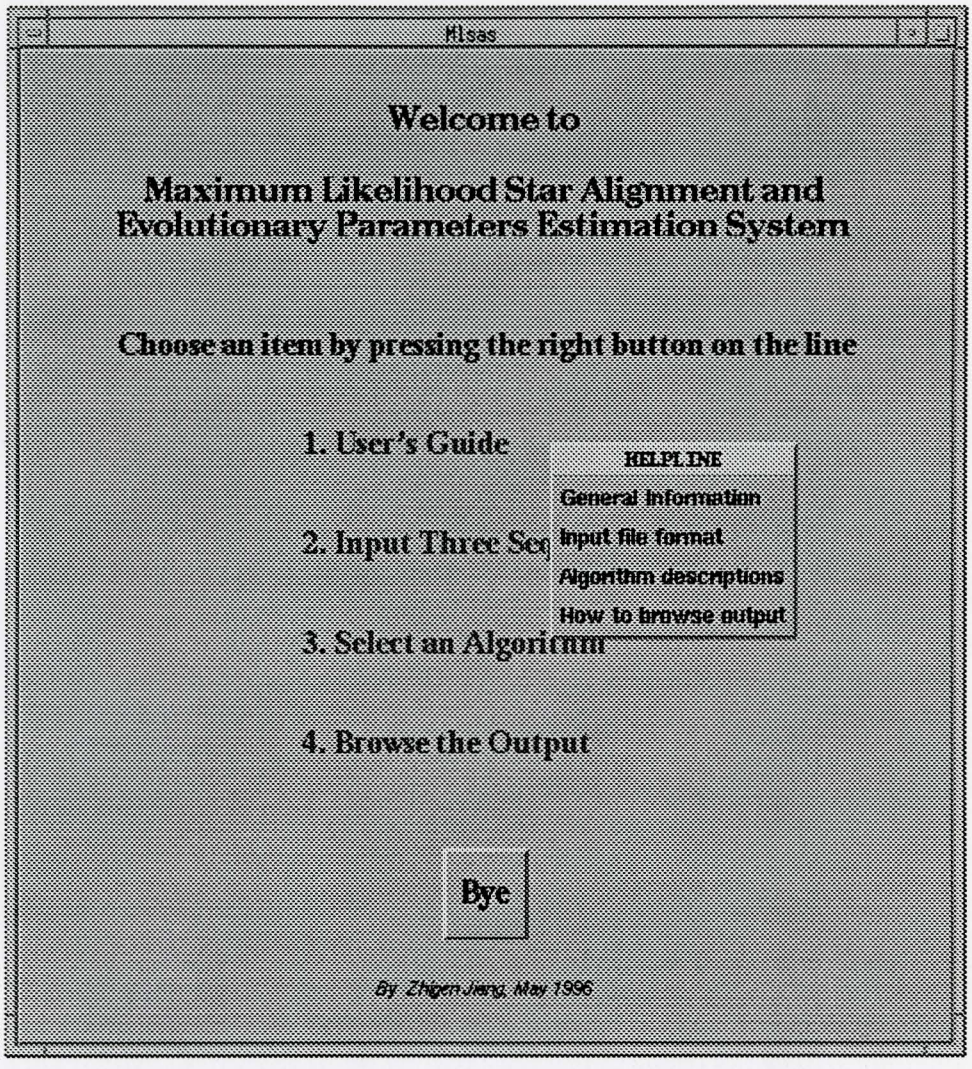


Figure 4.2: A popup menu on the line of "User's Guide"

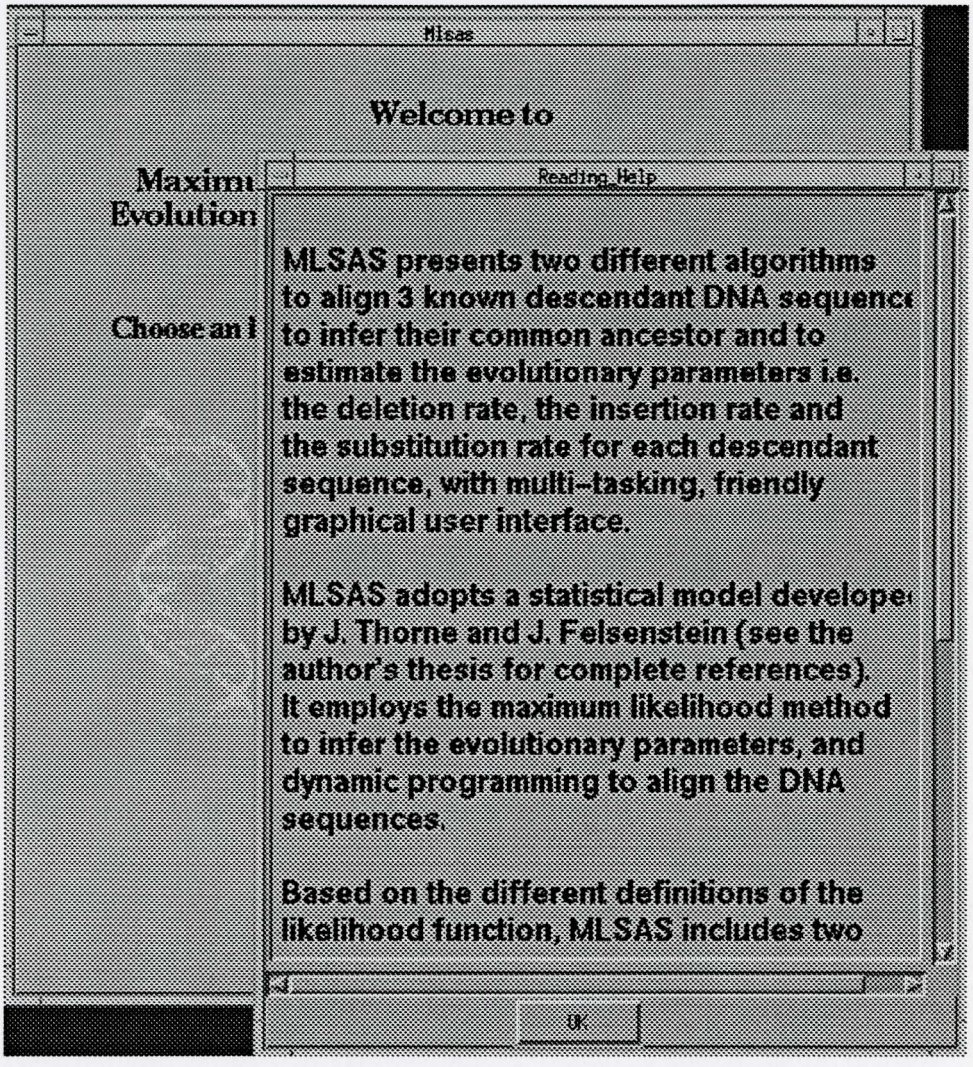


Figure 4.3: General information is displayed through the popup menu

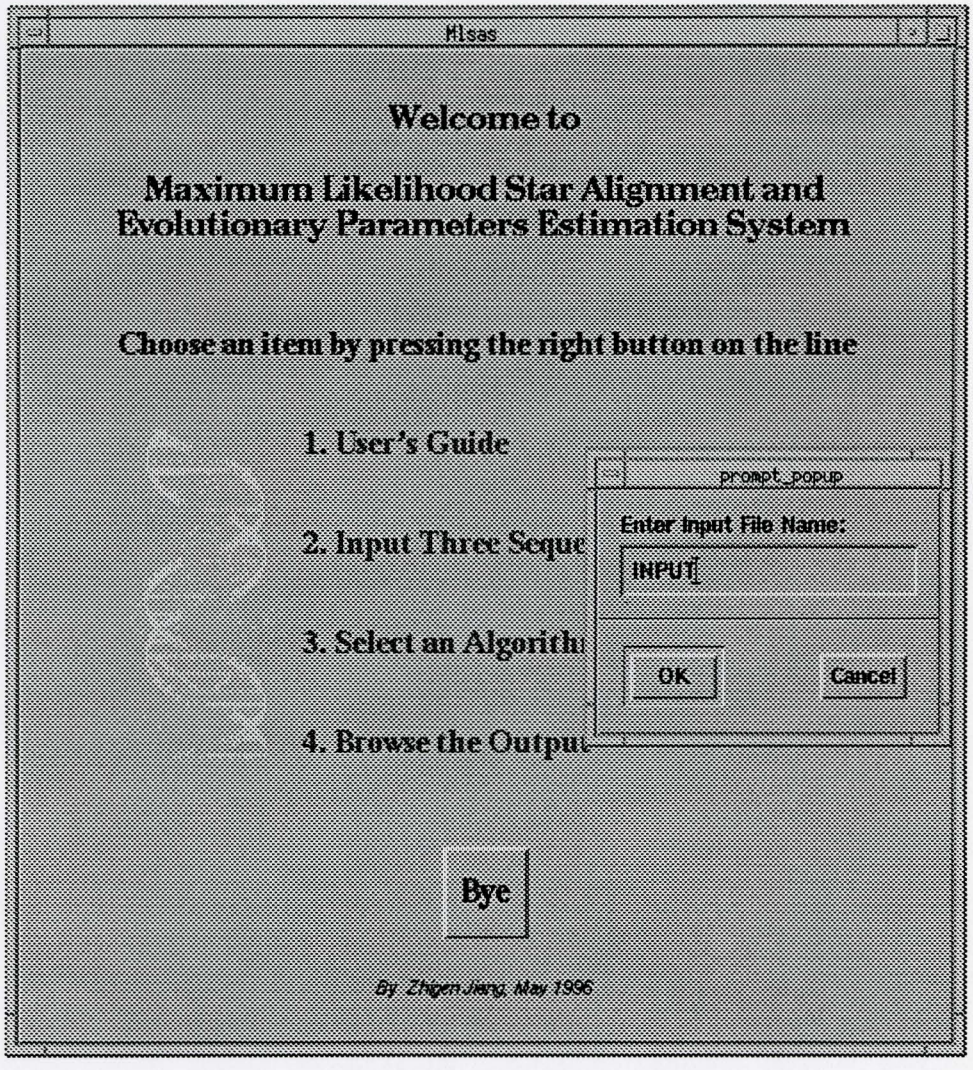


Figure 4.4: Prompt for input file name

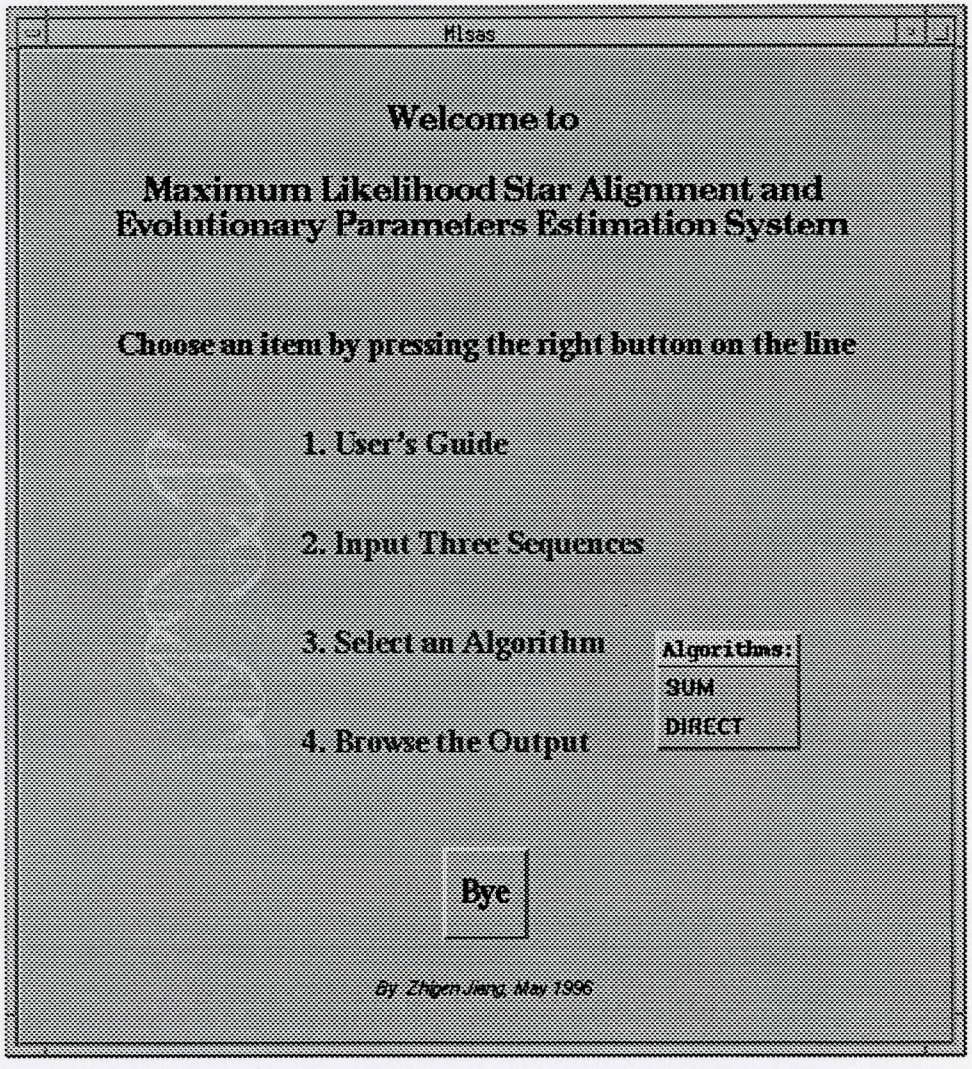


Figure 4.5: A Popup menu on the line of "Select an Algorithm"

Figure 4.6: A confirmation dialog for the choice of algorithm

is sure about her/his choice, just press the "RUN" button. Otherwise press the "QUIT" button to abandon the choice (see Figure 4.6). Note that MLSAS will fork a child process to execute the program in order to run multiple programs simultaneously within the MLSAS.

Once the "RUN" button is pressed, and an information dialog box is popped up to notify the user that the program is running in background. If the user changes his/her mind to stop the program, just press the "STOP" button. Press "QUIT" to remove the dialog box from the screen (see Figure 4.7).

As soon as a dispatched process terminates, a real-time notification window is popped up which shows how much time has elapsed since the process was started. "*Direct_Report.c*" and "*Sum_Report.c*" fulfill the notification task by creating a "Form" window as soon as the running process terminates. Figure 4.8 shows a notification window pops up when the sum program terminates.

The running results are stored in file "OutputDirect" for the direct alignment approach

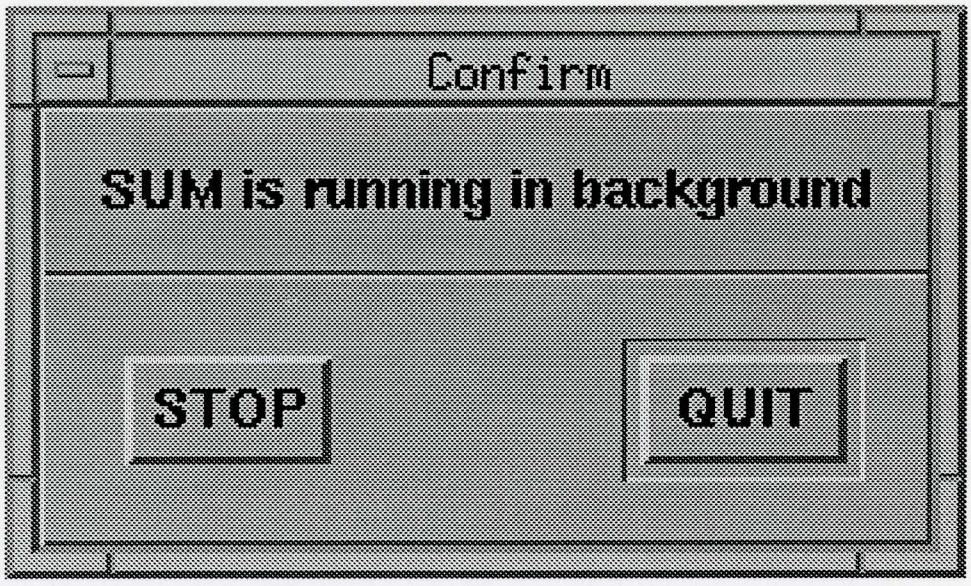


Figure 4.7: A dialog box to stop the running process

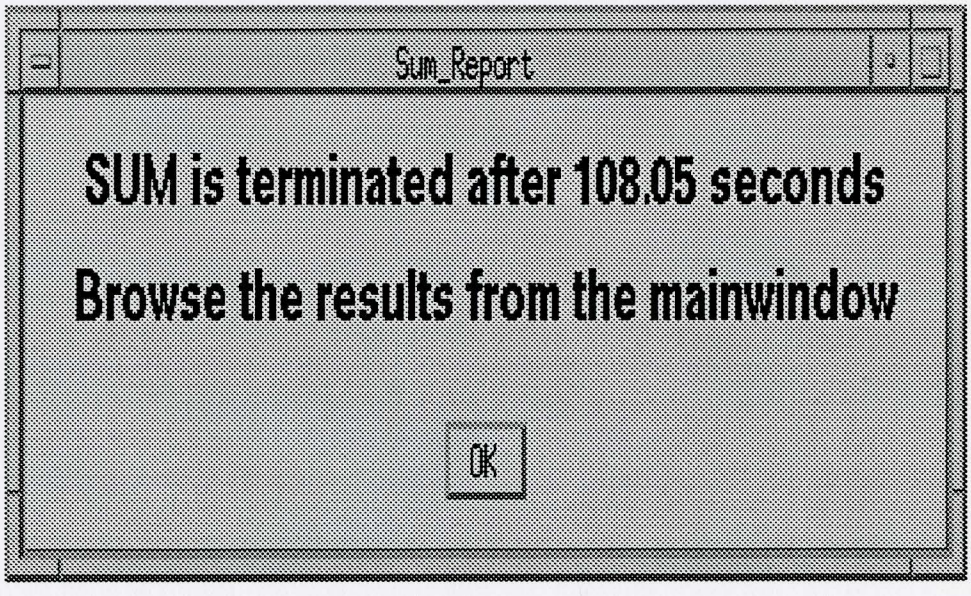


Figure 4.8: A notification window when a running process terminates

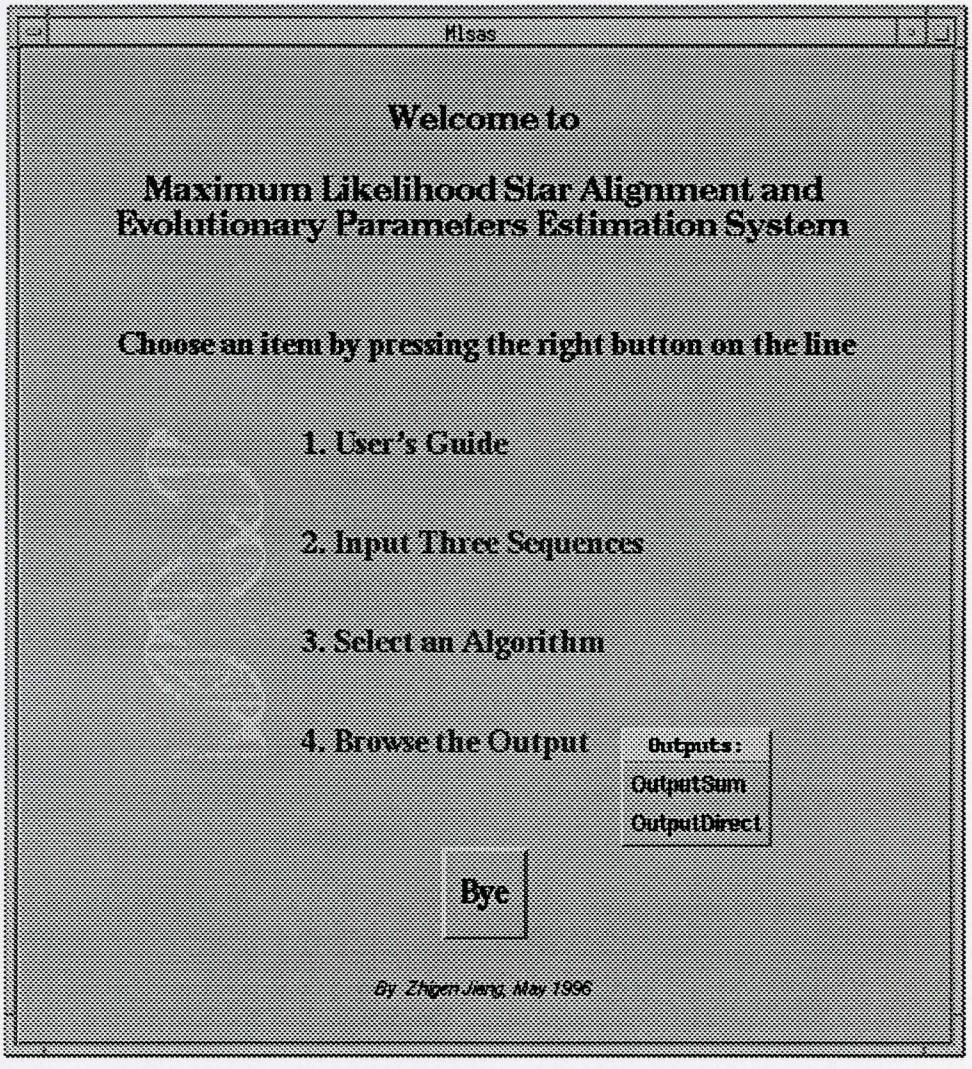


Figure 4.9: A popup menu on the line of "Browse the Output"

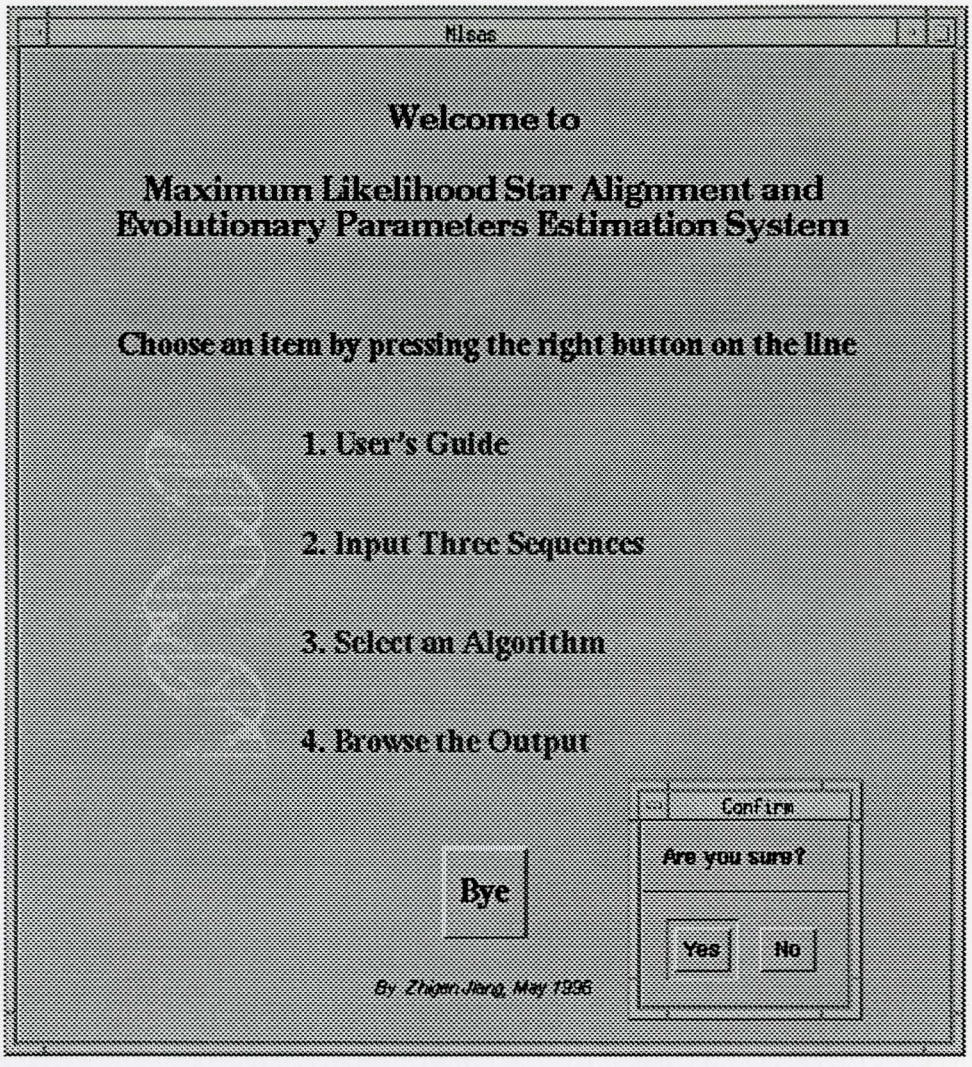


Figure 4.10: Exit MLSAS

and file "OutputSum" for the sum approach. "*BrowsePanel.c*" creates a popup menu on the line of "Browse the Output" (see Figure 4.9). The menu contains two items for the user to choose the file to be read. Once a file is chosen, the content of the file will be displayed in a read-only text widget in which the name of the file can be changed in order to avoid being over-written by MLSAS if it is running the same program.

To exit MLSAS, press the "Bye" button in the MLSAS main window (See Figure 4.10).

The callback functions used in the popup menu are defined in the following C files.

"*Editor.c*" creates an editor called in "*InputPanel.c*" for the user to edit an input file containing three molecular sequences. Its resource file is "./app-defaults/Editor".

"*File_Browser.c*" creates a read-only text widget to display running results. It is called in "*BrowsePanel.c*". Its resource file is "./app-defaults/File_browser".

"*Read_help.c*" builds a browser to display help information. It is called in "*GuidePanel.c*". Its resource file is "./app-defaults/Read_help".

Chapter 5

Simulations and Testing

In this chapter, we will approximately simulate the evolutionary process with pre-assigned parameters from a chosen ancestor to three descendants and then let MLSAS run on the three evolved sequences to see how precise and reliable MLSAS is.

Finally, we will provide a real example to find the ancestral sequence of three observed sequences.

We know that the sum approach is more unbiased than the direct alignment approach because its estimation of parameters is based on the summation of the likelihoods of all alignments rather than a single aliment with maximum likelihood although the direct alignment approach is faster. From the simulation results, we will see that the two approaches give very accurate parameter estimation as shown in Case Study 1 where the evolutionary distance is small. Also, we will see that the sum approach is more accurate (the smaller standard error) than the direct alignment approach when the evolutionary distance becomes larger (See Table 5.5 and Table 5.6).

5.1 Simulation Study

Suppose that we are given a DNA sequence X and a set of evolutionary parameters $\theta = (\lambda t, \mu t, st)$, where λ is the birth rate per normal or immortal link during the evolution from X to a descendant sequence Y, μ is the death rate per normal link during that evolution, s is the substitution rate during that evolution. We will simulate the evolutionary process to obtain the descendant sequence Y according to the model defined in Chapter 2.

Let i, j be two nucleotides. Let $L^D = \frac{\lambda}{\mu}p_0'(t)(= \lambda\beta(t))$, $L^N_{i,j} = \frac{\lambda}{\mu}f_{i,j}(t)p_1(t)$, $L^S_j = \frac{\lambda}{\mu}\pi_j p_1'(t)$, and $L^I_j = \pi_j\lambda\beta(t)$.

For each nucleotide i in X, define

$$L^T_i = L^D + \sum_{0\le j\le 3}(L^I_j + L^S_j + L^N_{i,j}).$$

Therefore the probability of a deletion of a nucleotide i in X is $P^D = \frac{L^D}{L^T_i}$. The probability of the normal substitution of i with j is $P^N_{i,j} = \frac{L^N_{i,j}}{L^T_i}$. The probability of the special substitution of i with j is $P^S_j = \frac{L^S_j}{L^T_i}$. The probability of the insertion of nucleotide j is $P^I_j = \frac{L^I_j}{L^T_i}$.

So for each nucleotide in X, there are 13 possible evolutionary events to happen. The 13 possible events are four normal substitutions, four special substitutions, four insertions and one deletion. We divide the interval [0, 1] into 13 subintervals according to the probabilities of the 13 events such that the length of each interval equals the probability of the corresponding event. In this way a one-to-one relationship between the 13 subintervals and the 13 evolutionary events is established. During the simulation, we call a C routine "`drand()`" which generates a double precision real number in [0, 1] uniformly. If the value of "`drand()`" falls in a subinterval, then we assume that an evolutionary event corresponding to the subinterval happens and create a column of the alignment. Keep calling "`drand()`" until the last nucleotide in X is considered. At the end, we get a descendant sequence Y

and an alignment with X.

In the simulation study, we fix 6 sets of the values of the parameters $\theta = (\lambda t, \mu t, st)$ with smaller values in first 3 sets and larger values in the last 3 sets. For each θ we generate 10 descendant sequences of around 50 nucleotides each. We group the 60 descendants into two cases. In Case 1, we study the 30 descendant sequences in 10 triples generated from the first 3 sets of parameters. In Case 2, we study the other 30 descendant sequences in 10 triples generated from the last 3 sets of parameters.

Case Study 1: Descendant sequences with small evolutionary distance

In this study, we choose 3 fixed sets of parameters of small values as shown in the row "Simulation" of Table 5.1 and Table 5.2 and a fixed ancestral sequence as shown in Table 5.3, and Table 5.4 and simulate the evolutionary process as described in the beginning of this section to get 30 descendant sequences and group them in 10 triples.

Table 5.1 and Table 5.2 list the estimated parameters for each sequence by the direct alignment algorithm and the sum algorithm respectively.

Table 5.1 and Table 5.2 also give the average values of all 9 estimated parameters and the standard errors.

Standard error is a measure of precision of an estimate (see [Blo80]). The smaller the standard error is, the more satisfactory the estimate is. Usually there is many ways to choose a satisfactory standard error. In this paper we choose the variance, $V(\theta^*)$, as the standard error, where θ^* is an estimated unknown parameter and

$$V(\theta^*) = \sqrt{\frac{\sum_{i=1}^{N} \theta_i^{*2}}{N}}.$$

Table 5.1: Case Study 1: Evolutionary Parameters Estimated by the *direct* algorithm

sample	seq.	$\frac{\lambda}{\mu}$	μt	st	$-log$ likelihood
1	P	0.9789	0.0175	0.0217	251.407
	Q	0.9796	0.0175	0.0542	
	R	0.9787	0.0542	0.1042	
2	P	0.9783	0.0170	0.0209	263.812
	Q	0.9783	0.0170	0.0539	
	R	0.9780	0.0539	0.1039	
3	P	0.9789	0.0100	0.0100	257.201
	Q	0.9792	0.0100	0.0500	
	R	0.9785	0.0500	0.1000	
4	P	0.9781	0.0153	0.0181	259.354
	Q	0.9781	0.0153	0.0528	
	R	0.9767	0.0530	0.1028	
5	P	0.9787	0.0100	0.0100	248.092
	Q	0.9792	0.0100	0.0500	
	R	0.9792	0.0500	0.1000	
6	P	0.9789	0.0213	0.0275	270.258
	Q	0.9787	0.0213	0.0563	
	R	0.9796	0.0563	0.1063	
7	P	0.9777	0.0102	0.0104	278.06
	Q	0.9774	0.0106	0.0508	
	R	0.9785	0.0504	0.0998	
8	P	0.9789	0.0133	0.0151	268.399
	Q	0.9792	0.0133	0.0518	
	R	0.9785	0.0518	0.1018	
9	P	0.9796	0.0138	0.0138	255.135
	Q	0.9794	0.0096	0.0538	
	R	0.9789	0.0481	0.1188	
10	P	0.9783	0.0102	0.0100	256.853
	Q	0.9783	0.0100	0.0498	
	R	0.9783	0.0501	0.0998	
Simulation	P	0.9950	0.0100	0.0100	
	Q	0.9950	0.0100	0.0500	
	R	0.9950	0.0500	0.1000	
Standard error	P	0.9786	0.0139	0.0158	
		±0.0005	±0.0037	±0.0058	
	Q	0.9787	0.0135	0.0523	
		± 0.0007	±0.0039	±0.0021	
	R	0.9785	0.0518	0.1037	
		± 0.0007	±0.0024	± 0.0055	

Table 5.2: Case Study 1: Evolutionary Parameters Estimated by the *sum* algorithm

sample	seq.	$\frac{\lambda}{\mu}$	μt	st	$-log$ likelihood
1	P	0.9748	0.0224	0.0281	247.379
	Q	0.9754	0.0225	0.0563	
	R	0.9745	0.0561	0.1059	
2	P	0.9721	0.0305	0.0272	258.121
	Q	0.9721	0.0289	0.0547	
	R	0.9719	0.0640	0.1031	
3	P	0.9766	0.0116	0.0140	252.66
	Q	0.9768	0.0147	0.0550	
	R	0.9761	0.0511	0.1027	
4	P	0.9622	0.0233	0.0093	251.398
	Q	0.9622	0.0194	0.0463	
	R	0.9608	0.0672	0.0926	
5	P	0.9787	0.0100	0.0100	245.879
	Q	0.9792	0.0100	0.0500	
	R	0.9792	0.0500	0.1000	
6	P	0.9718	0.0321	0.0271	264.103
	Q	0.9716	0.0302	0.0544	
	R	0.9724	0.0654	0.1026	
7	P	0.9656	0.0190	0.0138	270.883
	Q	0.9654	0.0101	0.0497	
	R	0.9664	0.0632	0.1034	
8	P	0.9756	0.0176	0.0205	265.267
	Q	0.9758	0.0191	0.0550	
	R	0.9752	0.0561	0.1045	
9	P	0.9791	0.0126	0.0124	252.625
	Q	0.9789	0.0100	0.0521	
	R	0.9785	0.0488	0.1110	
10	P	0.9788	0.0101	0.0100	252.513
	Q	0.9788	0.0100	0.0500	
	R	0.9788	0.0501	0.0999	
Simulation	P	0.9950	0.0100	0.0100	
	Q	0.9950	0.0100	0.0500	
	R	0.9950	0.0500	0.1000	
Standard error	P	0.9735	0.0189	0.0172	
		±0.0055	±0.0077	±0.0073	
	Q	0.9736	0.0175	0.0524	
		±0.0056	±0.0074	± 0.0031	
	R	0.9734	0.0572	0.1026	
		± 0.0056	±0.0068	±0.0045	

Table 5.3: Case Study 1: The ancestral DNA sequences estimated by the *direct* algorithm

Sample	Ancestor		m
1	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	-CTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	1
2	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTC-TCTAGCGAGTC	1
3	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	0
4	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCC-TCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	1
5	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	0
6	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTT-TAGCGAGTC	1
7	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	0
8	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CC-AACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	1
9	X	CCTAACACGCCCCC-TCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	1
10	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	0

Table 5.4: Case Study 1: The ancestral DNA sequences estimated by the *sum* algorithm

Sample	Ancestor		m
1	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	C-TAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	1
2	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCT-CTAGCGAG-C	1
3	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCA-CCTAACAGGCTCTTCTAGCGAGTC	1
4	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACG-CCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	1
5	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	0
6	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTT-TAGCGAGTC	1
7	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCT-ACAGGCTCTTCTAGCGAGTC	1
8	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CC-AACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	1
9	X	CCTAACACGCCCCC-TCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	1
10	X	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	
	X^*	CCTAACACGCCCCCTCTCTCATCCTAACAGGCTCTTCTAGCGAGTC	0

Table 5.3 and Table 5.4 show the estimated ancestral sequence for each group compared with the prefixed ancestral sequence. In the tables, X stands for the prefixed ancestral sequence, while X^* stands for the estimated ancestral sequence. The value of m stands for the number of mismatches between X and X^*.

Case Study 2: Descendant sequences with large evolutionary distance

In this study, we choose 3 fixed sets of parameters of not-so-small larger values as shown in the row "Simulation" of Table 5.5 and Table 5.6 and a fixed ancestral sequence as shown in Table 5.7 and Table 5.8, and simulate the evolutionary process as described in the beginning of this section to get 30 descendant sequences and group them in 10 triples.

Table 5.5 and Table 5.6 list the estimated parameters for each sequence by the direct alignment algorithm and the sum algorithm respectively.

Table 5.5 and Table 5.6 also give the average values of all 9 estimated parameters and the standard errors.

Table 5.7 and Table 5.8 show the estimated ancestral sequence for each group compared with the prefixed ancestral sequence. In the tables, again X stands for the prefixed ancestral sequence, X^* stands for the estimated ancestral sequence, and m stands for the number of mismatches between X and X^*.

5.2 Testing on Some Real DNA Sequences

In this section, we run MLSAS on two sets of real DNA sequences. In the first example, we compare the alignment obtained by the direct alignment algorithm of MLSAS, with the alignment obtained by a popular program called CLUSTAL V [HBF92]. In the second

Table 5.5: Case Study 2: Evolutionary Parameters Estimated by the *direct* algorithm

sample	seq.	$\frac{\lambda}{\mu}$	μt	st	$-log$ likelihood
1	P	0.9804	0.1250	0.2464	378.768
	Q	0.9808	0.1643	0.2178	
	R	0.9796	0.1250	0.2464	
2	P	0.9776	0.0841	0.3559	389.152
	Q	0.9776	0.0891	0.3286	
	R	0.9784	0.0993	0.3891	
3	P	0.9804	0.0694	0.5325	388.719
	Q	0.9808	0.0745	0.5233	
	R	0.9798	0.0694	0.5682	
4	P	0.9672	0.2120	0.2566	333.736
	Q	0.9661	0.1090	0.1957	
	R	0.9646	0.1078	0.2310	
5	P	0.9796	0.0850	0.3375	358.233
	Q	0.9787	0.0875	0.3075	
	R	0.9792	0.0850	0.3550	
6	P	0.9744	0.2044	0.2348	354.076
	Q	0.9724	0.1024	0.1732	
	R	0.9742	0.2056	0.2599	
7	P	0.9808	0.1877	0.2924	396.969
	Q	0.9808	0.1500	0.2470	
	R	0.9818	0.1877	0.3080	
8	P	0.9796	0.0850	0.3375	371.497
	Q	0.9794	0.0875	0.3075	
	R	0.9794	0.0850	0.3550	
9	P	0.9788	0.1066	0.1972	388.461
	Q	0.9792	0.1157	0.1838	
	R	0.9797	0.1662	0.2293	
10	P	0.9796	0.1125	0.2021	362.048
	Q	0.9794	0.1383	0.1666	
	R	0.9792	0.1125	0.2021	
Simulation	P	0.9950	0.0100	0.5000	
	Q	0.9950	0.1000	0.500	
	R	0.9950	0.5000	1.0000	
Standard error		0.9778	0.1272	0.2993	
	P	±0.0040	±0.0512	±0.0944	
		0.9775	0.1118	0.2651	
	Q	± 0.0045	±0.0285	±0.1032	
		0.9776	0.1243	0.3144	
	R	± 0.0047	±0.0442	± 0.1038	

Table 5.6: Case Study 2: Evolutionary Parameters Estimated by the *sum* algorithm

sample	seq.	$\frac{\lambda}{\mu}$	μt	st	$-log$ likelihood
1	P	0.9738	0.3680	0.4519	316.577
	Q	0.9742	0.2312	0.4493	
	R	0.9730	0.3746	0.5225	
2	P	0.9790	0.4168	0.4781	321.281
	Q	0.9790	0.1823	0.4397	
	R	0.9798	0.4182	0.5583	
3	P	0.9792	0.3827	0.4698	319.248
	Q	0.9796	0.2166	0.4571	
	R	0.9786	0.3837	0.5411	
4	P	0.9732	0.3944	0.4594	280.608
	Q	0.9721	0.2055	0.4375	
	R	0.9706	0.3991	0.5349	
5	P	0.9726	0.4323	0.4670	298.489
	Q	0.9717	0.1669	0.4161	
	R	0.9722	0.4393	0.5539	
6	P	0.9735	0.4814	0.4813	290.013
	Q	0.9714	0.1179	0.3938	
	R	0.9733	0.4875	0.5800	
7	P	0.9794	0.4279	0.4810	326.263
	Q	0.9794	0.1712	0.4343	
	R	0.9805	0.4292	0.5638	
8	P	0.9791	0.3993	0.4744	309.529
	Q	0.9789	0.2004	0.4497	
	R	0.9789	0.3997	0.5495	
9	P	0.9739	0.3933	0.4578	327.389
	Q	0.9742	0.2059	0.4363	
	R	0.9747	0.4000	0.5349	
10	P	0.9723	0.3927	0.4565	305.945
	Q	0.9721	0.2064	0.4352	
	R	0.9719	0.4000	0.5337	
Simulation	P	0.9950	0.0100	0.5000	
	Q	0.9950	0.1000	0.500	
	R	0.9950	0.5000	1.0000	
Standard error	P	0.9756	0.4089	0.4677	
		±0.0030	±0.0307	±0.0103	
	Q	0.9753	0.1904	0.4349	
		± 0.0034	±0.0307	±0.0173	
	R	0.9754	0.4131	0.5473	
		± 0.0035	±0.0309	± 0.0163	

Table 5.7: Case Study 2: The ancestral DNA sequences estimated by the *direct* algorithm

Sample		Ancestor	m
1	X	GCAGAAAGTGGTCATTGTGGATGATCT-CCTGGCCACTGGAGGTAAGGA	22
	X^*	--A-A-CGTG-TG-TTG-GGCT-ACATACCCGGC-AATGTAG-T-AG--	
2	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGA-GGTAAGGA	15
	X^*	GGTGC---TGGTAATTG-GGACGATCTCC-GACCA--GGCCGGTAAGGA	
3	X	GCAGAAAG-TGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	15
	X^*	GCAGT-AGCTGTTCGTTTTGGGTGATTCCCCGGCATCTG-AGG-AAGGG	
4	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	25
	X^*	GCAG---G-GG-C-GCGT-GAT---CTACT--CC---GCCGGATAGG-	
5	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	19
	X^*	-CA-AAAG-GGT--TTATGGAATTGCTTCTCG-CGCTGGGAGGAAATA	
6	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	21
	X^*	GC--AACGAGCT-GTTGTGG-TC-TCT---GGCC--T--AG-T--GGC	
7	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	19
	X^*	ACACAAA-T--TCATT-T-GAT--TCTA--GTC----GGCTGTCA-GA	
8	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	15
	X^*	CAAGAA-GCCG-CATT-TGACTGA-CTCCTAGGCAATGCAGCTAA-GA	
9	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	19
	X^*	G-TGA-AGTG-TCC-CG-A-ATG-T-GC-TGTCCCCAGGAGTTATGGA	
10	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	18
	X^*	GCCCAAAGT-GT--T-AA-AAGGATCTCCTT-CCAC-GG-CGTA-GG-	

Table 5.8: Case Study 2: The ancestral DNA sequences estimated by the *sum* algorithm

Sample		Ancestor	m
1	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	24
	X^*	--A-A---TG-T-GTTG--G-T-A-C-CCCGG--AATT---GT-AG--	
2	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	25
	X^*	G--G---GT--TGA----GG--GA-CTCC-G-CC---G-C--CA-GG-	
3	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	27
	X^*	-CAG--A--C-G--TTGTGG--GAT---A-G---A-T--AGG------	
4	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	27
	X^*	G--G---G-GG-C---GC-G-TG--CTCCCGGC---T--A-G-----	
5	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	25
	X^*	--A-A-AG--GT-A-T----AT--TCTC--G-CC---G-AGG-AAT--	
6	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	27
	X^*	-C-G-A------CATT-T---TG-T-T--T--CC---GG--GTAA---	
7	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	26
	X^*	-CA-AAA-----C-TTG---AT---CTCCTGGTCA----------GA	
8	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	28
	X^*	-C-G--AG-CC-CCTT-TG--T---CT---G---ACTG-C---A--GA	
9	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	27
	X^*	G--GAA--TC-TGATTG-C--TG--C-C-AG---AGT-----TT-G--	
10	X	GCAGAAAGTGGTCATTGTGGATGATCTCCTGGCCACTGGAGGTAAGGA	28
	X^*	GC-CA-AGT--TAA--A---A--A-C-CCTTG-----GGA-G---G--	

example, we compare the ancestral sequence and alignment estimated by MLSAS with the ancestral sequence and alignment found by a star alignment program provided by the software SAT (see [Che94]).

Case Study 1: Comparison of alignment construction with CLUSTAL V

We choose parts of three APRT (adenine phosphoribosyltransferase) genes from rodents Gc, Rn, and Ma. The names Gc, Rn and Ma stand for Gerbillus campestris, Rattus norvegicus and Mesocricetus auratus.

Figure 5.1 shows the alignment of the three DNA sequences given by CLUSTAL V.

Figure 5.2 shows the alignment of the three DNA sequences done by MLSAS with the estimated ancestral sequence X^*.

Note that gap penalties for opening and extending gaps are considered by CLUSTAL V while MLSAS does not introduce the gap penalty. This explains why spaces tend to stick together in Figure 5.1 while the spaces tend to scatter among the alignment in Figure 5.2.

Case Study 2: Comparison of ancestor and alignment estimate with SAT

SAT is a sequence analysist tool ([Che94]), which implements the approximation algorithm proposed in [JLW94] to solve the tree alignment problem for a given phylogeny with a good approximation ratio. In [SCL76] and [Che94], a phylogeny about nine organisms is shown in Figure 5.3. The 5S RNA sequences for all the nine organisms are known. SAT finds the 5S RNA sequences for the internal nodes, the ancestral species, by dividing the tree into seven 3-components as also shown in Figure 5.3 and then minimizing the cost of each 3-component repeatedly.

Figure 5.4 shows the ancestral 5S RNA sequences for *E.*coli, *P.*fluorescens and organism

```
Rn: AA-GCTTGTGCTAAACA---TGTGCACACCAGGCTCTGTGACTGAGATTCAGAAAC
Gc: AA-GCT----------------------CCAGGCTCCATGCGTGAGTTTCTGAAAC
Ma: AATTCTTGTGCTAAATAACTTTCACTTACCAGTG-CCAAGCACGGGCTTCAGAAAC

Rn: ACCCTGGGGTAGCTGAATGTCCACCAGGAGTGTCCAGA------GGGAGG-TGAAC
Gc: ACGCTAGGGTAGCTGAATGTCCACCAGGGGAGGCCAGA------GGGAGGGTGGGC
Ma: ACCCTAGGGTCGCTGAATGTCCACCAGGGGAGTC-AGACATGTCCAGAGGGTGAGA


Rn: ACCCCAGAGAACAGAGTGGCCCTCACAAGTGCTCAGGGACCACAGT-CCTTTTGCC
Gc: ACCCCA-------GGGTGGCCCTGGGAAATGCTCAGGGGCCAGAGT-ACTCGTGCC
Ma: ACCCCAGAGAATTCGGTAGCCCTGACATGTGCT--------ACAATTACTGATGCC


Rn: CACTTCACTTCCTATTGGTACCCCCTGACCATGCTGTAGAAATTAGGG--------
Gc: CACTTGACTTCCTGTTGGAACCCCCTGGCCATGCTCCAGAAATGAGGGTATGTATG
Ma: CACTT-----CCTACTGGTTCCTCCTGGCCATACCTCAGGAATTAGGGCATGCTTT

Rn: ----------
Gc: CATCTTTCCACTT
Ma: CTGCCTGCTACAG
```

Figure 5.1: Star alignment of APRT genes from Rn, Gc, Ma produced by CLUSTAL V

```
X*:  AA-GCTTGTGCT--A-AAC-T-T-CAC--ACCAG-GC-C-GTG-AC-G-G-TTCAG
Rn:  AA-GCTTGTGCT--A-AACATGTGCAC--ACCAG-GCTCTGTG-ACTGAGATTCAG
Gc:  AA-GCTC---C---A-GGC-T---C-C--A--TG--C---GTG-A--G-T-TTCTG
Ma:  AATTCTTGTGCTAAATAACTT-T-CACTTACCAGTGC-C-AAGCACGG-GCTTCAG

X*:  AAACACCCTAGGGTAGCTGAATGTCCACCAGGGGAGTC-CAGA-G---GGAGGGTG
Rn:  AAACACCCTGGGGTAGCTGAATGTCCACCAGGAGTGTC-CAGA-G---GGAGG-TG
Gc:  AAACACGCTAGGGTAGCTGAATGTCCACCAGGGGAGGC-CAGA-G---GGAGGGTG
Ma:  AAACACCCTAGGGTCGCTGAATGTCCACCAGGGGAGTCAGACATGTCCAGAGGGTG

X*:  AGCACCCCAGAGAA--C-G-GTGGCCCT--GAC-A-TGCTC--GGG-CCA-AGTAC
Rn:  AACACCCCAGAGAA--CAGAGTGGCCCT--CACAAGTGCTCA-GGGACCACAGTCC
Gc:  GGCACCCCA-G------G-GTGGCCCTGGGAA-A-TGCTCAGGGG-CCAGAGTAC-
Ma:  AGAACCCCAGAGAATTC-G-GTAGCCCT--GAC-A-TG-T---GCT-ACA-ATTAC

X*:  T-G-TGCCCACTT-C-AC---TTCCT-TTGG--A-ACC-C----CCT-GGCCATGC
Rn:  T-TTTGCCCACTT-C-AC---TTCCTATTGG--T-ACC-C----CCT-GACCATGC
Gc:  TCG-TGCCCACTT-G-AC---TTCCTGTTGG--A-ACC-C----CCT-GGCCATGC
Ma:  T-GATGCCCACTTCCTACTGGTTCCTCCTGGCCATACCTCAGGAATTAGGGCATGC

X*:  T---G---T-A--G-ATGT-T--A------CA--G
Rn:  T---G---T-A--G-AAAT-T--A------GG--G
Gc:  TCCAGAAATGACGGTATGTATGCATCTTTCCACTT
Ma:  T---T---TCT--GCCTGC-T--A------CA--G
```

Figure 5.2: Star alignment of APRT genes from Rn, Gc, Ma produced by MLSAS

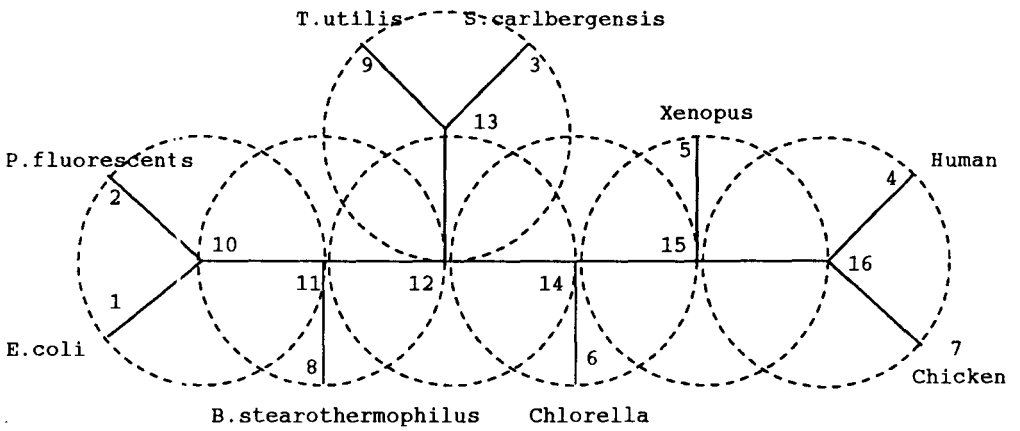


Figure 5.3: Phylogenetic relationship of nine organisms

```
10 :   UGCUUGCGGCCAUAGCAGCAGUGGAACACCUGACCCCAUCCCGAACUCAGAA
10*:   UGCUUGCGGCCGUAGC-GC--UGGAACACCUGACCCCAUCCCGAACUCAGAA

10 :   GUCAAACGCCGCAGCGCCGAUGGUAGUGUGGGGUCUCCCCAUGCGAGAGUAGG
10*:   GUCAAACGCCGCAGCGCCGAUGGUAGUGUGGGGUCUCCCCAUGCGAGAGUAGG

10 :   GAACCGCUAGGCAU
10*:   GA-CC-CUAGGCAU
```

Figure 5.4: The ancetral 5S RNAs for *E.*coli, *P.*fluorescens and organism 11 estimated by SAT and MLSAS

11 estimated by SAT and MLSAS. Sequence 10 is estimated by SAT and sequence 10* is estimated by MLSAS. The number of mismatches of the two estimated ancestral sequences is 6.

Figure 5.5 shows the alignment of *E.*coli, *P.*fluorescens and organism 11 estimated by MLSAS, and Figure 5.6 the alignment of *E.*coli, *P.*fluorescens and organism 11 estimated by SAT.

```
E.coli: UG--C-CUGGCG-GCCGUAGC-GCGGUG-GUCCCACCUGACCCCAUGCCGAAC
P.fluo: UGUUCUUUGACGAGUAGUAGC-AU--UG-G-AACACCUGAUCCCAUCCCGAAC
Org.11: -G--C-UUG-CG-GCCAUAGCAGC--AGAGAAACACCCGACCCCAUCCCGAAC

E.coli: UCAGAAGUGAAACGCCGUAGCGCCGAUGGUAGUGUGGGGUCUCCCCAUG-CGA
P.fluo: UCAGAGGUGAAACGAUGCAUCGCCGAUGGUAGUGUGGGGUUUCCCCAUGUCAA
Org.11: UCGGAAGUUAAGCUCCCCAGCGCCGAUGGUAGUGUGGGGUCACCCCCUG-CGA

E.coli: GAGUAGGGAACUGC-C-AG-GCAU
P.fluo: GA-UCUCG-AC--C-AUAGAGCAU
Org.11: GAGUAGGGUGC--CGCUAG-GC-U
```

Figure 5.5: Alignment of *E.*coli, *P.*fluorescens and organism 11 estimated by MLSAS

```
E.coli: UGCC-UGGCGGCCG-UAGC-GCGGUGGUCCCACCUGACCCCAUGCCGAACUCAGAA
P.fluo: UGUUCUU-UGACGAGUAGUAGCAUUGGAA-CACCUGAUCCCAUCCCGAACUCAGAG
Org.11: -GCU-UG-CGGCCA-UAGCAGCAGAGAAAC-ACCCGACCCCAUCCCGAACUCGGAA

E.coli: GUGAAACGCCGUAGCGCCGAUGGUAGUGUGGGGUCUCCCCAUG-CGAGAGUAGGGA
P.fluo: GUGAAACGAUGCAUCGCCGAUGGUAGUGUGGGGUUUCCCCAUGUCAAGA-UCUCGA
Org.11: GUUAAGCUCCCCAGCGCCGAUGGUAGUGUGGGGUCACCCCCUG-CGAGAGUAGGGU

E.coli: ACUGCCAG-GCAU
P.fluo: -CCA-UAGAGCAU
Org.11: GCCGCUAG-GC-U
```

Figure 5.6: Alignment of *E.*coli, *P.*fluorescens and organism 11 estimated by SAT

Chapter 6

Concluding Remarks

The maximum likelihood star alignment system, MLSAS, which is designed, implemented and tested in this thesis, is to alignment three molecualr sequences under an evolutionary model. MLSAS also estiamtes the evolutionary parameters and ancestral sequence. Chapter 5 demonstrates that MLSAS has a good performance when the three observed sequences are closely related. However, when the sequences are far related, the optimization routine for the maximum likelihood becomes inefficient and inaccurate to find the optimal points. One way to overcome this drawback is to reduce the number of estimated parameters from 9 to 6 by fixing the value of λ/μ's just as in the pairwise alignment case where $\frac{\lambda}{\mu}$ is fixed as $\frac{s_A+s_B}{s_A+s_B+2}$ (see [TKF91]).

MLSAS will demand high-speed computers with large memory when the input sequences are longer than 100 bases to maintain a good performance. J. Thorne [TC95] expects that one might be able to extend the EM method to multiple alignment after the parallel computing and the statistical resampling techniques are introduced.

Bibliography

[AY90] L. Allison and C. Yee. Minimum message length encoding and the comparison of macromolecules. *Bulletin of Mathematical Biology*,52:431-453, 1990.

[AWY92a] L. Allison, C. Wallace, and C. Yee. Finite-state models in the alignment of macromolecules. *Journal of Molecular Evolution*, 35:77-89, 1992.

[AWY92b] L. Allison, C. Wallace, and C. Yee. Minimum message length encoding, evolutionary trees and multiple alignment. *Hawaii International Conference on System Science*, 25 V1, 663-674, 1992.

[BT86] M. Bishop and E. Thompson. Maximum likelihood alignment of DNA sequences. *Journal of Molecular Biology*, 190:159–165, 1986.

[Blo80] G. Blom. Probability and statistics, theory and its applications. Springer-Verlag, New York, Berlin, Heidelberg, 1980.

[Che94] X. Chen. An interactive system for sequence analysis. M.S. Thesis, 1994.

[DLR77] A. Dempster, N. Laird and D. Robin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[Fell68] W. Feller. An Introduction to probability theory and its applications. McGraw-Hill, New York, Volume I, 480-481, 1968.

[Fel81] J. Felsenstein. Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, 17:368-376, 1981.

[FS83] W. Fitch and T. Smith. Optimal sequence alignments. *Proceedings of the National Academy of Sciences, U.S.A.*, 80,1382-1386.

[Got82] O. Gotoh. An improved algorithm for matching biological sequences. *Journal of Molecular Biology*, 162:705-708, 1982.

[HBF92] D. Higgins, A. Bleasby and R. Fuchs. CLUSTAL V: improved software for multiple sequence alignmnet CABIOS, Vol.8, No. 2, 189-191, 1992.

[HF94] D. Heller and P. Ferguson. Motif programming manual. O'Reilly & Associates, Inc.

[JLW94] T. Jiang, E. Lawler and L. Wang. Aligning sequences via an evolutinary tree: Complexity and approximation. *Proceedings of 23rd ACM Symposium on Theory of Computation,* 1994.

[NW70] S. Needleman and C. Wunsch. A general method applicable to the search for similarities in the amino acid sequences of two proteins. *Journal of Molecular Biology*, 48:444-453, 1970.

[NM65] J. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308-313, 1965.

[PFTV88] W. Press, B. Flannery, S. Teukolsky and W. Vettering. Numerical recipes in C. Cambridge University Press, New York, 305-309, 1988.

[RW73] T. Reichert, D. Cohen and A. Wong. An application of information theory to genetic mutations and the matching of polypeptide sequences. *Journal of Theoretical Biology*, 42:245-261, 1973.

[SCL76] D. Sankoff, R. Cedergren and G. Lapalme. Frequency of insertion-deletion, transversion, and transition in the evolution of 5S ribosomal RNA. *Journal of Molecular Biology*, 7:133-149, 1976.

[SK83] D. Sankoff and J. Kruskal. Time Warps, string edits, and macromolecules: the theory and practice of sequence comparison. Addison-Wesley, Reading MA, 1983.

[TC95] J. Thorne and G. Churchill. Estimation and reliability of molecular alignments. *Biomertrics*, 51:100–113, 1995.

[TKF91] J. Thorne, H. Kishino, and J. Felsenstein. An evolutionary model for maximum likelihood alignment of DNA sequences. *Journal of Molecular Evolution*, 33:114–124, 1991.

[TKF92] J. Thorne, H. Kishino, and J. Felsenstein. Inching toward reality: An improved likelihood model of sequence evolution. *Journal of Molecular Evolution*, 34:3–16, 1992.

[Wat84] M. Waterman. General method of sequence comparison. *Bulletin of Mathematical Biology*, 46:473–500, 1984.

[You95] D. Young. Object-Oriented programming with C++ and OSF/MOTIF. Prentice Hall, New Jersey, 1995.