Rare Events Simulation In Bacterial Genetic Evolution Models

by
Yingxue Su

A dissertation submitted to the Department of Mathematics,

College of Natural Sciences and Mathematics

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Mathematics

Chair of Committee: Robert Azencott

Committee Member: Mattew Nicol

Committee Member: Andreas Mang

Committee Member: Brett Geiger

University of Houston
May 2022

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Robert Azencott, for pushing me forward mathematically, for understanding me through sicknesses and difficult times and for inspiring me with life wisdom.

I would like to thank my committee members, Professor Mattew Nicol, Professor Andreas Mang and Professor Brett Geiger, for helping me complete my research computationally and financially.

I would like to thank Professor Alan Haynes for encouragements and recognition through my 5 years Ph.D. student life.

I would like to thank my husband, my mother, my friends and my other family members for sharing my happiness and struggles with me and for supporting me in all possible ways.

I complete this dissertation in memory of my grandpa and my friend, Guiping Zhong.

ABSTRACT

Rare events often disturb the dynamics of random systems dramatically, despite their low frequencies. It is crucial to study the possible paths leading to the happening of these rare events and the probabilities of these paths. We focus on the fixation events in bacterial genetic evolution models. Fixations happen when the frequencies of certain genotypes become unusually large in the population. We introduce two numerical algorithms to estimate the probabilities of fixations. The first algorithm is based on the large deviations theory and the importance sampling method. The second algorithm is inspired by the genealogy method introduced by physicists.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Stochastic Model of Bacterial Genetic Evolution

Rare events often have an important impact even though these events have extremely low occurrence. Tsunamis, finances crises, and earthquakes are some examples. If we are able to simulate the likely trajectories that realize these dangerous rare events, we could avoid the occurrence of the rare events by avoiding these trajectories. Constructing efficient numerical algorithms is the natural approach to simulate rare events. E. Vanden-Eijnden has studied the rare events simulation in different continuous time systems in a series of papers, see [16, 25, 26, 36, 64]. The book [6] summarizes the main theoretical and numerical results of rare events simulations. In this dissertation, we study the rare events in a discrete time random system. We introduce two numerical algorithms to estimate the probabilities of the rare events in the discrete time system.

We consider the stochastic model for genetic evolution of *Escherichia coli*, which was obtained from long-term laboratory experiments [4, 5, 9, 10]. This commonly used stochastic model contains three parts: deterministic growth, Poisson distributed mutations, and random selection. On day n, the population first grows with a deterministic vector F with the starting population size N. After the growth, random mutations happen simultaneously following Poisson distributions. We

then randomly extract N cells from the bacterial population to be the starting population of day $N+1$. The papers [38, 46, 65] introduce several computational methods to estimate the parameters in this stochastic model.

We are interested in the rare events where the frequencies of certain intermediate-strength genotypes become unusually large; such events are called fixations. It is important to understand the paths that lead to these fixations under the effects of two random steps, namely, random mutations and random selections. In the recent work [3], the large deviations theory results of the stochastic model are obtained. In this thesis, we derive a numerical method for estimating the probability of fixations, based on these large deviations theory results. We first use these results to search for the optimal path that realizes the fixation. Next, we estimate the fixation probability by forcing random trajectories to follow the optimal path. We also derive a second numerical method, based on the genealogy method of [55], which does not depend on the large deviations theory results. This genealogy method depends on a re-sampling trick to force the rare events to occur more often in the system.

1.2 Review of Large Deviations Theory

The large deviations theory studies the exponential decay of sequences of probability distributions asymptotically. The function that measures the rate of this decay is called the rate function. Since large deviations theory studies the tail behaviors of probability distributions, it becomes the common and useful tool for studying rare events in random systems.

The earliest work of large deviations theory was done by the Swedish mathematician Crámer. He studied the large deviations theory in order to solve actuary problems in 1938, see [12]. The result he obtained for i.i.d random variables is known as the Crámer theorem. Another important result for i.i.d random variables is Sanov's theorem which was obtained in 1957, see [59]. Sanov's theorem shows that the rate function of i.i.d Markov process is the Kullback-Leibler divergence between

the true probability distribution and the estimated probability distribution. From the late 1970s to the early 1980s, Donsker and Varadhan studied the large deviations theory of Markov processes much more thoroughly in a series of papers, see [17, 18, 19, 20]. Around the same time, Gärtner and Ellis extended Donsker's and Varadhan's results to more general settings, see [27, 28, 37]. One of their famous results is the Gärtner-Ellis theorem. Later on, Csiszár introduced the method of types technique to simplify some proofs of the classical large deviations theory results, see [13]. The books [2, 14] present the classic and important results of large deviations theory.

The large deviations theory results of the stochastic model of bacteria populations were obtained in paper [3]. In this thesis, we introduce an algorithm based on these large deviations theory results to simulate rare events and estimate the probabilities of rare events.

1.3 Review of Importance Sampling

Importance sampling is a Monte Carlo simulation technique. It uses a manipulated probability distribution instead of the true probability distribution of the system to simulate the events of our interest. Since we change the probability distribution of the system during simulations, we need to keep tracking the corrective ratio between the true probability distribution and the manipulated probability distribution. By using a "good" manipulated probability distribution, the rare events become the common events of the random system. Thus, importance sampling is one of the powerful tools to study and simulate rare events. In fact, the first use of the importance sampling was to study the rare events of nuclear particles penetrating shields in 1950, see [40]. In recent decades, people have developed multiple types of importance sampling methods such as, adaptive importance sampling [50, 51, 57, 22, 24], annealed importance sampling [45, 56], sequential importance sampling [15, 29], and multiple importance sampling [30]. [7, 63] review these developments of the importance sampling. We summarize the important developments here.

Let $p(x)$ be the true probability distribution of a random variable X. Let $q(x)$ be the changed

probability distribution. In general, not all the properties of the true distribution $p(x)$ are known. In these circumstances, it might be difficult to choose the changed distribution $q(x)$ that is practical to use and guarantees good accuracy. Several strategies were studied to solve this issue. We review three types of importance sampling methods here.

If some theoretical properties of $p(x)$ are known, one might be able to use these properties to determine the family of $q(x)$. We then select the optimal distribution $q(x)$ among this family based on certain measurements. This type of method is the adaptive importance sampling method. When a multivariate normal distribution is a good estimator of $p(x)$ asymptotically, one might select the corresponding multivariate normal distribution as $q(x)$, see [33]. The work [32] shows that a multivariate student density distribution is a better choice for $q(x)$. Later on, the work in [41, 49] reduces the restrictions on the asymptotic approximation of $p(x)$ and extended the candidates of $q(x)$.

Sequential importance sampling method is often used for sampling high dimensional random variables. Suppose the random variable is of dimension n, denote $x_{[1:t]} = (x_1, ..., x_t)$ for any $t \in [1, n]$. One can rewrite the true probability distribution $p(x)$ and the changed probability distribution $q(x)$ as

$$p(x) = p(x_1) \prod_{t=2}^{n} p(x_t | x_{[1:t-1]}),$$

and

$$q(x) = q(x_1) \prod_{t=2}^{n} q(x_t | x_{[1:t-1]}).$$

We select the proper $q(x_t|x_{[1:t-1]})$ for $p(x_t|x_{[1:t-1]})$ for $t = 2, ..., n$. Sequential importance sampling is often used for models with different states, see Chapters 3 and 4 in [43]. [44] applied this technique to study wireless communication. [54] proposed several applications in speech recognition by recognizing the chain like sequential distributions in the system. [53] analyzed the economical time series by applying this sequential importance sampling technique. [60] studied the prediction of the 3D structure of the protein by using this technique.

The work [48] introduces the anneal importance sampling method to sample the random variable sequentially, even if the random variable does not have a chain like structure. One first constructs a sequence of distributions $p_0(x), ..., p_d(x) = p(x)$, where p_0 is selected to be diffuse and

$$p_i(x) = p_0(x)^{1-b_i} p(x)^{b_i},$$

for $i = 1, ..., d$, $0 = b_0 \leq b_1 \leq ... \leq b_d = 1$. Next, one draws a sample $x(t)$ from the distribution $g_t(x'|x(t-1))$ for $t = 1, ..., d$ sequentially, where $g_t(x'|x(t-1))$ is a transition kernel satisfying

$$\int p_t(x) g_t(x'|x) dx = p_t(x').$$

[45] applied this anneal importance sampling method to study dileucine peptide. [35] studied the particle filtering problem by using this technique. People also proposed a sampling method of combining the sequential importance sampling method and the transition kernel idea, see [31, 39, 47]. This type of algorithm allows one to perform the dependent Markov chain sampling around the proper transition kernel. We will use this type of importance sampling structure to simulate rare paths in the bacterial genetic evolution stochastic model.

1.4 Exponential Shift Family and Genealogy Algorithm

If we have obtained the large deviations theory results of the true distribution $p(x)$, one of the commonly used families for the manipulated probability distributions is the exponential shift family, see [11, 21, 52, 61]. Moreover, the exponential shift distribution with the large deviation minimizer is the optimal manipulated distribution among this family. This is because the variance of the estimator is minimized asymptotically by this exponential shift distribution, see [23, 58]. The exponential shift family is constructed as follows. Let $\hat{p}(x)$ be the Laplace transform of the true distribution $p(x)$. The exponential shift family of $p(x)$ is defined as $q(x) = \frac{e^{\langle t,x \rangle} p(x)}{\hat{p}(t)}$ for $t \in \mathbb{R}$. The parameter t of the optimal manipulated distribution comes from the Cramer transform of $p(x)$.

The large deviations results in the path space of the bacterial genetic evolution model are obtained in paper [3]. The large deviations theory approach to simulate rare fixations is organized as follows in this dissertation. We first search for the optimal paths to realize the fixations, see Chapter 5. Next, we force the random trajectories to follow the optimal paths by applying the exponential shift technique, see Chapter 7. The methodology of estimating the probabilities of random trajectories is explained in Chapters 8 and 9. In Chapter 10, we estimate the probabilities of fixations by estimating the probabilities of random trajectories in thin tubes centered at the optimal paths.

Physicists propose the genealogy algorithm to study rare events in the overly complicated stochastic models. This genealogy algorithm does not depend on the large deviations theory results of the system. The computation of searching for the optimal paths to realize rare events is not needed. The genealogy algorithm simulates the likely paths that realize rare events by applying certain re-sampling tricks. The Giardina–Kurchan– Lecomte–Tailleur (GKLT) algorithm is one of the commonly used genealogy algorithms. This algorithm was originally designed to simulate the large deviation rate functions for infinite time limit, see [34, 62, 42]. Ragone, Wouters, and Bouchet apply this GKLT algorithm to a finite continuous time setting to study extreme weather, see [55]. In comparison, time is discrete in the stochastic model of our interest. Thus, we introduce the modified genealogy algorithm to simulate rare events in the bacterial genetic evolution model in Chapter 11.

Chapter 2

Stochastic Model for Genetic Evolution of Bacteria Population

There are g genotypes in the stochastic model of genetic evolution of bacteria populations. $H = [H(1), ..., H(g)] \in \mathbb{R}^g$ with coordinates $0 \leq H(j) \leq 1$, and $\sum_{j=1,...,g} H(j) = 1$ is a genetic histogram for this stochastic model. The set $\mathcal{H} \subset \mathbb{R}^g$ of all potential histograms is a compact convex set. We refer to the cells of genotype j as j-cells; j-cells have a rate of exponential growth $f_j > 0$. Define the growth factor for j-cells as $F_j = \exp(Df_j) > 1$, where D is the fixed duration of the deterministic growth. Denote $F = [F_1, F_2, ..., F_g]$. We order the F_j in an ascending order so that if $i < j$, then $F_i < F_j$. At the beginning of day n, the bacteria population starts with $pop_n = N$ and a genetic histogram H_n. In fact, we have $H_n \in \mathcal{H}_N$, where $\mathcal{H}_N = \{H \in \mathcal{H} | NH(j) \in \mathbb{Z} \text{ for } j = 1, ..., g\}$. The populations are allowed to grow deterministically till they reach a very large size $N\langle H_n, F\rangle$. After the growth, these cells of g genotypes are allowed to mutate simultaneously. At the end of day n, we select a random sample of size N as the starting population of day $n+1$. The precise model is stated as follows:

1. Phase 1: The j-cells colony grows deterministically and its population size grows to $[NH_n(j)F_j]$

for $j = 1, ..., g$.

2. Phase 2: Mutations occur simultaneously with a fixed small mutation rate m.

3. Phase 3: Randomly sample N cells from the population to be the starting population of day $n + 1$.

2.1 Deterministic Growth

Let N_{ter} denote the population size after the deterministic growth. The population size of j-cells is $NH_n(j)$ at the beginning of day n, then it reaches $siz_n(j) = [NH_n(j)F_j]$ after the deterministc growth, where $[z]$ gives the smallest integer greater or equal to z. Therefore, $N_{ter} = \sum_{j=1}^{g} siz_n(j) = [N\langle F, H_n\rangle]$. The frequency of j-cells becomes to $[NF_jH_n(j)]/[N\langle F, H_n\rangle]$.

Definition 2.1.1. Define the deterministic growth function $\Phi : \mathcal{H} \to \mathcal{H}$ by

$$\Phi_j(H) = F_jH_j/\langle F, H\rangle$$

for $j = 1, .., g$.

We use $\Phi_j(H_n)$ to estimate the frequency of j-cells after the deterministic growth. The distance between this estimation and the true frequency is of the order of $1/N$ for large N, see [3].

2.2 Random Mutations

Assume mutations occur simultaneously at the end of each growth period. Let a $g \times g$ matrix $M = mQ$ with entry $M_{i,j} = mQ_{i,j}$ be the mean mutation rates matrix, where m is a very small, fixed mutation rate and Q is the fixed transfer matrix with $Q_{j,k} \geq 0$ and $Q_{j,j} = 0$. The mutation rate m takes a value between 10^{-8} to 10^{-6}.

Let $R_n(j,k)$ denote the number of cells mutating from type j to type k on day n. $R_n(j,k)$ follows the Poisson distribution with mean $siz_n(j)M_{j,k}$. We have

$$\mathbb{P}(R_n(j,k) = R_{j,k}|H_n = H) = \exp(-siz_n(j)M_{j,k})(siz_n(j)M_{j,k})^{R_{j,k}}/R_{j,k}!, \quad j \neq k$$

$$\mathbb{E}(R_n(j,k)|H_n = H) = siz_n(j)M_{j,k}, \quad j \neq k.$$

Definition 2.2.1. A matrix A is N-rational if NA has only non-negative integer entries.

Definition 2.2.2. For each histogram H and each genotype j, define the constraint sets $K(j,H)$ to be all the $g \times g$ matrices r with non-negative entries such that

$$\begin{cases} \sum_{k=1}^{g} r_{j,k} < F_j H(j) & \text{when } H(j) > 0, \\ r_{j,k} = 0 & \text{when } H(j) = 0, \\ r_{j,k} = 0 & \text{when } Q_{j,k} = 0. \end{cases} \tag{2.1}$$

Definition 2.2.3. Let $\mathcal{Z}(N)$ be the set of all $g \times g$ N-rational matrices. Define $K(H) = \cap_{j=1}^{g} K(j,H)$ and $K_N(H) = K(H) \cap \mathcal{Z}(N)$.

Therefore, $R_n/N \in K_N(H)$. The population histogram J_n after the mutation is

$$J_n(j) = \frac{1}{[N\langle F, H_n\rangle]}([NF_jH_n(j)] - \sum_k R_n(j,k) + \sum_k R_n(k,j)).$$

Definition 2.2.4. For given $H_n = H \in \mathcal{H}_N$ and $r = R_n/N \in K_N(H)$, we define function Ψ by

$$\Psi_j(H,r) = \frac{1}{\langle F, H\rangle}(F_jH(j) - \sum_{k\neq j} r_{j,k} + \sum_k r_{k,j}),$$

for $j = 1, ..., g$.

The function Ψ is an estimator for the histogram J_n. The distance between this estimator and

the true frequency J_n is of the order $1/N$, see [3].

2.3 Random Selection

After the deterministic growth and random mutations, we extracts N cells from the population for cycle $n+1$. Let $\mu_{N,J}(V)$ be the multinomial distribution defined as

$$\mu_{N,J}(V) = N! \prod_j \frac{(J(j))^{V(j)}}{V(j)!}. \tag{2.2}$$

Definition 2.3.1. For any $H \in \mathcal{H}$, $spt(H)$ is the support of H and $b(H) > 0$ is its essential minimum defined as

$$spt(H) = \{j | H(j) > 0\} \text{ and } b(H) = \min_{j \in spt(H)} H(j).$$

Suppose $H_{n+1} = G \in \mathcal{H}_N$, then we have

$$\mathbb{P}(H_{n+1} = G | H_n, R_n) = \mathbb{P}(H_{n+1} = G | J_n) = \begin{cases} \mu_{N,J_n}(NG) & spt(G) \subset spt(J_n), \\ 0 & \text{otherwise .} \end{cases}$$

Definition 2.3.2. $Q(H,G) = \mathbb{P}(H_{n+1} = G | H_n = H)$ is the transition kernel defined as

$$Q(H,G) = \sum_{r \in K_N(H)} \mathbb{P}(H_{n+1} = G | R_n = Nr, H_n = H) \mathbb{P}(R_n = Nr | H_n = H). \tag{2.3}$$

Chapter 3

Sampling Method of Multinomial Random Vector with Large Population

Since the population size N of the bacteria evolution stochastic model is large, we will need to sample multinomial random vectors with this large population size N during the numerical simulation of the model. The current algorithm used for generating multinomial random vector in Matlab is not feasible when the population size N is large due to the storage issue and the efficiency issue. In this section, we implement an asymptotic sampling method for the multinomial random vector by using the normal approximation of the multinomial distribution.

3.1 Normal Approximation of Multinomial Distributions

Let X be a random vector in $\mathbb{R}^k$ following the multinomial distribution $\mu_{N,p}$ defined as formula (2.2). Notice that $p \in \mathbb{R}^k$ and $\sum_{i=1}^{k} p_i = 1$. The marginal distribution of X_j follows the binomial

distribution $Binomial(n, p_j)$. The mean and covariance matrix of X are

$$\mathbb{E}(X) = (np_1, ..., np_k)^T,$$

and

$$V(X) = \begin{pmatrix} np_1(1-p_1) & -np_1p_2 & ... & -np_1p_k \\ -np_1p_2 & np_2(1-p_2) & ... & -np_2p_k \\ \vdots & \vdots & \ddots & \vdots \\ -np_1p_k & -np_2p_k & ... & np_k(1-p_k) \end{pmatrix}.$$

Theorem 3.1.1. *$X \in \mathbb{R}^k$ follows the multinomial distribution $\mu_{N,p}$. As $N \to \infty$, the distribution of $\sqrt{N}(\frac{X}{N} - p)$ weakly converges to the multivariate Normal distribution $N(0, \Sigma)$, where*

$$\Sigma = \begin{pmatrix} p_1(1-p_1) & -p_1p_2 & ... & -p_1p_k \\ -p_1p_2 & p_2(1-p_2) & ... & -p_2p_k \\ \vdots & \vdots & \ddots & \vdots \\ -p_1p_k & -p_2p_k & ... & p_k(1-p_k) \end{pmatrix},$$

see [1].

3.2 Asymptotic Sampling Method

By using Theorem 3.1.1, we implement the following asymptotic sampling method for multinomial random vectors.

Algorithm 3.1: Asymptotic Sampling Method

Step 1. Sample a random vector Y following the multivariate normal distribution $N(0, I)$, where I is the $k \times k$ identity matrix.

Step 2. Compute $Z = \Sigma^{\frac{1}{2}} Y$ and $Z \sim N(0, \Sigma)$.

Table 3.1: Computing Time of the Multinomial Random Vector Sampling

N	Matlab *mnrnd*	Asymptotic sampling method
10^6	0.03(s)	0.005(s)
10^7	0.32(s)	0.005(s)
10^8	3.20(s)	0.005(s)
10^9	30.12(s)	0.005(s)

Computing time of the Matlab *mnrnd* function and the asymptotic sampling method on one node of the Opuntia cluster

Step 3. Compute $U = \sqrt{N}Z + Np$. Let $X_i = [U_i]$ for $i = 1, ..., k-1$ and $X_k = N - \sum_{i=1}^{k-1} X_i$. The vector X is the sampled multinomial random vector.

3.3 Example of Multinomial Distribution Sampling

Let $p = [0.5, 0.2, 0.3]$ and $k = 3$, we compare the computing time of sampling one multinomial random vector by using the Matlab function *mnrnd* and the asymptotic sampling method for $N = 10^6, 10^7, 10^8, 10^9$ in Table 3.1.

We see that as N increases by a factor of 10, the computing time of Matlab function *mnrnd* also increases by a factor of 10 approximately. When we sample a multinomial random vector with population size $N = 10^{10}$ by using the *mnrnd* function, we will also face the storage issue. In comparison, the computing time of the asymptotic sampling method is not increasing as N increases. In the following chapters, we will be using this asymptotic sampling method to sample mutinomial random vectors when the population size $N \geq 10^7$.

Chapter 4

Large Deviations in Path Space

The large deviations theory studies the exponential decay of probability distributions of random systems. The rate function is used to describe this exponential decay, which contains important information of the rare events occurring in the system. Thus, studying the large deviations theory of the random system is an important step to analyze rare events. In this chapter, we recall several important large deviations theory results in the path space of the bacterial genetic evolution model, which were studied in paper [3]. The proofs of the theorems in this chapter can be found in [3].

4.1 Large Deviations Asymptotics for Mutations

Definition 4.1.1. For all $H \in \mathcal{H}$ and $r \in K(H)$, define the $g \times g$ matrix $L(r, H)$ of Poissonian rate functions by

$$L_{j,k}(r,H) = \begin{cases} M_{j,k}F_jH(j) + r_{j,k}\log(\frac{r_{j,k}}{eM_{j,k}F_jH(j)}) & H(j)M(j,k) > 0, \\ 0 & H(j)M(j,k) = 0. \end{cases} \tag{4.1}$$

Thus, the large deviations rate function for mutations $mut(r, H)$ is

$$mut(r, H) = \sum_{j,k} L_{j,k}(r, H) \tag{4.2}$$

$$= \sum_{(j,k)|M_{j,k}H(j)>0} M_{j,k}F_jH(j) + r_{j,k}\log r_{j,k} - r_{j,k}\log(eM_{j,k}F_jH(j)) \tag{4.3}$$

with the convention $0 \times \log(0) = 0$. The $mut(r, H)$ is a finite, non-negative, and continuous strictly convex function of $r \in K(H)$, since each $L_{j,k}$ is strictly convex in $r_{j,k}$. $mut(r, H) = 0$ if and only if $r_{j,k} = M_{j,k}F_jH(j)$ for all $j \neq k$.

Definition 4.1.2. Define a fixed finite parameter set $\mathcal{P}$, namely:

1. the number $g \geq 2$ of genotypes.
2. the multiplicative daily growth factor F_j for genotype j with $j = 1, ..., g$.
3. the $g \times g$ transfer matrix Q.

Definition 4.1.3. For any fixed $0 < a < 1$, define the compact set of histogram $\mathcal{H}(a) \subset \mathcal{H}$ by

$$\mathcal{H}(a) = \{H \in \mathcal{H} | \min_{j=1,...,g} H(j) > a\}.$$

The large deviations result of the mutation stage is stated in the following Theorem 4.1.4.

Theorem 4.1.4. *Let R_n be the random mutation matrix of day n. Let $mut(r, H)$ be the mutations rate function defined by formula (4.2). Then for fixed $a > 0$ and the parameters $\mathcal{P}$, there is a constant $N_0 = N_0(a, \mathcal{P})$ such that for $N > N_0$, the large deviation formula*

$$\frac{1}{N}\log \mathbb{P}(R_n/N = r|H_n = H) = -mut(r, H) + o(N), \quad with \ \ |o(N)| \leq 4g^2 \log(N)/N \tag{4.4}$$

holds uniformly for $H \in \mathcal{H}(a) \cap \mathcal{H}_N$ and $r \in K_N(H)$.

4.2 Large Deviations Rate Function for Multinomial Sampling

Definition 4.2.1. The classical $Kullback - Leibler\ divergence$ between two histograms G and J is defined as

$$KL(G,J) = \begin{cases} \sum_{j \in spt(G)} G(j) \log \frac{G(j)}{J(j)} & \text{when } \ spt(G) \subset spt(J), \\ \infty & \text{otherwise.} \end{cases} \tag{4.5}$$

Recall that $KL(G,J) \geq 0$ for all G and J, and $KL(G,J) = 0$ if only if $G = J$. The large deviation result of the random selection stage is stated in the following Theorem 4.2.2.

Theorem 4.2.2. *If any histograms J and G with $spt(G) \subset spt(J)$ and G is N-rational, then the multinomial distribution $\mu_{N,J}$ defined by formula (2.2) verifies*

$$\frac{1}{N} \log \mu_{N,J}(NG) = -KL(G,J) + o(N) \quad with \ \ |o(N)| \leq 2(g+1) \log N/N. \tag{4.6}$$

4.3 Large Deviations for the One-step Transition Kernel

Definition 4.3.1. For any $H, G \in \mathcal{H}$ and $r \in K(H)$, define the composite transition rate $\tau(H,r,G)$ as

$$\tau(H,r,G) = mut(r,H) + KL(G, \Psi(H,r)). \tag{4.7}$$

Definition 4.3.2. Define the one-step cost function $C(H,G)$ by

$$C(H,G) = \min_{r \in K(H)} \tau(H,r,G) = \min_{r \in K(H)} [mut(r,H) + KL(G, \Phi(H,r))].$$

The large deviation result of the one-step transition kernel is stated in the following Theorem 4.3.3.

Theorem 4.3.3. *Fix any $0 < a < 1 < d$ and the parameters $\mathcal{P}$. One-step large deviations for Markov chain H_n are controlled as follows by two constants $c = c(d, a, \mathcal{P})$ and $N_0 = N_0(d, a, \mathcal{P})$. Consider any $H, G \in \mathcal{H}(a)$ with one-step transition cost $C(H, G) \leq d$. Then the transition kernel $Q(H, G)$ has a uniform large deviations approximation, valid for all $N > N_0$ and $H, G \in \mathcal{H}(a)$ as above,*

$$\frac{1}{N} \log Q(H, G) = -C(H, G) + o(N) \tag{4.8}$$

with $|o(N)| \leq c/\sqrt{N}$.

The explicit computation of the one-step cost function is stated in the following Theorem 4.3.4.

Theorem 4.3.4. *Fix $\mathcal{P}$ and any $0 < a < 1$. Let $\Gamma(a)$ be the set of interior histograms J with $b(J) > a$. There is a constant $c = c(a, \mathcal{P}) > 0$ such that for all $H, G \in \Gamma(a)$, and $0 \leq m \leq c$, the transition cost $C(H, G)$ is a finite C^∞ function and transition cost $C(H, G)$ has an explicit first order expansion in m, given by*

$$C(H, G) = KL(G, \Phi) + m \sum_{j,k} F_j H(j) Q_{j,k} (1 - U_k/U_j) + O(m^2) \tag{4.9}$$

where $U_j = \exp \frac{G_j}{F_j H(j)}$,

$$KL(G, \Phi) = \sum_j G(j) \log(G(j)/\Phi(j)) > 0 \quad \textit{and} \quad \Phi(j) = F_j H(j)/\langle F, H \rangle.$$

The first order expansion of $C(H, G)$ is obtained when

$$r_{j,k} \approx m F_j H(j) Q_{j,k} U_k / U_j, \tag{4.10}$$

for $j, k = 1, ..., g$ and $j \neq k$.

4.4 Large Deviations in Path Space

Definition 4.4.1. Define $\Omega_T = \mathcal{H}^T$ the path space of all histogram trajectories $\mathbf{H} = [H_0, ..., H_{T-1}]$ with all $H_n \in \mathcal{H}$ where T is the fixed time horizon. A trajectory $\mathbf{H} \in \Omega_T$ is N-rational if all the H_n are N-rational histograms. Define the essential minimum $b(\mathbf{H})$ of $\mathbf{H}$ as $b(\mathbf{H}) = \min_{n=0,...,T-1} b(H_n)$.

Definition 4.4.2. The large deviations rate function $\lambda : \Omega_T \to [0, +\infty]$ is defined for any $\mathbf{H} \in \Omega_T$ by

$$\lambda(\mathbf{H}) = \sum_{n=0,...,T-2} C(H_n, H_{n+1}).$$

Define the large deviations rate functional $\Lambda(F) \in [0, +\infty]$ for all subsets F of the path space Ω_T by

$$\Lambda(F) = \inf_{\mathbf{H} \in F} \lambda(\mathbf{H}).$$

The large deviations theory result of single trajectories is stated in the following Theorem 4.4.3.

Theorem 4.4.3. *For any path length $T \geq 2$, denote $\mathbf{H} = [H_0, ..., H_{T-1}]$ as the random trajectory of population histograms. Fix $\mathcal{P}$ and any positive constants $d > 0$ and $a > 0$. The $(a, d, \mathcal{P})$ determine positive constant c and N_0 such that the following properties hold. For any N-rational path $\mathbf{h} \in \Omega_T$ such that $\lambda(\mathbf{h}) \leq d$ and $b(\mathbf{h}) \geq a$, one has, for all $N > N_0$,*

$$\frac{1}{N} \log \mathbb{P}(\mathbf{H} = \mathbf{h} | H_1) = -\lambda(\mathbf{h}) + o(N) \quad \textit{with} \quad |o(N)| \leq Tc/\sqrt{N}. \tag{4.11}$$

4.5 Explicit Computation of Geodesics

Definition 4.5.1. A path $\mathbf{h}^* = [h_0^*, ..., h_{T-1}^*]$ from H to G is a geodesic from H to G if it minimizes the large deviation rate function $\lambda(\mathbf{h})$ over all $\mathbf{h} \in \Omega_T$ such that $h_0 = H$ and $h_T = G$.

When all $h_n^* \in \mathcal{H}^o$ for $n = 0, ..., T-1$, we call $\mathbf{h}^*$ an interior geodesic. The following Theorem 4.5.2 gives the explicit formula of generating a reverse geodesic recurrently.

Theorem 4.5.2. *Let* $\mathbf{h}^*$ *be any interior geodesic in* Ω_T *with* $T > 1$. *Let* $a = b(\mathbf{h}^*)$. *There is a constant* $m_0 = m_0(a, PAR)$ *such that for* $m < m_0$, *and any* $n \in 0, ..., T-3$, *then* $x := h_n^*$ *is fully determined by* $y := h_{n+1}^*$ *and* $z := h_{n+2}^*$. *Indeed,* $x = (m, y, z)$, *where is a* C^∞ *function of* (m, y, z) *for* $m < m_0$ *and* $y, z \in \mathcal{H}^o$. *Hence, for* $m < m_0$, $\mathbf{h}^*$ *is determined by its last two points* h_T^* *and* h_{T-1}^* *thanks to the reverse recurrence relation*

$$h_n^* = \chi(m, h_{n+1}^*, h_{n+2}^*) \quad for \quad 0 \leq n \leq T-3. \tag{4.12}$$

We call the histogram h_{T-1}^* *the penultimate point of the geodesic* $\mathbf{h}^*$. *Denote* $x_s = \hat{x}_s(1 + mw_s)$ *with* $s = 1, ..., g$ *and remainder of order* m^2 *the* $1st$ *order Taylor expansion of* $x = \chi(m, y, z)$ *in* m *for* $m < m_0$. *The interior histogram* $\hat{x}$ *and the vector* w *depend only on* y, z *and are given below by the explicit formulas*

$$\hat{x}_s = \frac{X_s}{\sum_t X_t}, \tag{4.13}$$

$$X_s = \frac{y_s}{F_s} \exp(\frac{F_s}{\langle F, y \rangle} - \frac{z_s}{y_s}), \tag{4.14}$$

$$w = u + v - \langle a, u + v \rangle, \tag{4.15}$$

$$u_s = \sum_k (Q_{s,k} e_{s,k} - \frac{F_k X_k}{F_s X_s} Q_{k,s} e_{k,s}), \tag{4.16}$$

$$e_{s,k} = \exp(-\frac{y_s}{F_s a_s} + \frac{y_k}{F_k a_k}), \tag{4.17}$$

$$v_s = F_s \sum_k Q_{s,k} - (F_s + \frac{z_s}{y_s}) \sum_k f_{s,k} Q_{s,k} - \frac{z_s}{F_s y_s^2} \sum_k F_k y_k Q_{k,s} f_{k,s}, \tag{4.18}$$

$$f_{s,k} = \exp(-\frac{z_s}{F_s y_s} + \frac{z_k}{F_k y_k}). \tag{4.19}$$

4.6 Mean Trajectories

The following Theorem 4.6.1 gives the explicit formula for computing the mean trajectories. The mean trajectories have zero cost.

Theorem 4.6.1. $\mathbf{H} = [H_0, ..., H_{T-1}]$ *is the random trajectory of population histograms. Given* $H_t = H$*, then* $\mathbb{E}(H_{t+1}|H_t = H) = \zeta(H)$*, where* $\zeta : \mathcal{H} \to \mathcal{H}$ *is defined as*

$$\zeta_j(H) = \frac{1}{\langle F, H\rangle}(F_j H(j) - m\sum_k Q_{j,k}F_j H(j) + m\sum_k Q_{k,j}F_k H(k)), \tag{4.20}$$

for $j = 1, ..., g$.

4.7 Sets of Thin Tubes of Realizing Rare Events

Definition 4.7.1. For $\eta > 0$ and $E^* \subset \Omega_T$, define the η-neighborhood $U_\eta(E^*)$ of E^* as the union of all open balls of radius η and center in E^*.

Definition 4.7.2. For any $\Gamma \in \Omega_T$, define the open neighborhood $V_N(\Gamma)$ as the union of all balls $V_N(\mathbf{H})$ with radius $\frac{2}{3N}$ and any arbitrary center $\mathbf{H} \in \Gamma$, where $V_N(\mathbf{H})$ is defined as the set

$$V_N(\mathbf{H}) = \{\mathbf{H}' \in \Omega_T \mid \max_{i=0,...,T-1} ||H_i' - H_i|| \leq \frac{2}{3N}\}.$$

Theorem 4.7.3. *Fix the path length* T *and an initial histogram* H*. Let* P_H *be the probability distribution of random histogram paths starting at* H*. Let* $E \subset \Omega_T$ *be any closed set of interior paths starting at* H *satisfying* $0 < \Lambda(E) < \infty$*. Let* E^* *be the set of all paths* $\mathbf{h}$ *minimizing the rate function* $\lambda(\mathbf{h})$ *over all* $\mathbf{h} \in E$*. Then* E^* *is a closed subset of* E*. For any fixed* $\eta > 0$*, the* η*-neighborhood* $U = U_\eta(E^*)$ *verifies*

$$\lim_{N\to\infty} P_H(\mathbf{H} \in U|\mathbf{H} \in V_N(E)) = 1.$$

From this theorem, we know that the probability of fixations is approximately equal to the probability of trajectories realizing the fixations through the thin tubes centered at the cost-minimizing paths for large population size N.

Chapter 5

Geodesic Computation

Let $A(G)$ be the set of trajectories starting with histogram H and ending with histogram G. We are interested in computing $\mathbb{P}(A(G)|H)$ which is the probability of the event that trajectories starting with histogram H end with histogram G. By the large deviation theory results in paper [3], we know that

$$\lim_{N\to\infty} -\frac{1}{N}\log \mathbb{P}(A(G)|H_0) = \Lambda(A(G)) = \inf_{\mathbf{H}\in A(G)} \lambda(\mathbf{H}).$$

Notice that $\inf_{\mathbf{H}\in A(G)} \lambda(\mathbf{H})$ brings up the definition of geodesics from H to G. We can roughly estimate the probability $\mathbb{P}(A(G)|H)$ by $\exp(-N \cdot \lambda(\mathbf{h}^*))$, where $\mathbf{h}^*$ is the geodesic from H to G. Thus, searching for geodesics from H to G is a very important aspect of studying the small probability $\mathbb{P}(A(G)|H)$. In this chapter, we will study several algorithms to search for geodesics from H to G. This chapter will be submitted as a paper with Brett Geiger, Andreas Mang, Ilya Timofeyev, and Robert Azencott.

5.1 Brute Force Simulation

Given a fixed starting histogram H and a fixed target histograms G, paper [3] states the brute force algorithm to search for the geodesic from H to G in detail. We summarize the basic steps of

this brute force simulation in the following Algorithm 5.1.

Algorithm 5.1: Brute Force Simulation

Step 1. H is the starting histogram. G is the target histogram. $Mesh$ is the mesh size. Let $PEN = \mathcal{H} \backslash G$. We use the mesh size $Mesh$ to discretize set PEN. Let $DPEN$ be the discretized penultimate points set. Denote the cardinal number of set $DEPN$ as $card(DPEN)$. Suppose $card(DPEN) = n$.

Step 2. For every histogram $P \in DPEN$, construct a reverse geodesic $RG(G, P)$ lying within the interior of $\mathcal{H}$ using formula (4.12). Let RG^1,...,RG^n be n such reverse geodesics and denote $RG^i = [RG^i_1 = G, RG^i_2, ..., RG^i_{T^i}]$, where $RG^i_2 \in DPEN$ and T^i is the length of reverse geodesic RG^i.

Step 3. Compute the mean trajectory $MT = [MT_0 = H, MT_1, MT_2, ..., MT_L]$ starting from H and lying within $\mathcal{H}^o$.

Step 4. For every reverse geodesic $RG = [RG_1 = G, ..., RG_t]$, we construct a broken geodesic of the form $BG = [MT_0 = H, ..., MT_I, RG_J, ..., RG_1 = G]$. We select the index I, J by minimizing

$$I, J = \arg \min_{i=0,...,L} \min_{j=1,...,t} C(MT_i, RG_j) + \sum_{k=1}^{j-1} C(RG_i, RG_{i+1}),$$

where $C(MT_i, RG_j)$ is the one step cost from a histogram MT_i in the mean trajectory to a histogram RG_j in the reverse geodesic, and $\sum_{k=1}^{j-1} C(RG_{i+1}, RG_i)$ is the rate function of the trajectory $[RG_j, RG_{j-1}, RG_{j-2}, ..., RG_1]$. Thus, we will get n broken geodesics $BG^1, ..., BG^n$ constructed from $RG^1, ..., RG^n$.

Step 5. The geodesic from H to G is

$$Geo = \arg \min_{\mathbf{H} \in \{BG^1, ..., BG^n\}} \lambda(\mathbf{H}).$$

Thus, the geodesic from H to G contains three pieces theoretically, truncated mean trajectory, one step jump from mean trajectory to a reverse geodesic and the truncated reverse geodesic.

However, from our numerical results, if $H, G \in \mathcal{H}$ satisfies $C(H,G) > 0.1$, $\min(H) > 0.01$ and $\min(G) > 0.01$, then the geodesic from H to G only contains two pieces, one step jump from H to a reverse geodesic and the truncated reverse geodesic. This is because the reverse geodesic that is used to construct the geodesic from H to G always shoots almost right at H. We will use this conclusion to construct the modified geodesic shooting method in Section 5.4.

If we want to compute the geodesic from H to G accurately by using this brute force simulation, we need to use a relatively small $Mesh$ ($\leq O(10^{-2})$) to discretize PEN, especially for the case when H or G is near to the boundary of $\mathcal{H}$. If we use $Mesh = 0.01$ for geodesic computing with $g = 4$, we would need to construct 10^6 reverse geodesics to find one geodesic for one pair of H and G. This computation takes about 40 seconds by using one core on our Opuntia cluster. However, for geodesic computing with $g = 6$, if we use $Mesh = 0.01$, we would need to construct 10^{10} reverse geodesics to obtain one geodesic for one pair of H and G. If we perform this computation on one core, the computing time would increase to around 4×10^5 seconds (more than 4 days), which is very computationally heavy.

Brute force simulation for searching for the geodesic from H to G works for cases with the number of genotypes $g \leq 5$. For geodesic computing with $g > 5$, we developed these following techniques to perform this computation, brute force simulation with parallel computing, brute force simulation with the quantile technique on multi-cores, and the modified geodesic shooting algorithm. Brute force simulation with parallel computing technique keeps the computing time for geodesic searching within an hour for cases when $g = 5, 6$. However, this parallel computing technique is not powerful enough to perform the geodesic searching computation within a reasonable time when $g = 7, 8$. Thus, we implement the quantile technique on multi-cores for these cases. For geodesic searching when $g \geq 9$, the modified geodesic shooting algorithm is the best choice. In the following sections of this chapter, we explain these algorithms in detail.

5.2 Parallel Computing Technique

The most natural approach to save computation time is to parallelize the computation task on multi-cores. The idea is to split the computation task on k cores to save the computing time by a factor of $1/k$. We explain the brute force simulation with parallel computing technique in the following algorithm.

Algorithm 5.2: Brute Force Simulation with Parallel Computing

Step 1. H is the starting histogram. G is the target histogram. $Mesh$ is the mesh size. PEN is the set of penultimate histograms. Discretize PEN with $Mesh$. $DPEN$ is the discretized set of penultimate histograms. k is the number of cores.

Step 2. Split the set $DPEN$ to k subsets, $DPEN_1, ..., DPEN_k$, where $\cup_{i=1}^{k} DPEN_i = DPEN$.

Step 3. We perform the brute force simulation with the set of penultimate histograms $DPEN_i$ on the $i-th$ core for $i = 1, ..., k$ following Algorithm 5.1. Let Geo^i be the geodesic found on the $i-th$ core for $i = 1, ..., k$.

Step 4. The geodesic from H to G is

$$Geo = \arg \min_{\mathbf{H} \in \{Geo^1, Geo^2, ..., Geo^k\}} \lambda(\mathbf{H}).$$

Theoretically, the computing time of parallel computing on k cores would be shortened by a factor of $1/k$. However, the actual computing time on each core is a little longer than the original computing time divided by k.

For the geodesic searching with $g \leq 6$, brute force simulation with parallel computing performs well. For the geodesic searching with $g = 7, 8$, brute force simulation with parallel computing technique still takes days to find one geodesic with access to 100 cores. Thus, we develop the following quantile technique to save the computing time by selecting the "good" penultimate histograms.

5.3 Quantile Technique

From our simulation experiments, we observe that the geodesic from H to G always has a relatively small one-step cost from the penultimate histogram to the target G for many different pairs of (H, G). By using this observation, we implement the following brute force simulation with the quantile technique on multicores.

Algorithm 5.3: Brute Force Simulation with Quantile Technique on Multicores

Step 1. H is the starting histogram, G is the target histogram, $PEN = \mathcal{H} - G$, $Mesh$ is the mesh size, $DPEN$ is the set of all penultimate histograms from discretizing PEN by $Mesh$ and $card(DPEN) = n$, k is the number of cores.

Step 2. Compute the one-step cost from every penultimate histograms in $DPEN$ to G. $Quan(p)$ is the p-quantile of all these one step cost.

Step 3. Split the set $DPEN$ to k subsets, $DPEN_1, ..., DPEN_k$, where $\cup_{i=1}^{k} DPEN_i = DPEN$.

Step 4. On the $i-th$ core, select the p-quantile penultimate histograms set as

$$PEN_i(p) = \{y \in DPEN_i | C(y, G) \leq Quan(p)\}. \tag{5.1}$$

Denote $PEN(p) = \cup_{i=1}^{k} PEN_i(p)$.

Step 5. We use Algorithm 5.1 to search for a geodesic from H to G using the penultimate points set $PEN_i(p)$ on the $i-th$ core for $i = 1, ..., k$. Let Geo^i be the geodesic found on the $i-th$ core for $i = 1, ..., k$.

Step 6. Select a geodesic from H to G as

$$Geo = \text{argmin}_{\mathbf{H} \in \{Geo^1, Geo^2, ..., Geo^k\}} \lambda(\mathbf{H}).$$

We denote the rate function of the geodesic found by this p-quantile technique as $\lambda(Geo) = \lambda(H, G, p)$.

For a fixed pair (H, G) of the starting histogram H and the target histogram G, if $\lambda(H, G, p) =$

$\lambda(H, G, 1)$, we say that set $PEN(p)$ is efficient to compute the geodesic from H to G. Since $card(PEN(p)) \approx p \cdot card(DPEN)$, the computing time of the geodesic search using Algorithm 5.3 will be shortened by a factor of p/k theoretically. Thus, for the geodesic computing when $g > 7$, if there exists a small percentage p such that $PEN(p)$ is efficient, the computing time of the geodesic search will be saved significantly by using the quantile technique. From several numerical examples we studied, we observe a relation between the number of genotypes g and the smallest percentage p such that $\lambda(H, G, p) = \lambda(H, G, 1)$. To describe this relation formally, we define the efficiency of $p - quantile$ technique as follows. Let $\mathcal{H} \times \mathcal{H}$ be the set of all pairs of the starting histogram H and the target histogram G, namely

$$\mathcal{H} \times \mathcal{H} = \{(H, G) | H, G \in \mathcal{H}\}.$$

Let the efficiency of $p - quantile$ technique be

$$EFF(p) = \frac{card(\{(H, G) \in \mathcal{H} \times \mathcal{H} | PEN(p) \text{ is efficient for } (H, G)\})}{card(\mathcal{H} \times \mathcal{H})}.$$

We conclude that the relation between g and $EFF(g)$ is as follows: as the number of genotypes g increases, the efficiency of $p - quantile$ technique $EFF(p)$ increases. To verify this conclusion, we first use the following algorithm to estimate the efficiency of the $p - quantile$ technique.

Algorithm 5.4: Estimate the Efficiency $EFF(p)$

Step 1. Randomly select n pairs of (H, G) satisfying that $\min(H) > 0.01$, $\min(G) > 0.01$ and $C(H, G) > 0.01$, where $H \in \mathcal{H}$ is the starting histogram and $G \in \mathcal{H}$ is the target histogram. Name this set HGP.

Step 2. For every pair of (H, G), compute the geodesic from H to G by using the brute force simulation with the $p - quantile$ technique. Recall that $PEN(p)$ is the $p - quantile$ penultimate points set.

Step 3. For every pair of (H, G), check whether $PEN(p)$ is efficient for (H, G) and we estimate

$EFF(p)$ by $eff(p)$ as

$$eff(p) = \frac{card(\{(H,G) \in HGP | PEN(p) \text{ is efficient for } (H,G)\})}{card(HGP)}$$

$$= \frac{card(\{(H,G) \in HGP | PEN(p) \text{ is efficient for } (H,G)\})}{n}.$$

We now analyze the accuracy of this estimation. When $eff(p) \in [\frac{4}{n}, 1 - \frac{4}{n}]$ and $n > 100$, $eff(p) - EFF(p)$ has the normal distribution with mean 0 and standard deviation $\sqrt{\frac{eff(p)(1-eff(p))}{n}}$ approximately. Thus, the 90% confidence interval for $EFF(p)$ is approximately

$$\left[eff(p) - 1.6\sqrt{\frac{eff(p)(1-eff(p))}{n}}, eff(p) + 1.6\sqrt{\frac{eff(p)(1-eff(p))}{n}} \right].$$

When $eff(p) \in (1 - \frac{4}{n}, 1]$ and $n > 100$, $n \cdot eff(p)$ has Poisson distribution with mean $n \cdot EFF(p)$ approximately. Thus, the 90% confidence interval of $EFF(p)$ is approximately

$$\left[eff(p) - 1.6\sqrt{\frac{eff(p)}{n}}, eff(p) + 1.6\sqrt{\frac{eff(p)}{n}} \right].$$

We tested the conclusion of the relation between g and $EFF(p)$ on 10000, 5000, 240 pairs of (H,G) when $g = 3, 4, 5$, respectively. The numerical results support this conclusion. We present these simulations in Section 5.3.1.1. Based on our simulations, we conclude that $PEN(5\%)$ is efficient for most pairs of (H,G) when $g = 7, 8$. Thus, for these cases, the quantile technique can shorten the computing time of geodesic computing by a factor of 20 approximately. We present two examples of the geodesic computing when $g = 7, 8$ using Algorithm 5.3 in the Section 5.3.1.2.

Table 5.1: Efficiency Computation for $g = 3$

p	1	0.75	0.5	0.25	0.1	0.05	0.01
Mean RT	170 s	146 s	119 s	87 s	67 s	54 s	48 s
Std RT	40 s	7 s	3 s	2 s	11 s	5 s	9 s
Max RT	203 s	168 s	126 s	90 s	101 s	62 s	66 s
$eff(p)$	100%	100%	100%	94.83%	54.43%	34.74%	19.78%
90% CI L	98.4%	98.4%	98.4%	94.48%	53.63%	33.98%	19.14%
90% CI R	100%	100%	100%	95.18%	55.23%	35.50%	20.42%

Efficiency computation for 10000 pairs of (H, G) when $g = 3$. Mean RT, Std RT, Max RT are the mean, the standard deviation and the maximum of runtime of geodesic computing for 10000 pairs of (H, G) among 20 nodes on the Opuntia cluster, respectively. 90% confidence interval of $EFF(p)$ is [90% CI L, 90% CI R].

5.3.1 Quantile Technique Simulations

5.3.1.1 Efficiency Computing

For $g = 3$, we randomly selected 10000 pairs of (H, G) satisfying $\min(H) > 0.05$, $\min(G) > 0.05$, and $C(H, G) > 0.01$. Let the growth factor be $F = [200, 200^{1.08}, 200^{1.12}]$, let the mutation rate be $m = 10^{-8}$ and let the mutation matrix be $Q_{i,j} = 0.5$ for $i \neq j$. For every fixed $p \in \{1, 0.75, 0.5, 0.25, 0.1, 0.05, 0.01\}$, we use the $p - quantile$ technique to compute the geodesic for every pair of (H, G) with $Mesh = 0.005$ and compute $eff(p)$. We also compute the 90% confidence intervals of $EFF(p)$ for every p. We perform this computation task on 20 nodes with 10 cores on each node on the Opuntia cluster. Hence, we compute the geodesics for 50 pairs of (H, G) on each core. We present the mean of the runtime (Mean RT), standard deviation of the runtime (Std RT), the maximum of the runtime (Max RT) among 20 nodes. We also present the $eff(p)$ and the 90% confidence intervals ([90% CI L, 90% CI R]) for $p = 1, 0.75, 0.5, 0.25, 0.1, 0.05, 0.01$ in Table 5.1.

For $g = 4$, we randomly selected 5000 pairs of (H, G) satisfying $\min(H) > 0.05$, $\min(G) > 0.05$, and $C(H, G) > 0.01$. Let the growth factor be

$$F = [200, 200^{1.08}, 200^{1.1}, 200^{1.12}],$$

Table 5.2: Efficiency Computation for $g = 4$

p	1	0.75	0.5	0.25	0.1	0.05	0.01
Mean RT	7108 s	6505 s	3411 s	417 s	177 s	135 s	92 s
Std RT	1968 s	1288 s	875 s	71 s	9 s	11 s	30 s
Max RT	10639 s	9132 s	4313 s	298 s	189 s	142 s	121 s
$eff(p)$	100%	100%	100%	100%	93.9%	66.29%	25.1%
90% CI L	97.74%	97.74%	97.74%	97.74%	93.36%	65.22%	24.12%
90% CI R	100%	100%	100%	100%	94.44%	67.36%	26.08%

Efficiency computation for 5000 pairs of (H, G) when $g = 4$. Mean RT, Std RT, Max RT are the mean, the standard deviation and the maximum of runtime of geodesic computing for 1000 pairs of (H, G) among 20 nodes on Opuntia cluster, respectively. 90% confidence interval of $EFF(p)$ is [90% CI L, 90% CI R]

let the mutation rate be $m = 10^{-8}$ and let the mutation matrix be $Q_{i,j} = 1/3$ for $i \neq j$. For every fixed $p \in \{1, 0.75, 0.5, 0.25, 0.1, 0.05, 0.01\}$, we use the $p - quantile$ technique to find the geodesic for every pair of (H, G) with $Mesh = 0.005$ and compute $eff(p)$. We also compute the 90% confidence interval of $EFF(p)$. We perform the geodesic computation for 1000 pairs of (H, G) on 20 nodes with 10 cores on each node on Opuntia cluster and repeat for 5 times. Hence, we compute the geodesics for 5 pairs of (H, G) on each core. We present the mean of the runtime (Mean RT), standard deviation of the runtime (Std RT), the maximum of the runtime (Max RT) among 20 nodes, $eff(p)$ and the 90% confidence intervals ([90% CI L, 90% CI R]) for $p = 1, 0.75, 0.5, 0.25, 0.1, 0.05, 0.01$ in Table 5.2.

For $g = 5$, we randomly selected 5000 pairs of (H, G) satisfying $\min(H) > 0.05$, $\min(G) > 0.05$, and $C(H, G) > 0.01$. Let the growth factor be

$$F = [200, 200^{1.08}, 200^{1.10}, 200^{1.11}, 200^{1.12}],$$

let the mutation rate be $m = 10^{-8}$ and let the mutation matrix be $Q_{i,j} = 0.25$ for $i \neq j$. For every fixed $p \in \{1, 0.75, 0.5, 0.25, 0.1, 0.05, 0.01\}$, we use the $p-quantile$ technique to find the geodesic for every pair of (H, G) with $Mesh = 0.01$ and compute $eff(p)$. We also compute the 90% confidence

Table 5.3: Efficiency Computation for $g = 5$

p	1	0.75	0.5	0.25	0.1	0.05	0.01
Mean RT	11610 s	7789 s	3331 s	2058 s	1056 s	1769 s	1600 s
Std RT	7698 s	4613 s	1691 s	378 s	282 s	375 s	246 s
Max RT	32274 s	18721 s	9842 s	2706 s	1613 s	2255 s	1997 s
$eff(p)$	100%	100%	100%	100%	100%	100%	70.83%
90% CI L	89.67%	89.67%	89.67%	89.64%	89.64%	89.64%	66.14%
90% CI R	100%	100%	100%	100%	100%	100%	75.52%

Efficiency computation for 240 pairs of (H, G) when $g = 5$. Mean RT, Std RT, Max RT are the mean, the standard deviation and the maximum of runtime of geodesic computing for 240 pairs of (H, G) among 20 nodes on Opuntia cluster, respectively. 90% confidence interval of $EFF(p)$ is [90% CI L, 90% CI R]

interval of $EFF(p)$. We perform the geodesic computation for 240 pairs of (H, G) on 20 nodes with 12 cores on each node on Opuntia cluster. Hence, we compute the geodesic for 1 pair of (H, G) on each core. We present the mean of the runtime (Mean RT), the standard deviation of the runtime (Std RT), the maximum of the runtime (Max RT) among 20 nodes, $eff(p)$ and the 90% confidence intervals ([90% CI L, 90% CI R]) for $p = 1, 0.75, 0.5, 0.25, 0.1, 0.05, 0.01$ in Table 5.3.

From Tables 5.3, 5.4, and 5.5, we see that for fixed p, as $g \in \{3, 4, 5\}$ increases, $eff(p)$ increases. Thus, we predict that $PEN(0.1)$ is efficient for most pairs of (H, G) for $g = 6, 7, 8$. In Section 5.3.1.2, we present one geodesic computing example with the quantile technique for $g = 7, 8$.

5.3.1.2 Geodesic Computing Example When $g = 7, 8$

For $g = 7, 8$, we search for the geodesic for one pair of (H, G) using Algorithm 5.3 with $p = 0.1, 0.2$. If the cost of the geodesics obtained by using $PEN(0.1)$ and $PEN(0.2)$ is same, which is $\lambda(H, G, 0.1) = \lambda(H, G, 0.2)$. We assume both $PEN(0.1)$ and $PEN(0.2)$ are efficient.

For $g = 7$, let the growth factor be

$$F = [200, 200^{1.07}, 200^{1.08}, 200^{1.09}, 200^{1.10}, 200^{1.11}, 200^{1.12}],$$

Table 5.4: Geodesic Example when $g = 7$

Geo_0	0.6000	0.1000	0.1000	0.0500	0.0500	0.0500	0.0500
Geo_1	0.5094	0.1013	0.1007	0.0699	0.0721	0.0960	0.0505
Geo_2	0.3827	0.1116	0.1110	0.0843	0.0852	0.1612	0.0640
Geo_3	0.2656	0.1145	0.1142	0.0948	0.0950	0.2382	0.0777
Geo_4	0.1700	0.1100	0.1100	0.1000	0.1000	0.3200	0.0900
Geo_5	0.1000	0.1000	0.1000	0.1000	0.1000	0.4000	0.1000

Geodesic from $H = [0.6, 0.1, 0.1, 0.05, 0.05, 0.05, 0.05]$ to $G = [0.1, 0.1, 0.1, 0.1, 0.1, 0.4, 0.1]$

the mutation rate be $m = 10^{-8}$, the mutation matrix be $Q_{i,j} = 1/6$ for $i \neq j$. The starting histogram is $H = [0.6, 0.1, 0.1, 0.05, 0.05, 0.05, 0.05]$, the target histogram is $G = [0.1, 0.1, 0.1, 0.1, 0.1, 0.4, 0.1]$ and $Mesh = 0.01$, we use $p = 0.1$ and $p = 0.2$ to perform the geodesic computing following Algorithm 5.3 on 16 nodes with 8 cores on each node. We obtained that $\lambda(H, G, 0.1) = \lambda(H, G, 0.2) = 0.0264$. The geodesic we found is presented in Table 5.4. The maximum computing time among all nodes is 3984 seconds and 4036 seconds for $p = 0.1$ and $p = 0.2$, respectively. The detailed computing time on each node is in Appendix.

For $g = 8$, let the growth factor be

$$F = [200, 200^{1.06}, 200^{1.07}, 200^{1.08}, 200^{1.09}, 200^{1.10}, 200^{1.11}, 200^{1.12}],$$

the mutation rate be $m = 10^{-8}$, the mutation matrix be $Q_{i,j} = 1/7$ for $i \neq j$. When the starting histogram is $H = [0.5, 0.1, 0.1, 0.1, 0.05, 0.05, 0.05, 0.05]$, the target histogram is $G = [0.05, 0.05, 0.1, 0.05, 0.05, 0.1, 0.5, 0.1]$ and $Mesh = 0.02$, we use $p = 0.1$ and $p = 0.2$ to perform the geodesic computing with the quantile technique on 32 nodes with 16 cores on each node. We obtained that $\lambda(H, G, 0.1) = \lambda(H, G, 0.2) = 0.0534$. The geodesic we found is in Table 5.5. The maximum computing time among all nodes is 5723 seconds and 5309 seconds for $p = 0.1$ and $p = 0.2$, respectively. The detailed computing time on each node is in Appendix.

From these two examples, we can see that the computing time of the geodesic computing on

Table 5.5: Geodesic Example when $g = 8$

Geo_0	0.5000	0.1000	0.1000	0.1000	0.0500	0.0500	0.0500	0.0500
Geo_1	0.4046	0.1027	0.0817	0.1155	0.0396	0.0632	0.1033	0.0893
Geo_2	0.2999	0.1009	0.0953	0.1125	0.0464	0.0771	0.1711	0.0969
Geo_3	0.2087	0.0927	0.1037	0.1025	0.0510	0.0886	0.2506	0.1023
Geo_4	0.1369	0.0799	0.1067	0.0871	0.0530	0.0963	0.3355	0.1047
Geo_5	0.0850	0.0650	0.1050	0.0688	0.0525	0.1000	0.4200	0.1038
Geo_6	0.0500	0.0500	0.1000	0.0500	0.0500	0.1000	0.5000	0.1000

Geodesic from $H = [0.5, 0.1, 0.1, 0.1, 0.05, 0.05, 0.05, 0.05]$ to $G = [0.05, 0.05, 0.1, 0.05, 0.05, 0.1, 0.5, 0.1]$

multi-cores with the quantile technique is reasonable for $g = 7, 8$. However, if we want to perform the geodesic computing for $g \geq 9$, parallel computing with quantile technique is still not efficient enough. Thus, we develop the following geodesic shooting algorithm for the geodesic computing when $g \geq 9$. The number of genotypes does not impact the computing time of this geodesic shooting algorithm strongly.

5.4 Modified Geodesic Shooting Method

Reverse geodesic shooting method is used to construct geodesics on Riemannian manifolds originally. Consider a smooth compact Riemannian manifold $\mathcal{M}$ of dimension r. Fix the time interval $[0, T]$, as well as the given starting and the terminal points $H, G \in \mathcal{M}$. Fix any initial vector Y in the r dimensional tangent space to $\mathcal{M}$ at point G. There exists a unique reverse geodesics $h(t)$ for $t \in (0, T)$, which is the solution of a second order differential equation

$$h''(t) = F(h'(t), h(t)) \quad \text{for} \quad 0 < t < T,$$

verifying $h(T) = G$ and $h'(T) = Y$. Moreover, $h(t) = h(t, G, Y)$ is a smooth function of (t, G, Y). The generic geodesic shooting algorithm on manifolds is to search for an initial vector Y tangent to

$\mathcal{M}$ at G, by minimizing the distance between $h(0, G, Y)$ and H. The main differences of geodesics searching in our model are that, time is discrete and the cost function is used to measure the distance between histograms. Thus, we implemented a modified geodesic shooting method by minimizing the cost of paths with a fixed number of steps from H to G.

We first explain the shooting algorithm for constructing the cost minimizing path with a fixed number of steps from the starting histogram H to the target histogram G.

Algorithm 5.4.1: Shooting Algorithm for Paths with a Fixed Length

Step 1. We fix the number of steps of all paths to be $T \geq 3$. Denote the secure set

$$SEC = \{H \in \mathcal{H} | \min(H) \geq 0.005\}.$$

$Y_1 \in SEC$ is the initial penultimate histogram.

Step 2. At iteration $t \geq 1$, the current penultimate histogram is Y_t. We construct a reverse geodesic

$$RG(Y_t) = [RG_1 = G, RG_2 = Y, RG_3, ..., RG_l],$$

by using the formula (4.12) and $RG_i \in SEC$ for $i = 1, ..., l$. If $l < T - 1$, we stop this computation. Otherwise, we continue to Step 3.

Step 3. We truncate $RG(Y_t)$ to construct a path with T steps from H to G as

$$FP_t = FP(Y_t) = [H, RG_{T-1}, RG_{T-2}, RG_{T-3}, ..., RG_1 = G] \tag{5.2}$$

Compute

$$K_t = \chi(m, RG_{T-1}, RG_{T-2}) \tag{5.3}$$

and $\lambda_t = \lambda(FP_t)$.

Step 4. We compute the gradient of the $\lambda(FP_t)$ with respect to Y_t as stated in Section 5.4.1. Name this gradient $GRD_t = GRD(Y_t) \in \mathbb{R}^{g-1}$. $-GRD_t$ is the search direction at iteration t.

Step 5. We use the line search method to find a suitable step length $\alpha_t \in \mathbb{R}$ for iteration t.

Step 6. Update the first $g-1$ coordinates of the penultimate histogram for iteration $t+1$ as

$$Y_{t+1}(1:g-1) = Y_t(1:g-1) - \alpha_t \cdot GRD_t.$$

Update $Y_{t+1}(g)$ as $Y_{t+1}(g) = 1 - \sum_{i=1}^{g-1} Y_t(i)$.

Step 7. We continue to Step 8 if $t > 500$ or $||H - Y_t|| < 10^{-4}$. Otherwise, we repeat Step 2 to Step 7.

Step 8. Suppose we stop this computation at iteration L, the cost minimizing path $FP(T)$ with T steps is selected as

$$FP(T) = \text{argmin}_{\mathbf{H} \in \{FP_1, \ldots, FP_L\}} \lambda(\mathbf{H}).$$

By using Algorithm 5.4.1, we are able to find a cost minimizing path $FP(T)$ with T steps from H to G. We now explain the modified geodesic shooting algorithm (Algorithm 5.4.2) for computing the geodesic from the starting histogram H to the target histogram G.

Algorithm 5.4.2: Modified Geodesic Shooting Algorithm

Step 1. We start with the number of steps in a path being $T = 3$.

Step 2. For the current number of steps T, we compute $FP(T)$ following Algorithm 5.4.1. Denote the rate function of $FP(T)$ as $\lambda(T) = \lambda(FP(T))$.

Step 3. We update the number of steps to be $T+1$, we compute $FP(T+1)$ following Algorithm 5.4.1. Denote the rate function of $FP(T+1)$ as $\lambda(T+1) = \lambda(FP(T+1))$.

Step 4. We stop the computation once $\lambda(T+1) > \lambda(T)$. Otherwise, we repeat Step 2 to Step 3.

5.4.1 Gradient of the Rate Function

5.4.1.1 Gradient of One-step Cost

Recall the one step cost from H to G,

$$C(H,G) = KL(G,\Phi) + m\sum_{j\neq k} F_j H(j) Q_{j,k}(1 - U_k/U_j),$$

where

$$KL(G,\Phi) = \sum_j G(j)\log(G(j)/\Phi(j)),$$

$$U_j = \exp\left(\frac{G(j)}{F_j H(j)}\right),$$

and

$$\Phi(j) = \frac{F_j H(j)}{\langle F, H\rangle}.$$

Since $G(g) = 1 - \sum_{j=1}^{g-1} G(j)$ and $H(g) = 1 - \sum_{j=1}^{g-1} H(j)$, $C(H,G)$ is a function of $H(1),...,H(g-1), G(1),...,G(g-1)$ and we have

$$\frac{\partial G(g)}{\partial G(j)} = -1,$$

$$\frac{\partial H(g)}{\partial H(j)} = -1,$$

for $j = 1,...,g-1$. Hence,

$$\frac{\partial \Phi(j)}{\partial H(j)} = \frac{F_j\langle F,H\rangle - F_j H(j)(F_j - F_g)}{\langle F,H\rangle^2}$$

for $j = 1,...,g-1$, and

$$\frac{\partial \Phi(j)}{\partial H(k)} = \frac{F_j H(j)(F_g - F_k)}{\langle F,H\rangle^2}$$

for $j = 1, ..., g-1$ and $k = 1, ..., g-1$, and

$$\frac{\partial \Phi(g)}{\partial H(j)} = \frac{-F_g \langle F, H \rangle - F_g H(g)(F_j - F_g)}{\langle F, H \rangle^2}$$

for $j = 1, ..., g-1$. Therefore,

$$\frac{\partial KL}{\partial H(j)} = \sum_k -\frac{G(k)}{\Phi(k)} \frac{\partial \Phi(k)}{H(j)},$$

$$\begin{aligned} \frac{\partial KL}{\partial G(j)} &= \frac{\partial}{\partial G(j)} G(j) \log \frac{G(j)}{\Phi(j)} - \frac{\partial}{\partial G(j)} G(g) \log \frac{G(g)}{\Phi(g)} \\ &= \log \frac{G(j)}{\Phi(j)} - \log \frac{G(g)}{\Phi(g)}, \end{aligned}$$

for $j = 1, ..., g-1$. We also have

$$\frac{\partial U_j}{\partial H(j)} = -\frac{U_j G(j)}{F_j H(j)^2},$$

$$\frac{\partial U_j}{\partial G(j)} = \frac{U_j}{F_j H(j)},$$

and

$$\frac{\partial U_g}{\partial G(j)} = \frac{U_g G(g)}{F_g H(g)^2},$$

$$\frac{\partial U_g}{\partial G(j)} = -\frac{U_g}{F_g H(g)},$$

for $j = 1, ..., g-1$. Let $DUH_{j,k} = \frac{\partial U_j}{\partial H_k}$ for $j = 1, ..., g$ and $k = 1, ..., g-1$. Denote

$$SS(H, G) = \sum_{j,k|j \neq k} m F_j H(j) Q_{j,k} \frac{U_k}{U_j}.$$

We rewrite $SS(H, G)$ as

$$SS(H, G) = \sum_{j \neq k, j,k=1,...,g-1} m F_j H(j) Q_{j,k} \frac{U_k}{U_j} + \sum_{k=1}^{g-1} \left(m F_g H(g) Q_{g,k} \frac{U_k}{U_g} + m F_k H(k) Q_{k,g} \frac{U(g)}{U(k)} \right).$$

Thus, we compute the derivative of $SS(H,G)$ with respect to $H(s)$ and $G(s)$ for $s = 1, ..., g-1$ as

$$\begin{aligned}\frac{\partial SS(H,G)}{\partial H_s} &= mF_gQ_{g,s}\left(-\frac{U_s}{U_g} + H(g)\frac{DUH_{s,s}U_g - U_sDUH_{g,s}}{U_g^2}\right) \\ &\quad + mF_sQ_{s,g}\left(\frac{U_g}{U_s} + H(s)\frac{DUH_{g,s}U_s - U_gDUH_{s,s}}{U_s^2}\right) \\ &\quad + \sum_{k=1,...,g-2|k\neq s}\left(mF_sQ_{s,k}(\frac{U_k}{U_s} - \frac{H(s)U_kDUH_{s,s}}{U_s^2}) + mF_kH(k)Q_{k,s}\frac{DUH_{s,s}}{U_k}\right) \\ &\quad + \sum_{k=1,...,g-2|k\neq s}\left(mF_gQ_{g,k}U_k(-\frac{1}{U_g} - \frac{H(g)DUH_{g,s}}{U_g^2}) + mF_kH(k)Q_{k,g}\frac{DUH_{g,s}}{U_k}\right),\end{aligned}$$

$$\begin{aligned}\frac{\partial SS(H,G)}{\partial G_s} &= \sum_{k\neq s,k=1,...,g-1}\left(-mQ_{s,k}\frac{U_k}{U_s} + mF_kH(k)Q_{k,s}\frac{U_s}{U_kF_sH(s)}\right) \\ &\quad + \sum_{k=1}^{g-1}\left(mQ_{g,k}\frac{U_k}{U_g} - mF_kH(k)Q_{k,g}\frac{U_g}{U_kF_gH_g}\right).\end{aligned}$$

Therefore, for $j = 1, ..., g-1$, we obtain

$$\frac{\partial C(H,G)}{\partial H(j)} = \frac{\partial KL}{\partial H(j)} - \frac{\partial SS(H,G)}{\partial H(j)}, \tag{5.4}$$

and

$$\frac{\partial C(H,G)}{\partial G(j)} = \frac{\partial KL}{\partial G(j)} - \frac{\partial SS(H,G)}{\partial G(j)}. \tag{5.5}$$

5.4.1.2 Gradient of One-step Geodesic

Recall that suppose the two histograms in a geodesic are $G_{T+1} = y$, $G_{T+2} = z$, then we have $G_T = x$ where

$$x_s = \hat{x}_s + m\hat{x}_sw_s,$$

$$X_s = \frac{y_s}{F_s}\exp(\frac{F_s}{\langle F, y\rangle} - \frac{z_s}{y_s}),$$

$$\hat{x}_s = \frac{X_s}{\sum_t X_t},$$

$$w_s = \alpha_s + \beta_s - \langle \hat{x}, \alpha + \beta \rangle,$$

$$\alpha_s = \sum_{k \neq s} (Q_{s,k} e_{s,k} - \frac{F_k X_k}{F_k X_s} Q_{k,s} e_{k,s}),$$

$$e_{s,k} = \exp(-\frac{y_s}{F_s \hat{x}_s} + \frac{y_k}{F_k \hat{x}_k}),$$

$$\beta_s = F_s \sum_k Q_{s,k} - (F_s + \frac{z_s}{y_s} \sum_k f_{s,k} Q_{s,k} - \frac{z_s}{F_s y_s^2} \sum_k F_k y_k Q_{k,s} f_{k,s}),$$

$$f_{s,k} = \exp(-\frac{z_s}{F_s y_s} + \frac{z_k}{F_k y_k}).$$

Since x, y and z are histograms, we only need to compute $\frac{\partial x_s}{\partial y_t}$ and $\frac{\partial x_s}{\partial z_t}$ for $s = 1, ..., g-1$ and $t = 1, ..., g-1$ and we have

$$\frac{\partial y_g}{\partial y_j} = -1,$$

$$\frac{\partial z_g}{\partial z_j} = -1,$$

for $j = 1, ..., g-1$. Hence, for $s = 1, ..., g-1$, we have

$$\frac{\partial X_s}{\partial y_s} = \exp\left(\frac{F_s}{\langle F, y \rangle} - \frac{z_s}{y_s}\right)\left(\frac{1}{F_s} + \frac{z_s}{F_s y_s} - \frac{y_s(F_s - F_g)}{\langle F, y \rangle^2}\right),$$

$$\frac{\partial X_s}{\partial z_s} = -\frac{1}{F_s} \exp\left(\frac{F_s}{\langle F, y \rangle} - \frac{z_s}{y_s}\right).$$

For $s = 1, ..., g$, $l = 1, ...g-1$, $s \neq l$, we have

$$\frac{\partial X_s}{\partial y_l} = \frac{y_s(F_g - F_l)}{\langle F, y \rangle^2} \exp\left(\frac{F_s}{\langle F, y \rangle} - \frac{z_s}{y_s}\right),$$

$$\frac{\partial X_s}{\partial z_l} = 0.$$

Denote $DXy_{s,l} = \frac{\partial X_s}{\partial y_l}$ for $s = 1, ..., g$, $l = 1, ..., g-1$. So, we obtain

$$\frac{\partial \sum_t X_t}{\partial y_k} = \exp\left(\frac{F_k}{\langle F, y\rangle} - \frac{z_k}{y_k}\right)\left(\frac{1}{F_k} + \frac{z_k}{F_k y_k} - \frac{y_k(F_k - F_g)}{\langle F, y\rangle^2}\right) + \sum_{s=1,...,g, s\neq k} \exp\left(\frac{F_s}{\langle F, y\rangle} - \frac{z_s}{F_s}\right)\frac{y_s(F_g - F_k)}{\langle F, y\rangle^2},$$

$$\frac{\partial \sum_t X_t}{\partial z_k} = \frac{\partial X_k}{\partial z_k},$$

for $k = 1, ..., g-1$. Denote $DSXy_k = \frac{\partial \sum_t X_t}{\partial y_k}$ for $k = 1, ..., g-1$. Therefore,

$$\frac{\partial \hat{x}_s}{\partial y_l} = \frac{DXy_{s,l}(\sum_t X_t) - X_s \cdot DSXy_l}{(\sum_t X_t)^2},$$

for $s = 1, ..., g$, $l = 1, ..., g-1$,

$$\frac{\partial \hat{x}_s}{\partial z_s} = \frac{\frac{\partial X_s}{\partial z_s}(\sum_{t\neq s} X_t)}{(\sum_t X_t)^2},$$

for $s = 1, ..., g-1$ and

$$\frac{\partial \hat{x}_s}{\partial z_l} = -\frac{X_s}{(\sum_t X_t)^2}\frac{\partial X_l}{\partial z_l}$$

for $s = 1, ..., g, l = 1, ..., g-1$, $s \neq l$. Denote $D\hat{x}y_{s,l} = \frac{\partial \hat{x}_s}{\partial y_l}$ and $D\hat{x}z_{s,l} = \frac{\partial \hat{x}_s}{\partial z_l}$ for $s = 1, ..., g$ and $l = 1, ..., g-1$. Recall that

$$e_{s,k} = \exp\left(-\frac{y_s}{F_s \hat{x}_s} + \frac{y_k}{F_k \hat{x}_k}\right).$$

Therefore, we have

$$Dey_{s,k,s} = \frac{\partial e_{s,k}}{\partial y_s} = e_{s,k}\left(\frac{y_s D\hat{x}y_{s,s} - \hat{x}_s}{\hat{x}_s^2} - \frac{y_k D\hat{x}y_{k,s}}{F_k \hat{x}_k^2}\right),$$

for $s = 1, ..., g-1$, $k = 1, ..., g$ with $s \neq k$,

$$Dey_{s,k,k} = \frac{\partial e_{s,k}}{\partial y_k} = e_{s,k}\left(\frac{y_s D\hat{x}y_{s,k}}{F_s \hat{x}_s^2} + \frac{\hat{x}_k - y_k D\hat{x}y_{k,k}}{\hat{x}_k^2}\right),$$

for $s = 1, ..., g$, $k = 1, ..., g-1$ with $s \neq k$,

$$Dey_{s,k,l} = \frac{\partial e_{s,k}}{\partial y_l} = e_{s,k}\left(\frac{y_s D\hat{x}y_{s,l}}{F_s\hat{x}_s^2} - \frac{y_k D\hat{x}y_{k,l}}{F_k\hat{x}_k^2}\right)$$

for $s = 1, ..., g$, $k = 1, ..., g$, $l = 1, ..., g-1$ with $l \neq s$ and $s \neq k$, and

$$Dez_{s,k,l} = \frac{\partial e_{s,k}}{\partial z_l} = e_{s,k}\left(\frac{y_s}{F_s\hat{x}_s^2} \cdot D\hat{x}z_{s,l} - \frac{y_k}{F_k\hat{x}_k^2} \cdot D\hat{x}z_{k,l}\right),$$

for $s, k = 1, ..., g$, and $l = 1, ..., g-1$. Recall

$$\alpha_s = \sum_{k \neq s}\left(Q_{s,k}e_{s,k} - \frac{F_k X_k}{F_s X_s}Q_{k,s}e_{k,s}\right).$$

Hence, we have

$$\frac{\partial \alpha_s}{\partial y_l} = \sum_{k \neq l} Q_{s,k}Dey_{s,k,l} - \sum_{k \neq s}\frac{F_k Q_{k,s}}{F_s}\left(e_{k,s}\frac{DXy_{k,l}X_s - X_k DXy_{s,l}}{X_s^2} + \frac{X_k Dey_{k,s,l}}{X_s}\right)$$

for $s = 1, ..., g$, $l = 1, ..., g-1$,

$$\frac{\partial \alpha_s}{\partial z_s} = \sum_{k \neq s}\left(Q_{s,k}Dez_{s,k,s} - \frac{F_k X_k Q_{k,s}}{F_s}\frac{Dez_{k,s,s}X_s - e_{k,s}\frac{\partial X_s}{\partial z_s}}{X_s^2}\right),$$

for $s = 1, ..., g-1$, as well as

$$\frac{\partial \alpha_s}{\partial z_l} = Q_{s,l}Dez_{s,l,l} - \frac{F_l Q_{l,s}}{F_s X_s}(e_{l,s}\frac{\partial X_l}{\partial z_l} + X_l Dez_{l,s,l}) + \sum_{k \neq s, k \neq l}(Q_{s,k}Dez_{s,k,l} - \frac{F_k X_k}{F_s X_s}Q_{k,s}Dez_{k,s,l}),$$

for $s = 1, ..., g$, $l = 1, ..., g-1$, $s \neq l$. Denote $D\alpha y_{s,l} = \frac{\partial \alpha_s}{\partial y_l}$ and $D\alpha z_{s,l} = \frac{\partial \alpha_s}{\partial z_l}$. Recall

$$f_{s,k} = \exp\left(-\frac{z_s}{F_s y_s} + \frac{z_k}{F_k y_k}\right).$$

Hence, we have

$$\frac{\partial f_{s,k}}{\partial y_s} = \frac{f_{s,k} z_s}{F_s y_s^2},$$

$$\frac{\partial f_{s,k}}{\partial z_s} = -\frac{f_{s,k}}{F_s y_s},$$

for $s = 1, ..., g-1$, $k = 1, ..., g$,

$$\frac{\partial f_{s,k}}{\partial y_k} = -\frac{f_{s,k} z_k}{F_k y_k^2},$$

$$\frac{\partial f_{s,k}}{\partial z_k} = \frac{f_{s,k}}{F_k y_k},$$

for $s = 1, ..., g$, $k = 1, ..., g-1$, as well as

$$\frac{\partial f_{s,k}}{\partial y_l} = 0,$$

$$\frac{\partial f_{s,k}}{\partial z_l} = 0,$$

for $l \neq s$ and $l \neq k$. Denote $Dfy_{s,k,l} = \frac{\partial f_{s,k}}{\partial y_l}$ and $Dfz_{s,k,l} = \frac{\partial f_{s,k}}{\partial z_l}$.

Recall that

$$\beta_s = F_s \sum_k Q_{s,k} - \left(F_s + \frac{z_s}{y_s} \sum_k f_{s,k} Q_{s,k} - \frac{z_s}{F_s y_s^2} \sum_k F_k y_k Q_{k,s} f_{k,s} \right).$$

Thus, we have

$$\frac{\partial \beta_s}{\partial y_s} = \frac{z_s}{y_s^2} \sum_k f_{s,k} Q_{s,k} - \left(F_s + \frac{z_s}{y_s} \right) \sum_{k \neq s} Dfy_{s,k,s} Q_{s,k} + \frac{2 z_s}{F_s y_s^3} \sum_k F_k y_k Q_{k,s} f_{k,s} - \frac{z_s}{F_s y_s^2} \sum_{k \neq s} F_k y_k Q_{k,s} Dfy_{k,s,s},$$

$$\frac{\partial \beta_s}{\partial z_s} = -\frac{1}{y_s}\sum_k f_{s,k}Q_{s,k} - \left(F_s + \frac{z_s}{y_s}\right)\sum_k Q_{s,k}Df z_{s,k,s} - \frac{1}{F_s y_s^2}\sum_k F_k y_k Q_{k,s} f_{k,s} - \frac{z_s}{F_s y_s^2}\sum_k F_k y_k Q_{k,s} Df z_{k,s,s}.$$

for $s = 1, ..., g-1$. For $s = 1, ..., g$, $l = 1, ..., g-1$ and $s \neq l$, we have

$$\frac{\partial \beta_s}{\partial y_l} = -\left(F_s + \frac{z_s}{y_s}\right) Q_{s,l} Df y_{s,l,l} - \frac{z_s F_l y_l Q_{l,s} Df y_{l,s,l}}{F_s y_s^2} - \frac{z_s F_l Q_{l,s} f_{l,s}}{F_s y_s^2},$$

$$\frac{\partial \beta_s}{\partial z_l} = -\left(F_s + \frac{z_s}{y_s}\right)\sum_{k\neq s} Q_{s,k} Df z_{s,k,l} - \frac{z_s}{F_s y_s^2}\sum_{k\neq s} F_k y_k Q_{k,s} Df z_{k,s,l}.$$

Denote $D\beta y_{s,l} = \frac{\partial \beta_s}{\partial y_l}$ and $D\beta z_{s,l} = \frac{\partial \beta_s}{\partial z_l}$, for $s = 1, ..., g$, $l = 1, ..., g-1$.

Recall that

$$w_s = \alpha_s + \beta_s - \langle \hat{x}, \alpha + \beta \rangle.$$

For $s = 1, ..., g-1$, we have

$$\frac{\partial \langle \hat{x}, \alpha + \beta \rangle}{\partial y_s} = \sum_t D\hat{x} y_{t,s}(\alpha_t + \beta_t) + \hat{x}_t(D\alpha y_{t,s} + D\beta y_{t,s}),$$

$$\frac{\partial \langle \hat{x}, \alpha + \beta \rangle}{\partial z_s} = \sum_t D\hat{x} z_{t,s}(\alpha_t + \beta_t) + \hat{x}_t(D\alpha z_{t,s} + D\beta z_{t,s}).$$

Denote $DSY_s = \frac{\partial \langle \hat{x}, \alpha+\beta \rangle}{\partial y_s}$ and $DSZ_s = \frac{\partial \langle \hat{x}, \alpha+\beta \rangle}{\partial z_s}$. Therefore,

$$\frac{\partial w_s}{\partial y_l} = D\alpha y_{s,l} + D\beta y_{s,l} - DSY_l.$$

$$\frac{\partial w_s}{\partial z_l} = D\alpha z_{s,l} + D\beta z_{s,l} - DSZ_l.$$

Denote $Dwy_{s,l} = \frac{\partial w_s}{\partial y_l}$ and $Dwz_{s,l} = \frac{\partial w_s}{\partial z_l}$.

Recall that

$$x_s = \hat{x}_s + m\hat{x}_s w_s.$$

Thus, we have

$$\frac{\partial x_s}{\partial y_l} = D\hat{x}y_{s,l}(1 + mw_s) + \hat{x}_s mDwy_{s,l}, \tag{5.6}$$

$$\frac{\partial x_s}{\partial z_l} = D\hat{x}z_{s,l}(1 + mw_s) + \hat{x}_s mDwz_{s,l}, \tag{5.7}$$

For $s = 1, ..., g-1$, $l = 1, ..., g-1$.

5.4.1.3 Gradient of Rate Functions of Truncated Reverse Geodesics

Given a reverse geodesic $RG = [RG_1, ..., RG_t]$, RG is determined by the target histogram $RG_1 = G$ and the penultimate histogram $RG_2 = Y$. We truncate this reverse geodesic to a trajectory with T steps as $TRG = [RG_T, RG_{T-1}, ..., RG_2 = Y, RG_1 = G]$. We want to compute the derivative of $\lambda(TRG)$ with respect to $Y(1), ..., Y(g-1)$. First, we clarify the notations of derivatives of one step cost function and reverse geodesic function χ. Recall that $C(H, G)$ is the one step cost from H to G, denote

$$D1C(H,G) = \frac{\partial C(H,G)}{\partial H} = [\frac{\partial C(H,G)}{\partial H(1)}, ..., \frac{\partial C(H,G)}{\partial H(g-1)}],$$

$$D2C(H,G) = \frac{\partial C(H,G)}{\partial G} = [\frac{\partial C(H,G)}{\partial G(1)}, ..., \frac{\partial C(H,G)}{\partial G(g-1)}].$$

Recall that if $\mathbf{h}^* = [h_0, h_1, h_2, ...]$ is a geodesic and we have $h_{t+1} = y$, $h_{t+2} = z$, then $h_t = x = \chi(m, y, z)$, where $t \geq 0$ and $\chi(m, y, z)$ is defined by formula (4.12). We denote the derivatives of x with respect to y and z as

$$D1\chi(m,y,z) = \frac{\partial x}{\partial y} = \begin{bmatrix} \frac{\partial x_1}{\partial y_1} & \frac{\partial x_1}{\partial y_2} & \cdots & \frac{\partial x_1}{\partial y_{g-1}} \\ \frac{\partial x_2}{\partial y_1} & \frac{\partial x_2}{\partial y_2} & \cdots & \frac{\partial x_2}{\partial y_{g-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial x_{g-1}}{\partial y_1} & \frac{\partial x_{g-1}}{\partial y_2} & \cdots & \frac{\partial x_{g-1}}{\partial y_{g-1}} \end{bmatrix},$$

$$D2\chi(m,y,z) = \frac{\partial x}{\partial z} = \begin{bmatrix} \frac{\partial x_1}{\partial z_1} & \frac{\partial x_1}{\partial z_2} & \cdots & \frac{\partial x_1}{\partial z_{g-1}} \\ \frac{\partial x_2}{\partial z_1} & \frac{\partial x_2}{\partial z_2} & \cdots & \frac{\partial x_2}{\partial z_{g-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial x_{g-1}}{\partial z_1} & \frac{\partial x_{g-1}}{\partial z_2} & \cdots & \frac{\partial x_{g-1}}{\partial z_{g-1}} \end{bmatrix}.$$

Recall that

$$\lambda(TRG) = \sum_{i=1}^{T-1} C(RG_{i+1}, RG_i).$$

Thus,

$$\frac{\partial \lambda(TRG)}{\partial Y} = \sum_{i=1}^{T-1} \frac{C(RG_{i+1}, RG_i)}{\partial Y} \tag{5.8}$$

$$= \sum_{i=1}^{T-1} \frac{\partial C(RG_{i+1}, RG_i)}{\partial RG_{i+1}} \frac{\partial RG_{i+1}}{\partial Y} + \frac{\partial C(RG_{i+1}, RG_i)}{\partial RG_i} \frac{\partial RG_i}{\partial Y}. \tag{5.9}$$

We can compute $\frac{\partial C(RG_{i+1}, RG_i)}{\partial RG_{i+1}}$ and $\frac{\partial C(RG_{i+1}, RG_i)}{\partial RG_i}$ as formula (5.4) and formula (5.5) for $i = 1, ..., T$.

We also know

$$\frac{\partial RG_1}{\partial Y} = \frac{\partial G}{\partial Y} = \text{Zero Matrix}, \tag{5.10}$$

$$\frac{\partial RG_2}{\partial Y} = \frac{\partial Y}{\partial Y} = \text{Identity Matrix}. \tag{5.11}$$

For $i = 1, ..., T-2$, we have that $RG_{i+2} = \chi(m, RG_{i+1}, RG_i)$. Thus,

$$\frac{\partial RG_{i+2}}{\partial Y} = D1\chi(m, RG_{i+1}, RG_i)\frac{\partial RG_{i+1}}{\partial Y} + D2\chi(m, RG_{i+1}, RG_i)\frac{\partial RG_i}{\partial Y}, \tag{5.12}$$

for $i = 1, ..., T-2$. Thus, we can compute the gradient $\frac{\partial \lambda(TRG)}{\partial Y}$ by using formula (5.8) to formula (5.12).

5.4.2 Modified Geodesic Shooting Example

In this section, we present two computational examples by using the modified geodesic shooting method. One is the same example we used to present the brute force simulation with quantile technique for $g = 8$. We use this same example to show that we can obtain a geodesic for this example in a much shorter time by using the modified geodesic shooting method. This geodesic is also very close to the geodesic we obtained by using the brute force simulation. The other example is a geodesic computation example for $g = 10$ by using the modified geodesic shooting method. We use this example to show that the computing time of this modified geodesic shooting method is not impacted by the number of genotypes strongly. Therefore, this modified shooting algorithm can perform well for geodesic computing when $g \geq 9$.

5.4.2.1 Modified Geodesic Shooting Example 1

Recall the example with $g = 8$ in Section 5.3.1.2. The growth factor is

$$F = [200, 200^{1.06}, 200^{1.07}, 200^{1.08}, 200^{1.09}, 200^{1.10}, 200^{1.11}, 200^{1.12}],$$

the mutation rate is $m = 10^{-8}$, the mutation matrix is $Q_{i,j} = 1/7$ for $i \neq j$, the starting histogram is

$$H = [0.5, 0.1, 0.1, 0.1, 0.05, 0.05, 0.05, 0.05],$$

the target histogram is

$$G = [0.05, 0.05, 0.1, 0.05, 0.05, 0.1, 0.5, 0.1]$$

and the mesh size is $Mesh = 0.02$. By using the brute force simulation with the quantile technique on multicores, we estimated the rate function of the geodesic from H to G to be 0.0534 and the

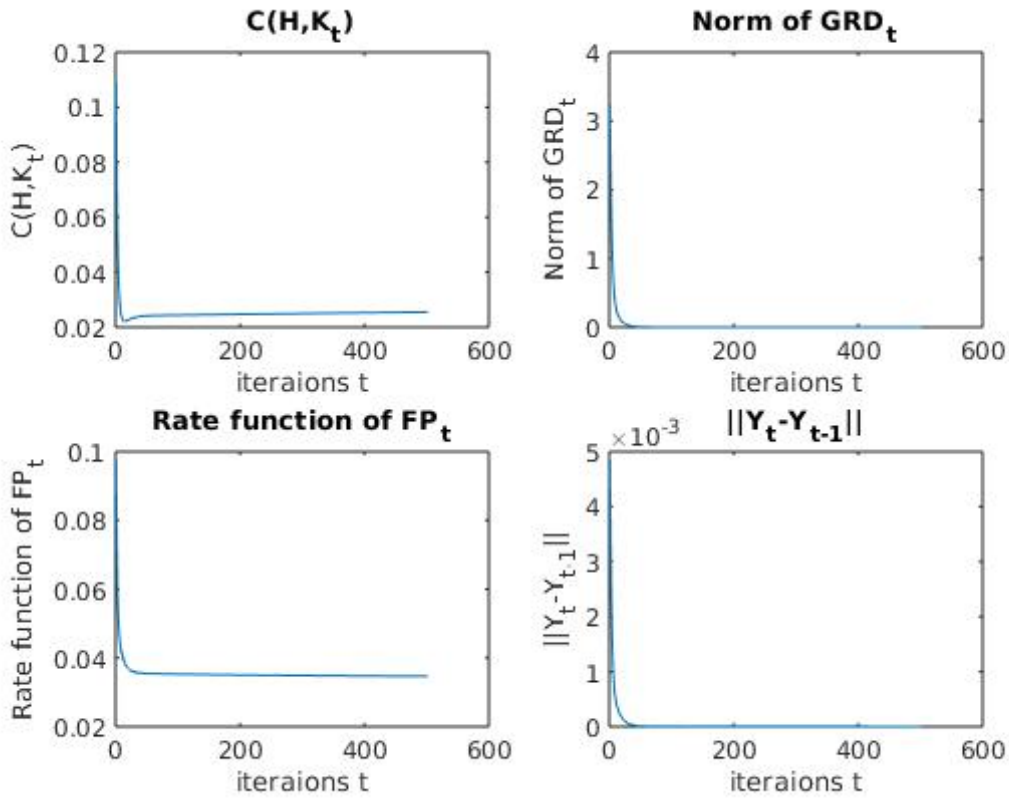

Figure 5.1: Modified Geodesic Shooting Example 1 with $g = 8$

Top left figure presents the one step cost $C(H, K_t)$ from H to K_t at iteration $t = 1, ..., 500$. Top right figure presents the norm of the gradient GRD_t at iteration $t = 1, ..., 500$. Bottom left figure presents the rate function $\lambda(FP_t)$ at iteration $t = 1, ..., 500$. Bottom right figure presents the norm $||Y_t - Y_{t-1}||$ at iteration $t = 2, ..., 500$.

penultimate histogram of this geodesic to be

$$[0.0850, 0.0650, 0.1050, 0.0688, 0.0525, 0.1000, 0.4200, 0.1038].$$

The maximum computing time among all nodes is about an hour. If we use the modified geodesic shooting algorithm to search for the geodesic from H to G, we find that the estimated rate function is 0.0350 and the estimated penultimate histogram is

$$[0.0837, 0.0650, 0.1129, 0.0663, 0.0566, 0.0999, 0.4173, 0.0982].$$

The total computing time is around 20 seconds by using one node on the Opuntia cluster. In Figure 5.1, we present the one step cost $C(H, K_t)$, where K_t is defined as formula (5.3), the norm of gradient $GRD_t = \frac{\partial \lambda(FP_t)}{\partial Y_t}$, the rate function $\lambda(FP_t)$ and the norm of $Y_t - Y_{t-1}$ for $t = 1, ..., 500$, when we fix the number of the steps in the path to be 7. From this figure, we can see that $\lambda(FP_t)$ decreases very fast using this modified geodesic shooting method. By using this modified geodesic shooting algorithm, we are also able to find a geodesic in a much shorter time.

5.4.2.2 Modified Geodesic Shooting Example 2

We now present a geodesic computing example when $g = 10$ by using the modified geodesic shooting method. This example shows that this shooting algorithm performs very well for geodesic searching when $g \geq 9$.

We first define the parameters of this example. Let the growth factor be

$$F = [200, 200^{1.04}, 200^{1.05}, 200^{1.06}, 200^{1.07}, 200^{1.08}, 200^{1.09}, 200^{1.10}, 200^{1.11}, 200^{1.12}].$$

Let the mutation rate be $m = 10^{-8}$. Let the mutation matrix be $Q_{i,j} = 1/9$ for $i \neq j$. Let the starting histogram be $H = [0.45, 0.10, 0.10, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05]$. Let the target histogram be $G = [0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.10, 0.45, 0.10]$. We use the modified geodesic shooting algorithm to search for a geodesic from H to G and find that the estimated rate function of this geodesic is 0.0277 and the estimated penultimate histogram is

$$[0.0741, 0.0616, 0.0626, 0.0561, 0.0567, 0.0566, 0.0555, 0.0974, 0.3818, 0.0977].$$

The total computing time is around 40 seconds on one node of the Opuntia cluster. In Figure 5.2, we present the one step cost $C(H, K_t)$, where K_t is defined as formula (5.3), the norm of gradient $GRD_t = \frac{\partial \lambda(FP_t)}{\partial Y_t}$, the rate function $\lambda(FP_t)$ and the norm of $Y_t - Y_{t-1}$ for $t = 1, ..., 500$ when we fix the number of the steps in the geodesic to be 8.

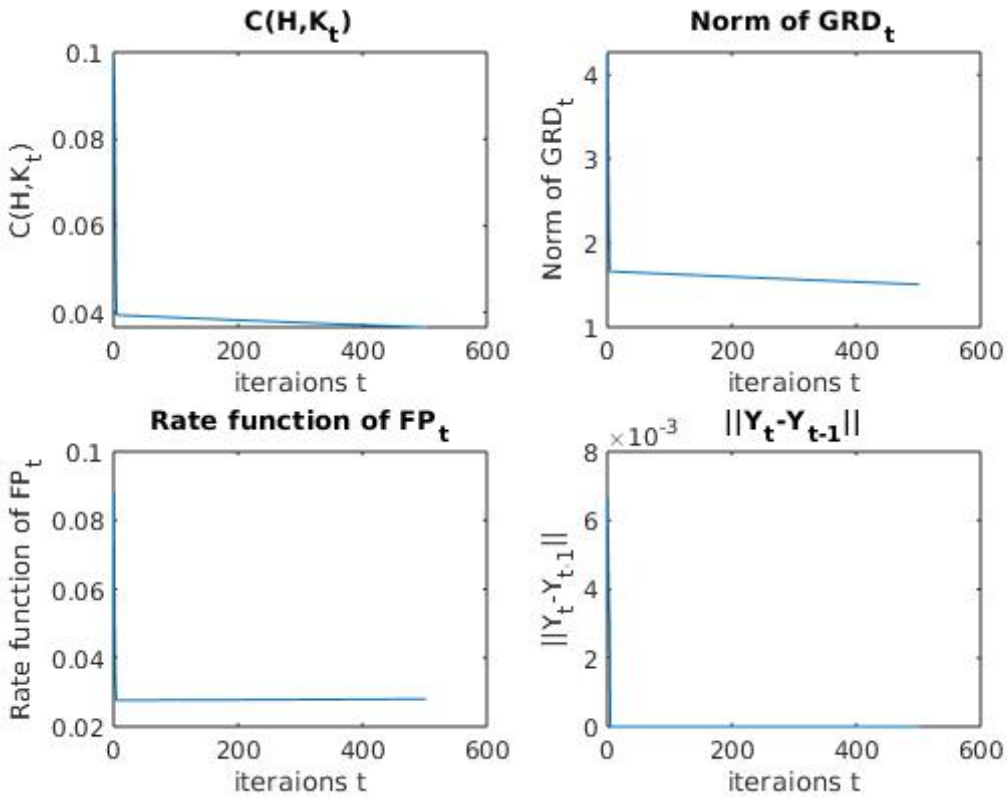

Figure 5.2: Modified Geodesic Shooting Example 2 with $g = 10$

Top left figure presents the one step cost $C(H, K_t)$ from H to K_t at iteration $t = 1, ..., 500$. Top right figure presents the norm of the gradient GRD_t at iteration $t = 1, ..., 500$. Bottom left figure presents the rate function $\lambda(FP_t)$ at iteration $t = 1, ..., 500$. Bottom right figure presents the norm $||Y_t - Y_{t-1}||$ at iteration $t = 2, ..., 500$.

From this example, we can see that the computing time of geodesic computing using the modified shooting method is not impacted strongly by the number of genotypes g. When we increase g from 8 to 9, the computing time increases from 20 seconds to 40 seconds. In comparison, the computing time of the geodesic searching using the brute force simulation would increase at least by a factor of 10 when g increases 1. Thus, this modified geodesic shooting method has a much better efficiency for the geodesic computing when $g \geq 9$.

Chapter 6

Fixations

The rare events of our interest are due to the fact that the frequencies of certain intermediate-strength genotypes become large in the bacterial population. Such rare events are called fixations. For a fixed genotype $J < g$ and a level β, define a target set $TAR(\beta) = \{H \in \mathcal{H} | H(J) > \beta\}$. For a fixed large T, $\mathbf{H} = [H_0, ..., H_T] \in \Omega_{T+1}$ is a histogram trajectory of length $T+1$, where $H_i \in \mathcal{H}$. Since the g-cells have the largest growth factor F_g, $H_T(g)$ will tend to 1 as $T \to \infty$ for N large. We say that $\mathbf{H}$ enters the target set $TAR(\beta)$ if there exists a step $1 \leq t$ such that $H_t(J) \geq \beta$, and denote this event as $Eve(J, \beta)$. This event is a rare event when β is relatively large and this event is the fixation of genotype J. Let $\mathbb{P}_{N,H_0}(Eve(J, \beta))$ be the probability of the event that trajectories starting with histogram H_0 and population size N realize the event $Eve(J, \beta)$. In fact, when the population size N is large, the fixation of genotype J happens very rarely even for $\beta > 0.2$. We will show this fact by presenting an example in Section 6.1.

6.1 Example of Fixations

We now discuss a fixation example with $g = 3$ genotypes. We are interested in the fixation of genotype $J = 2$ of trajectories starting from the histogram $H_0 = [0.9, 0.05, 0.05]$. We first analyze

the highest percentage of genotype 2 that the mean trajectory starting from H_0 reaches. We

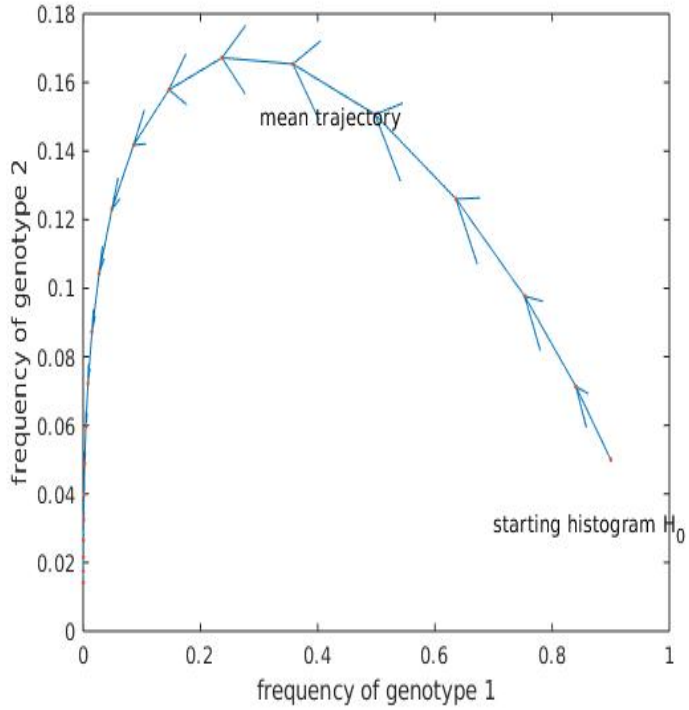

Figure 6.1: Mean Trajectory

Figure 6.1: mean trajectory starting from $h_0 = [0.9, 0.05, 0.05]$

present the mean trajectory **h** starting from $[0.9, 0.05, 0.05]$ is in Figure 6.1. The highest frequency of genotype 2 of the mean trajectory **h** is

$$\max_{t=0,1,..} h_t(2) = 0.167180.$$

For large population size N, the random trajectories will follow this mean trajectory **h** in a thin tube. We set $\beta = 0.167182$, which is slightly larger than $\max_t h_t(2)$, we use the brute force simulation to estimate the frequency of the event $Eve(J, \beta)$ for $N = 10^{10}, 10^{11}, 10^{12}, 10^{13}$ in the following Table 6.1. We can see that when $N = 10^{13}$, the event $Eve(J, \beta)$ already becomes rare. Thus, the event $Eve(J, \beta)$ is a rare event when $\beta > 0.2$ and $N \geq 10^{13}$.

When $N = 10^{13}$, the brute force simulation takes 7 minutes to generate $5 \cdot 10^7$ trajectories on

Table 6.1: Brute Force Simulation of $Eve(J,\beta)$

N	num of simulation	frequency of $Eve(J,\beta)$	stand error
10^{10}	10^6	0.4456	$4.9 \cdot 10^{-4}$
10^{11}	10^6	0.3369	$4.7 \cdot 10^{-4}$
10^{12}	10^6	0.0928	$2.9 \cdot 10^{-4}$
10^{13}	5×10^7	2.8×10^{-6}	$2.4 \cdot 10^{-7}$

Brute force simulation for different population size starting from $H_0 = [0.9, 0.05, 0.05]$.

50 cores on Opuntia cluster. If we want to simulate a much rarer event ($\beta > 0.3$), brute force simulation becomes too costly. Thus, we implement the importance sampling method and the genealogy method to study the rare fixations in this dissertation.

6.2 Optimal Trajectory $\mathcal{G}$ Realizing $Eve(J,\beta)$

Paper [3] obtained the following large deviation result,

$$\lim_{N\to\infty} -\frac{1}{N}\log \mathbb{P}_{N,H_0}(Eve(J,\beta)) = \Lambda(Eve(J,\beta)) = \inf_{\mathbf{H}\in Eve(J,\beta)} \lambda(\boldsymbol{H})$$

Therefore, one approach of estimating $\mathbb{P}_{N,H_0}(Eve(J,\beta))$ is to estimate $\inf_{\mathbf{H}\in Eve(J,\beta)} \lambda(\boldsymbol{H})$. Hence, we want to find an optimal trajectory that realize the event $Eve(J,\beta)$ with the smallest cost. We develop the following algorithms to search for this optimal trajectory $\boldsymbol{\mathcal{G}}$.

6.2.1 Algorithms for Searching for $\mathcal{G}$

Algorithm 6.1: Brute Force Simulation for Searching for $\mathcal{G}$

Step 1. $TAR = TAR(\beta)$ is the set of target histograms. $Mesh$ is the mesh size. We use the mesh size $Mesh$ to discretize the set TAR. Let $DTAR$ be the discretized set of target histograms. H_0 is the starting histogram.

Step 2. For every target histogram $G \in TAR$, $PEN(G) = \{H \in \mathcal{H} | H \neq G\}$ is the set of

penultimate histograms set for G. We use the brute force simulation Algorithm 5.1 in Section 5.1 to search for the geodesic from H_0 to G. Suppose $card(DTAR) = n$, let $\mathbf{P}_1, ..., \mathbf{P}_n$ be the geodesics from H_0 to all the target histograms in $DTAR$.

Step 3. The optimal path $\boldsymbol{\mathcal{G}}$ for realizing $Eve(J, \beta)$ is

$$\boldsymbol{\mathcal{G}} = \text{argmin}_{\mathbf{H} \in \{\mathbf{P}_1, ..., \mathbf{P}_n\}} \lambda(\mathbf{H}).$$

In Step 2 of Algorithm 6.1, we can also use the parallel computing technique, the quantile technique, and the modified shooting algorithm instead of the brute force simulation algorithm to search for the geodesic from H_0 to a target histogram. We also implement the following multi-scale algorithm to improve the efficiency of the algorithm for searching for the optimal trajectory $\boldsymbol{\mathcal{G}}$.

Algorithm 6.2: Multi-scale Algorithm for Searching for $\boldsymbol{\mathcal{G}}$

Step 1. We start with the set of target histograms TAR, the set of penultimate histograms $PEN = \mathcal{H} - TAR$ and the mesh size $Mesh = 0.01$.

Step 2. Given the current TAR, PEN and $Mesh$, we find the optimal trajectory $\boldsymbol{\mathcal{G}} = [\mathcal{G}_0 = H_0, ..., \mathcal{G}_{T-1}, \mathcal{G}_T]$ by using Algorithm 6.1.

Step 3. Update the set of penultimate histograms TAR, the set of penultimate histograms PEN and the mesh size $Mesh$ as

$$new\_TAR = \{H || H - \mathcal{G}_T|_\infty < 3 \cdot mesh\},$$

$$new\_PEN = \{H || H - \mathcal{G}_{T-1}|_\infty < 3 \cdot mesh\},$$

$$new\_Mesh = 0.1 \cdot Mesh.$$

Step 4. Given the new_TAR, new_PEN and new_Mesh, find the new optimal trajectory $\boldsymbol{\mathcal{G}'}$ by using Algorithm 6.1.

Step 5. Stop the algorithm if $|\lambda(\boldsymbol{\mathcal{G}}) - \lambda(\boldsymbol{\mathcal{G}'})| \leq 10^{-3} \cdot \lambda(\boldsymbol{\mathcal{G}})$. The optimal trajectory for realizing

the event $Eve(J, \beta)$ is $\boldsymbol{\mathcal{G}'}$.

6.2.2 Examples of the Multi-scale Algorithm

We present two examples of searching for the optimal trajectory $\boldsymbol{\mathcal{G}}$ for realizing the event $Eve(J, \beta)$ by using the multi-scale algorithm. By presenting these two examples, we show that the multi-scale algorithm is much more efficient than the brute force Algorithm 6.1.

6.2.2.1 Example 1

Let $F = [200, 200^{1.08}, 200^{1.20}]$, $m = 10^{-7}$ and

$$Q = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0.5 & 0 \end{bmatrix}.$$

Select $H_0 = [0.8, 0.1, 0.1]$, $J = 2$, $\beta = 0.7$. We start with

$$TAR = \{G \mid G(1) \in [0.1,\ 0.69], G(2) = 0.7\},$$

$$PEN = \{G \mid G(1) \in [0.05,\ 0.3], G(2) \in [0.1,\ 0.7)\}$$

and the $Mesh = 0.01$, we generate 7×10^6 reverse geodesics to find the optimal geodesic $\boldsymbol{\mathcal{G}}$ to realize the event $Eve(2, 0.7)$ by using the multi-scale algorithm. The optimal trajectory $\boldsymbol{\mathcal{G}}$ is presented in Table 6.2 and $\lambda(\boldsymbol{\mathcal{G}}) = 0.2169$. The mesh size decrease to 10^{-5} from 0.01 through the computation. This computation takes 120 seconds on one core on Opuntia. If we use the brute force simulation algorithm with $Mesh = 10^{-5}$, we will need to generate 10^{20} reverse geodesics, which is 10^{14} more costly than the multi-scale algorithm.

We also find the optimal trajectories when $\beta = 0.3, 0.5$ using the multi-scale algorithm. The

Table 6.2: Optimal Trajectory $\boldsymbol{\mathcal{G}}$

$\mathcal{G}_0$	0.8000	0.1000	0.1000
$\mathcal{G}_1$	0.7579	0.1927	0.0494
$\mathcal{G}_2$	0.6566	0.3071	0.0363
$\mathcal{G}_3$	0.5253	0.4345	0.0402
$\mathcal{G}_4$	0.3847	0.5574	0.0579
$\mathcal{G}_5$	0.2532	0.6530	0.0938
$\mathcal{G}_6$	0.1450	0.7000	0.1550

Optimal trajectory for $Eve(2, 0.7)$ given $H_0 = [0.8, 0.1, 0.1]$.

Table 6.3: Optimal Trajectory $\boldsymbol{\mathcal{G}}$

$\mathcal{G}_0$	0.8000	0.1000	0.1000
$\mathcal{G}_1$	0.6383	0.1805	0.1812
$\mathcal{G}_2$	0.4266	0.2578	0.3156
$\mathcal{G}_3$	0.2233	0.3000	0.4767

Optimal trajectory for $Eve(2, 0.3)$ given $H_0 = [0.8, 0.1, 0.1]$.

Table 6.4: Optimal Trajectory $\boldsymbol{\mathcal{G}}$

$\mathcal{G}_0$	0.8000	0.1000	0.1000
$\mathcal{G}_1$	0.6759	0.2068	0.1173
$\mathcal{G}_2$	0.5002	0.3325	0.1673
$\mathcal{G}_3$	0.3107	0.4413	0.2480
$\mathcal{G}_4$	0.1514	0.5000	0.3486

Optimal trajectory for $Eve(2, 0.5)$ given $H_0 = [0.8, 0.1, 0.1]$.

optimal trajectories for event $Eve(2, 0.3)$ and $Eve(2, 0.5)$ are in Tables 6.3 and 6.4, respectively. We plot these three optimal trajectories in Figure 6.2. We can see these three optimal trajectories shoot much higher than the mean trajectory. As β increases, the optimal trajectories take more steps to realize the rare event $Eve(J, \beta)$. However, the total number of steps in the optimal trajectory of the rare events in this discrete time model remains less than 15 steps most of the time. In comparison, the optimal trajectory in a continuous time stochastic model often have many more steps.

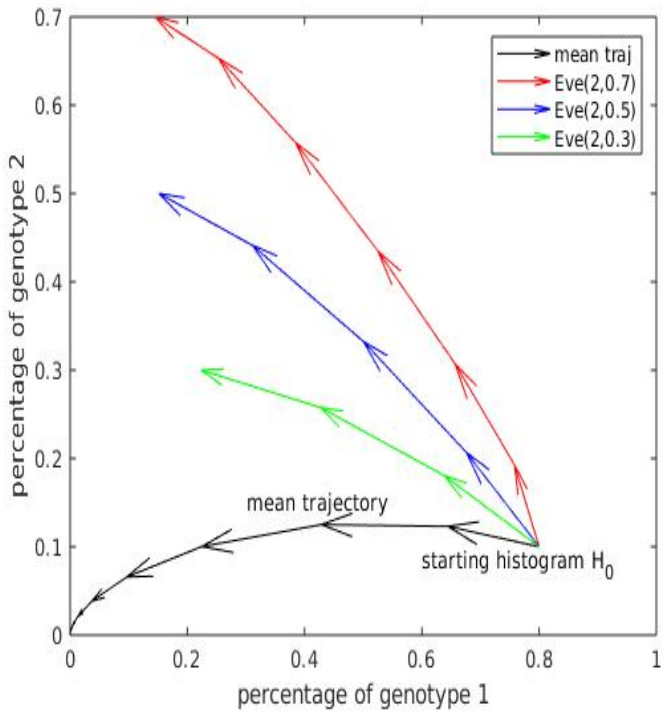

Figure 6.2: Optimal Trajectories and the Mean Trajectory

$H_0 = [0.8, 0.1, 0.1]$, the black path is the mean trajectory. Green, blue, and red paths are the optimal trajectories realizing $Eve(2, 0.3)$, $Eve(2, 0.5)$ and $Eve(2, 0.7)$, respectively.

6.2.2.2 Example 2

The starting histogram is $H_0 = [0.7, 0.1, 0.1, 0.1]$, the growth vector is

$$F = [200, 200^{1.08}, 200^{1.10}, 200^{1.20}],$$

and the mutation matrix is

$$Q = \begin{bmatrix} 0 & 0.3 & 0.3 & 0.4 \\ 0.3 & 0 & 0.3 & 0.4 \\ 0.3 & 0.3 & 0 & 0.4 \\ 0.3 & 0.3 & 0.4 & 0 \end{bmatrix},$$

the mutation rate is 10^{-7}. We use the multi-scale algorithm to search for the optimal trajectories $\boldsymbol{\mathcal{G}}$ for realizing the event $Eve(3, 0.3)$, $Eve(3, 0.5)$ and $Eve(3, 0.7)$. We present these three optimal trajectories in Appendix. We plot these optimal trajectories in Figure 6.3. The computation of searching for $\boldsymbol{\mathcal{G}}$ by using this multi-scale algorithm takes about 400 seconds on one core of Opuntia for each of these three events. The mesh size decreases to 10^{-4} from 0.01. We generated around $3 \cdot 10^7$ inverse geodesics to find the optimal trajectory $\boldsymbol{\mathcal{G}}$. In comparison, we need to generate around 10^{24} reverse geodesics to search for the optimal trajectory by using the brute force simulation with mesh size 10^{-4} and when $g = 4$. So the multi-scale algorithm improves the efficiency of the computation of searching for the optimal trajectory $\boldsymbol{\mathcal{G}}$ significantly.

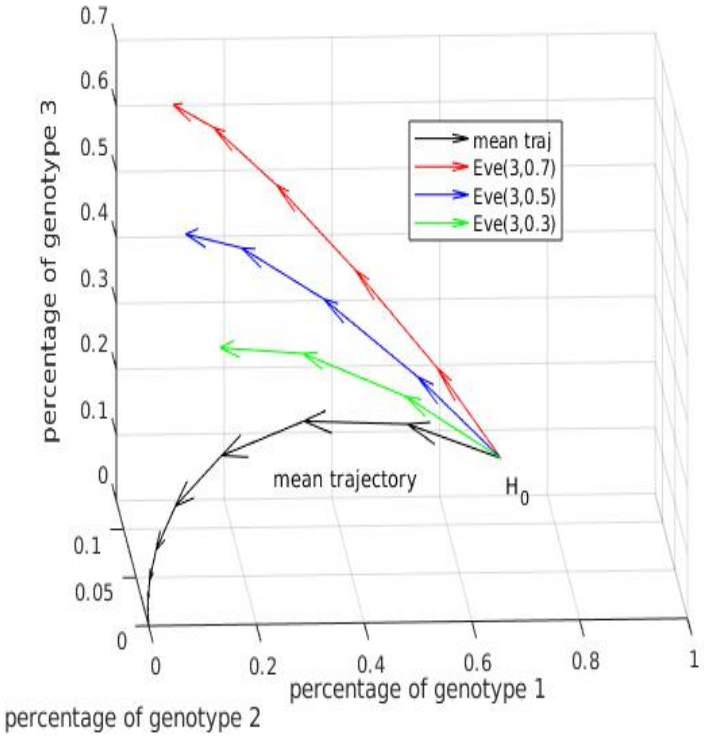

Figure 6.3: Optimal Trajectories and the Mean Trajectory

$H_0 = [0.7, 0.1, 0.1, 0.1]$, the black path is the mean trajectory. Green, blue, and red paths are the optimal trajectories realizing $Eve(3, 0.3)$, $Eve(3, 0.5)$ and $Eve(3, 0.7)$, respectively

Chapter 7

Importance Sampling in Path Space

7.1 Background of Importance Sampling

Importance sampling is an important Monte Carlo simulation technique which increases the frequency of rare events happening in the stochastic model by using a manipulated probability distribution. Consider a rare event with probability $p = 10^{-6}$, if we use the brute force simulation, we at least need a sample size of 10^8 to guarantee that the standard error $\sqrt{p(1-p)/N}$ satisfies $\sqrt{p(1-p)/N} \leq 10\%p$. If we use the importance sampling method, the rare event becomes a common event of the stochastic model. A random sample with a much smaller size is needed to guarantee the same accuracy.

One of the commonly used families of manipulated probability distributions is the exponential shifts family. We have introduced the large deviation theory results of the path space of the stochastic model in Chapter 4. The exponential shift with the large deviation minimizer is the optimal manipulated distribution among the exponential shift families. We study this importance sampling technique in details in this chapter. We first compute the Cramer transforms of Poisson distribution and multinomial distribution by following the definitions in book [2].

7.2 Cramer Transform of Poisson Distribution and Multinomial Distribution

Definition 7.2.1. μ is a probability density function on $\mathbb{R}$. $\hat{\mu} : \mathbb{R} \to [0, \infty]$ defined as

$$\hat{\mu}(t) = \int_{\mathbb{R}} e^{tx} \mu(x) dx.$$

is the Laplace transform of μ.

Definition 7.2.2. Define the Cramer transform of the measure μ in dimension 1 as $\lambda : \mathbb{R} \to [0, \infty]$

$$\lambda(x) = \sup_{t \in \mathbb{R}} [tx - \log \hat{\mu}(t)] \tag{7.1}$$

for all $x \in \mathbb{R}$.

Let X be a random variable following the Poisson distribution μ_{poi} with mean a. Hence, $\mu_{poi}(r) = a^r \times e^{-a}/r!$. The Laplace transform of μ_{poi} is

$$\hat{\mu}_{poi}(t) = \sum_{k=0}^{\infty} e^{tk} a^k e^{-a}/k! = e^{-a} \sum_{k=0}^{\infty} (e^t a)^k/k! = e^{a(e^t - 1)}. \tag{7.2}$$

By Definition 7.2.2, we compute the Cramer transform of Poisson distribution

$$\lambda_{Poi}(x) = \sup_{t \in \mathbb{R}} (tx - a(e^t - 1)) = x \log(x/a) - x + a, \tag{7.3}$$

which is also called the large deviations rate function for the probability μ_{poi}.

Definition 7.2.3. $\boldsymbol{\mu}$ is a probability density on $\mathbb{R}^k$. $\hat{\boldsymbol{\mu}} : \mathbb{R}^k \to [0, \infty]$ defined by

$$\hat{\boldsymbol{\mu}}(t) = \int_{\mathbb{R}^k} e^{\langle x, t \rangle} \boldsymbol{\mu}(x) dx.$$

is the Laplace transform of $\boldsymbol{\mu}$.

Definition 7.2.4. Define the Cramer transform of the measure $\boldsymbol{\mu}$ in dimension k as, $\boldsymbol{\lambda} : \mathbb{R}^k \to [0, \infty]$

$$\boldsymbol{\lambda}(x) = \sup_{t \in \mathbb{R}^k} [\langle x, t \rangle - \log \hat{\boldsymbol{\mu}}(t)] \tag{7.4}$$

for all $x \in \mathbb{R}^k$.

Fix any $p = [p_1, ..., p_k]$ with $p_j \geq 0$ and $p_1 + ... + p_k = 1$. For a random vector $Y = (Y_1, Y_2, ..., Y_k)$ following the multinomial distribution $\boldsymbol{\mu}_{mul} = \mu_{n,[p_1,...,p_k]}$, we have for any $y = (y_1, ..., y_k)$,

$$\boldsymbol{\mu}_{mul}(y) = \mu_{n,[p_1,...,p_k]}(y)$$

$$= \frac{n!}{y_1! y_2! ... y_k!} p_1^{y_1} p_2^{y_2} ... p_k^{y_k},$$

and the Laplace transform of multinomial probability distribution $\boldsymbol{\mu}_{mul}$ is

$$\hat{\boldsymbol{\mu}}_{mul}(t) = (\sum_{i=1}^{k} p_i e^{t_i})^n. \tag{7.5}$$

Thus, $\log \hat{\boldsymbol{\mu}}_{mul} = n \log(p_1 e^{t_1} + ... + p_k e^{t_k})$. The Cramer transform of multinomial distribution is

$$\boldsymbol{\lambda}_{mul}(x) = \sup_t [\langle t, y \rangle - \log \hat{\boldsymbol{\mu}}_{mul}(t)] = \sum_{i=1}^{k} y_i \log(y_i / p_i) - n \log n. \tag{7.6}$$

7.3 Exponential Shift Distribution of the Poisson Distribution

In Section 7.2, we computed the Cramer transform of Poisson distribution as formula (7.3). Recall that the Laplace transform of Poisson distribution is $\hat{\mu}_{poi}(t) = \sum_{n=0}^{\infty} e^{tn} \mu_{poi}(n)$. The Cramer transform of Poisson distribution is obtained when $t^* = \log(x/a)$, which is the solution of

$$\frac{\partial}{\partial t}(tx - \log \hat{\mu}_{poi}(t)) = x - \frac{\hat{\mu}'_{poi}(t)}{\hat{\mu}_{poi}(t)} = 0.$$

Define a function $\gamma_t(n) = \frac{e^{tn}\mu_{poi}(n)}{\hat{\mu}_{poi}(t)}$. Since $\sum_{n=0}^{\infty}\gamma_t(n) = 1$, γ_t is a probability mass function. Moreover, for $t^* = \log\frac{x}{a_1}$,

$$\gamma_{poi}(n) = \gamma_{t^*}(n) = \frac{e^{tn}\mu_{poi}(n)}{\hat{\mu}_{poi}(t)} = \frac{x^n}{e^x n!},$$

which is the probability mass function of Poisson distribution with mean x. γ_{poi} is the optimal exponential shift distribution of the Poisson distribution μ_{poi}. Recall that $\mathbb{E}_{\mu_{poi}}(X) = a$. If we change the distribution of random variable X from μ_{poi} to γ_{poi}, we have

$$\mathbb{E}_{\gamma_{poi}}(X) = \sum_n n \cdot \gamma_{poi}(n) = \frac{\hat{\mu}'_{poi}}{\hat{\mu}_{poi}} = x,$$

and the corrective weight,

$$W_{poi}(k) = \frac{\mu_{poi}(k)}{\frac{e^{tk}\mu_{poi}(k)}{\hat{\mu}_{poi}(t)}} = \hat{\mu}_{poi}(t)e^{-tk} = (\frac{a}{x})^k \exp(x-a). \tag{7.7}$$

In conclusion, if we want to sample a Poisson variable $X \sim \mu_{poi}$ to be around $x \neq \mathbb{E}_{\mu_{poi}}(X)$, we can change the probability mass function of X to γ_{poi}. By this change, we will be able to sample a value very close to x easily by following the distribution γ_{poi}, since $\mathbb{E}_{\gamma_{poi}}(X) = x$.

7.4 Exponential Shift Distribution of the Multinomial Distribution

Similar as above, let Y follow the multinomial distribution $\boldsymbol{\mu}_{mul}(y) = \mu_{n,p}(y)$. The Laplace transform of this multinomial distribution $\boldsymbol{\mu}_{mul}$ is

$$\hat{\boldsymbol{\mu}}_{mul}(t) = \sum_y e^{\langle y,t\rangle}\boldsymbol{\mu}_{mul}(y) = (\sum_{i=1}^{k} p_i e^{t_i})^n.$$

The Cramer transform of multinomial distribution is obtained when $t_i^* = \log \frac{y_i}{p_i}$, by solving

$$\frac{\partial}{\partial t_i}(\langle t, y\rangle - \log \hat{\boldsymbol{\mu}}_{mul}(t))) = y_i - \frac{\partial \hat{\boldsymbol{\mu}}_{mul}}{\partial t_i}\frac{1}{\hat{\boldsymbol{\mu}}_{mul}} = 0.$$

Let

$$\begin{aligned}\boldsymbol{\gamma}_{mul}(z = (z_1, ..., z_k)) &= \frac{e^{\langle \boldsymbol{t}^*, z\rangle}\boldsymbol{\mu}_{mul}(z)}{\hat{\boldsymbol{\mu}}_{\boldsymbol{mul}}(\boldsymbol{t}^*)} \\ &= \frac{e^{t_1^* z_1 + ... + t_k^* z_k}\boldsymbol{\mu}_{mul}(z_1, ..., z_k)}{\hat{\boldsymbol{\mu}}_{mul}(\boldsymbol{t}^*)} \\ &= \frac{n!}{z_1!...z_k!}(\frac{y_1}{n})^{z_1}...(\frac{y_k}{n})^{z_k}.\end{aligned}$$

$\boldsymbol{\gamma}_{mul}$ is the optimal exponential shift distribution of the multinomial distribution $\boldsymbol{\mu}_{mul}$ and it is also a multinomial distribution. If we change the probability mass function of Y from $\boldsymbol{\mu}_{mul}$ to $\boldsymbol{\gamma}_{mul}$, we have $\mathbb{E}_{\boldsymbol{\gamma_{mul}}}(Y_i) = \frac{y_i}{n}$ for $i = 1, ..., k$. Suppose we sample $Y = z$ by using the distribution $\boldsymbol{\gamma}_{mul}$, we have the following corrective weight

$$W_{mul}(z) = \frac{\boldsymbol{\mu}_{mul}(z)}{\boldsymbol{\gamma}_{mul}(z)} = \hat{\boldsymbol{\mu}}_{mul}(\boldsymbol{t}^*)e^{-\langle z, \hat{\boldsymbol{t}}^*\rangle} = n^n(\frac{p_1}{y_1})^{z_1}...(\frac{p_k}{y_k})^{z_k}. \tag{7.8}$$

In conclusion, if we want to sample a multinomial random variable Y to be close to $y \neq np$, we can change the probability mass function of Y to $\boldsymbol{\gamma}_{mul}$. By this change, $\mathbb{E}_{\boldsymbol{\gamma}_{poi}}(Y) = y$ and we will be able to sample a value very close to y.

7.5 Forced Simulation of One-step Transition

We have introduced the optimal exponential shift distributions of the Poisson distribution and the multinomial distribution. We now introduce the forced simulation of the one-step transition from H to G by using these two exponential shift distributions.

On day n, we start with histogram $H_n = H$ and we want to force $H_{n+1} = G$ for day $n + 1$.

However, $\mathbb{P}(H_{n+1} = G|H_n = H)$ is a very small probability if G is not close to $\mathbb{E}(H_{n+1}|H_n = H)$. Therefore, we implement the following forced simulation algorithm.

Algorithm 7.1: Forced Simulation of One-step Transition

Step 1. We start with the histogram $H_n = H$ and population size N. The population first grows with the growth factor $F = [F(1), ..., F(g)]$ deterministically. Compute the sizes of the g colonies of cells as $siz = (siz(1), ..., siz(g))$ with $siz(i) = NH(i)F(i)$ for $i = 1, ..., g$.

Step 2. Compute the optimal mutation matrix R^* from $H_n = H$ to $H_{n+1} = G$ by using the following formula,

$$R^*_{j,k} = [r^*_{j,k}N] = [mNF_jH(j)Q_{j,k}U_k/U_j], \tag{7.9}$$

where $U_j = \exp(\frac{G_j}{F_jH(j)})$. Next, we want to force the random mutation matrix R_n of day n to be close to R^*.

Step 3. $R_n = (R_{j,k})$ is the random mutation matrix on day n. $R_{j,k}$ is the number of mutations from j-cells to k-cells. p_λ is the probability mass function of Poisson distribution with mean λ. Recall that $R_{j,k}$ follows the Poisson distribution $p_{NL_{j,k}}$ with mean

$$NL_{j,k} = m \cdot siz(j)Q_{j,k} = mNH_jF_jQ_{j,k}.$$

In order to force $R_{j,k}$ to be close to $R^*_{j,k}$, we change the probability distribution of $R_{j,k}$ to $p_{R^*_{j,k}}$. Suppose we sample $R_{j,k} = z_{j,k} = Nr_{j,k}$, the corrective weight for mutation $W_{poi}(z_{j,k})$ is

$$W_{poi}(z_{j,k}) = \frac{p_{NL_{j,k}}(Nr_{j,k})}{p_{Nr^*_{j,k}}(Nr_{j,k})} = (\frac{L_{j,k}}{r^*_{j,k}})^{z_{j,k}} \exp(N(r^*_{j,k} - L_{j,k})). \tag{7.10}$$

We compute

$$\frac{1}{N} \log W_{poi}(z_{j,k}) = (r_{j,k} \log \frac{L_{j,k}}{r^*_{j,k}}) + (r^*_{j,k} - L_{j,k}). \tag{7.11}$$

Let P_A denote a probability mass function defined as

$$P_A(R) = \mathbb{P}(R_n = R|H_n = H),$$

where $A = (A_{i,j})_{g\times g} \in \mathbb{R}^{g\times g}$, $A_{i,i} = 0$ and $R_{j,k}$ follows $p_{A_{j,k}}$. Denote $Siz = (NL_{j,k})_{g\times g}$. The corrective weight for mutation $W_{mut}(z)$ for the mutation stage is

$$W_{mut}(z) = \frac{P_{Siz}(z)}{P_{R^*}(z)} = \prod_{j,k|j\neq k} W_{poi}(z_{j,k}). \tag{7.12}$$

We also know

$$\frac{1}{N}\log P_{Siz}(z) = \frac{1}{N}\log P_{R^*}(z) + \sum_{j,k|j\neq k}(r_{j,k}\log\frac{L_{j,k}}{r^*_{j,k}} + r^*_{j,k} - L_{j,k}). \tag{7.13}$$

Step 4. After the random mutations of step 3, the population histogram is transformed to $J = (J_1, ..., J_g)$ given by the formula

$$J_j = \frac{NF_jH_j - \sum_k(R_{j,k} - R_{k,j})}{NF_jH_j}.$$

We need to randomly select N cells from the population after growth and mutations. Let $Y = [Y_1, Y_2, ..., Y_g]$ be a random vector where Y_j is the random number of j-cells being selected. Recall that Y follows the multinomial distribution $\mu_{N,J}$ and $\mathbb{E}_{\mu_{N,J}}(Y_i/N) = J_i$. We want Y/N to be close to $G \neq J$. So we change the distribution of Y to $\mu_{N,G}$. We sample $Y = x = [x_1, ..., x_g]$ for the selection stage, the actual histogram we get for day $n+1$ is $H_{n+1} = x/N$. Compute the following corrective weights for selection W_{sel} for the selection stage by formula (7.8) as

$$W_{sel}(x/N) = \frac{\mu_{N,J}(x)}{\mu_{N,G}(x)} = N^N(\frac{J_1}{NG_1})^{x_1}...(\frac{J_g}{NG_g})^{x_g} = \prod_j(\frac{J_j}{G_j})^{x_j}. \tag{7.14}$$

Thus, let $G' = x/N$, we have

$$\frac{1}{N}\log\mathbb{P}(H_{n+1} = G'|R_n = z, H_n = H) = \frac{1}{N}\log\mu_{N,J}(x)$$

$$= \frac{1}{N}\log\mu_{N,G}(x) + \frac{1}{N}\log W_{sel}(x) \tag{7.15}$$

Step 5. The corrective weight for one-step transition $H_n = H \to R_n = z \to H_{n+1} = G' = x/N$ is

$$W_{step}(H, z, G') = \frac{P_{Siz}(z)}{P_{R^*}(z)} \cdot \frac{\mu_{N,J}(x)}{\mu_{N,G}(x)} = W_{mut}(z) \cdot W_{sel}(G').$$

Since $W_{step}(H, z, G')$ takes extremely small values for large N, we compute

$$LW(H, z, G') = \frac{1}{N} \log(W_{step}(H, z, G'))$$

$$= \sum_j G'_j \log(\frac{J_j}{G_j}) + \sum_{j,k|j\neq k} (r_{j,k} \log \frac{L_{j,k}}{r^*_{j,k}} + r^*_{j,k} - L_{j,k}). \tag{7.16}$$

We formulate the extremely small probability $\mathbb{P}(H_{n+1} = G', R_n = z|H_n = H)$ as

$$\mathbb{P}(H_{n+1} = G', R_n = z|H_n = H) = P_{R^*}(z) \cdot \mu_{N,G}(G'N) \cdot \exp(N \cdot LW(H, z, G')).$$

We obtain the following lemma from the formulations of Algorithm 7.1.

Proposition 7.5.1. *Let N be the size of the population. p_λ is the probability mass function of the Poisson distribution with mean λ. For any $H, G \in \mathcal{H}_N$ and any $R/N = r \in K_N(H)$, we can formulate the probability $\mathbb{P}(H_{n+1} = G, R_n = R|H_n)$ as*

$$\mathbb{P}(H_{n+1} = G, R_n = R|H_n) = P_{R^*}(R) \cdot \mu_{N,G}(NG) \cdot \exp(N \cdot LW(H, R, G)).$$

where

$$LW(H, rN, G) = \sum_j G(j) \log(\frac{J(j)}{G(j)}) + \sum_{j,k|j\neq k} \left(r_{j,k}(\frac{G(j)}{F_j H(j)} - \frac{G(k)}{F_k H(k)}) + r^*_{j,k} - F_j H(j) M_{j,k} \right),$$

$$J_j = \frac{[NF_j H(j)] - \sum_k (R_{j,k} - R_{k,j})}{[NF_j H(j)]},$$

$$r^*_{j,k} = mF_jH(j)Q_{j,k}U_k/U_j,$$

$$U_j = \exp(\frac{G(j)}{F_jH(j)}),$$

and $P_{R^*}(R) = \prod_{j \neq k} p_{R^*_{j,k}}(R_{j,k})$.

Chapter 8

Estimation of the One-step Transition Kernel

8.1 Estimator of the One-step Transition Kernel

Recall the probability of one step transition from H to G is defined as

$$Q(H,G) = \mathbb{P}(H_{n+1} = G | H_n = H) = \sum_{R/N \in K_N(H)} \mathbb{P}(H_{n+1} = G, R_n = R | H_n = H).$$

If we use the forced simulation distribution, we can formulate $Q(H,G)$ as

$$Q(H,G) = \sum_{R/N \in K_N(H)} P_{R^*}(R) \cdot \mu_{N,G}(NG) \cdot \exp(N \cdot LW(H,R,G))$$

$$= \mu_{N,G}(GN) \sum_{R/N \in K_N(H)} \exp(N \cdot LW(H,R,G)) P_{R^*}(R).$$

Therefore,

$$Q(H,G) = \mu_{N,G}(GN) \mathbb{E}_{P_{R^*}}(X(R_n)).$$

where

$$X(R_n) = \exp(N \cdot LW(H, R_n, G))$$

and $P_{R^*}(R) = \prod_{j \neq k} p_{R^*_{j,k}}(R_{j,k})$.

Since $X(R_n) \geq 0$ and $\mu_{N,G}(GN) > 0$,

$$\mathbb{E}_{R^*}(|X(R_n)|) = \mathbb{E}_{R^*}(X(R_n)) = \frac{Q(H,G)}{\mu_{N,G}(GN)} < \infty,$$

where $\mathbb{E}_{R^*}(X(R_n))$ is the expectation of $X(R_n)$ when $R_n(j,k)$ follows the Poisson distribution with mean $R^*_{j,k}$. By the strong law of large number, we obtain the following proposition.

Proposition 8.1.1. *Let N be the size of the bacterial population. For any $H, G \in \mathcal{H}_N$, let R^* be the optimal mutation matrix from H to G computed as formula (7.9). $Q(H,G)$ is the one-step transition kernel from H to G. Let $R^1, ..., R^K$ be i.i.d random matrices with $R^i_{j,k}$ following the Poisson distribution with mean $R^*_{j,k}$ for $i = 1, ..., K$ and $j \neq k = 1, .., g$. Denote*

$$X_i = \exp(N \cdot LW(H, R^i, G)),$$

where $LW(H, R^i, G)$ is defined in Lemma 7.5.1. Then,

$$S_K = \frac{X_1 + ... + X_K}{K} \to \frac{Q(H,G)}{\mu_{N,G}(NG)}$$

a.s. as $K \to \infty$.

□

8.2 Algorithm for Estimating $Q(H,G)$

We estimate the one step transition kernel $Q(H,G)$ by using the following algorithm.

Algorithm 8.1: Estimating the One-step Transition kernel $Q(H,G)$

Step 1. Sample the random mutation matrices $R^1, ..., R^K$, where $R^k_{i,j}$ follows the Poisson distribution with mean $R^*_{i,j}$ for $k = 1, ..., K$, $i, j = 1, ..., g$.

Step 2. Compute $LW_i = LW(H, R^i, G)$ defined in Lemma 7.5.1 for $i = 1, ..., K$.

Step 3. Estimate $Q(H, G)$ by $Q_K(H, G)$ where

$$Q_K(H, G) = \frac{\mu_{N,G}(NG)}{K} \sum_i \exp(N \cdot LW_i). \tag{8.1}$$

Since $\exp(N \cdot LW_i)$ takes extremely small values, we use the following Algorithm 8.2 to compute the summation $\sum_i \exp(N \cdot LW_i)$.

8.3 Estimation of the Summation of Extremely Small Values

We want to estimate $\sum_{i=1}^K Z_i$, where $Z_1, ..., Z_K \geq 0$ are extremely small numbers, we implement the following algorithm.

Algorithm 8.2: Estimation of the Summation of Extremely Positive Small Values

Step 1. Compute $LZ_i = \log Z_i$ for $i = 1, ..., K$.

Step 2. Sort LZ_i in an increasing order and name it $SLZ_1 \leq SLZ_2 \leq ... \leq SLZ_K$.

Step 3. Let $a = \log 100$. Construct a series of intervals

$$(SLZ_K - j \cdot a, \quad SLZ_K - (j-1) \cdot a],$$

for $j = 1, ..., J = [\frac{SLZ_K - SLZ_1}{a}] + 1$.

Step 4. For $j = 1, ..., J = [\frac{SLZ_K - SLZ_1}{a}] + 1$, count the number of SLZ_i that belongs to the interval $(SLZ_K - j \cdot a, SLZ_K - (j-1) \cdot a]$ and name it n_j.

Step 5. We estimate the summation $\sum_i^K Z_i = \sum_i \exp(LZ_i)$ as

$$\sum_i^K Z_i \approx \exp(SLZ_K)(n_1 + \frac{n_2}{100} + ... \frac{n_J}{100^{J-1}}).$$

The accuracy of this estimation is

$$\frac{1}{100} \leq \frac{\sum_i Z_i}{\exp(SLZ_K)(n_1 + \frac{n_2}{100} + ...\frac{n_J}{100^{J-1}})} \leq 1.$$

8.4 Concentration Properties of $LW(H, R, G)$

Recall that

$$LW(H, R_n, G) = \frac{1}{N} \log(\frac{P_{Siz}(R_n)}{P_{R^*}(R_n)} \cdot \frac{\mu_{N,J}(GN)}{\mu_{N,G}(GN)}),$$

R^* is the optimal mutation matrix from H to G, J is the histogram after mutation. We will derive a concentration property of $LW(H, R, G)$ by using the Poisson tail bounds derived in [8].

Lemma 8.4.1. *Let X be a random variable following the Poisson distribution with mean λ. Then we have*

$$\mathbb{P}(X \geq z) \leq \exp(-z \log \frac{z}{\lambda} + z - \lambda), \tag{8.2}$$

for some $z > \lambda$.

Proof. For $t > 0$, we have

$$\mathbb{P}(X \geq z) = \mathbb{P}(Xt \geq zt) = \mathbb{P}(\exp(Xt) \geq \exp(zt)).$$

By Markov's inequality, we have

$$\mathbb{P}(\exp(Xt) \geq \exp(zt)) \leq \frac{\mathbb{E}(\exp(Xt))}{\exp(zt)} = \exp(\lambda(e^t - 1) - zt).$$

Since $\lambda(e^t - 1) - zt \geq z - \lambda - z \log \frac{z}{\lambda}$, we have

$$\mathbb{P}(X \geq z) \leq \exp(-z \log \frac{z}{\lambda} + z - \lambda).$$

□

We define a function $h : [-1, \infty) \to \mathbb{R}$ as

$$h(u) = 2\frac{(1+u)\log(1+u) - u}{u^2},$$

$$h(0) = 1, h(-1) = 2.$$

We can reformulate the Lemma 8.4.1 as

$$\mathbb{P}(X \geq \lambda + x) \leq \exp(-\frac{x^2}{2\lambda}h(\frac{x}{\lambda})) \tag{8.3}$$

for $x > 0$. We can also obtain a similar result for $0 < x < \lambda$ that

$$\mathbb{P}(X \leq \lambda - x) \leq \exp(-\frac{x^2}{2\lambda}h(-\frac{x}{\lambda})) \tag{8.4}$$

Lemma 8.4.2. *Let X be a random variable following the Poisson distribution with mean λ. Then we have*

$$\mathbb{P}(|X - \lambda| \geq x) \leq 2\exp(-\frac{x^2}{2(\lambda + x)}), \tag{8.5}$$

for $x > 0$.

Proof. We first prove that $h(u) \geq \frac{1}{1+u}$ for $u \geq 0$. We define a function $g(u) = (1+u)h(u)$. We compute the derivative of $g(u)$ and obtain

$$g'(u) = \frac{2}{u^3}(-2\frac{1+u}{u^3}\log(1+u) + \frac{2+u}{u^2}) \geq 0.$$

Thus,

$$g(u) \geq g(0) = h(0) = 1,$$

for $u \geq 0$. Therefore,

$$h(u) \geq \frac{1}{1+u}.$$

for $u \geq 0$. Thus,

$$\mathbb{P}(X - \lambda \geq x) \leq \exp(-\frac{x^2}{2(\lambda + x)}),$$

for $x > 0$. Similarly, we can obtain

$$\mathbb{P}(X \leq \lambda - x) \leq \exp(-\frac{x^2}{2(\lambda + x)}),$$

for $0 < x < \lambda$. Therefore, we obtain

$$\mathbb{P}(|X - \lambda| \geq x) \leq 2\exp(-\frac{x^2}{2(\lambda + x)}),$$

for $x > 0$. □

We apply Lemma 8.4.2 to the random mutation stage of our stochastic model and obtain the following result.

Lemma 8.4.3. *Let N be the size of bacterial population. For any histograms $H, G \in \mathcal{H}_N$, let $R^* = r^*N$ be the optimal mutation matrix from H to G defined as formula (7.9). Let $R_{j,k} = r_{j,k}N$ be the random number of mutations from $j-$cells to $k-$cells and it follows the Poisson distribution with mean $R^*_{j,k}$. For some constant $c > 0$, we obtain*

$$\mathbb{P}(|r_{j,k} - r^*_{j,k}| \geq cN^{\alpha-1}) < 2\exp(-\frac{1}{2}\frac{c^2N^{2\alpha}}{r^*_{j,k}N + cN^{\alpha}}). \tag{8.6}$$

for any $\alpha \in \mathbb{R}$. When $\alpha = \frac{2}{3}$ and $N \geq c^3/{r^*_{j,k}}^3$, we have

$$\mathbb{P}(|r_{j,k} - r^*_{j,k}| \geq cN^{-\frac{1}{3}}) < 2\exp(-\frac{c^2}{4}\frac{N^{\frac{1}{3}}}{r^*_{j,k}}) \tag{8.7}$$

Proof. We apply Lemma 8.4.2 to $R_{j,k}$ and select $x = cN^{\alpha}$. We obtain

$$\mathbb{P}(|r_{j,k} - r^*_{j,k}| \geq cN^{\alpha-1}) < 2\exp(-\frac{1}{2}\frac{c^2N^{2\alpha}}{r^*_{j,k}N + cN^{\alpha}}).$$

If we select $\alpha = 2/3$ and $N \geq c^3/{r^*_{j,k}}^3$, $r^*_{j,k}N + N^{\alpha} \leq 2r^*_{j,k}N$, then we obtain

$$\mathbb{P}(|r_{j,k} - r^*_{j,k}| \geq cN^{-\frac{1}{3}}) < 2\exp(-\frac{c^2}{4}\frac{N^{\frac{1}{3}}}{r^*_{j,k}}).$$

□

Lemma 8.4.4. *Let N be the size of bacterial population. For any histograms $H, G \in \mathcal{H}_N$, let $R^* = r^*N$ be the optimal mutation matrix from H to G defined as formula (7.9). Let $R_{j,k} = r_{j,k}N$ be the random number of mutations from $j-$cells to $k-$cells and it follows the Poisson distribution with mean $R^*_{j,k}$. Let X be a standard normal random variable. Then*

$$\frac{(r_{j,k} - r^*_{j,k})\sqrt{N}}{\sqrt{r^*_{j,k}}} \to X$$

in distribution as $N \to \infty$.

Proof. Since $R_{j,k}$ follows the Poisson distribution $p_{R^*_{j,k}}$, then $\mathbb{E}(r_{j,k}) = r^*_{j,k}$ and $Var(r_{j,k}) = \frac{r^*_{i,j}}{N}$. Hence, the Laplace transform of $\frac{(r_{j,k}-r^*_{j,k})\sqrt{N}}{\sqrt{r^*_{j,k}}}$ is

$$\mathbb{E}(\exp(t\sqrt{N}\frac{r_{j,k} - r^*_{j,k}}{\sqrt{r^*_{j,k}}})))$$

$$= \mathbb{E}(\exp(t\frac{R_{j,k}}{\sqrt{Nr^*_{j,k}}} - t\sqrt{r^*_{j,k}N}))$$

$$= \exp(-t\sqrt{r^*_{j,k}N})\mathbb{E}(\frac{t}{\sqrt{Nr^*_{j,k}}}R_{j,k})$$

$$= \exp(-t\sqrt{Nr^*_{j,k}}) \exp(r^*_{j,k}N(e^{\frac{t}{\sqrt{Nr^*_{j,k}}}} - 1))$$

$$= \exp(r^*_{j,k}N(e^{\frac{t}{\sqrt{Nr^*_{j,k}}}} - \frac{t}{\sqrt{Nr^*_{j,k}}} - 1))$$

$$= \exp(\frac{t^2}{2} + O(\frac{t^3}{\sqrt{Nr^*_{j,k}}}))$$

Therefore, $\frac{(r_{j,k}-r^*_{j,k})\sqrt{N}}{\sqrt{r^*_{j,k}}}$ converges to a standard normal random variable as $N \to \infty$. □

Lemma 8.4.5. *Let N be the size of bacterial population. For any histograms $H, G \in \mathcal{H}_N$, let $R^* = r^*N$ be the optimal mutation matrix from H to G defined as formula (7.9). Let $R_{j,k} = r_{j,k}N$ be the random number of mutations from $j-$cells to $k-$cells and it follows the Poisson distribution $p_{R^*_{j,k}}$. J_j is the percentage of the $j-$cells after mutation. Then for some small constant ϵ, we have*

$$\mathbb{P}(J_j < \epsilon) \leq \sum_{k \neq s} \exp\left(-\frac{(C - r^*_{j,k}N)^2}{2C}\right)$$

where $C = (1-\epsilon)\frac{[NF_jH(j)]}{g-1}$.

Proof. Recall that

$$J_j = \frac{[NF_jH(j)] - \sum_k (R_{j,k} - R_{k,j})}{[NF_jH(j)]}.$$

Thus, we have

$$\mathbb{P}(J_j < \epsilon) = \mathbb{P}\left(\sum_{k \neq j}(R_{j,k} - R_{k,j}) \geq (1-\epsilon)[NF_jH(j)]\right)$$

$$\leq \mathbb{P}(\text{at least one } R_{j,k} \geq \frac{(1-\epsilon)[NF_jH(j)]}{g-1})$$

$$\leq \sum_{k \neq j} \mathbb{P}(R_{j,k} \geq \frac{(1-\epsilon)[NF_jH(j)]}{g-1}))$$

$$\leq \sum_{k \neq j} \exp\left(-\frac{(C - r^*_{j,k}N)^2}{2C}\right) = O(\exp(-N)),$$

where $C = \frac{(1-\epsilon)[NF_jH(j)]}{g-1}$. □

From Lemma 8.4.5, we know that the probability of $J_j < \epsilon$ is of the order $\exp(-N)$. This probability is much smaller than the probability of the rare events we are interested in.

Recall the corrective weight of the transition $H \to R = rN \to G$ is

$$LW(H, rN, G) = \sum_j G(j) \log\left(\frac{J(j)}{G(j)}\right) + \sum_{j,k|j\neq k}\left(r_{j,k}(\frac{G(j)}{F_jH(j)} - \frac{G_k}{F_kH(k)}) + r^*_{j,k} - F_jH(j)M_{j,k}\right).$$

Let R^* be the optimal mutation matrix from H to G. When $R = R^*$, we have

$$LW(H, R^*, G) = \sum_j G(j) \log \frac{V_j}{G(j)} + \sum_{j\neq k}(r^*_{j,k}(h_{j,k}+1) - F_jH(j)M_{j,k}), \tag{8.8}$$

where

$$V_j = \frac{F_jH(j)}{\langle F, H\rangle} + \frac{1}{\langle F, H\rangle}(\sum_k (r^*_{k,j} - r^*_{j,k})), \tag{8.9}$$

and

$$h_{j,k} = \frac{G(j)}{F_jH(j)} - \frac{G(k)}{F_kH(k)}. \tag{8.10}$$

We denote $LW^*(H, G) = LW(H, R^*, G)$. When we fix the histogram H and G, we want to study the difference between the random variable $LW(H, R, G)$ and constant $LW^*(H, G)$. To study this, we first reformulate $LW(H, R, G)$ as the following proposition.

Proposition 8.4.6. *Let N be the population size. For any histograms $H, G \in \mathcal{H}_N$, let $R^* = r^*N$ be the optimal mutation matrix from H to G. Let $R = rN = (R_{j,k})_{g\times g}$ be the random mutation matrix where $R_{j,k}$ follows the Poisson distribution with mean $R^*_{j,k}$. Recall that $LW(H, R, G)$ is defined as in Lemma 7.5.1. $LW^*(H, G)$ is defined as formula (8.8). V_j is defined as formula (8.9).*

Then we have

$$LW(H, rN, G) = LW^*(H, G) + \sum_{j \neq k} A_{j,k} t_{j,k} + Rem, \tag{8.11}$$

where

$$t_{j,k} = r_{j,k} - r^*_{j,k}, \tag{8.12}$$

$$A_{j,k} = \frac{G(j)}{F_j H(j)} - \frac{G(k)}{F_k H(k)} - \frac{G(j)}{V_j \langle F, H \rangle} + \frac{G(k)}{V_k \langle F, H \rangle}, \tag{8.13}$$

$$Rem = \sum_j G(j) rem_j, \tag{8.14}$$

$$rem_j = \log\left(1 + \frac{\sum_k (t_{k,j} - t_{j,k})}{V_j \langle F, H \rangle}\right) - \frac{\sum_k (t_{k,j} - t_{j,k})}{V_j \langle F, H \rangle}. \tag{8.15}$$

Proof. Recall that

$$\begin{aligned} J(j) &= \frac{F_j H(j) - \sum_k (r_{j,k} - r_{k,j})}{F_j H(j)} \\ &= V_j + \frac{\sum_k (t_{k,j} - t_{j,k})}{\langle F, H \rangle} \\ &= V_j (1 + \frac{\sum_k (t_{k,j} - t_{j,k})}{V_j \langle F, H \rangle}). \end{aligned}$$

where $t_{j,k} = r_{j,k} - r^*_{j,k}$. Let $b_j = \frac{1}{V_j \langle F, H \rangle}$ and $z_j = \sum_k (t_{k,j} - t_{j,k})$. Then we have

$$\log J(j) = \log V_j + b_j z_j + \log(1 + b_j z_j) - b_j z_j.$$

Denote $rem_j = \log(1 + b_j z_j) - b_j z_j$. Thus, we have

$$\log J(j) = \log V_j + b_j z_j + rem_j.$$

Then we compute $LW(H, R, G) - LW^*(H, G)$ as

$$LW(H, R, G) - LW^*(H, G)$$

$$= \sum_j G(j)(\log J(j) - \log V_j) + \sum_{j \neq k} h_{j,k}(r_{j,k} - r^*_{j,k})$$

$$= \sum_j G(j)(\frac{\sum_k (t_{k,j} - t_{j,k})}{V_j \langle F, H \rangle} + rem_j) + \sum_{j \neq k} h_{j,k} t_{j,k}$$

$$= \sum_{j, \neq k} t_{j,k}(\frac{G(k)}{V_k \langle F, H \rangle} - \frac{G(j)}{V_j \langle F, H \rangle} + h_{j,k}) + \sum_j G(j) rem_j$$

where $h_{j,k}$ is defined as formula (8.10). □

We now analyze the term $\sum_{j \neq k} A_{j,k} t_{j,k}$ and the term Rem in the following lemmas.

Lemma 8.4.7. *Let N be the population size. For any histograms $H, G \in \mathcal{H}_N$, let $R^* = r^* N$ be the optimal mutation matrix from H to G. Let $R = rN = (R_{j,k})_{g \times g}$ be the random mutation matrix where $R_{j,k}$ follows the Poisson distribution with mean $R^*_{j,k}$. $A_{j,k}$ is defined as formula (8.13) and $t_{j,k} = r_{j,k} - r^*_{j,k}$ for $j, k = 1, ..., g$. For some constant $c > 0$, we obtain*

$$\mathbb{P}(|\sum_{j,k} A_{j,k} t_{j,k}| \geq c N^{-\frac{1}{3}}) \leq 2g(g-1) \exp(-\frac{C^2}{4} \frac{N^{1/3}}{r^*}), \tag{8.16}$$

where $C = \min_{j,k} \frac{c}{g(g-1)|A_{j,k}|}$ and $r^* = \max_{j,k} r^*_{j,k}$. We also have

$$A_{j,k} = \frac{\sum_k (r^*_{k,j} - r^*_{j,k})}{(F_j H(j))^2} - \frac{\sum_j (r^*_{j,k} - r^*_{k,j})}{(F_k H(k))^2} + O(m^2).$$

Proof. For some constant $c > 0$, we have

$$\mathbb{P}(|\sum_{j,k} A_{j,k} t_{j,k}| \geq c N^{-1/3})$$

$$\leq \mathbb{P}(\sum_{j,k} |A_{j,k} t_{j,k}| \geq c N^{-1/3})$$

$$\leq \mathbb{P}(\text{at least one } |A_{j,k} t_{j,k}| \text{ is larger than } \frac{c N^{-1/3}}{g(g-1)})$$

$$\leq \sum_{j,k} \mathbb{P}(|t_{j,k}| \geq \frac{cN^{-1/3}}{g(g-1)|A_{j,k}|})$$

$$\leq \sum_{j,k} \mathbb{P}(|t_{j,k}| \geq CN^{-1/3})$$

where $C = \min_{j,k} \frac{c}{g(g-1)|A_{j,k}|}$. Denote $r^* = \max_{j,k} r^*_{j,k}$. We have

$$\mathbb{P}(|\sum_{j,k} A_{j,k}t_{j,k}| \geq cN^{-1/3})$$

$$\leq \sum_{j,k} 2\exp(-\frac{C^2}{4}\frac{N^{1/3}}{r^*_{j,k}})$$

$$\leq 2g(g-1)\exp(-\frac{C^2}{4}\frac{N^{1/3}}{r^*}).$$

Recall that $A_{j,k} = h_{j,k} - \frac{G(j)}{V_j\langle F,H\rangle} + \frac{G(k)}{V_k\langle F,H\rangle}$. Since

$$\frac{1}{\langle F,H\rangle V_j} = \frac{1}{F_jH(j) + \sum_k (r^*_{k,j} - r^*_{j,k})},$$

$$= \frac{1}{F_jH(j)} - \frac{\sum_k (r^*_{k,j} - r^*_{j,k})}{(F_jH(j))^2} + O(m^2),$$

then

$$A_{j,k} = h_{j,k} - \frac{G(j)}{V_j\langle F,H\rangle} + \frac{G(k)}{V_k\langle F,H\rangle}$$

$$= \frac{\sum_k G(j)(r^*_{k,j} - r^*_{j,k})}{(F_jH(j))^2} - \frac{\sum_j G(k)(r^*_{j,k} - r^*_{k,j})}{(F_kH(k))^2} + O(m^2)$$

□

Lemma 8.4.8. *We use the same conditions in Lemma 8.4.7. Denote*

$$T(H,R,G) = \sqrt{N}\sum_{j,k} A_{j,k}t_{j,k}. \tag{8.17}$$

Then we have $T(H,R,G) \to Y$ in distribution as $N \to \infty$, where Y follows $N(0,\sigma^2)$ and

$$\sigma^2(H,G) = \sum_{j,k} A_{j,k}^2 r_{j,k}^* = O(m^3). \tag{8.18}$$

Proof. The Laplace transform of $T(H,R,G)$ is

$$\mathbb{E}(\exp(t \cdot T(H,R,G)))$$

$$= \mathbb{E}(\exp(t\sqrt{N} \sum_{j \neq k} A_{j,k} t_{j,k}))$$

$$= \prod_{j \neq k} \mathbb{E}(\exp(t\sqrt{N} A_{j,k} t_{j,k}))$$

$$= \prod_{j \neq k} \mathbb{E}(\exp(t\sqrt{N} A_{j,k} (r_{j,k} - r_{j,k}^*)))$$

$$= \prod_{j \neq k} \exp(-t\sqrt{N} A_{j,k} r_{j,k}^*) \mathbb{E}(\exp(\frac{tA_{j,k}}{\sqrt{N}} R_{j,k}))$$

$$= \prod_{j \neq k} \exp(-t\sqrt{N} A_{j,k} r_{j,k}^*) \exp(r_{j,k}^* N \cdot (\exp(\frac{tA_{j,k}}{\sqrt{N}} - 1)))$$

$$= \prod_{j \neq k} \exp(-t\sqrt{N} A_{j,k} r_{j,k}^* + r_{j,k}^* N \exp(\frac{tA_{j,k}}{\sqrt{N}}) - r_{j,k}^* N)$$

$$= \prod_{j \neq k} \exp(r_{j,k}^* N (\exp(\frac{tA_{j,k}}{\sqrt{N}}) - \frac{tA_{j,k}}{\sqrt{N}} - 1))$$

$$= \prod_{j \neq k} \exp(r_{j,k}^* N (\frac{t^2 A_{j,k}^2}{2N} + O(\frac{t^3 A_{j,k}^3}{6(\sqrt{N})^3})))$$

$$= \prod_{j \neq k} \exp(\frac{t^2 A_{j,k}^2 r_{j,k}^*}{2} + O(\frac{t^3 A_{j,k}^3 r_{j,k}^*}{\sqrt{N}}))$$

$$= \exp(\frac{t^2}{2} (\sum_{j \neq k} A_{j,k}^2 r_{j,k}^*) + O(\frac{t^3}{\sqrt{N}} \sum_{j \neq k} A_{j,k}^3 r_{j,k}^*)).$$

Therefore, $T(H,R,G)$ convergence to the normal distribution $N(0,\sigma^2)$, where

$$\sigma^2 = \sum_{j,k} A_{j,k}^2 r_{j,k}^* = O(m^3).$$

□

In the bacterial evolution stochastic model, the population size N is very large and the mutation rate m is very small. Generally, we have $N \geq 10^{12}$ and $m \leq 10^{-6}$. The variance of $T(H,R,G)$ will be of the order less $O(10^{-18})$, which is extremely small. This means that $T(H,R,G)$ will concentrate around 0. Notice the remainder term $O(\frac{t^3}{\sqrt{N}} \sum_{j\neq k} A_{j,k}^3 r_{j,k}^*)) = O(t^3 \cdot \frac{m^4}{\sqrt{N}}) \leq O(t^3 \cdot 10^{-36})$, which takes extremely small value for small t. Thus, $T(H,R,G)$ is approximately normal in this stochastic model.

Lemma 8.4.9. *We use same conditions as in Proposition 8.4.6. Recall that* $Rem = \sum_j G(j) rem_j$ *where* rem_j *is defined as formula (8.15). For some constant* $c > 0$*, we have*

$$\mathbb{P}(|Rem| \geq cN^{-\frac{2}{3}}) \leq 2g^2(g-1)\exp(-C1\frac{N^{1/3}}{r^*}), \tag{8.19}$$

where $r^* = \max_{j,k} r_{j,k}^*$ *and* $C1 = \min_j \frac{c}{16g} V_j^2 \langle F, H\rangle^2$.

Proof. Recall that $rem_j = \log\left(1 + \frac{\sum_k (t_{k,j} - t_{j,k})}{V_j\langle F,H\rangle}\right) - \frac{\sum_k (t_{k,j} - t_{j,k})}{V_j\langle F,H\rangle}$. Let

$$X_j = \frac{\sum_k (t_{k,j} - t_{j,k})}{V_j\langle F,H\rangle} = \frac{\sum_k (t_{k,j} - t_{j,k})}{F_j H(j) + \sum_k r_{k,j}^* - r_{j,k}^*}.$$

Since we have proved that $|t_{k,j}| \leq cN^{-1/3}$ with ultra high probability, we assume that $X_j \geq -0.5$. We can prove that $x^2 - x + \log(1+x) \geq 0$ and $x - \log(x+1) \geq 0$ for $x > -0.5$. Thus,

$$|rem_j| = -\log(1+X_j) + X_j \leq X_j^2,$$

for $j = 1, ..., g$. Therefore, we have

$$\mathbb{P}(|rem_j| > cN^{-2/3}) \leq \mathbb{P}(X_j^2 > cN^{-2/3}) = \mathbb{P}(|X_j| > \sqrt{c}N^{-1/3}).$$

We use Lemma 8.4.3 and obtain

$$\mathbb{P}(|rem_j| > cN^{-2/3}) \leq 2g(g-1)\exp(-C'\frac{N^{1/3}}{r^*}),$$

where $C' = \min_{j,k} \frac{cV_j^2\langle F,H\rangle^2}{16}$ and $r^* = \max_{j,k} r_{j,k}^*$. Therefore, we have

$$\mathbb{P}(|Rem| > cN^{-2/3})$$

$$\leq \mathbb{P}(\sum_j |rem_j| > cN^{-2/3})$$

$$\leq 2g^2(g-1)\exp(-C1\frac{N^{1/3}}{r^*}),$$

where $C1 = \frac{c}{16g}V_j^2\langle F,H\rangle^2$. □

We know that $A_{j,k} = O(m)$ from Lemma 8.4.7. In the following proposition, we show that we can control the term $\sum_{j,k} A_{j,k}t_{j,k}$ and the term *Rem* simultaneously.

Proposition 8.4.10. *We use same conditions as in Proposition 8.4.6. m is the mutation rate. Recall that $Rem = \sum_j G(j)rem_j$ where rem_j is defined as formula (8.15). For some constant $c > 0$ and $\alpha \in (0.5, 1)$, we are able to control*

$$\mathbb{P}(|\sum_{j,k} A_{j,k}t_{j,k}| \geq cN^{\alpha-1}) \leq C_1\exp(-\frac{N^{2\alpha-1}}{r^*}) \tag{8.20}$$

and

$$\mathbb{P}(|Rem| \geq cN^{2\alpha-2}) \geq C_2\exp(-\frac{N^{2\alpha-1}}{r^*}), \tag{8.21}$$

at the same time, where C_1, C_2 are constants depending only on c, H and G.

Proof. We can prove this Proposition by using Lemma 8.4.3, Lemma 8.4.7, and Lemma 8.4.9. □

Theorem 8.4.11. *Let N be the population size. For any histograms $H, G \in \mathcal{H}_N$, let $R^* = r^*N$ be the optimal mutation matrix from H to G. Let $R = rN = (R_{j,k})_{g\times g}$ be the random mutation matrix where $R_{j,k}$ follows the Poisson distribution with mean $R^*_{j,k}$. Recall that $LW(H,R,G)$ is defined as in Lemma 7.5.1. $LW^*(H,G)$ is defined as formula (8.8). Then for some constant $c > 0$ and $\alpha \in (0.5, 1)$, we obtain*

$$\mathbb{P}(|LW(H,R,G) - LW^*(H,G)| \geq cN^{\alpha-1} + cN^{2\alpha-2}) \leq C' \exp(-\frac{N^{2\alpha-1}}{r^*}),$$

for $N \geq O(\max_{j,k} {r^*_{j,k}}^{\frac{1}{\alpha-1}})$, where C' depends only on c, H and G, $r^* = \max_{j,k} r^*_{j,k}$.

Proof. We know that

$$LW(H, rN, G) = LW^*(H,G) + \sum_{j\neq k} A_{j,k}t_{j,k} + Rem,$$

Thus, $LW(H,R,G) - LW^*(H,G) = \sum_{j\neq k} A_{j,k}t_{j,k} + Rem$. By using Proposition 8.4.10, we have

$$\mathbb{P}(|\sum_{j\neq k} A_{j,k}t_{j,k} + Rem| \geq cN^{\alpha-1} + cN^{2\alpha-2})$$

$$\leq \mathbb{P}(|\sum_{j\neq k} A_{j,k}t_{j,k}| + |Rem| \geq cN^{\alpha-1})$$

$$\leq \mathbb{P}(|\sum_{j\neq k} A_{j,k}t_{j,k}| \geq \frac{c}{2}N^{\alpha-1}) + \mathbb{P}(|Rem| \geq \frac{c}{2}N^{\alpha-1})$$

$$\leq \mathbb{P}(|\sum_{j\neq k} A_{j,k}t_{j,k}| \geq \frac{c}{2}N^{\alpha-1}) + \mathbb{P}(|Rem| \geq \frac{c}{2}N^{2\alpha-2})$$

$$\leq C' \exp(-\frac{N^{2\alpha-1}}{r^*}),$$

where $r^* = \max_{j,k} r^*_{j,k}$ for $N \geq O(\max_{j,k} {r^*_{j,k}}^{\frac{1}{\alpha-1}})$. □

8.5 Accuracy of $Q_K(H, G)$

We have shown that the estimator $Q_K(H, G) \to Q(H, G)$ almost surely as $K \to \infty$ in Proposition 8.1.1. In the following theorem , we obtain a confidence interval for $Q_K(H, G)$ by assuming that $T(H, R, G)$ follows the normal distribution $N(0, \sigma^2)$, where σ^2 is defined in formula (8.18). We first prove a lemma about the standard normal random variable.

Lemma 8.5.1. *Let $Z_1, ..., Z_k$ be the i.i.d random variables that follow the standard normal distribution $N(0, 1)$. Then for any constant $c > 0$, we have*

$$\mathbb{P}(\max_{i=1,...,k} |Z_i| > c) \leq 2k \exp(-\frac{c^2}{2}). \tag{8.22}$$

Proof. Let $MZ = \max_{i=1,..,k} Z_i$. Then, for $c > 0$

$$\mathbb{P}(MZ > c) = \mathbb{P}(\exp(t \cdot MZ) > \exp(tc))$$

$$\leq \exp(-tc)\mathbb{E}(\exp(t \cdot MZ))$$

$$\leq \exp(-tc)\mathbb{E}(\sum_i \exp(t \cdot Z_i))$$

$$= k \exp(\frac{t^2}{2} - tc).$$

Since $\frac{t^2}{2} - tc \geq -c^2/2$,

$$\mathbb{P}(\max_i Z_i > c) \leq k \exp(-\frac{c^2}{2}).$$

Similarly, for any $c < 0$

$$\mathbb{P}(\min_i Z_i < c) \leq k \exp(-\frac{c^2}{2}).$$

Thus, for any constant $c > 0$, we have

$$\mathbb{P}(\max_{i=1,...,k} |Z_i| > c) \leq 2k \exp(-\frac{c^2}{2}).$$

□

Theorem 8.5.2. *Let N be the population size. For any histograms $H, G \in \mathcal{H}_N$, let $R^* = r^*N$ be the optimal mutation matrix from H to G. Let $R = rN = (R_{j,k})_{g\times g}$ be the random mutation matrix where $R_{j,k}$ follows the Poisson distribution with mean $R^*_{j,k}$. $Q(H,G)$ is the one-step transition kernel from H to G. Let $R^1, ..., R^K$ be a random sample of R. Let $Q_K(H,G)$ be the estimator defined as formula (8.1). $LW^* = LW^*(H,G)$ is defined as formula (8.8). Assume that $T(H,R,G)$ defined as formula (8.18) follows the normal distribution $N(0,\sigma)$, where σ is defined as formula (8.17). Then for any constant $c_1 > 0, c_2 > 0$, we obtain that*

$$L_K(H,G) \leq \frac{Q_K(H,G)}{\mu_{N,G}(GN)} \leq U_K(H,G) \tag{8.23}$$

with probability above

$$p(c_1, c_2, H, G) = 0.99 p1(c_1, H, G)^K p2(c_2) \tag{8.24}$$

for $K \geq \max\{10^3, 9N\sigma^2\}$, where

$$U_K(H,G) = \exp(N \cdot LW^* + c_1 N^{1/3})(1 + \frac{3\sqrt{N}\sigma}{\sqrt{K}} + N\sigma^2 \exp(c_2\sqrt{N}\sigma)), \tag{8.25}$$

$$L_K(H,G) = \exp(N \cdot LW^* - c_1 N^{1/3})(1 - \frac{3\sqrt{N}\sigma}{\sqrt{K}}), \tag{8.26}$$

$$p1(c_1, H, G) = 1 - 2g^2(g-1)\exp(-C1\frac{N^{1/3}}{r^*}), \tag{8.27}$$

$$C1 = \min_j \frac{c_1}{16g} V_j^2 \langle F, H\rangle^2,$$

$$p(c_2) = 1 - 2K\exp(-\frac{c_2^2}{2}). \tag{8.28}$$

Proof. For every R^i, we compute $LW_i = LW(H, R^i, G)$ defined in Lemma 7.5.1 for $i = 1, ..., K$ and we have

$$Q_K(H,G) = \mu_{N,G}(GN) \sum_i \frac{\exp(N \cdot LW_i)}{K}.$$

By our assumption, $\sqrt{N}(LW_i - LW^*(H,G)) = \sigma Z_i + \sqrt{N}Rem_i$, where Z_i follows the normal distribution $N(0,1)$. Thus,

$$\frac{Q_K(H,G)}{\mu_{N,G}(GN)} = \sum_i \exp(N \cdot LW_i)/K$$

$$= \exp(N \cdot LW^*) \sum_i \frac{\exp(\sqrt{N}\sigma Z_i)\exp(N \cdot Rem_i)}{K}.$$

Recall that

$$\mathbb{P}(|Rem| \geq cN^{-\frac{2}{3}}) \leq 2g^2(g-1)\exp(-C1\frac{N^{1/3}}{r^*}),$$

where $r^* = \max_{j,k} r^*_{j,k}$ and $C1 = \min_j \frac{c}{16g} V_j^2 \langle F, H\rangle^2$. Denote

$$p1(c) = 1 - 2g^2(g-1)\exp(-C1\frac{N^{1/3}}{r^*}).$$

where $r^* = \max_{j,k} r^*_{j,k}$ and $C1 = \min_j \frac{c}{16g} V_j^2 \langle F, H\rangle^2$. Thus, for some constant $c_1 > 0$, we have

$$\mathbb{P}(\max_i |Rem_i| \leq c_1 N^{-\frac{2}{3}}) \geq p1(c_1)^K.$$

Therefore, we have

$$\exp(N \cdot LW^* - c1N^{1/3}) \sum_i \frac{\exp(\sqrt{N}\sigma Z_i)}{K} \leq \frac{Q_K(H,G)}{\mu_{N,G}(GN)}$$
$$\leq \exp(N \cdot LW^* + c1N^{1/3}) \sum_i \frac{\exp(\sqrt{N}\sigma Z_i)}{K} \quad (8.29)$$

with probability above $p1(c_1)^K$. We now analyze the term $\sum_i \frac{\exp(N\sigma Z_i)}{K}$. By using the Taylor expansion of $\exp(\sqrt{N}\sigma Z_i)$, we get

$$\sum_i \exp(\sqrt{N}\sigma Z_i)/K = 1 + \sqrt{N}\sigma \sum_i \frac{Z_i}{K} + N\sigma^2 \sum_i \frac{Z_i^2}{K} \frac{\exp(\sqrt{N}\sigma q_i)}{2},$$

where q_i is in between 0 and Z_i. For a constant c_2, we have

$$\mathbb{P}(\max_{i=1,...,k} |Z_i| > c_2) \leq 2K \exp(-\frac{c_2^2}{2}).$$

Therefore,

$$\exp(-\sqrt{N}\sigma c_2) \leq \exp(\sqrt{N}\sigma q_i) \leq \exp(\sqrt{N}\sigma c_2),$$

holds for $i = 1, ..., K$ with probability above $p2(c_2) = 1 - 2K \exp(-\frac{c_2^2}{2})$. Notice that

$$|\frac{\sum_i Z_i}{K}| \leq \frac{3}{\sqrt{K}},$$

with probability 0.99, and

$$\mathbb{P}(\frac{\sum_i Z_i^2}{K} \leq 2) = F(2K),$$

is almost 1 for $K \geq 10^3$, where F is the cdf for chi-square distribution $\chi^2(K)$. Therefore,

$$1 - \frac{3\sqrt{N}\sigma}{\sqrt{K}} \leq \sum_i \exp(\sqrt{N}\sigma Z_i)/K \leq 1 + \frac{3\sqrt{N}\sigma}{\sqrt{K}} + N\sigma^2 \exp(c_2\sqrt{N}\sigma), \tag{8.30}$$

with probability above $0.99p2(c_2)$. Combine inequality 8.29 and inequality 8.30, we obtain

$$\begin{aligned} \exp(N \cdot LW^* - c_1 N^{1/3})(1 - \frac{3\sqrt{N}\sigma}{\sqrt{K}}) &\leq \frac{Q_K(H,G)}{\mu_{N,G}(GN)} \\ &\leq \exp(N \cdot LW^* + c_1 N^{1/3})(1 + \frac{3\sqrt{N}\sigma}{\sqrt{K}} + N\sigma^2 \exp(c_2\sqrt{N}\sigma)) \end{aligned} \tag{8.31}$$

with probability above $0.99p1(c_1)^K p2(c_2)$ for $K \geq \max\{10^3, 9N\sigma^2\}$. □

In the following corollary, we explain how to select the constant c_1 and c_2 when we fix the sample size K and the probability of the confidence interval.

Corollary 8.5.3. *Given a probability $p < 0.99$, let $p' = (\frac{p}{0.99})^{\frac{1}{K+1}}$, we can select*

$$c_1 = \frac{16gr^*}{\langle F, H\rangle^2 \min_j V_j^2 N^{\frac{1}{3}}} \log \frac{2g^2(g-1)}{1-p'} \tag{8.32}$$

$$c_2 = \sqrt{2 \log \frac{2K}{1-p'}} \tag{8.33}$$

so that

$$L_K(H, G) \leq \frac{Q_K(H, G)}{\mu_{N,G}(GN)} \leq U_K(H, G)$$

holds above probability p.

Proof. If we set $p1(c_1, H, G) = p2(c_2) = p'$, then we have $0.99p1(c_2)^K p2(c_2) = p$. By solving $p1(c_1) = p'$ and $p2(c_2) = p'$, we obtain formula (8.32) and formula (8.33). □

When we fix the sample size K and the probability of the confidence interval, we can obtain the confidence interval of the estimator $Q_K(H, G)$ by using Theorem 8.5.2 and Corollary 8.5.3 without any simulations.

8.6 Example of Estimating the $Q(H, G)$

Let $F = [200, 200^{1.08}, 200^{1.12}]$, $m = 10^{-7}$ and

$$Q = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0.5 & 0 \end{bmatrix}.$$

Let $H = [0.8, 0.1, 0.1]$ and $G = [0.7, 0.2, 0.1]$. We compute

$$\sigma^2(H, G) = 1.0750 \cdot 10^{-10}.$$

Thus, we select the sample size $K = 10^3$. By using Algorithm 8.1, we obtain

$$Q_K(H, G) = \exp(-2.9269 \cdot 10^{10}).$$

If we want to obtain a 90% confidence interval of $Q_K(H, G)$, we select

$$c_1 = 5.2931 \cdot 10^{-10},$$

and

$$c_2 = 5.8069,$$

by using Corollary 8.5.3. Then the confidence we obtain by using the Theorem 8.5.2 satisfies

$$\log \frac{Q_K(H, G)}{\mu_{N,G}(GN)} - \log L_K(H, G) = 38.3567,$$

$$\log U_K(H, G) - \log \frac{Q_K(H, G)}{\mu_{N,G}(GN)} = 30.6412.$$

Chapter 9

Estimating the Probabilities of Random Trajectories

9.1 Estimator for the Probability of a Random Trajectory

Let $\mathbf{H} = [H_0, H_1, ..., H_t] \in \Omega_{t+1}$ be a random trajectory starting from H_0. Let $R^*(i)$ be the optimal mutation matrix from H_i to H_{i+1}. We can use the forced simulation method in Chapter 8 to estimate $Q(H_i, H_{i+1})$ by $Q_K(H_i, H_{i+1})$ for every step, where

$$s_i = \sqrt{N\sigma^2(H_i, H_{i+1})} \tag{9.1}$$

$$K_i = \max\{10^3, 9s_i^2\} \tag{9.2}$$

$$K = \max_i K_i, \tag{9.3}$$

$$Q_K(H_i, H_{i+1}) = \frac{\mu_{N,H_{i+1}}(NH_{i+1})}{K} \sum_{j=1}^{K} \exp(N \cdot LW(H_i, R^j(i), H_{i+1})), \tag{9.4}$$

and $R^1(i), ..., R^K(i)$ are the random samples of $R^*(i)$. Therefore, the estimator for $\mathbb{P}(\mathbf{H}|H_0)$ is

$$P_K(\mathbf{H}) = \prod_{i=1}^{t-1} Q_K(H_i, H_{i+1}).$$

9.2 Accuracy of the Estimation

Proposition 9.2.1. *For any random trajectory $\mathbf{H} = [H_0, ..., H_t] \in \Omega_{t+1}$, let $R^*(i)$ be the optimal mutation matrix from H_i to H_{i+1} for $i = 0, ..., t-1$. $Q(H_i, H_{i+1})$ is the one-step transition kernel from H_i to H_{i+1}. Let $R^1(i), ..., R^K(i)$ be a random sample of $R^*(i)$, where K is defined as formula (9.1-9.3). Let*

$$Q_K(H_i, H_{i+1}) = \frac{\mu_{N,H_{i+1}}(NH_{i+1})}{K} \sum_{j=1}^{K} \exp(N \cdot LW(H_i, R^j(i), H_{i+1})),$$

where $LW(H_i, R^j(i), H_{i+1})$ is defined in Proposition 7.5.1. Then,

$$P_K(\mathbf{H}) := \prod_{i=0}^{t-1} Q_K(H_i, H_{i+1}) \to \mathbb{P}(\mathbf{H}|H_0)$$

a.s. as $K \to \infty$ for $i = 0, ..., t-1$.

Proof. By Proposition 8.1.1, $Q_K(H_i, H_{i+1}) \to Q(H_i, H_{i+1})$ a.s. as $K \to \infty$. Thus, $P_K(\mathbf{H}) \to \mathbb{P}(\mathbf{H}|H_0)$ a.s. as $K \to \infty$. □

We can obtain a confidence interval of $P_K(\mathbf{H})$ in the following theorem.

Theorem 9.2.2. *Given the random trajectory $\mathbf{H} = [H_0, ..., H_t] \in \Omega_{t+1}$ and population size N, $Q(H_i, H_{i+1})$ is the one-step transition kernel from H_i to H_{i+1} for $i = 0, ..., t-1$. Let $R^*(i) = r^*(i)N$ be the optimal mutation matrix from H_i to H_{i+1}. Let $R^1(i), ..., R^K(i)$ be a random sample of $R^*(i)$. $P_K(\mathbf{H})$ is the estimator for $\mathbb{P}(\mathbf{H}|H_0)$ defined in Proposition 9.2.1 . Let $s_i = \sqrt{N\sigma^2(H_i, H_{i+1})}$ where*

$\sigma^2(H_i, H_{i+1})$ *is defined as formula (8.18). Let* $K = \max_{i=0,...,t-1}\{10^3, 9s_i^2\}$. *Denote*

$$\mu(N, \mathbf{H}) = \prod_{i=0}^{t-1} \mu_{N,H_{i+1}}(NH_{i+1}). \tag{9.5}$$

We obtain that

$$\prod_{i=0}^{t-1} L_K(H_i, H_{i+1}) \leq \frac{P_K(\mathbf{H})}{\mu(N, \mathbf{H})} \leq \prod_{i=0}^{t-1} U_K(H_i, H_{i+1}) \tag{9.6}$$

with probability above

$$p(C^0, .., C^{t-1}, C_2, \mathbf{H}) = 0.99^t p_2(C_2)^t \prod_{i=0}^{t-1} p1^K(C^i, H_i, H_{i+1}) \tag{9.7}$$

where $U_K(H_i, H_{i+1})$, $L_K(H_i, H_{i+1})$, $p1(C^i, H_i, H_{i+1})$, *and* $p2(C_2)$ *are defined in formula (8.25), formula (8.26), formula (8.27), and formula (8.28).*

Proof. Since the one step transition $H_i \to H_{i+1}$ are independent for $i = 0, ..., t-1$, we can obtain this theorem by applying the Theorem 8.5.2. □

The following corollary explains how to select the constant $C^0, ..., C^{t-1}, C_2$ for Theorem 9.2.2.

Corollary 9.2.3. *Given a trajectory* $\mathbf{H} = [H_0, ..., H_t]$ *and a probability* $p < 0.99^t$, *let* $R^*(i) = r^*(i)N$ *are the optimal mutation matrix from* H_i *to* H_{i+1} *for* $i = 0, ..., t-1$. *Let* $p' = (\frac{p}{0.99})^{\frac{1}{t(K+1)}}$, *we can select*

$$C^i = \frac{16gr^*(i)}{\langle F, H_i\rangle^2 \min_j V_j^2(H_i, H_{i+1}) N^{\frac{1}{3}}} \log \frac{2g^2(g-1)}{1-p'} \tag{9.8}$$

where $r^*(i) = \max_{j,k} r^*(i)_{j,k}$,

$$V_j(i) = \frac{F_j H_i(j)}{\langle F, H_i\rangle} + \frac{1}{\langle F, H_i\rangle}(\sum_k (r^*_{k,j}(i) - r^*_{j,k}(i))), \tag{9.9}$$

for $i = 0, ..., t-1$ *and*

$$C_2 = \sqrt{2 \log \frac{2K}{1-p'}} \tag{9.10}$$

Table 9.1: Optimal Trajectory $\boldsymbol{\mathcal{G}}$

$\mathcal{G}_0$	0.8000	0.1000	0.1000
$\mathcal{G}_1$	0.6383	0.1805	0.1812
$\mathcal{G}_2$	0.4266	0.2578	0.3156
$\mathcal{G}_3$	0.2233	0.3000	0.4767

Optimal trajectory for $Eve(2, 0.3)$ given $H_0 = [0.8, 0.1, 0.1]$.

so that

$$\prod_{i=0}^{t-1} L_K(H_i, H_{i+1}) \leq \frac{P_K(\mathbf{H})}{\mu(N, \mathbf{H})} \leq \prod_{i=0}^{t-1} U_K(H_i, H_{i+1})$$

holds above probability p.

Proof. If we set $p1(C^i, H_i, H_{i+1}) = p2(C_2) = p'$, then we have

$$p = 0.99^t p2(C_2)^t \prod_{i=0}^{t-1} p_1^K(C^i, H_i, H_{i+1}).$$

By solving $p1(C^i, H_i, H_{i+1}) = p'$ and $p2(C_2) = p'$, we obtain formula (9.8) and formula (9.10). □

9.3 Examples of Estimating $\mathbb{P}(\mathbf{H}|H_0)$

Recall the example in Section 6.2.2.1, $F = [200, 200^{1.08}, 200^{1.12}]$, $m = 10^{-7}$,

$$Q = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0.5 & 0 \end{bmatrix}.$$

and $H_0 = [0.8, 0.1, 0.1]$. The optimal trajectory $\boldsymbol{\mathcal{G}}$ for realizing the event $Eve(2, 0.3)$ is in Table 9.1. We want to estimate the $\mathbb{P}(\boldsymbol{\mathcal{G}}|H_0)$ when $N = 10^{12}$.

We use Algorithm 8.1 to estimate the one-step transition kernel $Q(\mathcal{G}_i, \mathcal{G}_{i+1})$ for $i = 0, ..., 2$.

Table 9.2: Forced Simulation Results

i	$Q_K(\mathcal{G}_i, \mathcal{G}_{i+1})$	$\sigma^2(\mathcal{G}_i, \mathcal{G}_{i+1})$	C^i	C_2	$left(i)$	$right(i)$
0	$\exp(-1.7826 \times 10^{11})$	$9.9067 \cdot 10^{-11}$	$3.1518 \cdot 10^{-10}$	6.7528	32.73	189.59
1	$\exp(-1.3909 \times 10^{11})$	$5.9854 \cdot 10^{-11}$	$7.7143 \cdot 10^{-10}$	6.7528	31.45	141.48
2	$\exp(-1.3996 \times 10^{11})$	$8.4287 \cdot 10^{-11}$	$7.3950 \cdot 10^{-10}$	6.7528	32.09	172.75

$Q_K(\mathcal{G}_i, \mathcal{G}_{i+1})$ is the estimated one step transition kernel. $\sigma^2(\mathcal{G}_i, \mathcal{G}_{i+1})$ is defined as formula 8.18. C^i and C_2 are computed as in Corollary 9.2.3. $left(i)$ and $right(i)$ are computed as formula 9.11 and formula 9.12

We compute $\sigma^2(\mathcal{G}_i, \mathcal{G}_{i+1})$ and present these values in Table 9.2 and select $K = 10^4$. We present $Q_K(\mathcal{G}_i, \mathcal{G}_{i+1})$ in Table 9.2. When we select $p = 0.9$, we compute C^i and C_2 using Corollary 9.2.3 and present these values in Table 9.2. We also present

$$left(i) = \log \frac{Q_K(\mathcal{G}_i, \mathcal{G}_{i+1})}{\mu_{N,\mathcal{G}_{i+1}}(N\mathcal{G}_{i+1})} - \log L_K(\mathcal{G}_i, \mathcal{G}_{i+1}) \tag{9.11}$$

and

$$right(i) = \log U_K(\mathcal{G}_i, \mathcal{G}_{i+1}) - \log \frac{Q_K(\mathcal{G}_i, \mathcal{G}_{i+1})}{\mu_{N,\mathcal{G}_{i+1}}(N\mathcal{G}_{i+1})} \tag{9.12}$$

in Table 9.2. From our computation, we have

$$\mathbb{P}(\boldsymbol{\mathcal{G}}|H_0) \approx P_K(\boldsymbol{\mathcal{G}}) = \prod_{i=0}^{2} Q(\mathcal{G}_i, \mathcal{G}_{i+1}) = \exp(-4.5731 \cdot 10^{11}).$$

Chapter 10

Estimating $\mathbb{P}(\mathbf{H}$ in a thin tube) by Importance Sampling

From the large deviations theory result Theorem 4.7.3, we know that the probability of fixations is approximately equal to the probability of trajectories realizing the fixation by following the optimal trajectory through a thin tube for large population size N. Therefore, in this chapter, we study how to force random trajectories to follow the optimal trajectory through a thin tube and estimate the probability of these forced trajectories.

10.1 Forced Trajectories in a Thin Tube

We want to force a random trajectory $\mathbf{H}$ to follow the optimal path $\boldsymbol{\mathcal{G}} = [\mathcal{G}_0 = H_0, ..., \mathcal{G}_T]$ through a thin tube. The basic ideas are:

1. Start with H_0, we aim for $\mathcal{G}_1$, suppose R_0 is the forced mutation matrix and we get to H_1 with $LW_1 = LW(H_0, R_0, H_1)$.

2. Start with H_1, we aim for $\mathcal{G}_2$, suppose R_1 is the forced mutation matrix and we get to H_2 with $LW_2 = LW(H_1, R_1, H_2)$.

...

T-1. Start with H_{T-1}, we aim for $\mathcal{G}_T$, suppose R_{T-1} is the forced mutation matrix and we get to H_T with $LW_{T-1} = LW(H_{T-1}, R_{T-1}, H_T)$.

By performing the algorithm above, we get a forced trajectory $\mathbf{H}$ following the optimal geodesic $\boldsymbol{\mathcal{G}}$. Denote $\boldsymbol{R} = [R_0, ..., R_{T-1}]$. Let the logarithm of the corrective weight of the forced trajectory $\mathbf{H}$ be

$$LW(\boldsymbol{H}, \boldsymbol{R}) = \sum_{i=0}^{T-1} LW(H_i, R_i, H_{i+1}). \tag{10.1}$$

We also know that $(H_i - \mathcal{G}_i) \cdot \sqrt{N}$ follows approximately multivariate normal distribution $N(0, \Sigma)$, where

$$\Sigma = \begin{pmatrix} \mathcal{G}_1(1-\mathcal{G}_1) & -\mathcal{G}_1\mathcal{G}_2 & ... & -\mathcal{G}_1\mathcal{G}_k \\ -\mathcal{G}_1\mathcal{G}_2 & \mathcal{G}_2(1-\mathcal{G}_2) & ... & -\mathcal{G}_2\mathcal{G}_k \\ \vdots & \vdots & \ddots & \vdots \\ -\mathcal{G}_1\mathcal{G}_k & -\mathcal{G}_2\mathcal{G}_k & \vdots & \mathcal{G}_k(1-\mathcal{G}_k) \end{pmatrix}.$$

Therefore,

$$|H_i - \mathcal{G}_i|_\infty \leq \max_{\{j=1,...,g\}} \frac{3\sqrt{\mathcal{G}_i(j)(1-\mathcal{G}_i(j))}}{\sqrt{N}} \leq \frac{3}{2\sqrt{N}},$$

with probability above 0.99 for large N.

10.2 Estimator of $\mathbb{P}(\mathbf{H}$ in a thin tube$)$

Definition 10.2.1. For any histogram $H \in \mathcal{H}$, define a ball centered at a histogram H with radius $\rho \in \mathbb{R}$ as

$$ball(H, \rho) = \{G \in \mathcal{H} \mid |G(i) - H(i)|_\infty < \rho\}.$$

Definition 10.2.2. For any random trajectory $\mathbf{H} = [H_0, ..., H_T] \in \Omega_{T+1}$, define a thin tube

centered at trajectory $\mathbf{H}$ with radius vector $\boldsymbol{\rho} = [\rho_0, ..., \rho_T] \in \mathrm{R}^{T+1}$ as

$$Tube(\mathbf{H}, \boldsymbol{\rho}) = [ball(H_0, \rho_0), ball(H_1, \rho_1), ..., ball(H_T, \rho_T)].$$

We say a trajectory $\mathbf{G} \in Tube(\mathbf{H}, \rho)$ if $G_i \in ball(H_i, \rho_i)$ for $i = 0, ..., T$.

Conjecture 10.2.3. Let N be the population size. Let $\boldsymbol{\mathcal{G}} = [\mathcal{G}_0 = H_0, ..., \mathcal{G}_t]$ be the optimal trajectory that realizes the event $Eve(J, \beta)$. Let $\mathbf{H} = [H_0, ..., H_t]$ be the forced trajectories following $\boldsymbol{\mathcal{G}}$. Let $\boldsymbol{\epsilon} = [0, \frac{3}{2\sqrt{N}}, ..., \frac{3}{2\sqrt{N}}]$. Then,

$$\mathbb{P}(\mathbf{H} \in Tube(\boldsymbol{\mathcal{G}}, \boldsymbol{\epsilon})) \geq 0.99^t.$$

We select the radius of a thin tube of the optimal trajectory $\boldsymbol{\mathcal{G}} = [\mathcal{G}_0, ..., \mathcal{G}_T]$ as

$$\epsilon = \frac{3}{2\sqrt{N}}.$$

Let $\boldsymbol{\epsilon} = [0, \epsilon, ..., \epsilon] \in \mathbb{R}^{T+1}$. We can estimate the probability $\mathbb{P}(\mathbf{H} \in Tube(\boldsymbol{\mathcal{G}}, \boldsymbol{\epsilon}) \cap Eve(J, \beta))$ which is the probability of realizing $Eve(J, \beta)$ through the thin tube $Tube(\boldsymbol{\mathcal{G}}, \boldsymbol{\epsilon})$ by the following algorithm.

Algorithm 10.1: Forcing $\mathbf{H} \in Tube(\boldsymbol{\mathcal{G}}, \boldsymbol{\epsilon}) \cap Eve(J, \beta)$

Step 1. Generate S forced trajectories $\boldsymbol{tra}^1, ..., \boldsymbol{tra}^S$ by following the optimal trajectory $\boldsymbol{\mathcal{G}}$.

Step 2. For each forced trajectories $\boldsymbol{tra}^i = [tra_0^i, ..., tra_T^i]$, let $\boldsymbol{R}^i = [R_0^i, ..., R_{T-1}^i]$ be the forced mutation matrix vector. Compute $LW(\boldsymbol{tra}^i, \boldsymbol{R^i})$ as formula (10.1).

Step 3. Let $Set = Tube(\boldsymbol{\mathcal{G}}, \boldsymbol{\epsilon}) \cap Eve(J, \beta)$. We can estimate the probability $\mathbb{P}(\mathbf{H} \in Set)$ as

$$P_S(\mathbf{H} \in Set) = \frac{1}{S} \sum_{i=1}^{S} \mathbf{1}_{\{\boldsymbol{tra}^i \in Set\}} \exp(N \cdot LW(\boldsymbol{tra}^i, \boldsymbol{R}^i)),$$

where $\mathbf{1}_{\{\boldsymbol{tra}^i \in Set\}} = 1$ if $\boldsymbol{tra}^i \in Set$, and $\mathbf{1}_{\{\boldsymbol{tra}^i \in Set\}} = 0$, otherwise.

Since $\exp(N \cdot LW(\mathbf{tra}^i, \mathbf{R}^i))$ takes extremely small values, we use Algorithm 8.2 to compute

the following summation

$$\sum_{i=1}^{S} \mathbf{1}_{\{\boldsymbol{tra}^i \in Set\}} \exp(N \cdot LW(\boldsymbol{tra}^i, \boldsymbol{R}^i)).$$

10.3 Accuracy of the Estimation

Recall for any trajectory $\mathbf{H} = [H_0, ..., H_T]$,

$$\mathbb{P}(\mathbf{H}) = \prod_{i=0}^{T-1} \{\mu_{N,H_{i+1}}(NH_{i+1}) \mathbb{E}(\exp(N \cdot LW(H_i, R_i, H_{i+1}))\}$$

$$= \mathbb{E}(\exp(N \sum_{i=0}^{T-1} LW(H_i, R_i, H_{i+1}))) \prod_{i=0}^{T-1} \mu_{N,H_{i+1}}(NH_{i+1})$$

By the change of probability, we have

$$\mathbb{P}(\mathbf{H} \in Set) = \sum_{\mathbf{h} \in Set} \mathbb{P}(\mathbf{H} = \mathbf{h})$$

$$= \sum_{\mathbf{h} \in Set} \mathbb{E}(\exp(N \sum_{i=0}^{T-1} LW(h_i, R_i, h_{i+1}))) \prod_{i=0}^{T-1} \mu_{N,h_{i+1}}(Nh_{i+1})$$

$$= \sum_{\mathbf{h} \in Set} \mathbb{E}(\exp(N \sum_{i=0}^{T-1} LW(H_i, R_i, H_{i+1}))|\mathbf{H} = \mathbf{h}) \prod_{i=0}^{T-1} \mu_{N,h_{i+1}}(Nh_{i+1})$$

$$= \mathbb{E}(\mathbb{E}(\exp(N \sum_{i=0}^{T-1} LW(H_i, R_i, H_{i+1}))|\mathbf{H}))$$

$$= \mathbb{E}(\exp(N \sum_{i=0}^{T-1} LW(H_i, R_i, H_{i+1})))$$

$$= \mathbb{E}(\exp(N \cdot LW(\mathbf{H}, \mathbf{R})))$$

Lemma 10.3.1. *Let $\boldsymbol{\mathcal{G}} = [\mathcal{G}_0, ..., \mathcal{G}_T] \in Set$ be an optimal geodesic. $\mathbf{tra}^1, ..., \mathbf{tra}^S$ are forced trajectories following $\boldsymbol{\mathcal{G}}$. Let $\mathbf{R}^i = [R_0^i, ..., R_{T-1}^i]$, where R_j^i is the forced mutation matrix from tra_j^i to*

tra^i_{j+1}. *$LW(\boldsymbol{tra}^i, \boldsymbol{R}^i)$ is defined as formula (10.1). Then*

$$P_S(\mathbf{H} \in Set) := \frac{1}{S}\sum_{i=1}^{S} \mathbf{1}_{\{\boldsymbol{tra}^i \in Set\}} \exp(N \cdot LW(\boldsymbol{tra}^i, \boldsymbol{R}^i)) \to \mathbb{P}(\mathbf{H} \in Set)$$

a.s. as $S \to \infty$.

Proof. Since

$$\mathbb{E}(|\exp(N \cdot LW(\mathbf{H}, \mathbf{R}))|) = \mathbb{E}(\exp(N \cdot LW(\mathbf{H}, \mathbf{R}))) = \mathbb{P}(\mathbf{H} \in Set) < \infty,$$

by the strong law of large number, we have

$$P_S(\mathbf{H} \in Set) \to \mathbb{P}(\mathbf{H} \in Set),$$

a.s. as $S \to \infty$. □

10.4 Example of Estimating $\mathbb{P}(\mathbf{H}$ in a thin tube$)$

Let $F = [200, 200^{1.08}, 200^{1.12}]$, $m = 10^{-7}$ and

$$Q = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0.5 & 0 \end{bmatrix}.$$

Let $H_0 = [0.8, 0.1, 0.1]$, $\beta = 0.3$. The optimal path for realizing the rare event $Eve(2, 0.3)$ is in Table 9.1. We generate 10^4 forced trajectories following the optimal geodesic $\boldsymbol{\mathcal{G}}$ for different population size $N = 10^{10}, 10^{12}, 10^{14}$. Let num_1 be the number of forced trajectories realizing the rare event $Eve(2, 0.3)$. Let num_2 be the number of forced trajectories realizing this rare event though the thin tube $Tube(\boldsymbol{\mathcal{G}}, \boldsymbol{\epsilon})$ where $\boldsymbol{\epsilon} = [0, \epsilon, ..., \epsilon]$. We present the radius of the tube $\epsilon = \frac{3}{2\sqrt{N}}$, num_1, num_2,

Table 10.1: Results of Forced Simulation of Trajectories

N	ϵ	num_1	num_2	$P_S(\mathbf{H} \in Set)$
10^{10}	$1.5 \cdot 10^{-5}$	5086	5058	$10^{-3.9727 \cdot 10^8}$
10^{12}	$1.5 \cdot 10^{-6}$	5033	5017	$10^{-3.9727 \cdot 10^{10}}$
10^{14}	$1.5 \cdot 10^{-7}$	4956	4934	$10^{-3.9727 \cdot 10^{12}}$

$\epsilon = 3/(2\sqrt{N})$. num_1 is the number of forced trajectories realizing the rare event. num_2 is the number of forced trajectories realizing the rare event through the thin tube. $P_S(\mathbf{H} \in Set)$ is the estimator of $\mathbb{P}(\mathbf{H} \in Set)$.

and $P_S(\mathbf{H} \in Set)$ in Table 10.1

We can see that about half of the forced trajectories realize the rare event $Eve(2, 0.3)$. Around 99% forced trajectories that realize the rare event $Eve(2, 0.3)$ stay in the thin tube $Tube(\boldsymbol{\mathcal{G}}, \boldsymbol{\epsilon})$ for different population size N. When N increases by a factor 10^n, $\log P_S(\mathbf{H} \in Set)$ decrease by a factor of $1/10^n$.

When $N = 10^4$, we plot the optimal trajectory $\boldsymbol{\mathcal{G}}$ and the histograms in 10^4 forced trajectories in Figure 10.1. The blue path is the optimal trajectory. The green clouds are formed by the histograms in the forced trajectories. The radius of each green clouds is of the order $1/\sqrt{N}$. As N increase, the radius of the cloud decreases. Therefore, the forced trajectories concentrate around the optimal trajectory for large N.

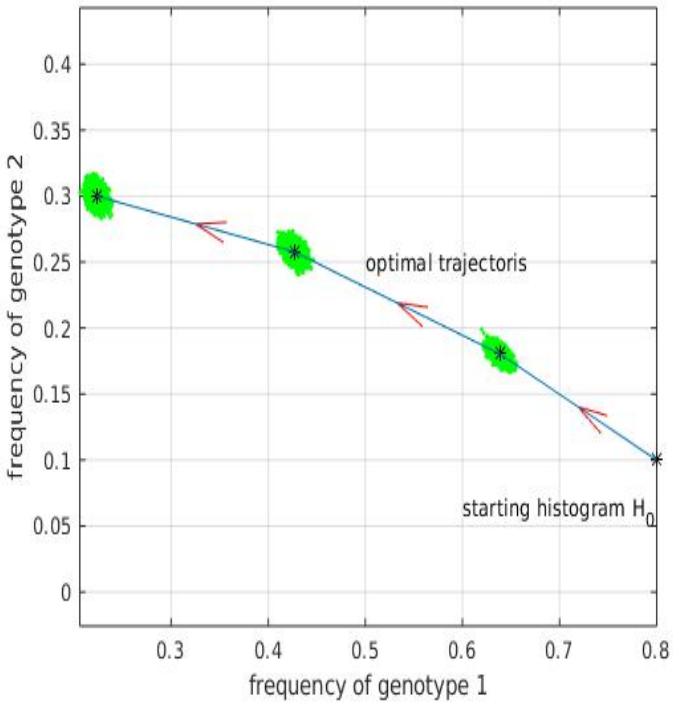

Figure 10.1: Forced Trajectories Around $\boldsymbol{\mathcal{G}}$

$N = 10^4$. The blue path is the optimal trajectory $\boldsymbol{\mathcal{G}}$. The green clouds are formed by histograms in the 10^4 forced trajectories.

Chapter 11

Genealogy Forced Trajectory Simulation

Searching for the optimal trajectory $\boldsymbol{\mathcal{G}}$ for realizing the event $Eve(J, \beta)$ could become very computational heavy when the number of genotypes g becomes larger than 8. Therefore, we develop this genealogy simulation method to force the rare event to happen more frequently without following any paths. To do so, we favor the trajectories with higher frequencies of J-cell by implementing a re-sampling trick.

11.1 One-step Transition from $H_n = H$ to $H_{n+1} = G$

We follow the stochastic model described in Chapter 2 to simulate H_{n+1} given $H_n = H$. We implement the following Algorithm 11.1.

Algorithm 11.1: One-step Transition

Step 1. We start with the population N and histogram H for day n. The population grows with the growth factor $F = [F(1), ..., F(g)]$. Compute the size of the g colonies of cells as $siz(j) = [NH(j)F(j)]$ for $j = 1, ..., g$.

Step 2. Recall that $R_n(j,k)$ is the random mutation number from j-cell to k-cell for day n. We sample $R_n(j,k)$ following the Poisson distribution with mean $siz(j)M_{j,k}$ for $j,k = 1,...,g$ and $j \neq k$. Suppose we get the mutation matrix $R_n = R$, compute the histogram of population $J = (J_1,...,J_g)$ as

$$J_j = \frac{1}{[N\langle F,H\rangle]}([NF_jH(j)] - \sum_k R_{j,k} + \sum_k R_{k,j}).$$

Step 3. After the mutation stage, we sample the random vector V following the multinomial distribution $\mu_{N,J}$. Suppose we get $V = y$, then compute $H_{n+1} = \frac{y}{N}$.

11.2 Generate a Set of Trajectories $Q(t) = \{\mathbf{H}^1,...,\mathbf{H}^P\}$

Let T denote the duration. We will generate a set of random trajectories $Q(t) = \{\mathbf{H}^1,...,\mathbf{H}^P\}$ at time t, where P is the number of trajectories in set $Q(t)$ and $\mathbf{H}^i = [H_0^i,...,H_T^i]$ is a trajectory in Ω_{T+1}.

Algorithm 11.2: Generate a Set of Trajectories $Q(t) = \{\mathbf{H}^1,...,\mathbf{H}^P\}$

Given the starting histogram H_0^i of every random trajectory $\mathbf{H}^i$, generate the random trajectory $\mathbf{H}^i$ step by step following Algorithm 11.1.

Define a function $level_J$ of genotype J on a random trajectory set $Q = \{\mathbf{H}^1,...,\mathbf{H}^P\}$ where $\mathbf{H}^j$ as

$$level_J(Q) = \max_{j=1,...,P, i=1,...,T} H_i^j(J).$$

11.3 Generate $Q(t+1)$ Given $Q(t)$

Given the trajectory set $Q(t)$ at time t, we generate the trajectory set $Q(t+1)$ at time $t+1$ by the re-sampling technique to favor the trajectories in $Q(t)$ which have higher percentages of J-cell at step T as the following algorithm.

Algorithm 11.3: Generate $Q(t+1)$ **Given** $Q(t)$

Step 1. Re-sampling Define a weight function on set $Q = \{\mathbf{H}^1, ..., \mathbf{H}^P\}$ where $\mathbf{H}^j = [H_0^j, H_1^j, ..., H_T^j]$, as $Wei(Q) = w = (w_1, ..., w_P) \in \mathbb{R}^P$, where

$$w_i = \frac{H_T^i(J)}{\sum_{j=1}^P H_T^j(J)}.$$

Apply this weight function on set $Q(t) = \{H^1, ..., H^P\}$ and compute $W = Wei(Q(t))$, where

$$W_i = \frac{H_T^i(J)}{\sum_{j=1}^P H_T^j(J)}$$

for $i = 1, .., P$. Then we use this weight vector W and P as parameters of multinomial distribution $\mu_{P,W}$ where

$$\mu_{P,W}(X = (x_1, ..., x_P)) = \frac{P!}{x_1! \cdot ... \cdot x_P!} W_1^{x_1} \cdot ... \cdot W_P^{x_P}.$$

We sample a vector $q = (q_1, ..., q_P)$ with $\sum_{j=1}^P q_j = P$ following the distribution $\mu_{P,W}$. Clone q_j copies of histogram H_T^j for $j = 1, ..., P$. Let $Copy(t) = \{h_1, ..., h_P\}$ be the set of these copied histograms, where

$$h_1 = ... = h_{q_1} = H_T^1,$$

$$h_{q_1+1} = ... = h_{q_1+q_2} = H_T^2,$$

$$...$$

$$h_{q_1+...+q_{P-1}+1} = h_{q_1+...+q_P} = H_T^P.$$

Notice that q_i might be 0 for some $i = 1, ..., P$. If $q_i = 0$, then there is no copy of H_T^i in set $Copy(t)$.

Step 2. Generating $Q(t+1)$**.** To generate set $Q(t+1) = \{\mathbf{S}^1, ..., \mathbf{S}^P\}$ where $\mathbf{S}^i = [S_0^i, ..., S_T^i]$, let $S_0^i = h_i$ for $i = 1, ..., P$. Then we generate the whole set $Q(t+1)$ following Algorithm 11.2.

We call the trajectory $\mathbf{H} \in Q(t)$ the ancestor of trajectory $\mathbf{S} \in Q(t+1)$ if $H_T = S_1$. Define the ancestor function Anc on a trajectory $\mathbf{S} \in Q(t+1)$ and set $Q(t)$ as $Anc(\mathbf{S}, Q(t)) = (\mathbf{H}, n)$ where $\mathbf{H} \in Q(t)$ is the ancestor of $\mathbf{S}$ and n is the number of ancestors of $\mathbf{S}$ in $Q(t)$.

11.4 Genealogy Simulation of Forced Trajectory

Given the starting histogram H_0, we want to simulate random trajectories that realize the rare event $Eve(J, \beta)$. We propose the following genealogy simulation algorithm.

Algorithm 11.4: Genealogy Simulation

Step 1. Generate the set of trajectories $Q(1) = \{\mathbf{H}^1, ..., \mathbf{H}^P\}$ given $H_0^i = H_0$ for $i = 1, ..., P$ at the time $t = 1$.

Step 2. If $level_J(Q(1)) \geq \beta$, we stop the simulation. The trajectories in $Q(1)$ which enter the target set TAR are the rare trajectories that realize the event $Eve(J, \beta)$. If $level_J(Q(1)) < \beta$, we follow the steps 3-4.

Step 3. We generate $Q(t+1)$ given $Q(t)$ following Algorithm 11.3 for $t \geq 1$.

Step 4. If $level_J(Q(t+1)) \geq \beta$ or $t > 50$, we stop the simulation. If $level_J(Q(t+1)) \geq \beta$, the trajectories in $Q(t+1)$ which enter the target set TAR realize the rare event $Eve(J, \beta)$. Otherwise, we repeat step 3-4.

Suppose the simulation stops at time t, $Q(t) = \{\mathbf{S}^1, ..., \mathbf{S}^P\}$ and $level_J(Q(t)) \geq \beta$. We select the trajectories in $Q(t)$ that enter the target set TAR. We name these l trajectories $\mathbf{ST}^1, ..., \mathbf{ST}^l$. For every $\mathbf{ST}^i$, we use the ancestor function Anc to construct a vector $n^i = [n_1^i, ..., n_{t-1}^i]$ where

$$(\mathbf{Y}_{t-1}, n_{t-1}^i) = Anc(\mathbf{ST}^i, Q(t-1)),$$

$$(\mathbf{Y}_{t-2}, n_{t-2}^i) = Anc(\mathbf{Y}_{t-1}, Q(t-2)),$$

...

$$(\mathbf{Y}_1, n_1^i) = Anc(\mathbf{Y}_2, Q(1)).$$

$\mathbf{Y}_i$ is the ancestor of $\mathbf{Y}_{i+1}$ for $i = 1, ..., t-2$ and $\mathbf{Y}_{t-1}$ is the ancestor of $\mathbf{ST}^i$. Then the estimator

for $\mathbb{P}(Eve(J,\beta))$ is

$$P_{J,\beta} = \frac{1}{P}(\sum_{i=1}^{l} \frac{1}{\prod_{j=1}^{t-1} n_j^i}).$$

11.5 Genealogy Forced Trajectory Simulation Example

Let $F = [200, 200^{1.08}, 200^{1.12}]$, $m = 10^{-7}$, $N = 10^8$ and

$$Q = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0.5 & 0 \end{bmatrix}.$$

Let $P = 10000$, $T = 3$, and starting histogram $H_0 = [0.8, 0.1, 0.1]$. We use Algorithm 11.4 to simulate the trajectories that realize $Eve(2, 0.2085)$. Among these 10000 trajectories, 14 trajectories realized the event $Eve(2, 0.2085)$. We plot these 14 trajectories in Figure 11.1. Since the difference between these 14 trajectories is of the order 10^{-4}, these trajectories look like one trajectory in this figure. The estimated probability of event $Eve(2, 0.2085)$ is

$$\mathbb{P}(Eve(2, 0.2085)) \approx 8.70 \cdot 10^{-4}.$$

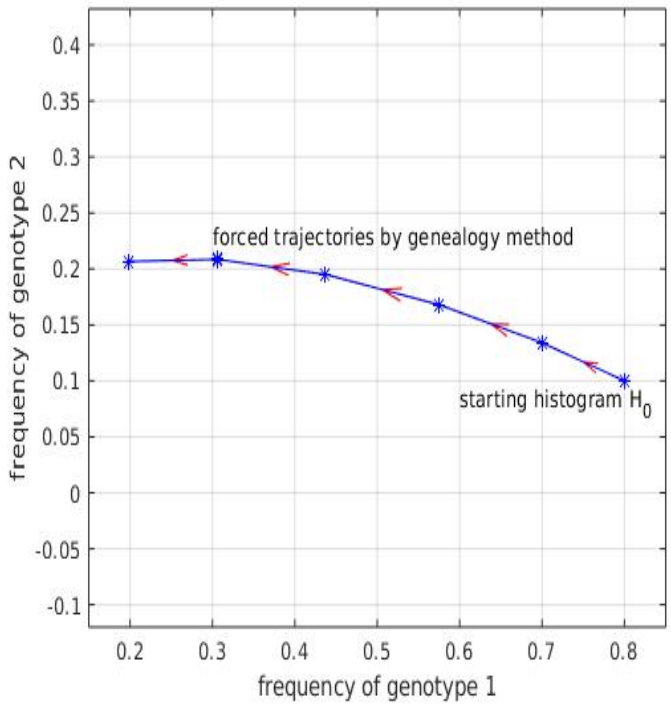

Figure 11.1: Forced Trajectories by Genealogy Method

Blue paths are the forced trajectories by using Algorithm 11.4. Red arrows represent the direction.

Appendix A

Appendix

A.1 Computing Time of Examples in Section 5.3.1.2

Recall the geodesic computing example with $g = 7$ in Section 5.3.1.2. The growth factor is

$$F = [200, 200^{1.07}, 200^{1.08}, 200^{1.09}, 200^{1.10}, 200^{1.11}, 200^{1.12}],$$

the mutation rate is $m = 10^{-8}$, the mutation matrix is $Q_{i,j} = 1/6$ for $i \neq j$. The starting histogram is $H = [0.6, 0.1, 0.1, 0.05, 0.05, 0.05, 0.05]$, the target histogram is $G = [0.1, 0.1, 0.1, 0.1, 0.1, 0.4, 0.1]$ and $Mesh = 0.01$, we use $p = 0.1$ and $p = 0.2$ to perform the geodesic computing following Algorithm 5.3 on 16 nodes with 8 cores on each node. Recall that $PEN_i(p)$ is the penultimate histograms set using the $p-quantile$ technique on the $i-th$ node. We present the computing time $CT_i(p)$ on each node $i = 1, ..., 16$ and the cardinality of $PEN_i(p)$ on each node $i = 1, ..., 16$ in the following Table A.1. We can see that all the penultimate histograms belonging to $PEN(0.1)$ and $PEN(0.2)$ are in the first node. It is hard to split the geodesic computing task evenly among all nodes. This is because we have to check whether a penultimate histogram y belongs to $PEN(p)$ on the fly. Otherwise, we have storage issues saving all the penultimate histograms on each core.

Table A.1: Computing Time for Example when $g = 7$

$0.1 - quantile$	$0.1 - quantile$	$0.2 - quantile$	$0.2 - quantile$
$CT_i(0.1)$ (sec)	$card(PEN_i(0.1))$	$CT_i(0.2)$ (sec)	$card(PEN_i(0.2))$
3984	1043949	4036	2119180
4142	0	3875	0
4044	0	3994	0
3772	0	4066	0
3760	0	3809	0
4070	0	3721	0
3997	0	3664	0
3704	0	4334	0
3973	0	4286	0
4060	0	4132	0
4037	0	4097	0
3849	0	4152	0
3652	0	3980	0
4416	0	4557	0
3934	0	3665	0
4070	0	3845	0

$H = [0.6, 0.1, 0.1, 0.05, 0.05, 0.05, 0.05]$, $G = [0.1, 0.1, 0.1, 0.1, 0.1, 0.4, 0.1]$, $CT_i(p)$ is the computing time of geodesic search from H to G using Algorithm 5.3 on node i for $p = 0.1, 0.2$, $card(PEN_i(p))$ is the cardinality of set $PEN_i(p)$ on node i for $p = 0.1, 0.2$.

Recall the geodesic computing example with $g = 8$ in Section 5.3.1.2. The growth factor is

$$F = [200, 200^{1.06}, 200^{1.07}, 200^{1.08}, 200^{1.09}, 200^{1.10}, 200^{1.11}, 200^{1.12}],$$

the mutation rate is $m = 10^{-8}$, the mutation matrix is $Q_{i,j} = 1/7$ for $i \neq j$. When the starting histogram is $H = [0.5, 0.1, 0.1, 0.1, 0.05, 0.05, 0.05, 0.05]$, the target histogram is $G = [0.05, 0.05, 0.1, 0.05, 0.05, 0.1, 0.5, 0.1]$ and $Mesh = 0.02$, we use $p = 0.1$ and $p = 0.2$ to perform the geodesic computing with quantile technique on 32 nodes with 16 cores on each node. We present the computing time $CT_i(p)$ on each node $i = 1, ..., 32$ and the cardinality of $PEN_i(p)$ on each node $i = 1, ..., 32$ in Table A.2.

From Tables A.1 and A.2, we see that we did not split the geodesic computing task evenly among all nodes. This is because we have to check whether a penultimate histogram belongs to $PEN(p)$ on the fly. Otherwise, we have storage issues saving all the penultimate histograms on each core.

Table A.2: Computing Time for Example when $g = 8$

$0.1 - quantile$	$0.1 - quantile$	$0.2 - quantile$	$0.2 - quantile$
$CT_i(0.1)$ (sec)	$card(PEN_i(0.1))$	$CT_i(0.2)$ (sec)	$card(PEN_i(0.2))$
5723	14424498	5309	16058299
189	855292	208	1033565
157	590376	189	812038
88	1027	92	2581
223	1047416	227	1225831
94	5083	99	7887
94	2073	96	4650
84	0	89	0
138	442650	165	673772
88	329	88	1386
93	7	87	307
84	0	81	0
77	899	111	2799
106	0	86	0
97	0	82	0
86	0	108	0
162	366985	158	601778
90	122	93	842
86	0	87	115
118	0	81	0
117	462	96	1884
112	0	83	0
126	0	86	0
91	0	111	0
88	0	113	21
116	0	86	0
117	0	103	0
104	0	88	0
84	0	98	0
90	0	83	0
87	0	88	0
82	0	87	0

$CT_i(p)$ is the computing time of geodesic search by using Algorithm 5.3 on node $i = 1, ..., 32$ for $p = 0.1, 0.2$, $card(PEN_i(p))$ is the cardinality of set $PEN_i(p)$ on node $i = 1, ..., 32$ for $p = 0.1, 0.2$.

Table A.3: Optimal trajectory $\boldsymbol{\mathcal{G}}$

$\mathcal{G}_0$	0.7000	0.1000	0.1000	0.1000
$\mathcal{G}_1$	0.5287	0.1116	0.1789	0.1808
$\mathcal{G}_2$	0.3352	0.1056	0.2550	0.3042
$\mathcal{G}_3$	0.1708	0.0822	0.3000	0.4470

Optimal trajectory for $Eve(3, 0.3)$ given $H_0 = [0.7, 0.1, 0.1, 0.1]$.

Table A.4: Optimal trajectory $\boldsymbol{\mathcal{G}}$

$\mathcal{G}_0$	0.7000	0.1000	0.1000	0.1000
$\mathcal{G}_1$	0.5514	0.1130	0.2065	0.1291
$\mathcal{G}_2$	0.3760	0.1107	0.3302	0.1832
$\mathcal{G}_3$	0.2151	0.0923	0.4365	0.2561
$\mathcal{G}_4$	0.0983	0.0643	0.5000	0.3374

Optimal trajectory for $Eve(3, 0.5)$ given $H_0 = [0.7, 0.1, 0.1, 0.1]$.

A.2 Optimal Trajectory $\mathcal{G}$ in Section 6.2.2.2

Recall that the example in Section 6.2.2.2, the starting histogram is $H_0 = [0.7, 0.1, 0.1, 0.1]$, the growth vector is

$$F = [200, 200^{1.08}, 200^{1.10}, 200^{1.20}],$$

and the mutation matrix is

$$Q = \begin{bmatrix} 0 & 0.3 & 0.3 & 0.4 \\ 0.3 & 0 & 0.3 & 0.4 \\ 0.3 & 0.3 & 0 & 0.4 \\ 0.3 & 0.3 & 0.4 & 0 \end{bmatrix},$$

the mutation rate is 10^{-7}. The optimal trajectories for realizing the event $Eve(3, 0.3)$, $Eve(3, 0.5)$ and $Eve(3, 0.7)$ are in Tables A.3, A.4, A.5.

Table A.5: Optimal trajectory $\boldsymbol{\mathcal{G}}$

$\mathcal{G}_0$	0.7000	0.1000	0.1000	0.1000
$\mathcal{G}_1$	0.5887	0.1159	0.2156	0.0798
$\mathcal{G}_2$	0.4393	0.1198	0.3597	0.0812
$\mathcal{G}_3$	0.2877	0.1099	0.5058	0.0966
$\mathcal{G}_4$	0.1621	0.0887	0.6251	0.1241
$\mathcal{G}_5$	0.0741	0.0622	0.7000	0.1637

Optimal trajectory for $Eve(3, 0.7)$ given $H_0 = [0.7, 0.1, 0.1, 0.1]$.

Bibliography

[1] N. K. Arenbaev. Asymptotic behavior of the multinomial distribution. *Theory of Probability & Its Applications*, 21(4):805–810, 1977.

[2] R. Azencott, M. Freidlin, and S. R. S. Varadhan. *Large deviations at Saint-Flour*. Springer New York, 2013.

[3] R. Azencott, B. Geiger, and I. Timofeyev. Rare events analysis in stochastic models for bacterial evolution. *arXiv*, 2018.

[4] J. E. Barrick and R. E. Lenski. Genome dynamics during experimental evolution. *Nature Reviews*, 14(12):827–839, 2013.

[5] J. E. Barrick, C. C. Strelioff, R. E. Lenski, and M. R. Kauth. Escherichia coli rpob mutants have increased evolvability in proportion to their fitness defects. *Molecular Biology and Evolution*, 27(6):1338–1347, 2010.

[6] J. Bucklew. *Introduction to rare event simulation*. Springer New York, 1st edition, 2010.

[7] M. Bugallo, V. Elvira, L. Martino, D. Luengo, J. Miguez, and P. Djuric. Adaptive importance sampling: the past, the present, and the future. *IEEE Signal Processing Magazine*, 34:60–79, 2017.

[8] C. Canonne. A short note on poisson tail bounds. *Retrieved from the Website: http://www. cs. columbia. edu/ ccanonne*, 2017.

[9] T. Cooper, D. Rozen, and R. Lenski. Parallel changes in gene expression after 20,000 generations of evolution in escherichia coli. *Proceedings of the National Academy of Sciences of the United States of America*, 100:1072–7, 03 2003.

[10] V. S. Cooper, D. Schneider, M. Blot, and R. E. Lenski. Mechanisms causing rapid and parallel losses of ribose catabolism in evolving populations of Escherichia coli. *Journal of Bacteriology*, 183:2834–2841, 2001.

[11] M. Cottrell, J. -C. Fort, and G. Malgouyres. Large deviations and rare events in the study of stochastic algorithms. *IEEE Transactions on Automatic Control*, 28(9):907–920, 1983.

[12] H. Cram'er and H. Touchette. Sur un nouveau théorème-limite de la théorie des probabilités. *Actualit'es Scientifiques et Industrielles*, 736:5 – 23, 1938.

[13] I. Csiszar. The method of types information theory. *IEEE Transactions on Information Theory*, 44(6):2505–2523, 1998.

[14] A. Dembo and O. Zeitouni. *Large deviations techniques and applications*. Springer, Berlin, Heidelberg, 2010.

[15] P. M. Djuric, T. Lu, and M. F. Bugallo. Multiple particle filtering. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 3, pages III–1181–III–1184, 2007.

[16] C. R. Doering, K. V. Sargsyan, L. M. Sander, and E. Vanden-Eijnden. Asymptotics of rare events in birth–death processes bypassing the exact solutions. *Journal of Physics: Condensed Matter*, 19(6):065145, 2007.

[17] M. D. Donsker and S. R. S. Varadhan. Asymptotic evaluation of certain Markov process expectations for large time, i. *Communications on Pure and Applied Mathematics*, 28(1):1–47, 1975.

[18] M. D. Donsker and S. R. S. Varadhan. Asymptotic evaluation of certain Markov process expectations for large time, ii. *Communications on Pure and Applied Mathematics*, 1975.

[19] M. D. Donsker and S. R. S. Varadhan. Asymptotic evaluation of certain Markov process expectations for large time iii. *Communications on Pure and Applied Mathematics*, 29(4):389–461, 1976.

[20] M. D. Donsker and S. R. S. Varadhan. Asymptotic evaluation of certain Markov process expectations for large time. iv. *Communications on Pure and Applied Mathematics*, 36(2):183–212, 1983.

[21] P. Dupuis and H. J. Kushner. Stochastic systems with small noise, analysis and simulation; a phase locked loop example. *SIAM Journal on Applied Mathematics*, 47(3):643–661, 1987.

[22] P. Dupuis, A. Devin Sezer, and H. Wang. Dynamic importance sampling for queueing networks. *The Annals of Applied Probability*, 17(4):1306–1346, 2007.

[23] P. Dupuis and H. Wang. Importance sampling, large deviations, and differential games. *Stochastics An International Journal of Probability and Stochastic Processes*, 76:37, 2004.

[24] P. Dupuis and H. Wang. Dynamic importance sampling for uniformly recurrent Markov chains. *Annals of Applied Probability*, 15, 2005.

[25] W. Ee, W. Ren, and E. Vanden-Eijnden. String method for the study of rare events. *Physical Review B*, 66, 2002.

[26] W. Ee and E. Vanden-Eijnden. Transition-path theory and path-finding algorithms for the study of rare events. *Annual Review of Physical Chemistry*, 61:391–420, 2008.

[27] R. S. Ellis. Large deviations for a general class of random vectors. *The Annals of Probability*, 12(1):1–12, 1984.

[28] R. S. Ellis. *Entropy, large deviations, and statistical mechanics*, volume 271. Springer, Berlin, Heidelberg, 2006.

[29] V. Elvira, L. Martino, M. F. Bugallo, and P. M. Djuric. Elucidating the auxiliary particle filter via multiple importance sampling [lecture notes]. *IEEE Signal Processing Magazine*, 36(6):145–152, 2019.

[30] V. Elvira, L. Martino, D. Luengo, and M. F. Bugallo. Generalized multiple importance sampling. *Statistical Science*, 34(1):129 – 155, 2019.

[31] P. Fearnhead. Computational methods for complex stochastic systems: A review of some alternatives to mcmc. *Statistics and Computing*, 18:151–171, 2008.

[32] J. Geweke. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57(6):1317–1339, 1989.

[33] J. K. Ghosh, M. Delampady, and T. Samanta. *An introduction to Bayesian analysis: theory and methods*. Springer Texts in Statistics. Springer New York, 2007.

[34] C. Giardinà, J. Kurchan, and L. Peliti. Direct evaluation of large-deviation functions. *Phys. Rev. Lett.*, 96:120603, 2006.

[35] S. J. Godsill and T. Clapp. Improvement strategies for Monte Carlo particle filters. *Sequential Monte Carlo Methods in Practice*, pages 139–158, 2001.

[36] T. Grafke and E. Vanden-Eijnden. Numerical computation of rare events via large deviation theory. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29:063118, 06 2019.

[37] J. Gärtner. On large deviations from the invariant measure. *Theory of Probability & Its Applications*, 22(1):24–39, 1977.

[38] C. J. R. Illingworth and V. Mustonen. A method to infer positive selection from marker dynamics in an asexual population. *Bioinformatics*, 28:831 – 837, 2012.

[39] A. Jasra, D. A. Stephens, and C. C. Holmes. Population-based reversible jump Markov chain Monte Carlo. *Biometrika*, 94(4):787–807, 2007.

[40] H. Kahn. Random sampling (Monte Carlo) techniques in neutron attenuation problems–ii. *Nucleonics*, 6 6:60–5, 1950.

[41] T. Kloek and H. K. van Dijk. Bayesian estimates of equation system parameters: an application of integration by Monte Carlo. *Econometrica*, 46(1):1–19, 1978.

[42] V. Lecomte and J. Tailleur. A numerical approach to large deviations in continuous-time. *Journal of Statistical Mechanics Theory and Experiment*, page P03004, 2007.

[43] J. S. Liu. *Monte Carlo strategies in scientific computing*. Springer Series in Statistics. Springer New York, 2013.

[44] J. S. Liu and R. Chen. Blind deconvolution via sequential imputations. *Journal of the American Statistical Association*, 90(430):567–576, 1995.

[45] E. Lyman and D. Zuckerman. Annealed importance sampling of peptides. *The Journal of Chemical Physics*, 127:065101, 09 2007.

[46] J. A. M. de Sousa, P. R. A. Campos, and I. Gordo. An abc method for estimating the rate and distribution of effects of beneficial mutations. *Genome Biology and Evolution*, 5(5):794–806, 2013.

[47] P. D. Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 68(3):411–436, 2006.

[48] R. M. Neal. Annealed importance sampling. *Statistics and Computing*, 11:125–139, 2001.

[49] M.-S. Oh and J. O. Berger. Adaptive importance sampling in Monte Carlo integration. *Journal of Statistical Computation and Simulation*, 41(3-4):143–168, 1992.

[50] M.-S. Oh and J. O. Berger. Integration of multimodal functions by Monte Carlo importance sampling. *Journal of the American Statistical Association*, 88(422):450–456, 1993.

[51] A. Owen and Y. Zhou. Safe and effective importance sampling. *Journal of the American Statistical Association*, 95(449):135–143, 2000.

[52] S. Parekh and J. Walrand. A quick simulation method for excessive backlogs in network of queues. *Automatic Control, IEEE Transactions on*, 34:54 – 66, 1989.

[53] M. K. Pitt and N. Shephard. Filtering via simulation: auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

[54] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[55] F. Ragone, J. Wouters, and F. Bouchet. Computation of extreme heat waves in climate models using a large deviation algorithm. *Proceedings of the National Academy of Sciences*, 115(1):24–29, 2018.

[56] J. Richard and W. Zhang. Annealed importance sampling. *Statistics and Computing*, 11, 2001.

[57] J. Richard and W. Zhang. Efficient high-dimensional importance sampling. *Journal of Econometrics*, 141:1385–1411, 2007.

[58] J. Sadowsky and J. Bucklew. Large deviations theory techniques in Monte Carlo simulation. IEEE Computer Society, 1989.

[59] I. N. Sanov. On the probability of large deviations of random variables. *Matematicheskij Sbornik*, 42:11-44, 1958.

[60] S. Schmidler, J. Liu, and D. Brutlag. Bayesian segmentation of protein secondary structure. *Journal of Computational Biology*, 7:233–248, 2000.

[61] D. Siegmund. Importance sampling in the Monte Carlo study of sequential tests. *The Annals of Statistics*, 4(4):673 – 684, 1976.

[62] J. Tailleur and J. Kurchan. Probing rare physical trajectories with lyapunov weighted dynamics. *Nature Physics*, 3:203, 2006.

[63] S. Tokdar and R. Kass. Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2:54 – 60, 2010.

[64] E. Vanden-Eijnden and J. Weare. Rare event simulation of small noise diffusions. *Communications on Pure and Applied Mathematics*, 65, 12 2012.

[65] W. Zhang, V. Sehgal, D. Dinh, R. Azevedo, T. Cooper, and R. Azencott. Estimation of the rate and effect of new beneficial mutations in asexual populations. *Theoretical Population Biology*, 81:168–78, 2011.