2010-05-06

Stochastic Modeling and Simulation of Gene Networks

Zhouyi Xu
University of Miami, z.xu@umiami.edu

UNIVERSITY OF MIAMI

STOCHASTIC MODELING AND SIMULATION OF GENE NETWORKS

By

Zhouyi Xu

Submitted to the Faculty
of the University of Miami
in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

Coral Gables, Florida

May 2010

UNIVERSITY OF MIAMI

A dissertation submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

STOCHASTIC MODELING AND SIMULATION OF GENE NETWORKS

Zhouyi Xu

Approved:

Xiaodong Cai, Ph.D.
Assistant Professor of Electrical and Computer Engineering

Terri A. Scandura, Ph.D.
Dean of the Graduate School

Kamal Premaratne, Ph.D.
Professor of Electrical and Computer Engineering

James W. Modestino, Ph.D.
Professor of Electrical and Computer Engineering

Dimitris Papamichail, Ph.D.
Assistant Professor of Computer Science

Akmal Younis, Ph.D.
Associate Professor of Electrical and Computer Engineering

XU, ZHOUYI (Ph.D., Electrical and Computer Engineering)

Stochastic Modeling and Simulation of Gene Networks (May 2010)

Abstract of a dissertation at the University of Miami.

Dissertation supervised by Professor Xiaodong Cai.
No. of pages in text. (168)

Recent research in experimental and computational biology has revealed the necessity of using stochastic modeling and simulation to investigate the functionality and dynamics of gene networks. However, there is no sophisticated stochastic modeling techniques and efficient stochastic simulation algorithms (SSA) for analyzing and simulating gene networks. Therefore, the objective of this research is to design highly efficient and accurate SSAs, to develop stochastic models for certain real gene networks and to apply stochastic simulation to investigate such gene networks.

To achieve this objective, we developed several novel efficient and accurate SSAs. We also proposed two stochastic models for the circadian system of *Drosophila* and simulated the dynamics of the system.

The K-leap method constrains the total number of reactions in one leap to a properly chosen number thereby improving simulation accuracy. Since the exact SSA is a special case of the K-leap method when $K=1$, the K-leap method can naturally change from the exact SSA to an approximate leap method during simulation if necessary. The hybrid τ/K-leap and the modified K-leap methods are particularly suitable for simulating gene networks where certain reactant molecular species have a small number of molecules.

Although the existing τ-leap methods can significantly speed up stochastic simulation of certain gene networks, the mean of the number of firings of each reaction channel is not equal to the true mean. Therefore, all existing τ-leap methods produce biased results, which limit simulation accuracy and speed. Our unbiased τ-leap methods remove the bias in simulation results that exist in all current leap SSAs and therefore significantly improve simulation accuracy without sacrificing speed.

In order to efficiently estimate the probability of rare events in gene networks, we applied the importance sampling technique to the next reaction method (NRM) of the SSA and developed a weighted NRM (wNRM). We further developed a systematic method for selecting the values of importance sampling parameters. Applying our parameter selection method to the wSSA and the wNRM, we get an improved wSSA (iwSSA) and an improved wNRM (iwNRM), which can provide substantial improvement over the wSSA in terms of simulation efficiency and accuracy.

We also develop a detailed and a reduced stochastic model for circadian rhythm in *Drosophila* and employ our SSA to simulate circadian oscillations. Our simulations showed that both models could produce sustained oscillations and that the oscillation is robust to noise in the sense that there is very little variability in oscillation period although there are significant random fluctuations in oscillation peeks. Moreover, although average time delays are essential to simulation of oscillation, random changes in time delays within certain range around fixed average time delay cause little variability in the oscillation period. Our simulation results also showed that both models are robust to parameter variations and that oscillation can be entrained by light/dark circles.

To my beloved mother, wife, sister, and deceased father

for their endless love, understanding, support and encouragement

Acknowledgements

I want to express my sincerest gratitude to Professor Xiaodong Cai, who introduced me into the field of stochastic modeling and simulation of gene networks. His continuous encouragement and inspiration always helped me break through the difficulties I encountered in my research. I am also thankful to my other committee members, Dr. James W. Modestino, Dr. Kamal Premaratne, Dr. Akmal Younis and Dr. Dimitris Papamichail, for their insightful comments which improved my thesis.

More than everyone else, I am indebted to my mother and sister for their enthusiastic encouragement and unlimited supports throughout all the stages of my life. I want to thank my wife, Ruxin, for her understanding, support and encouragement during my studies.

I would like to extend my thanks to the faculty and staff members of the Electrical and Computer Engineering Department at the University of Miami for their help with my education and for their efforts to make various resources available to me. Especially, I want to thank Dr. Mohamed Abdel-Mottaleb for his advice and support during my studies.

In addition, I would like to thank my laboratory members, especially, Dr. Feng Niu, Dr. Kasun Wickramaratna, Dr. Baoyuan Wang, Dian Fan, Ji Wen, Yu Zhang, Zongxing Xie, Bing Chen, Kefei Lu, Jing Liu, Jindan Zhou, Shaminda Subasingha and all my other friends for their help and collaboration during my Ph.D studies.

Zhouyi Xu

University of Miami

May 2010

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Background

Genes in living cells regulate various cellular biochemical processes mainly through proteins that they express. It appears that genes and their products including RNA and proteins, as well as other molecular substances, interact with each other, which composed of complicated gene networks [1]. Despite growing knowledge about the molecular components of the cell, the dynamics of gene networks are not well understood.

Along with experimental investigation, appropriate computational models and tools for gene network can substantially help researchers to uncover the mechanism underlying gene regulation and understand gene functionality [2]. To explore the dynamics of gene networks, several computational approaches with different levels of modeling detail have been developed [3–5]. A Boolean network provides a coarse model, able to predict certain dynamic behavior of a biochemical system or gene network [6, 7]. At a more detailed level, chemically reacting systems or more specifically gene networks are viewed as deterministic systems, whose dynamics are entirely predictable given sufficient knowledge of the state of the system. The time evolution of the system is described by a set of coupled, ordinary differential equations (ODEs).

These equations characterize the system dynamics as a continuous and deterministic process. However, the ODE-based models have at least two problems. First, the number of molecules of each species in a gene network is an integer, but the ODE-based model treat it as a real number that can take any nonnegative values. When the number of molecules is large, this may be an acceptable approximation. However, in many biological systems, such as gene networks, certain molecular species have a very small number of molecules. For example, a gene typically has only two DNA molecules. In such cases, it is apparent that ODE-based model gives a very poor approximation to the system states. Second, it turns out that stochasticity exists in many biological systems [8–10]. In particular, stochasticity in gene expression stems from fluctuations in transcription and translation. More specifically, in gene expression, a series of events involve a small number of molecules of DNA, RNA and proteins. As each of these molecular events is subject to significant thermal fluctuations, the amount of mRNA and protein expressed from a gene expression is a stochastic process, which is called noise by biologists [11]. Such a process can result in very different rates of synthesis of a specific protein in genetically identical cells in the same environment [12–14].

Although stochastic gene expression was discovered decades ago [15–18], only recently it received much attention since advances in technology for single-cell analysis have provided an impetus for novel investigations, which sequentially result in new insights [2, 19–24]. It has been convincingly demonstrated that stochasticity is significant in gene expression and corresponding gene networks [25]. Understanding how stochasticity contributes to cellular process is important to the understanding of how cells work. Gene expression noise can explain many biological phenomena, such as in intrinsic property of randomness of intracellular networks [26–28], phenotypic variations in cells or organisms with the same genes and in the same environment [20]. However, there are many questions related to gene expression noise that remain unanswered [19].

While biological investigations of expression noise of a single gene or in a simple gene network have revealed some of the mechanisms by which cells control and exploit noise, a computational approach to modeling and simulating relatively large gene networks will elucidate many unanswered questions.

1.2 Motivation and Objectives

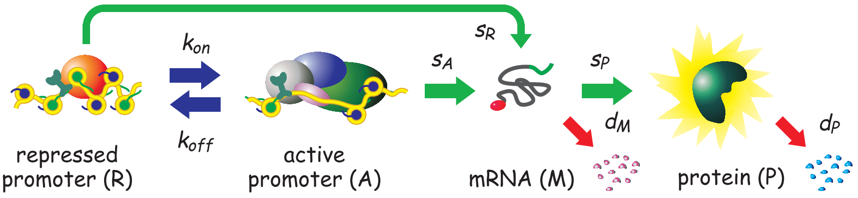


Figure 1.1: A model of the expression of a single gene.

We here use a simple example to illustrate the need of stochastic modeling in analyzing gene networks. Figure 1.1 depicts a model of the expression of a single gene [21], which elucidate several main steps in gene expression. In order to initiate transcription, an RNA polymerase needs to bind to the promoter of a gene. In eucaryotes, an RNA polymerase requires a large set of proteins called transcription factors to position itself correctly at the promoter, and open the two strands of DNA [29, 30]. As DNA in eucaryotes is packed into nucleosomes and higher order forms of chromatin structure, chromatin-modifying enzymes are also required to remodel chromatin so that an RNA polymerase can access the promoter. Consequently, the promoter is either in a repressed state in which an RNA polymerase cannot effectively bind to the promoter, or in an active state in which an RNA polymerase can bind to the promoter and efficiently initiate transcription. As activities of transcription factors and chromatin-modifying enzymes are subject to thermal fluctuations, the promoter randomly switches between these two states. A procaryotic gene, such as the lacZ gene in bacteria E. coli, can be controlled by an activator or/and a repressor [29]. As a result, a procaryotic gene can also randomly stay in either an inactive or active state. The parameters k_{on} and k_{off} in Figure 1.1 are deterministic rate constants used in conventional deterministic kinetics modeling the initiation of transcription. As we will discuss in Section 2, the transition probability between two states is related to these deterministic rate constants. As shown in Figure 1.1, the gene is transcribed with a probability s_A per unit time, when the promoter is active, and with a much lower rate s_R per unit time, when the promoter is repressed. Due to the randomness present in the initiation of transcription and transcription process itself, the number of mRNA molecules transcribed from the gene is random.

Since mRNA does not need to be processed or transported in prokaryotes, ribosomes can bind to the mRNA as soon as it is accessible behind the transcribing RNA polymerase and start translation. On the other hand, in eukaryotes, mRNA molecules are processed and transported from nucleus into cytoplasm so as to be translated there. Meanwhile, the mRNA can be bound and degraded by a multi-enzyme complex called degradosome. Therefore, an mRNA molecule is randomly translated to protein peptides by ribosomes, or, is degraded by degradosomes with certain probability that determines the rate constants s_P and d_M as shown in Figure 1.1. Finally, a functional protein can be targeted by a small polypeptide called ubiquitin, and be degraded by the proteasome. It is apparent that the amount of protein expressed from a gene is a random number, since the number of mRNA molecules is random as we discussed earlier, and the degradation and translation of the mRNA, as well as the degradation of protein itself, are random events. The model in Figure 1.1 is a simplified stochastic model for gene expression. More sophisticated models can be developed to characterize the gene expression in real cells, taking into account many additional factors, such as sequential assembly of the core transcription apparatus, pulsatile mRNA production due to reinitiation [14], and the scanning mechanism of ribosomes including leaky scanning and reinitiation in initiation of translation [31].

The gene expression example in Figure 1.1 can be modeled as the following reaction channels:

$$\begin{aligned} R &\overset{c_{\mathrm{on}}}{\rightarrow} A, \\ A &\overset{c_{\mathrm{off}}}{\rightarrow} R, \\ A &\overset{s_{\mathrm{A}}}{\rightarrow} A + M, \\ R &\overset{s_{\mathrm{R}}}{\rightarrow} R + M, \\ M &\overset{s_{\mathrm{P}}}{\rightarrow} M + P, \\ M &\overset{d_{\mathrm{M}}}{\rightarrow} \emptyset, \\ P &\overset{d_{\mathrm{P}}}{\rightarrow} \emptyset, \end{aligned} \tag{1.1}$$

where R, A, M, P and $\emptyset$ represent the gene's repressed promoter, the gene's active promoter, mRNA, protein and degraded molecules, respectively. Probability rate constants c_{on} and c_{off} determine the probability that the promoter is in the activated or repressed state; s_{A} and s_{R} are the transcriptional rates of active promoter and repressed promoter; s_{P} is the translational rate of mRNA; d_{M} and d_{P} are the degradation rate of mRNA and protein.

We simulated the gene expression process depicted in Figure 1.1 and modeled in (1.1). In our simulation, we assumed that two copies of the genes were initially in the repressed state, and thus the molecular number of R is 2 and the molecular number of all other species is zero. Similar to [21], we used the following probability rate constants: $c_{\mathrm{on}} = c_{\mathrm{off}} = 0.7$ per minute; $s_{\mathrm{A}} = 5$ per minute; $s_{\mathrm{R}} = 0.5$ per minute; $s_{\mathrm{P}} = 0.2$ per minute; $d_{\mathrm{M}} = 0.1$ per minute and $d_{\mathrm{P}} = 0.05$ per minute.

After running the simulation using ODE-based deterministic modeling method and SSA stochastic modeling method, we got the trajectory of mRNA and protein within time interval [0 1200], which is shown in Figure 1.2. It is seen that both mRNA and protein molecular numbers in one simulation run have large fluctuation, which demonstrated the stochasticity inside the gene networks. As reported in many experiments [12–14, 19–22], gene expression exhibits stochastic fluctuations similar to the stochastic trajectories depicted in Figure 1.2.

As stochasticity in gene expression has been clearly demonstrated in experiments, it is apparent that this stochasticity should be taken into consideration by precise modeling and simulation of a gene network. Stochastic kinetics can describe the time evolution of a biochemically reacting system as an overtly discrete, stochastic process, evolving in real continuous time. It tries to do this in a way that accurately reflects how chemical reactions physically occur at the molecular level and illustrate the stochastic behavior of coupled reactions [32, 33]. It was also shown to have a rigorous physical base [34]. Therefore, stochastic kinetics can be employed to characterize and simulate the dynamics of chemical reactions in gene expression. Since stochastic modeling of gene network from both biological and computational perspectives is still at its infancy, development of efficient stochastic simulation algorithms has been an intensive research topic.

Recently, several gene networks have been simulated [35–37] using Gillespie's exact SSA [32,33]. However, Gillespie's SSA requires large computational power and quickly becomes unmanageable when the reaction system becomes relatively large. Several approximate SSAs, including Poisson τ-leap method [38,39], modified τ-leap method [40], the binomial and multinomial τ-leap methods [41–43] and the midpoint Poisson [38] and binomial τ-leap [41], have been developed. Although these approximate SSAs improve simulation speed, at the price of sacrificing simulation accuracy, more efficient and accurate SSAs are needed for simulating large gene networks.

Therefore, the objective of this research is to develop highly efficient and accurate SSAs to accelerate stochastic simulation and improve simulation accuracy, and to apply stochastic simulation to simulate several real gene networks including the gene network of circadian clock in *Drosophila*.

1.3 Contributions

The major contributions of this dissertation are listed in the following:

- Developed a novel algorithm of K-leap method to accelerate stochastic simulation speed and improve simulation accuracy.

- Introduced a hybrid τ/K-leap method and a modified K-leap method to speed up simulation without losing accuracy when dealing with systems with small and large number of reactant molecules.

- Developed innovative unbiased Poisson and Binomial τ-leap methods to remove the bias in all existing τ-leap methods, including Poisson τ-leap, binomial, multinomial and modified τ-leap methods, thereby significantly improving simulation accuracy.

- Designed a weighted next reaction method (wNRM) for estimating the probability of rare events in chemical reaction systems, which is more efficient than the weighted SSA [44,45].

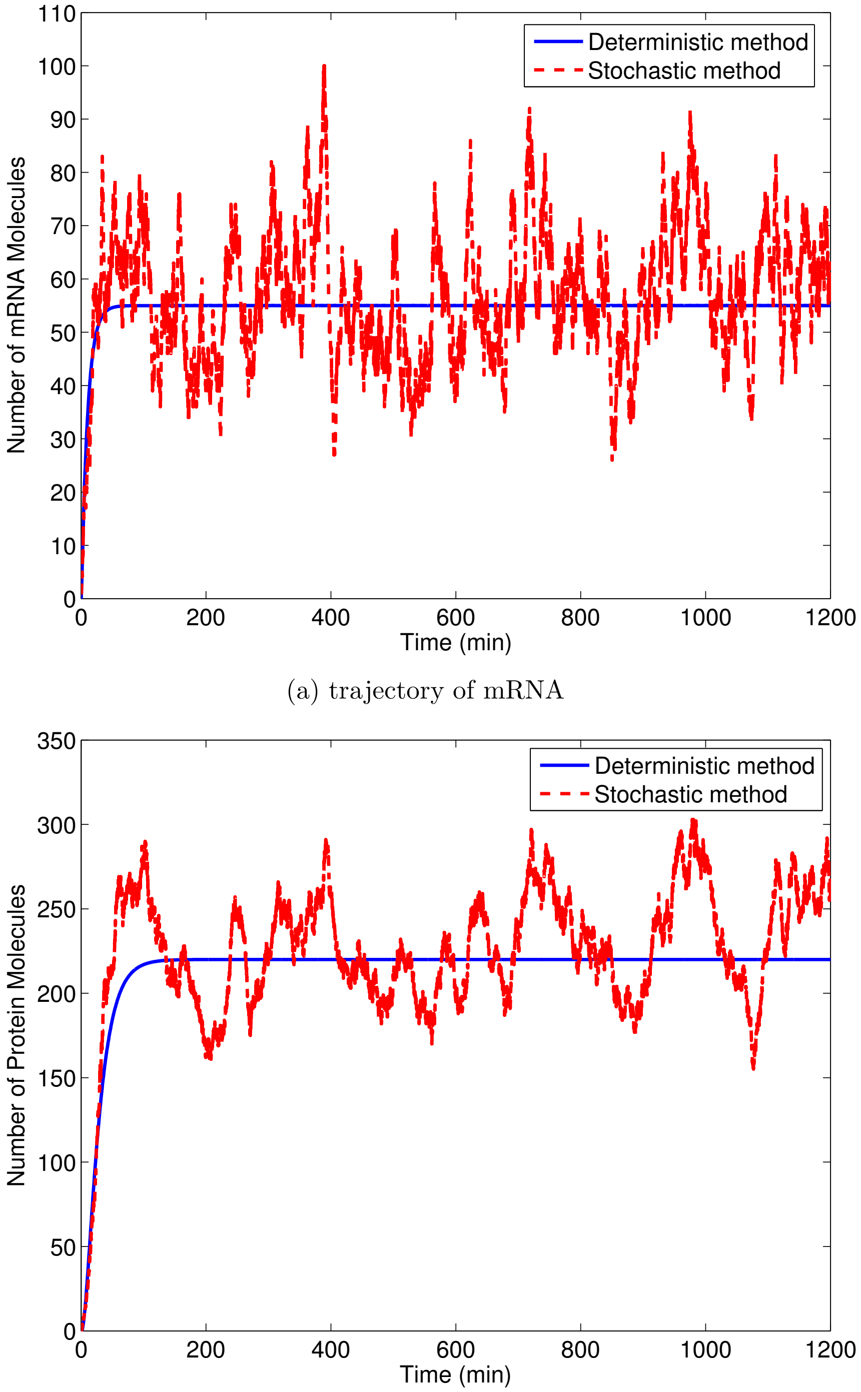
- Developed an improved weighted SSA (iwSSA) and an improved weighted next reaction method (iwNRM), which offer substantial improvement over the wSSA for estimating the probability of rare event in gene networks.

- Proposed a detailed and reduced stochastic model for the circadian rhythm of gene networks in *Drosophila* and applied exact stochastic simulation algorithm (SSA) with delays to simulate the model.

1.4 Dissertation Outline

The rest of this proposal is organized as follows. In Chapter 2, we briefly review the existing SSAs including the exact SSA [32,33], the Poisson τ-leap method [38,39], the modified τ-leap method [40], the binomial τ-leap method [41,42] and the midpoint Poisson [38] and binomial τ-leap [41]. In Chapter 3, we develop our K-leap SSA and demonstrate that our K-leap SSA outperforms existing leap methods. In Chapter 4, we develop a hybrid τ/K-leap method and a modified K-leap method for the gene network systems with small number of reactant molecules. In Chapter 5, we first show that all existing τ-leap methods produce biased results, which results in large simulation errors. We then develop unbiased Poisson, binomial and Poisson/Gaussian/Binomial τ-leap methods. We further show that our unbiased τ-leap methods significantly improve simulation accuracy without sacrificing speed. In

Chapter 6, we propose the wNRM algorithm to accelerate the simulation speed for estimating the probability of rare events in gene networks. We further introduce a systematic parameter selection method for the wSSA and the wNRM and develop the iwSSA and the iwNRM to improve the accuracy of the estimation of the probability of the rare event. In Chapter 7 we propose two stochastic models with delays for circadian rhythms in *Drosophila* and simulate the system dynamics. Finally, the summary and possible future work are presented in Chapter 8.



(a) trajectory of mRNA

(b) trajectory of protein

Figure 1.2: The trajectory of mRNA and protein using the ODE-based deterministic model method and SSA stochastic modeling method within time interval of [0 1200] for gene expression model of Figure 1.1.

CHAPTER 2

Stochastic Simulation Algorithms

In this chapter, we model a gene network as a chemically reacting system and then review several existing SSAs for simulating the dynamics of the system.

2.1 System Description

Suppose that gene expression and other activities in a gene network involve $N \geq 1$ molecular species $\{S_1, \cdots, S_N\}$ that chemically interact through $M \geq 1$ reaction channels $\{R_1, \cdots, R_M\}$. We describe the dynamic state of this chemical system by the state vector $\mathbf{X}(t) = [X_1(t), \cdots, X_N(t)]^T$, where $X_n(t),\ n = 1, \cdots, N$, is the number of S_n molecules at time t, and $[\cdot]^T$ denotes the transpose of the vector in the bracket. We assume the system is confined to a constant volume Ω and is in thermal (but not chemical) equilibrium at some constant temperature. Given the system at state $\mathbf{X}(t_0) = \mathbf{x}_0$ at initial time t_0, our goal is to obtain the system information of the state vector $\mathbf{X}(t)$.

Following Gillespie [33, 38, 39, 46], we define the dynamics of reaction R_m by a state-change vector $\boldsymbol{\nu}_m = [\nu_{1m}, \cdots, \nu_{Nm}]^T$, where ν_{nm} gives the changes in the S_n molecular population produced by one R_m reaction, and a propensity function $a_m(\mathbf{x})$ together with the fundamental premise of stochastic chemical kinetics:

$$a_m(\mathbf{x})dt \overset{\triangle}{=} \text{the probability, given } \mathbf{X}(t) = \mathbf{x}, \text{ that one reaction } R_m \\ \text{will occur in the next infinitesimal time interval } [t, t + dt). \tag{2.1}$$

It is instructive to consider the following simple example involving $N = 6$ molecular species and $M = 3$ reactions:

$$R_1 : \ S_1 + S_1 \rightarrow S_2, \ R_2 : \ S_3 + S_4 \rightarrow S_5 + S6, \ R_3 : \ S_6 \rightarrow S_1. \tag{2.2}$$

In this example, we have $\nu_1 = [-2, +1, 0, 0, 0, 0]^T$, $\nu_2 = [0, 0, -1, -1, +1, +1]^T$, and $\nu_3 = [+1, 0, 0, 0, 0, -1]^T$,

In the example of gene expression in Figure 1.1, the system involves $N = 4$ molecular species and $M = 7$ reactions. If we label $S_1 = R$, S_2=A, $S_3 = M$ and $S_4 = P$, the system can be described as:

$$R_1 : S_1 \rightarrow S_2, \ R_2 : S_2 \rightarrow S_1, \ R_3 : S_1 \rightarrow S_3, \\ R_4 : S_2 \rightarrow S_3, \ R_5 : S_3 \rightarrow S_4, \ R_6 : S_3 \rightarrow \emptyset, \ R_7 : S_4 \rightarrow \emptyset. \tag{2.3}$$

Then we have $\nu_1 = [-1, +1, 0, 0]^T$, $\nu_2 = [+1, -1, 0, 0]^T$, $\nu_3 = [-1, 0, +1, 0]^T$, $\nu_4 = [0, -1, +1, 0]^T$, $\nu_5 = [0, 0, -1, +1]^T$, $\nu_6 = [0, 0, -1, 0]^T$, and $\nu_7 = [0, 0, 0, -1]^T$.

If we define the probability rate constant c_m as the probability that a randomly selected combination of R_m reactant molecules reacts in a unit time period and let $h_m(\mathbf{x})$ be the number of distinct combinations of R_m reactant molecules in the system at time t, then the propensity function is given by $a_m(\mathbf{x}) = c_m h_m(\mathbf{x})$ [33]. Gillespie showed that for the bimolecular reaction R_m of the form $S_1 + S_2 \rightarrow$ product(s), we have $a_m(\mathbf{x}) = c_m x_1 x_2$. For the bimolecular reaction R_m of the form $S_1 + S_1 \rightarrow$product(s), we have $a_m(\mathbf{x}) = c_m x_1(x_1 - 1)/2$. For a monomolecular reaction R_m: $S_1 \rightarrow$ product(s), we have $a_m(\mathbf{x}) = c_m x_1$.

As argued in [34, 47], we typically only need to consider elementary reactions including bimolecular and monomolecular reactions, such as those in example (2.2), since trimolecular reactions in a fluid are usually the combined result of two bimolecular reactions and one monomolecular reaction. The probability rate constant c_m can be calculated from the conventional deterministic reaction rate k_m [32]. For monomolecular reactions, we have $c_m = k_m$, and for bimolecular reactions, we have $c_m = k_m/\Omega$, when two reactants are from different molecular species as in R_2 of example (2.2), and $c_m \approx 2k_m/\Omega$, when two reactants are the same as in R_1 of example (2.2), where Ω is the volume of the system.

2.2 Chemical Master Equation

The traditional way of investigating the stochastic time evolution of a chemically reacting system is to set up and solve the chemical master equation (CME) for the system. As the probability of a reaction occurs in the infinitesimal time interval $[t, t + dt)$ is only dependent upon the state $\mathbf{X}(t)$ at time t, it is clear that $\mathbf{X}(t)$ is a Markov process with discrete states, or a jump Markov process. The time evolution of the state probability mass function (PMF) $P(\mathbf{x}, t)$ of this Markov process is governed by the CME [33]:

$$\frac{\partial P(\mathbf{x}, t)}{\partial t} = \sum_{m=1}^{M} \Big[a_m(\mathbf{x} - \boldsymbol{\nu}_m) P(\mathbf{x} - \boldsymbol{\nu}_m, t) - a_m(\mathbf{x}) P(\mathbf{x}, t) \Big]. \tag{2.4}$$

The CME essentially says that the rate of change in $P(\mathbf{x}, t)$ is equal to the probability of entering the state $\mathbf{x}$ minus the probability of leaving the state $\mathbf{x}$ in unit time. A rigorous derivation of the CME is given in [33, 48], based on the fundamental premise (2.1). Even though the CME exactly describes the evolution of $P(\mathbf{x}, t)$ with time, unfortunately, we can solve the CME to obtain $P(\mathbf{x}, t)$ only in rare case.

2.3 Exact Stochastic Simulation Algorithm

Since it is difficult to solve the CME in general, Gillespie developed an SSA to simulate the Markov process $\mathbf{X}(t)$. Based on the fundamental premise (2.1), Gillespie developed an exact SSA to simulate the occurrence of every reaction when the time evolves [33]. Therefore, the realizations of $\mathbf{X}(t)$ generated from Gillespie's SSA adhere to a probability model identical to that obtained by the CME. For this reason, Gillespie's simulation algorithm is called the *exact* SSA.

There are three different but statistically equivalent methods for exact SSA: Gillespie's direct method (DM) [33], Gillespie's first reaction method (FRM) [32], and the next reaction method (NRM) of Gibson and Bruck [49].

2.3.1 The Direct Method

To elucidate the system state information for a chemical system in a given state, we need to know when will the next reaction occur and which reaction will occur. Gillespie's DM SSA answers these two questions. Specifically, the SSA simulates the occurrence of the following event:

$$\begin{aligned} E: &\text{ no reaction occurs in the time interval } [t, t+\tau], \text{ and a reaction } R_\mu \\ &\text{ occurs in the infinitesimal time interval } (t+\tau, t+\tau+d\tau). \end{aligned} \tag{2.5}$$

It has been shown by Gillespie [32, 33] that τ and μ are two independent random variables, and have the following probability density functions (PDF), respectively,

$$p(\tau) = a_0(\mathbf{x}) \exp(-a_0(\mathbf{x})\tau), \ \tau > 0, \tag{2.6}$$

and

$$p(\mu) = a_\mu(\mathbf{x})/a_0(\mathbf{x}), \ \mu = 1, \cdots, M, \tag{2.7}$$

where $a_0(\mathbf{x}) = \sum_{m=1}^{M} a_m(\mathbf{x})$. It is easy to generate τ and μ from two independent uniform random variables according to (2.6) and (2.7), respectively. Therefore, Gillespie's exact SSA generates a realization of τ and μ in each step of the simulation, and then updates the system state as $\mathbf{X}(t+\tau) = \mathbf{x} + \boldsymbol{\nu}_\mu$.

The SSA based on the DM can be summarized as follows:

Algorithm 1 (Exact SSA – Direct Method [33])

1. *Initialization (set the initial number of molecules, set* $t \leftarrow 0$*).*
2. *Calculate the propensity function,* $a_m(\mathbf{x})$*,* $m = 1, \cdots, M$*.*
3. *Generate* τ *and* μ *according to their pdf's in* (2.6) *and* (2.7)*.*
4. *Set* $t \leftarrow t + \tau$*, and update the state vector* $\mathbf{X}(t) \leftarrow \mathbf{X}(t) + \boldsymbol{\nu}_\mu$*.*
5. *Go to step 2, or else stop.*

2.3.2 The First Reaction Method

Another way of exact SSA is the first reaction method (FRM). Let us consider M independent events:

$$\begin{aligned} E_m : &\text{ no reaction } R_m \text{ occurs in the time interval } [t, t+\tau_m], \text{ and an } R_m \\ &\text{occurs in the infinitesimal time interval } (t+\tau_m, t+\tau_m+d\tau_m),\ m = 1, \cdots, M. \end{aligned} \tag{2.8}$$

Notice that in the event E_m in (2.8), it is possible that a reaction other than R_m occurs in the time interval $[t, t+\tau_m]$, while in the event E in (2.5), no reaction occurs in $[t, t+\tau]$. The pdf of τ_m can be easily found to be an exponential distribution with parameter $a_m(\mathbf{x})$, i.e., $p(\tau_m) = a_m(\mathbf{x}) \exp(-a_m(\mathbf{x})\tau_m)$, $\tau_m > 0$. If we independently generate τ_m, $m = 1, \cdots, M$, and take $\tau = \min\{\tau_1, \cdots, \tau_M\}$ and $\mu = \arg\min_m\{\tau_1, \cdots, \tau_M\}$, we essentially generate the event E in (2.5). Therefore, the FRM is equivalent to the DM. Compared with the DM, the FRM is not efficient, because it needs to generate more random variables, especially when number of reaction channels, M, is very large. However, Gibson and Bruck transformed the FRM into an equivalent and more efficient NRM for large N and large M [49].

2.3.3 The Next Reaction Method

The NRM is essentially a heavily revised version of the FRM and improves the efficiency of the FRM by exploiting the following two observations: i) each $a_m(\mathbf{x})$ is only affected by a few reactions and can be efficiently calculated in each step, and ii) τ_m, $m = 1, \cdots, M$, generated in a step can be reused in the next step. The NRM saves the putative next firing times of all reaction channels in an indexed binary tree priority queue, which is constructed so that the firing time of each parent node is always earlier than the firing times of its daughter nodes. The time and index of the next occurring reaction are therefore always available at the top node of the queue. A data structure called *dependency graph* is defined to tell precisely which $a_m(\mathbf{x})$ should be updated after a reaction occurs. The detailed description of the NRM can be found in [49]. After incorporating these two mechanisms into the NRM, it is argued in [49] that the FRM is more efficient than the DM, for loosely coupled chemical reaction systems where the firing of one reaction channel does not affect many other reactions.

However, a detailed analysis of CPU cost of both NRM and DM in [50] shows that maintaining and updating the data structure of the indexed priority queue in the NRM may require significantly large cost for some practical systems.

2.3.4 Improvement Strategies of Exact SSA

Several improvements have been developed for exact SSA. Lok and Brent [51] proposed a stochastic simulation software package, named Moleculizer, that uses a slightly simplified version of the NRM, but with a unique technique, where reaction channels and species are introduced only when they are need and removed when they are not needed.

Cao *et al.* [50] proposed an optimized direct method (ODM) to improve the efficiency of the DM. The ODM incorporates the dependency graph used in the NRM into the DM to reduce the cost of calculating the propensity functions, $a_m(\mathbf{x})$. It also properly reorders the index of reaction channels to reduce the cost of generating the reaction index μ. With these two optimization steps, the ODM is much more efficient than the original DM. It is argued in [50] that in practical systems that almost always have the multiscale nature, the ODM is preferable to the NRM. McCollum *et al.* [52] also introduced an improvement on the DM, which they called the sorting direct method (SDM). Similar to the ODM, SDM seeks to index the reaction cheannels in order of decreasing values of their propensity functions so as to optimize the search of the reaction index μ.

Li and Petzold [53] have recently developed the logarithmic direct method, to improve the direct method. Specifically, storing the partial sums of the propensity functions during the computation of $a_0(\mathbf{x})$, the value of μ can be obtained rapidly by conducting a binary search over those partial sums.

Improvements to the SSA are certainly beneficial, but any algorithm that simulates every reaction event one at a time will have a expected time between two consecutive reactions to be $E[\tau] = 1/a_0(\mathbf{x})$. Since $a_0(\mathbf{x})$ is at least linear and more commonly quadratic in the reactant population, $a_0(\mathbf{x})$ can be very large, and $E[\tau]$

correspondingly very small. Therefore, the exact SSA will inevitably be too slow for many practical applications.

2.4 Tau-leap Methods

Although the exact SSA produces realizations of $\mathbf{X}(t)$ with correct statistics, it requires huge computation, when the system population and/or the number of reaction channels are relatively large. To reduce computation burden, several approximate methods have been developed to significantly speed up simulation by giving up some of the exactness of the SSA. The basic idea behind these approximate methods is that instead of simulating a single reaction per step, a number of reactions can occur in each simulation step. As one step leaps over many reactions, these approximate methods are known as leap methods including the τ-leap method and mid-point τ-leap method [38, 39], the modified τ-leap method [40] and the binomial τ-leap method [41, 42]. Since the exact SSA is based on the fundamental premise (2.1), one would expect that a leap method can provide an excellent approximation to the exact SSA, if the propensity functions $a_m(\mathbf{x})$ remain approximately constant in each leap.

2.4.1 The Poisson τ-leap Method

The τ-leap method proposed by Gillespie [38,39] attempts to accelerate stochastic simulation by allowing each reaction channel to fire more than one times during a time interval of duration τ. The deterministic value τ is also referred to as the step size of a leap and is selected to satisfy the following leap condition: [38, 39]

C1 *The change in the state during $[t, t+\tau]$ is so slight that no propensity function will suffer an appreciable change in its value, i.e., $a_m(\mathbf{X}(t')) \approx a_m(\mathbf{x})$, $\forall t' \in [t,\ t+\tau]$, $\forall m \in [1, M]$.*

Let $K_m(\mathbf{x}, \tau)$, for any $\tau > 0$, be the number of R_m reactions that occur in the time interval $[t, t+\tau]$. If the leap condition C1 is satisfied, it can be shown that each $K_m(\mathbf{x}, \tau)$ is an independent Poisson random variable with mean $a_m(\mathbf{x})\tau$ [38, 39]. Therefore, Gillespie's τ-leap method generates a realization of $K_m(\mathbf{x}, \tau)$, $m = 1, \cdots, M$, according to the Poisson distribution, and then update the state after a leap as follows:

$$\mathbf{X}(t+\tau) = \mathbf{x} + \sum_{m=1}^{M} \boldsymbol{\nu}_m K_m(\mathbf{x}, \tau). \tag{2.9}$$

The question now is how to select the value of τ to satisfy the leap condition. Letting $\Delta a_m(\tau; \mathbf{x}) \overset{\triangle}{=} a_m\big(\mathbf{X}(t+\tau)\big) - a_m(\mathbf{x})$, Gillespie imposed the following constraint to satisfy the leap condition C1: [38]

$$|\Delta a_m(\tau; \mathbf{x})| \leq \epsilon a_0(\mathbf{x}),\ \forall m = 1, \cdots, M, \tag{2.10}$$

where ϵ is a prespecified error control parameter satisfying $0 < \epsilon \ll 1$. Since $\Delta a_m(\tau; \mathbf{x})$ is a random variable, it is difficult to find a τ directly satisfying (2.10). Gillespie proposed to use the first-order Taylor expansion of $\Delta a_m(\tau; \mathbf{x})$ to approximate $\Delta a_m(\tau; \mathbf{x})$, and then calculate τ by bounding the absolute mean and standard deviation of this approximate $\Delta a_m(\tau; \mathbf{x})$ by $\epsilon a_0(\mathbf{x})$ [38, 39], which leads to the following formula for determining τ: [38, 39]

$$\tau = \min_{m \in [1, M]} \left\{ \frac{\epsilon a_0(\mathbf{x})}{|\eta_m(\mathbf{x})|}, \frac{\epsilon^2 a_0^2(\mathbf{x})}{\sigma_m^2(\mathbf{x})} \right\}. \tag{2.11}$$

where

$$\eta_m(\mathbf{x}) \overset{\triangle}{=} \sum_{m'=1}^{M} f_{mm'}(\mathbf{x}) a_{m'}(\mathbf{x}),\ m = 1, \cdots, M, \tag{2.12}$$

$$\sigma_m^2(\mathbf{x}) \stackrel{\triangle}{=} \sum_{m'=1}^{M} f_{mm'}^2(\mathbf{x}) a_{m'}(\mathbf{x}),\ m = 1, \cdots, M, \tag{2.13}$$

and

$$f_{mm'}(\mathbf{x}) \stackrel{\triangle}{=} \left[\frac{\partial a_m(\mathbf{x})}{\partial \mathbf{x}}\right]^T \boldsymbol{\nu}_{m'},\ m, m' = 1, \cdots, M. \tag{2.14}$$

After calculating τ from (2.11), We can generate K_m, a realization of $K_m(\mathbf{x}, \tau)$, $m = 1, \cdots, M$, according to the Poisson distribution, and update the state after a leap as follows:

$$\mathbf{X}(t + \tau) = \mathbf{X}(t) + \boldsymbol{\nu}\mathbf{K}, \tag{2.15}$$

where $\boldsymbol{\nu} = [\boldsymbol{\nu}_1, \cdots, \boldsymbol{\nu}_M]$ and $\mathbf{K} = [K_1, \cdots, K_M]^T$

We summarize the Poisson τ-leap algorithm in the following:

Algorithm 2 (Poisson τ-Leap)

1. *Initialization (set the initial number of molecules, set $t \leftarrow 0$).*

2. *Calculate the propensity function, $a_m(\mathbf{x})$, $m = 1, \cdots, M$.*

3. *Calculate τ from* (2.11).

4. *Generate K_m, $m = 1, \cdots, M$, according to the Poisson distribution with mean $a_m(\mathbf{x})\tau$.*

5. *Set $t \leftarrow t + \tau$, and update the state vector $\mathbf{X}(t) \leftarrow \mathbf{X}(t) + \boldsymbol{\nu}\mathbf{K}$.*

6. *Go to step 2 until reaching the end time t_{end}.*

A more efficient method of selecting τ was developed by Cao *et al.* [54]. Instead of using (2.10) to satisfy the leap condition C1, the authors of [54] propose to bound the *relative* change in all the propensity function by the same amount ε: $|\Delta a_m(\tau; \mathbf{x})| \leq$

$\varepsilon a_m(\mathbf{x})$, $\forall m = 1, \cdots, M$. They further show that these inequalities are approximately equivalent to the following set of inequalities: $|\Delta X_n(\tau)| \leq \max\{\varepsilon_n x_n, 1\}$, $n = 1, \cdots, N$, where ε_n can be found from ε as discussed in [54]. Then, they choose the step size τ to satisfy the above inequalities appreciably as follows:

$$\tau = \min_{m\in[1,M]} \left\{ \frac{\max\{\epsilon x_n/g_n, 1\}}{|\mu_m(\mathbf{x})|}, \frac{\max\{\epsilon x_n/g_n, 1\}^2 a_0^2(\mathbf{x})}{\sigma_m^2(\mathbf{x})} \right\}, \tag{2.16}$$

where g_n is a constant defined in [54] for a specific type of reaction and

$$\begin{aligned} \mu_n(\mathbf{x}) &\triangleq \sum_{m=1}^{M} \nu_{nm}(\mathbf{x}) a_m(\mathbf{x}),\ n = 1, \cdots, N, \\ \sigma_n^2(\mathbf{x}) &\triangleq \sum_{m=1}^{M} \nu_{nm}^2(\mathbf{x}) a_m(\mathbf{x}),\ n = 1, \cdots, N. \end{aligned} \tag{2.17}$$

2.4.2 The Modified τ-leap Method

As realizations of a Poisson random variable can be any nonnegative integer, we always run the risk that some reaction channels fire so many times during one leap that the leap condition is violated by overly large population changes. In the extreme case, more molecules of some reactants will be consumed than those are actually available. When this occurs, the numbers of molecules of those reactants become negative, which is clearly undesirable. Cao *et al.* developed a modified τ-leap method [40] to avoid the problem of negative population.

In order to avoid negative number of molecules, the maximum number of times that reaction channel m can fire during a leap is given by

$$k_{m,\max} = \min_{n\in[1,N],\, \nu_{nm}<0} \left\lfloor \frac{x_n}{|\nu_{nm}|} \right\rfloor, \tag{2.18}$$

where $\lfloor x \rfloor$ denotes the greatest integer that is less than or equal to x. Cao *et al.* classify the reaction channels into two categories: critical and noncritical reaction channels. If $k_{m,\max}$ is less than or equal to some critical value n_c, then R_m is critical;

otherwise, it is noncritical. Typical value for n_c is chosen to be between 2 and 20 [40]. Let us denote $\mathcal{R}_c$ and $\mathcal{R}_{nc}$ as the set of indices of the critical and noncritical reaction channels, respectively, and the number of critical and noncritical reactions as M_c and M_{nc}, respectively. A tentative step size τ' is calculated from (2.11) with the reaction index m running over $\mathcal{R}_{nc}$. Another tentative step size τ'' is generated from an exponential distribution with parameter $1/a_0^c(\mathbf{x})$, where $a_0^c(\mathbf{x}) = \sum_{m\in\mathcal{R}_c} a_m(\mathbf{x})$. Then the actual step size is chosen as $\tau = \min\{\tau', \tau''\}$. For noncritical reactions, we generate K_m, $m \in \mathcal{R}_{nc}$, from a Poisson random variable as in the τ-leap method. For critical reactions, if $\tau'' > \tau'$, then $K_m = 0$ for all $m \in \mathcal{R}_c$; otherwise, a $\mu \in \mathcal{R}_c$ is generated using the exact SSA and then we set $K_\mu = 1$ and $K_m = 0$ for $m \in \mathcal{R}_c$ and $m \neq \mu$. Finally, the state after a leap is updated using (2.15). Essentially, the modified τ-leap method applies the τ-leap method to the noncritical reactions and the exact SSA to the critical reactions and chooses the step size τ as the minimum of the step sizes calculated from the τ-leap method and generated from the exact SSA.

2.4.3 The Binomial and Multinomial τ-leap Methods

Tian and Burrage [41], and independently Chatterjee *et al.* [42], proposed the binomial τ-leap method to deal with the problem of negative population. The binomial τ-leap method approximates the Poisson random variable K_m, $m = 1, \cdots, M$, used in the Poisson τ-leap method by a binomial random variable, $\mathcal{B}(k_{m,\max}, p_m)$, with parameters $k_{m,\max}$ and $p_m = [a_m(\mathbf{x})\tau]/k_{m,\max}$. If every molecular species is a reactant of only one reaction channel, the problem of negative population can be avoid by choosing $k_{m,\max}$ as of (2.18). However, a molecular species may be the reactant of several reaction channels. Since several reaction channels consume the molecules of the same species, using $k_{m,max}$ in (2.18) as one parameter of the binomial random variable will still have the risk of causing negative number of molecules.

Chatterjee *et al.* [42] propose to handle this problem by generating a realization k_m of the binomial random variable $\mathcal{B}(k_{m,\max}, p_m)$ for each reaction channel that involves the same molecule species in succession, decreasing the common reactant population on the right hand side of (2.18) appropriately after k_m is chosen [42]. However, there is a bias in this strategy that makes its outcome dependent on the arbitrary order in which the reactions are considered: earlier considered reaction channels tend to fire more often than later considered reaction channels. Chatterjee *et al.* attempts to correct this bias by randomly changing the order in which reaction channels fire from one leap to the next.

Tian and Burrage proposed a different approach [41]. If two reaction channels R_j and R_m both consume one molecule of a common reactant species, Tian and Burrage first generate a sample k_{jm} from a binomial random variable $\mathcal{B}\big(k_{jm,\max}, p_{jm}\big)$, where $p_{jm} = [(a_j(\mathbf{x}) + a_m(\mathbf{x}))\tau]/k_{jm,\max}$ and $k_{jm,\max} = \min\{k_{j,\max}, k_{m,\max}\}$ with $k_{j,\max}$ and $k_{m,\max}$ calculated from (2.18). Then, they generate a sample k_j from a binomial random variable $\mathcal{B}\big(k_{jm}, a_j(\mathbf{x})/[a_j(\mathbf{x}) + a_m(\mathbf{x})]\big)$ for R_j, and let $k_m = k_{jm} - k_j$ for R_m. However, Tian and Burrage did not address the situations where there are more than two coupled reaction channels, such as the following case: $R_1 : S_1 + S_2 \rightarrow S_5$, $R_2 : S_2 + S_3 \rightarrow S_6$, and $R_3 : S_3 + S_4 \rightarrow S_7$.

Basically, Chatterjee *et al.* employ the binomial random variable $\mathcal{B}(k_{m,\max}, p_m)$ to approximate the Poisson random variable K_m, while Tian and Burrage use binomial random variables $\mathcal{B}\big(k_{jm,\max}, p_{jm}\big)$ and $\mathcal{B}\big(k_{jm}, a_j(\mathbf{x})/[a_j(\mathbf{x}) + a_m(\mathbf{x})]\big)$ to approximate Poisson random variables $K_j + K_m$ and K_j, respectively. It is well known that a Poisson random variable can be well approximated by a binomial random variable $\mathcal{B}(n, p)$

only when $n \gg 1$ and $p \ll 1$ [55]. If the condition $n \gg 1$ and $p \ll 1$ is not satisfied, then the binomial τ-leap method may introduce considerable simulation inaccuracies in addition to that caused by changes in propensity functions. Specifically, when p_m and p_{jm}, calculated from the formulas discussed previously, are greater than one, we have to reduce τ so that $p_m \leq 1$ and $p_{jm} \leq 1$. Chatterjee *et al.* propose to choose τ such that $p_m = 1$, while Tian and Burrage do not specify how to choose τ, when the above situation occurs. Since we do not want to increase simulation time too much, we typically reduce τ to a value so that $p_{jm} \leq 1$ and close to 1.

Recently, Pettigrew and Resat proposed the multinomial τ-leap method [43], where K_m, $m = 1, \cdots, M$, are generated from multinomial random variables with mean $a_m(\mathbf{x})\tau$. Since the marginal distribution of multinomial random variables follows a binomial distribution, the multinomial τ-leap method essentially also generates K_m, $m = 1, \cdots, M$ from binomial distributions, but it was shown by Pettigrew and Resat that the multinomial τ-leap method can generate K_m, $m = 1, \cdots, M$ more efficiently.

2.4.4 The Midpoint Poisson and Binomial τ-leap Methods

As we discussed earlier, the Poisson, binomial and multinomial τ-leap methods generate K_m, $m = 1, \cdots, M$, with mean $a_m(\mathbf{x})\tau$, where $a_m(\mathbf{x})$ is calculated from the system state at t, $\mathbf{x}$. Since the propensity functions change during the time interval $[t, t+\tau]$, it may be better to use the state at the midpoint of $[t, t+\tau]$, $\mathbf{x}'$, to calculate $a_m(\mathbf{x}')$, $m = 1, \cdots, M$, and then generate K_m, $m = 1, \cdots, M$ with mean $a_m(\mathbf{x}')\tau$. Towards this end, Gillespie proposed an estimated midpoint Poisson τ-leap method. [38] Gillespie first estimated the expected state change in the time interval $[t, t+\tau]$ as $\overline{\Delta \mathbf{x}} = \sum_{m=1}^{M} a_m(\mathbf{x})\tau\boldsymbol{\nu}_m$, and then estimated the state at the midpoint as

$\mathbf{x}' = \mathbf{x} + \lceil \overline{\Delta \mathbf{x}}/2 \rceil$, where $\lceil x \rceil$ denotes the smallest integer greater than x. Gillespie's midpoint Poisson τ-leap method therefore has the same steps as Algorithm 2, but the mean used to generate K_m in step 4 is replaced by $a_m(\mathbf{x}')\tau$.

Tian and Burrage also proposed a midpoint binomial τ-leap method. [41] They used the same method as in the midpoint Poisson τ-leap method to calculate $\mathbf{x}'$, and then modify the binomial τ-leap method by changing one of the parameters of the binomial random variable $\mathcal{B}(k_{m,\max}, p_m)$ to $p_m = a_m(\mathbf{x}')\tau / k_{m,\max}$.

2.5 Accuracy Measurement of Stochastic Simulation

As we discussed earlier, exact SSA simulates every reaction occurring when the system evolves with time. Therefore, exact SSA simulates the stochastic dynamics of the system accurately. Approximate SSA methods aim to improve simulation speed with acceptable accuracy. Since the system state $\mathbf{X}(t)$ that is simulated is a random process, a natural question arises: how can we measure the accuracy of various approximate SSAs? In this thesis, we use the density distance proposed by Cao and Petzold [56] as a figure of merit for simulation accuracy.

The density distance for two random variable X and Y is defined as follows:

$$D(X, Y) = \int |p_X(s) - p_Y(s)|\, ds, \tag{2.19}$$

where p_X and p_Y are probability density functions (PDF) of X and Y. If X and Y are two discrete random variables, the density distance is defined as:

$$D(X, Y) = \sum_z |P(X = z) - P(Y = z)| \tag{2.20}$$

where $P(X)$ and $P(Y)$ are the PMF of X and Y, respectively.

To compare the accuracy of different approximate SSAs, we run simulations for a specific system using exact SSA and approximate SSAs. The histogram of the interested variables are obtained from simulation results. Using histogram as an approximation of the PDF in (2.19) or PMF in (2.20), we can calculate the density distance between the results of exact SSA and those approximate SSAs. The density distance calculated in this way is also referred to as histogram distance (HD).

CHAPTER 3

The K-Leap Method for Accelerating Stochastic Simulation

3.1 Motivation

The exact SSA can simulate the time evolution of a chemical reaction system with exact statistical properties stipulated by the CME, but it requires huge computation. Therefore, in many practical systems, approximate SSAs are widely used. As we discussed earlier, the Poisson τ-leap method [38,39] use Poisson random variable, whose realization can be any nonnegative integer so that there always certain probability that the leap condition is violated and extremely, negative number of molecules occurred. The binomial τ-leap method [41,42] attempts to improve simulation accuracy by avoiding negative numbers of molecules. However, the number of firings of each reaction channel during a leap still can be large enough to violate the leap condition. Keep in mind that when negative number of molecules occurs, the leap condition has been severely violated. Hence, a better way of improving simulation accuracy and avoiding negative number of molecules is to design a leap method that can enforce the leap condition at the beginning.

This motivates us to develop a new leap method, called K-leap method [57], where we constrain the total number of reactions occurred during a leap to be a constant K, calculated from the leap condition. As the number of times that each reaction fires during a leap is bounded by K, the leap condition can be better satisfied, thereby improving simulation accuracy. We will show that given the above constraint, the time τ leaped over during a step follows a Gamma distribution, while the joint distribution of the numbers of reactions occurred during a leap for each channel follows a multinomial distribution. We will also developed several methods of determining K according to the leap condition. Since the exact SSA is a special case of our K-leap method when K=1, our K-leap method can naturally change from the exact SSA to an approximate leaping method during simulation, whenever the leap condition allows to do so.

3.2 K-leap Method

Recall that the exact SSA simulates the occurrence of a single reaction in each step, and the time between two steps is a random variable. However, in the τ-leap method, the time τ leaped over in a step is a deterministic number preselected to satisfy the leap condition; once τ is selected, the number of reactions occurred during a leap is a random variable that can take very large values. The dilemma of the τ-leap method is: how can a preselected τ, without knowing at least an upper bound on the number of reactions that will occur in the next leap, well satisfy the leap condition? After all, it is the number of reactions occurred during a leap, not the time leaped over, that affects the propensity functions.

Our K-leap method [58] avoid this dilemma by mimicking the exact SSA: simulate the occurrence of $K \geq 1$ reactions during each leap, where K is a *deterministic* number chosen to satisfy the leap condition. After K is chosen, the time, τ, that is leaped over in each step, is a random variable. Mathematically, we impose the constraint $\sum_{m=1}^{M} K_m(\mathbf{x}, \tau) = K$ on total number reactions occurred during each leap. For notational brevity, we will denote $K_m(\mathbf{x}, \tau)$ as K_m in the remaining of the thesis. Although K is a deterministic number, K_m, $m = 1, \cdots, M$ are random variables taking integer values in $[0, K]$.

In order to simulate τ and K_m, $m = 1, \cdots, M$, we first need to find the joint PDF of τ and K_m, $m = 1, \cdots, M$, given the constraint $\sum_{m=1}^{M} K_m = K$. Denoting this conditional PDF as $p(K_1, \cdots, K_M, \tau | \sum_{m=1}^{M} K_m = K)$, we show in the Appendix A that τ is independent of $K_1, \cdots, K_M$ under the above constraint, that is $p(K_1, \cdots, K_M, \tau | \sum_{m=1}^{M} K_m = K) = p(\tau | \sum_{m=1}^{M} K_m = K) p(K_1, \cdots, K_M | \sum_{m=1}^{M} K_m = K)$, where $p(\tau | \sum_{m=1}^{M} K_m = K)$ is the conditional PDF of τ, and $p(K_1, \cdots, K_M | \sum_{m=1}^{M} K_m = K)$ is the joint conditional PDF of K_m, $m = 1, \cdots, M$, given $\sum_{m=1}^{M} K_m = K$. Moreover, we prove in the Appendix A that $p(\tau | \sum_{m=1}^{M} K_m = K)$ is a Gamma PDF given by

$$p\big(\tau | \sum_{m=1}^{M} K_m = K\big) = \frac{a_0(\mathbf{x}) \exp(-a_0(\mathbf{x})\tau)(a_0(\mathbf{x})\tau)^{K-1}}{(K-1)!}, \tau > 0, \tag{3.1}$$

while $p(K_1, \cdots, K_M | \sum_{m=1}^{M} K_m = K)$ is a multinomial PDF given by

$$p\big(K_1, \cdots, K_M | \sum_{m=1}^{M} K_m = K\big) = \frac{K!}{\prod_{m=1}^{M} K_m!} \prod_{m=1}^{M} \theta_m^{K_m},\ K_m \geq 0, \sum_{m=1}^{M} K_m = K \tag{3.2}$$

where $\theta_m = a_m(\mathbf{x})/a_0(\mathbf{x})$, $m = 1, \cdots, M$.

To implement the K-leap method in simulating a practical chemical reaction system, we need some way of quickly determining the value of K so that the leap condition C1 can be well satisfied. We next propose three methods of calculating the value of K.

3.2.1 K-selection Method I

The most straightforward way is to employ the same approach used by Gillespie and Petzold to selecting τ in the τ-leap method [39]. Since K_m, $m = 1, \cdots, M$ follow a multinomial distribution given in (3.2), the mean and variance of K_m are $E[K_m] = K\theta_m$ and $\text{var}[K_m] = K\theta_m(1-\theta_m)$, respectively, and the covariance of K_m and $K_{m'}$ is $\text{cov}[K_m, K_{m'}] = -K\theta_m\theta_{m'}$, $m \neq m'$. Let us define $\boldsymbol{\theta} \stackrel{\triangle}{=} [\theta_1, \cdots, \theta_M]^T$, and $\mathbf{K} \stackrel{\triangle}{=} [K_1, \cdots, K_M]^T$, then we have $E[\mathbf{K}] = K\boldsymbol{\theta}$. If we define a matrix $\mathbf{C}$ with $[\mathbf{C}]_{mm'} = -\theta_m\theta_{m'}$, for $m \neq m'$, and $[\mathbf{C}]_{mm} = \theta_m(1-\theta_m)$, where $[\mathbf{C}]_{mm'}$ denotes the entry on the mth row and the m'th column of $\mathbf{C}$, the covariance matrix of $\mathbf{K}$ is given by $\text{cov}[\mathbf{K}] = K\mathbf{C}$.

The first order Taylor expansion of $\Delta a_m(K; \mathbf{x})$ can be found as [38]

$$\Delta a_m(K; \mathbf{x}) \approx \sum_{m'=1}^{M} f_{mm'}(\mathbf{x}) K_{m'}, \tag{3.3}$$

where

$$f_{mm'}(\mathbf{x}) \stackrel{\triangle}{=} \left[\frac{\partial a_m(\mathbf{x})}{\partial \mathbf{x}}\right]^T \boldsymbol{\nu}_{m'}, \ m, m' = 1, \cdots, M. \tag{3.4}$$

Letting $\mathbf{f}_m = [f_{m1}, \cdots, f_{mM}]^T$, $m = 1, \cdots, M$, and

$$\begin{aligned} \eta_m(\mathbf{x}) &\stackrel{\triangle}{=} \mathbf{f}_m^T \boldsymbol{\theta}, \ m = 1, \cdots, M, \\ \sigma_m^2(\mathbf{x}) &\stackrel{\triangle}{=} \mathbf{f}_m^T \mathbf{C} \mathbf{f}_m, \ m = 1, \cdots, M, \end{aligned} \tag{3.5}$$

we obtain from (3.3) the following:

$$\begin{aligned} E[\Delta a_m(K; \mathbf{x})] &\approx \eta_m(\mathbf{x}) K, \\ \text{var}[\Delta a_m(K; \mathbf{x})] &\approx \sigma_m^2(\mathbf{x}) K. \end{aligned} \tag{3.6}$$

If we impose the requirements $|E[\Delta a_m(K; \mathbf{x})]| < \epsilon a_0(\mathbf{x})$ and $\sqrt{\text{var}[\Delta a_m(K; \mathbf{x})]} < \epsilon a_0(\mathbf{x})$ as in the τ-selection method of Gillespie and Petzold [39], using (3.6), and considering that the minimum value of K is 1, we obtain the value of K hat approximately satisfies the leap condition (2.10):

$$K = \max\left\{ \min_{m\in[1,M]} \left\{ \frac{\epsilon a_0(\mathbf{x})}{|\eta_m(\mathbf{x})|}, \frac{\epsilon^2 a_0^2(\mathbf{x})}{\sigma_m^2(\mathbf{x})} \right\}, 1 \right\}. \tag{3.7}$$

Note that (3.7) is in a form similar to the formula for determining τ in the τ-leap method [39], but $\eta_m(\mathbf{x})$ and $\sigma_m^2(\mathbf{x})$ in (3.5) are calculated differently.

Cao *et al.* has proposed a more efficient method of determining τ for the τ-leap method [54]. Based on the approach of Cao *et al.*, we next develop two efficient methods of calculating K for our K-leap method.

3.2.2 K-selection Method II

Although (2.10) limits the changes in the propensity functions during a leap as required by the leap condition, it does not satisfy the leap condition very well for those propensity functions that are relatively small compared to $a_0(\mathbf{x})$. This is because (2.10) will allow a large relative change in those small propensity functions, which could result in simulation inaccuracies. To better satisfy the leap condition C1, we bound the *relative* change in all the propensity function by the same among ϵ [54]:

$$|\Delta a_m(\tau;\mathbf{x})| \leq \epsilon a_m(\mathbf{x}), \ \forall m = 1, \cdots, M. \tag{3.8}$$

Let us define

$$\Delta X_n \overset{\triangle}{=} X_n(t+\tau) - x_n = \sum_{m=1}^{M} K_m \nu_{nm}. \tag{3.9}$$

Instead of directly working on (3.8) to calculate K, we find K that satisfy the following condition [54]:

$$|\Delta X_n| \leq \max\{\epsilon_n x_n, 1\}, \ n = 1, \cdots, N, \tag{3.10}$$

where ϵ_n can be found from ϵ to satisfy or approximately satisfy the condition in (3.8) as follows [54]. For a monomolecular reaction R_m involving molecular specie

S_n, the propensity function is $a_m(\mathbf{x}) = c_m x_n$, and the relative change in $a_m(\mathbf{x})$ is given by $\Delta a_m(\mathbf{x})/a_m(\mathbf{x}) = \Delta x_n/x_n$. Therefore, we can choose $\epsilon_n = \epsilon$ so that (3.10) is equivalent to (3.8).

For a bimolecular reaction R_m with two different reactants S_{n_1} and S_{n_2}, we have $a_m(\mathbf{x}) = c_m x_{n_1} x_{n_2}$. We can reasonably approximate $\Delta a_m(\mathbf{x})$ by $\Delta a_m(\mathbf{x}) \approx c_m \Delta x_{n_1} x_{n_2} + c_m x_{n_1} \Delta x_{n_2}$, where the typically small term $c_m \Delta x_{n_1} \Delta x_{n_2}$ has been discarded. Thus, the relative change in $a_m(\mathbf{x})$ is given by $\Delta a_m(\mathbf{x})/a_m(\mathbf{x}) \approx \Delta x_{n_1}/x_{n_1} + \Delta x_{n_1}/x_{n_1}$, and we can select $\epsilon_{n_1} = \epsilon_{n_2} = \epsilon/2$ to approximately satisfy (3.8). Similarly, for a bimolecular reaction R_m with a single molecular specie S_n, we can choose $\epsilon_n = \epsilon/[2 + 1/(x_n - 1)]$. Although formulas for selecting ϵ_n for trimolecular reactions are also derived by Cao *et al.* [54], we typically only need to consider elementary reactions including bimolecular and monomolecular reactions, as argued by Gillespie [47]. When one molecular specie S_n is the reactant of several reaction channels, ϵ_n takes the smallest value calculated from these reaction channels.

If we define $\tilde{\boldsymbol{\nu}}_n \overset{\triangle}{=} [\nu_{n1}, \cdots, \nu_{nM}]^T$, $n = 1, \cdots, N$, the mean and variance of ΔX_n can be found from (3.9) as

$$\begin{aligned} E[\Delta X_n] &= \tilde{\boldsymbol{\nu}}_n^T \boldsymbol{\theta} K, \\ \text{var}[\Delta X_n] &= \tilde{\boldsymbol{\nu}}_n^T \mathbf{C} \tilde{\boldsymbol{\nu}}_n K. \end{aligned} \tag{3.11}$$

We can regard the condition in (3.10) to be "substantially satisfied", if both the absolute mean and the standard derivation of ΔX_n are bounded by the right hand side of (3.10) [54]:

$$\begin{aligned} |E[\Delta X_n]| &\leq \max\{\epsilon_n x_n, 1\},\ n = 1, \cdots, N, \\ \sqrt{\text{var}[\Delta X_n]} &\leq \max\{\epsilon_n x_n, 1\},\ n = 1, \cdots, N. \end{aligned} \tag{3.12}$$

Substituting (3.11) into (3.12) and noticing that the minimum value of K is one, we obtain

$$K = \max\left\{ \min_{n\in[1,N]} \left\{ \frac{\max\{\epsilon_n x_n, 1\}}{|\tilde{\boldsymbol{\nu}}_n^T \boldsymbol{\theta}|}, \frac{\max\{\epsilon_n x_n, 1\}^2}{\tilde{\boldsymbol{\nu}}_n^T \mathbf{C} \tilde{\boldsymbol{\nu}}_n}, \right\}, 1 \right\}. \tag{3.13}$$

Note that calculating $\eta_m(\mathbf{x})$ and $\sigma_m^2(\mathbf{x})$, $m = 1, \cdots, M$, in (3.5), that is used in the K-selection formula (3.7), requires $f_{mm'}(\mathbf{x})$ which is calculated from (3.4), while the K-selection formula (3.13) does not need $f_{mm'}(\mathbf{x})$. Therefore, the K-selection based on (3.13) is more efficient, especially when M is comparable to or larger than N.

We obtain K in (3.13) by mimicking the τ-selection method proposed by Cao *et al.* [54]: bounding the absolute mean and the standard derivation of ΔX_n. In the τ-leap method, since K_m, $m = 1, \cdots, M$ are Poisson random variable and can take any nonnegative values, ΔX_n, $n = 1, \cdots, N$, are unbounded. Hence it is reasonable to obtain τ to "substantially" satisfy (3.10) by bounding the absolute mean and the standard derivation of ΔX_n. However, in our K-leap method, we have $0 \leq K_m \leq K$, $m = 1, \cdots, M$, i.e. random variables K_m, $m = 1, \cdots, M$ are bounded, which implies that we can choose K to satisfy (3.10) *strictly.* This leads to a more accurate and efficient method of determining the value of K as follows.

3.2.3 K-selection Method III

Since $\Delta X_n = \sum_{m=1}^{M} \nu_{nm} K_m$, taking into account the constraint $\sum_{m=1}^{M} K_m = K$, we have

$$|\Delta X_n| \leq K \max_{m\in[1,M]} \{|\nu_{nm}|\}. \tag{3.14}$$

Letting the right hand side of (3.14) be less than or equal to the right hand side of (3.10), we can satisfy the leap condition (3.10), which yields the following formula for selecting K:

$$K = \max\left\{ \min_{n\in[1,N]} \left\{ \frac{\max\{\epsilon_n x_n, 1\}}{\max_{m\in[1,M]}\{|\nu_{nm}|\}} \right\}, 1 \right\}. \tag{3.15}$$

It is clear that (3.15) requires less computation than (3.13). Moreover, our K-leap method using K calculated from (3.15) can strictly satisfy the leap condition (3.10). Of course, ϵ_n in (3.15) is approximately obtained from ϵ for reactions with an order greater than 1, thus, satisfying (3.10) only approximately satisfies (3.8). Nevertheless, our K-leap method, that uses anyone of (3.7), (3.13), and (3.15) to select K, can satisfy the leap condition C1 by properly choosing ϵ, since the total number of reactions occurred during one leap is a deterministic number K, and the times that each reaction channel fires are upper bounded by K. In contrast, the τ-leap method always has certain probability that the leap condition cannot be satisfied.

3.3 K-leap Simulation Algorithm

When $K = 1$, our K-leap method becomes the exact SSA. Hence, our K-leap method can adaptively change from the exact SSA to an approximate leap method, whenever the leap condition allows to do so. The SSA based on our K-leap method is summarized in the following algorithm:

Algorithm 3 (K-Leap SSA)

1. *Initialization (set the initial number of molecules, set $t \leftarrow 0$).*

2. *Calculate the propensity function, $a_m(\mathbf{x})$, $m = 1, \cdots, M$.*

3. *Calculate K from (3.7), (3.13), or (3.15).*

4. *If $K = 1$, execute an exact SSA step, and go to step 6.*

5. *If $K > 1$, generate τ according to the Gamma PDF (3.1), and generate K_m, $m = 1, \cdots, M$, according to the multinomial PDF (3.2).*

6. *Set $t \leftarrow t + \tau$, and update the state vector $\mathbf{X}(t) \leftarrow \mathbf{X}(t) + \sum_{m=1}^{M} \boldsymbol{\nu}_m k_m$.*

7. *Go to step 2.*

Algorithm 3 gives the outline of the K-leap simulation method. Several specific implementation issues, including the method of generating multinomial random variables and issues related to negative numbers of molecules, will be discussed in the ensuing section.

3.4 Implementation Issues

3.4.1 Efficient Generation of Multinomial Random Variables

The multinomial random variables K_m, $m = 1, \cdots, M$, can be generated from $M - 1$ binomial random variables as follows [59]. A realization of K_1, k_1, is first generated from a binomial random variable $\mathcal{B}(K, \theta_1)$, then, a realization of K_m, k_m, $m = 2, \cdots, M - 1$, is generated from $\mathcal{B}(K - \sum_{j=1}^{m-1} k_j, \theta_m / \sum_{j=m}^{M} \theta_j)$, and finally,

$k_M = K - \sum_{j=1}^{M-1} k_j$. To efficiently generate a binomial random variable $\mathcal{B}(n, p)$, we can employ the direct method [60] for small np, say $np < 10$, and use rejection method [61] for large np. Computational complexity of the direct method is proportional to np, while that of the rejection method for $np > 10$ is greater than that of the direct method for $np = 10$. To save computation, we have two strategies. One tactic is to order $\theta_1, \cdots, \theta_M$ in descending order, and generate K_m sequentially starting from the K_m with the largest θ_m. In some cases, the mean of the K_m with small propensity function is close to 0. After we generate those k_m with large θ_m, the remaining total number of firings is to be 0. Then we don't need to generate the remaining k_m. Thus, the average number of binomial random variable generators used to generate K_m will likely be less than the number of reaction channels in the system. This further reduces the computation in random number generation. In some cases, the reaction channels are easily been partitioned into two sets $\mathcal{R}_a$ and $\mathcal{R}_b$ so that $\sum_{j \in \mathcal{R}_a} \theta_j \ll \sum_{j \in \mathcal{R}_b} \theta_j$. The θ_m in each set changes quickly but those channels didn't change to the other set. Since θ_m changes quickly, it requires us to reorder them frequently, which consumes time. Then we can implement another tactic using these two groups without sorting them every step during simulation. We can first generate a number k_a from $\mathcal{B}(K, \sum_{j \in \mathcal{R}_a} \theta_j)$, then generate $K_m \in \mathcal{R}_a$ as multinomial random variables with parameters k_a, $\theta_m / \sum_{j \in \mathcal{R}_a} \theta_j$, and similarly generate $K_m \in \mathcal{R}_b$ as multinomial random variables with parameters $K - k_a$, $\theta_m / \sum_{j \in \mathcal{R}_b} \theta_j$. The value of k_a will most likely be a very small, close or equal to zero. Thus, the average number of binomial random variable generators used to generate $K_m \in \mathcal{R}_a$, M_a, will likely be less than the number of reaction channels in $\mathcal{R}_a$, $|\mathcal{R}_a|$, i.e., $M_a < |\mathcal{R}_a|$. This further reduces the computation in random number generation. In contrast, the binomial τ-leap method [41,42] requires M independent binomial random variable generators, and thus, it does not have the structure of multinomial random variables that can be explored to reduce computation.

3.4.2 Negative Numbers of Molecules

The third method of choosing K in (3.15) can avoid negative number of molecules for any $0 < \epsilon < 1$, since K is chosen to satisfy condition (3.14), which implies that we have $\Delta X_n \leq \epsilon_n x_n$. If the step size K determined using the first or the second method given in (3.7) or (3.13) is relatively large, there may be certain probability that negative numbers of molecules occur. However, we can simply reduce K to a proper value to avoid this problem. On the contrary, Gillespie's τ-leap method theoretically cannot avoid negative numbers of molecules by reducing the step size τ, since the realizations of a Poisson random variable are unbounded.

The step size K calculated from (3.15) is typically smaller than that determined from (3.7) or (3.13), which may slow simulation. Also, it may be inefficient to avoid negative numbers of molecules simply by reducing K, if (3.7) or (3.13) is used to determine K. This motivates us to develop a more efficient method without reducing K to deal with the problem of negative numbers of molecules, which we refer to as the partition method, as depicted in the following.

We partition the reaction channels into M_s sets: $\mathcal{R}_1, \cdots, \mathcal{R}_{M_s}$, where reaction channels in the same set can have some common reactants, but the channels in different sets does not share any reactants. Let $K_{\mathcal{R}_j}$ be the total number of firings of all channels in $\mathcal{R}_j$ during a leap. Clearly, $K_{\mathcal{R}_1}, \cdots, K_{\mathcal{R}_{M_s}}$ are multinomial random variables with parameters K, $\theta_{\mathcal{R}_1}, \cdots, \theta_{\mathcal{R}_{M_s}}$, where $\theta_{\mathcal{R}_j} = \sum_{m \in \mathcal{R}_j} \theta_m$. In each step, we use (3.7) or (3.13) to determine K, and generate realizations of $K_{\mathcal{R}_1}, \cdots, K_{\mathcal{R}_{M_s}}$ according to the multinomial distribution. To avoid negative numbers of molecules, we choose a number $k_{\max,\mathcal{R}_j} = \min_{m \in \mathcal{R}_j} \min_{\nu_{nm}<0} \{\lfloor x_n/|\nu_{nm}| \rfloor\}$ as an upper bound on $K_{\mathcal{R}_j}$. For example, if $\mathcal{R}_1$ has the following three coupled reaction channels: $S_1 + S_1 \rightarrow S_{10}$, $S_1 + S_2 \rightarrow S_{11}$, and $S_2 + S_3 \rightarrow S_{12}$, we have $k_{\max,\mathcal{R}_1} = \min\{\lfloor x_1/2 \rfloor, x_2, x_3\}$. If a realization of $K_{\mathcal{R}_j}$, say $k_{\mathcal{R}_j}$, is greater than $k_{\max,\mathcal{R}_j}$, we set $k_{\mathcal{R}_j} = k_{\max,\mathcal{R}_j}$. Then, for each set $\mathcal{R}_j$, we generate $k_{\mathcal{R}_j,1}, \cdots, k_{\mathcal{R}_j,M_{R_j}}$, where M_{R_j} is the total number of reaction channels in $\mathcal{R}_j$, for the number of firings of each channel in $\mathcal{R}_j$, according to a multinomial distribution with parameters $k_{\mathcal{R}_j}$, $\theta_m / \sum_{\mu \in \mathcal{R}_j} \theta_\mu, m \in \mathcal{R}_j$. Since K determined by (3.7) or (3.13) is typically larger than that determined by (3.15), this method is more efficient than the K-leap method with K calculated from (3.15).

In summary, taking into account the issue of negative numbers of molecules, we have three methods to run simulation: (i) use (3.7) or (3.13) to choose K and the partition method to avoid negative numbers of molecules, (ii) use (3.7) or (3.13) to choose K and the method by reducing K to avoid negative numbers of molecules, and (iii) use (3.15) to choose K. These three methods provide different tradeoffs between the simulation accuracy and time. Generally, for a given ϵ, method (i) offers the shortest simulation time but worst accuracy, method (iii) offers the longest simulation time but best accuracy, and method (ii) offers simulation time and accuracy in between those of methods (i) and (iii).

3.5 Numerical Examples

To demonstrate the accuracy and efficiency of our K-leap method, we now simulate three chemical reaction systems. Since selection of the step size is very important in leap methods, we will consider several different methods of selecting step size: K-selection according to (3.7), (3.13) and (3.15) for our K-leap method, and the τ-selection procedures given in (6) of Gillespie [39] and in (33) of Cao *et al.* [54] for the τ-leap method. To assess the accuracies of different simulation methods, we first obtain the histograms of the populations of each molecular species at the end of simulation from a series of repeated exact SSA runs. We then simulate the same chemical reaction system over the same time interval by the same number of runs, using the K- and τ-leap methods. As discussed in Sec. 2.5, the histogram distances (HD) between the results of the exact SSA and those of a leap method is employed to measure the simulation accuracy.

An important issue for the efficiency of stochastic simulations is generation of random numbers. In the τ-leap method, we use the Poisson random number generator in Press et al [62]. In our K-leap method, we employ the Gamma random number generator in Press et al. [62], and use $M-1$ binomial random number generators to generate M multinomial random numbers. In generating a binomial random variable $\mathcal{B}(n, p)$, we employ the BTPE algorithm [61] for $np > 10$, and the BG algorithm [60] for $np \leq 10$. All simulations are run in Matlab on a PC with a 2.99 GHz CPU and 1G-byte memory running Windows XP.

3.5.1 A System with Two Independent Reaction Channels

This simple example has two reaction channels:

$$S_1 \xrightarrow{c_1} \emptyset, \; S_2 + S_3 \xrightarrow{c_2} S_4. \tag{3.16}$$

These two reactions channels are independent, but we need to consider both reaction channels when selecting a step size in a leap SSA. If the initial number of S_1 molecules at $t = 0$ is $X_1(0) = \tilde{x}_1$, the pdf of $X_1(t)$ can be obtained from the CME as [33]

$$p(X_1(t) = x) = \frac{x}{x!(\tilde{x}_1 - x)!}[e^{-c_1 x}]^t[1 - e^{-ct}]^{\tilde{x}_1 - x}, \; x = 0, \cdots, \tilde{x}_1. \tag{3.17}$$

Hence, we can compare the histogram of $X_1(t)$ generated from a leap SSA with the pdf of $X_1(t)$ in (3.17). In our simulation, we set $X_1(0) = 3000$, $X_2(0) = 3000$, and $X_3(0) = 10^4$, choose $c_1 = 1$, $c_2 = 10^{-4}$. We run simulation 5×10^4 times, and each time starts from $t = 0$ and ends at $t = 2$. We then calculate the histogram of $X_1(2)$ and the distance between the histogram and the pdf of $X_1(2)$ using the histogram distance formula [56].

Figure 3.1 depicts the histogram distance versus CPU time. It is seen that our K-leap method outperforms both Gillespie's original τ-leap method [39] and the modified τ-leap method of Cao *et al.*, [54] since our K-leap method produces smaller histogram distance for a given CPU time, or, our K-leap method takes less CPU time for a give histogram distance. In our K-leap method, K-selection method 3 using (3.15) offers better performance than K-selection method 1 using (3.7) and method 2 using (3.13), while method 2 outperforms method 1, as expected.

To understand how our K-leap method improves performance, we list the average number of steps, CPU time, and the histogram distance for these leap methods in Table 3.1. We compare the performance of Gillespie τ-leap method and our $K-$leap

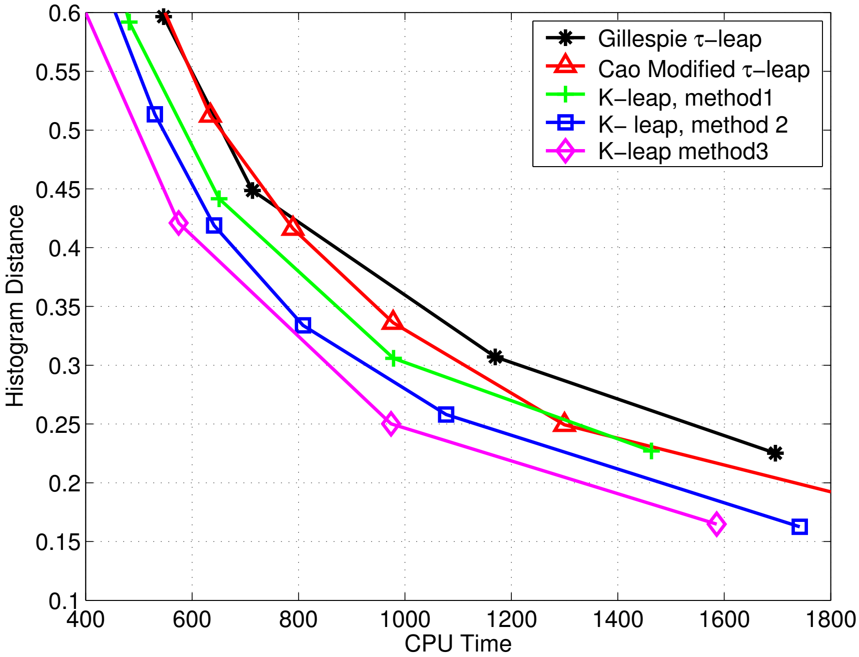


Figure 3.1: Histogram distance of $X_1(t)$ at $t = 2$ versus CPU time for two-channel reactions (3.16) with $c_1 = 1$, $c_2 = 10^{-4}$, $X_1(0) = 3000$, $X_2(0) = 3000$, and $X_3(0) = 10^4$. The τ-leap method uses (6) of Gillespie [39] to determine τ; Cao modified τ-leap method uses (33) of Cao *et al.* [54] to calculate τ; and K-leap method 1, 2, and 3, employ (3.7), (3.13), and (3.15), respectively, to calculate K. The histogram is obtained after 5×10^4 simulation runs. The CPU time is the total time (in seconds) of 5×10^4 runs.

method 1 in Table 3.1 (a), since these two methods use the similar approach to select the step size. For a given ϵ, it is seen that these two methods requires almost the same number of steps, and yield almost the same histogram distance, but our K-leap method needs less time. We then compare the performance of the τ-leap method of Cao *et al.* and our $K-$leap method 2 in Table 3.1 (b), since these two methods use the similar approach to select the step size. Similar observation to that from Table 3.1 (a) is seen from 3.1 (b): compared to Cao modified τ-leap method, our K-leap method takes less time to provide almost the same histogram distance. These observations suggest that for a given ϵ, our K-leap method does not significantly reduce histogram distance, but reduces simulation time. It is worth to emphasize that although our

Table 3.1: Average number of steps of one simulation run, CPU time of 5×10^4 runs (in seconds), and histogram distance of $X_1(2)$ for the example of two-channel reactions

(a) Gillespie τ-leap (G-τ leap) and K-leap method 1 (K-leap I)

	$\epsilon = 0.03$			$\epsilon = 0.0175$			$\epsilon = 0.0058$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
G-τ leap	38.3	389.9	0.866	65.2	649.7	0.526	195.6	2087.6	0.179
K-leap I	38.4	320.8	0.868	65.7	550.6	0.519	200.6	1809.1	0.169

*The CPU time for the exact SSA is 4438.9 second.

(b) Cao modified τ-leap (C-τ leap) and K-leap method 2 (K-leap II)

	$\epsilon = 0.1$			$\epsilon = 0.06$			$\epsilon = 0.02$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
C-τ leap	40.0	387.8	0.823	67.0	634.3	0.512	200.0	2034.6	0.165
K-leap, II	40.7	314.0	0.823	68.0	530.1	0.513	206.1	1741.9	0.162

(c) K-leap method 3 (K-leap III)

	$\epsilon = 0.1$			$\epsilon = 0.06$			$\epsilon = 0.02$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
K-leap, III	41.3	285.4	0.806	68.74	480.8	0.501	209	1585.8	0.163

K-leap method does not significantly reduces the histogram distance, it can avoid the small probability event where the number that a reaction channel fires in a leap is very large. In contrast, the τ-leap cannot avoid such small probability event, since the number of firings of a channel is a Poisson random variable. Therefore, our K-leap method increases simulation accuracy by avoiding such small probability event, which is not observable from the histogram distance. Comparing Table 3.1 (c) with (a) and (b), we see that our K-leap method 3 can further reduce simulation time while provide similar histogram distance.

The reduction of simulation time in our K-leap method 1 and 2 is primarily due to the random variable generators, while the reduction of simulation time in our K-leap method 3 is due to not only the binomial random variable generator, but also the simplicity of calculating K using (3.15). In our K-leap method, we use a

Gamma random variable generator and $M-1$ binomial random variable generator, while in the τ-leap method, M Poisson random variables are required. The binomial random variable generator we used is more efficient than the Poisson random variable generator, while the Gamma random variable generator offers similar speed to that of the Poisson random variable generator. We used the most efficient and accurate binomial and Poisson random variable generators that we are aware of. Investigation of efficient random variable generators is an important issue in stochastic simulation, but it is beyond the scope of this thesis.

Comparing Table 3.1 (a), (b), and (c), column by column, we can see the effect of the parameter ϵ has on the simulation time and accuracy. Since (3.8) satisfies the leap condition C1 better than (2.10), the modified τ-leap method and our K-leap method 2 and 3 that employ (3.8) to select the step size can use a relatively large ϵ, while still offer comparable performance to that of the τ-leap method and our K-leap method 1 that use (2.10) to select the step size.

3.5.2 Decaying-Dimerizing Reactions

This example is originally used by Gillespie [38,39] to test the τ-leap method. It includes the following four reactions:

$$\begin{aligned} &S_1 \xrightarrow{c_1} \emptyset,\ S_1 + S_1 \xrightarrow{c_2} S_2, \\ &S_1 + S_1 \xleftarrow{c_3} S_2,\ S_2 \xrightarrow{c_4} S_3. \end{aligned} \tag{3.18}$$

In these reactions, a monomer S_1 reversibly dimerizes to an unstable form S_2, which can convert to a stable form S_3. We simulate these reactions using the same values of rate constant and initial conditions as Gillespie [39]:

$$c_1 = 1,\ c_2 = 0.002,\ c_3 = 0.5,\ c_4 = 0.04, \tag{3.19}$$

Table 3.2: Average number of steps of one simulation run, CPU time of 5×10^4 runs (in seconds), and histogram distance (HD) of $X_1(10)$ for the example of decaying-dimerizing reactions

(a) Gillespie τ-leap (G-τ leap) and K-leap method 1 (K-leap I)

	$\epsilon = 0.04$			$\epsilon = 0.024$			$\epsilon = 0.012$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
G-τ leap	67.1	1170.4	0.557	120.7	2081	0.303	383.46	6722.8	0.082
K-leap I	67.1	972.4	0.551	120.7	1750.4	0.288	396.4	6125.6	0.074

* The CPU time for the exact SSA is 327,271 second.

(b) Cao τ-leap (C-τ leap) and K-leap method 2 (K-leap II)

	$\epsilon = 0.1$			$\epsilon = 0.06$			$\epsilon = 0.03$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
C-τ leap	68.1	1168.2	0.516	123.9	2119.4	0.259	399.3	6956.9	0.0752
K-leap II	68.1	942.2	0.516	123.9	1725.5	0.263	399.5	5948.8	0.0722

(c) K-leap method 3 (K-leap III)

	$\epsilon = 0.8$			$\epsilon = 0.6$			$\epsilon = 0.4$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
K-leap III	417.7	5877.5	0.089	556.88	8000.1	0.066	835.3	12510	0.044

and

$$X_1(0) = 4150,\ X_2(0) = 39565,\ X_3(0) = 3445. \tag{3.20}$$

We run simulation 5×10^4 times, and each time starts at $t = 0$ and ends at $t = 10$. We then calculate the histogram distance between the results of a leap method and those of the exact SSA.

Figure 3.2 depicts the histogram distance of $X_1(10)$ versus CPU time. It is seen that our K-leap method 1 outperforms the Gillespie's original τ-leap method, [39] while our K-leap method 2 outperforms both the Gillespie τ-leap method and the Cao τ-leap method of Cao *et al* [54]. These observations are also shown in Table 3.2 (a) and (b), where the average number of steps, CPU time, and the histogram distance for these leap methods are listed. As shown in Table 3.2 (c), although our K-leap

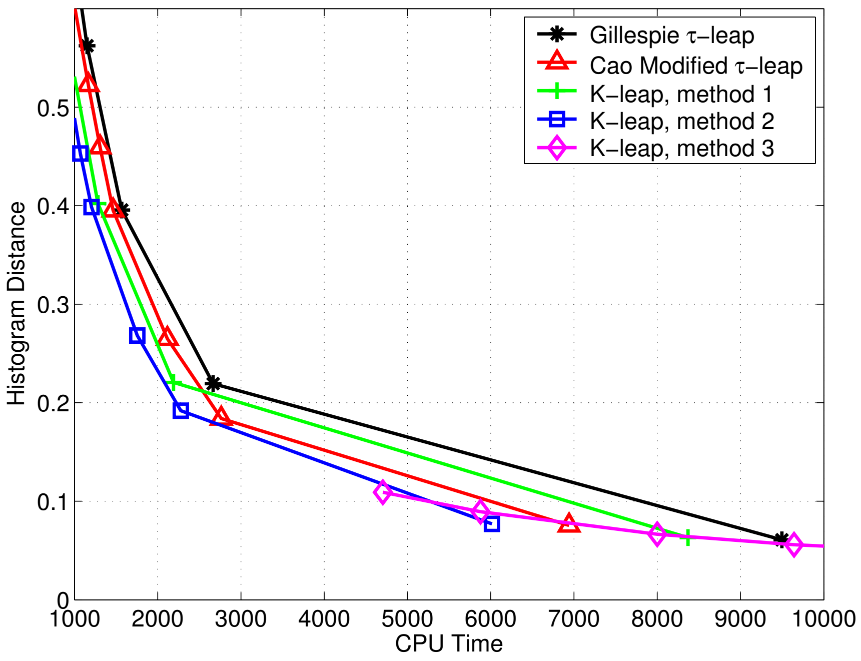


Figure 3.2: Histogram distance of $X_1(t)$ at $t = 10$ versus CPU time for decaying-dimerizing reactions (3.18) with rate constants (3.19) and the initial condition (3.20). The τ-leap method uses (6) of Gillespie [39] to determine τ; Cao Modified τ-leap method uses (33) of Cao *et al.* [54] to calculate τ; and K-leap method 1, 2, and 3, employ (3.7), (3.13), and (3.15), respectively, to calculate K. The histogram is obtained after 5×10^4 simulation runs. The CPU time is the total time (in seconds) of 5×10^4 runs.

method 3 uses a relatively large ϵ, it produces a small histogram distance as offered by other leap methods with a much smaller ϵ. For this reason, we only show the small histogram distance for our K-leap method 3 in Figure 3.2. It seems strange that using an ϵ as large as 0.8, our K-leap method 3 still produces a small histogram distance. A careful look at (3.15) and reactions (3.18) reveals that K is determined by bounding the worst case change in $a_2(\mathbf{x})$: K is selected so that the change in $a_2(\mathbf{x})$ caused by K firings of the second reaction channel is upper bounded by $\epsilon a_2(\mathbf{x})$. However, the event that the second reaction channel fires K times while other channels do not fire is a small probability event. Particularly, $a_3(\mathbf{x})$ is comparable to $a_2(\mathbf{x})$; hence, it is more often that reaction channels 2 and 3 fire comparable times in a leap, and the reduction of $X_2(t)$ caused by firings of the second reaction channel will be compensated for by the increase of $X_2(t)$ caused by firings of the third reaction channel. Essentially, although using (3.15) can strictly satisfy condition (3.10), it is too conservative for the situation where the number of a molecular species can be increased and decreased by several reaction channels with comparable probability. As a result, we can select a relatively large ϵ, while achieving relatively accurate simulation results.

3.5.3 Expression of LacZ/LacY Genes and Activities of the Proteins

The example of this system is the lactose operon model, which depicts the expression of LacZ and LacY genes and activity of LacZ and LacY proteins in *E.coli*, described in detail by Kierzek [37,63,64]. This model has 22 reaction channels, listed in Table 3.3. We run simulation using the exact SSA, starting at $t = 0$ with initial condition given by Kierzek, and ending at $t = 600$. We then use the result of exact SSA as the initial condition, and run simulation 10^4 times using exact SSA, our K-leap method and the binomial τ-leap methods of Chatterjee *et al.* [42], and of Tian and Burrage [41]. Each run starts at $t = 600$ and ends at $t = 601$. As depicted by Tian and Burrage [41], in each step, the number of RNAP molecules is generated from a Gaussian random variable with mean 35 and standard deviation 3.5, while the number of Ribosome molecules is generated from a Gaussian random variable with mean 350 and standard deviation 35. In our K-leap method, we used the method discussed in Section 3.4.2 to avoid negative numbers of molecules. We also partition the reaction channels into two set: $\mathcal{R}_a$ contains reaction channels 16, 17, 20, 21, 22, while $\mathcal{R}_b$ contains the remaining channels, since $\sum_{m\in\mathcal{R}_b}\theta_m \ll \sum_{m\in\mathcal{R}_a}\theta_m$, and then use the method in Section 3.4.1 to reduce computation.

Table 3.3: A full list of reaction channels and deterministic reaction rates of LacZ/LacY gene expression and protein activity

	Reaction channel	Reaction rate
R1	PLac + RNAP →PLacRNAP	0.17
R2	PLacRNAP →PLac + RNAP	10
R3	PLacRNAP →TrLacZ1	1
R4	TrLacZ1 →RbsLacZ+PLac+TrLacZ2	1
R5	TrLacZ2→TrLacY1	0.015
R6	TrLacY1→RbsLacY+TrLacY2	1
R7	TrLacY2→RNAP	0.36
R8	Ribosome+RbsLacZ→RbsRibosomeLacZ	0.17
R9	Ribosome+RbsLacY→RbsRibosomeLacY	0.17
R10	RbsRibosomeLacZ →Ribosome+RbsLacZ	0.45
R11	RbsRibosomeLacY →Ribosome+RbsLacY	0.45
R12	RbsRibosomeLacZ→TrRbsLacZ+RbsLacZ	0.4
R13	RbsRibosomeLacY →TrRbsLacY+RbsLacY	0.4
R14	TrRbsLacZ→LacZ	0.015
R15	TrRbsLacY→LacY	0.036
R16	LacZ→∅	6.42×10^{-5}
R17	LacY→∅	6.42×10^{-5}
R18	RbsLacZ→∅	0.3
R19	RbsLacY→∅	0.3
R20	LacZ+lactose→LacZlactose	9.52×10^{-5}
R21	LacZlactose→ product+ LacZ	431
R22	LacY→lactose+LacY	14

The histogram distance of the molecular number of the species product at $t = 601$ is depicted in Figure 3.3. It is seen that our K-leap method outperforms the binomial τ-leap methods. For other species, such as RbsLacY and TrRbsLacY, we observed, from our simulation results that are not shown here, that our K-leap method and the binomial τ-leap methods offer similar performance at $t = 601$. The reason why the performance difference of different leap methods only reflects in product is explained as follows. Since the propensity functions of reaction channel R_{20}, R_{21} and R_{22} are two order of magnitude higher than the propensity functions of other reaction channels, these three channels fire much more frequently than other channels, which implies

Table 3.4: CPU time (in seconds) of 10^4 simulation runs and speedup over Gillespie's exact SSA of the K-leap method for the example of LacZ/LacY. The speedup is defined as the CPU time of the K-leap method divided by the CPU time of the exact SSA.

Method	CPU time	Speedup
K-leap ($\epsilon = 0.01$)	13704.0	20.1
K-leap ($\epsilon = 0.0125$)	8908.2	30.9
K-leap ($\epsilon = 0.015$)	6243.9	44.0
K-leap ($\epsilon = 0.0175$)	4658.5	59.0
K-leap ($\epsilon = 0.02$)	3660.3	75.1
Exact SSA	2.75×10^5	1

that the numbers of reactions occurred during a leap for these three channels are much larger than the numbers of reactions of other channels. Therefore, different leap methods offer similar performance for the species involved in channels other than R_{20}, R_{21} and R_{22}. For the species involved in R_{20}, R_{21} and R_{22}, only product appears at one side of the reaction, and other species appear at both sides of these reactions. Therefore, it is the product that can reveal the performance difference of different leap methods.

The comparison between the simulation time of the K-leap method and Gillespie's exact SSA is shown in Table 3.4. Comparing to the exact SSA, the K-leap method reduces the simulation time by about 20 times when $\epsilon = 0.01$, and about 75 times when $\epsilon = 0.02$, while it produces an histogram distance of 0.062 when $\epsilon = 0.01$, and 0.174 when $\epsilon = 0.02$. Therefore, the K-leap method significantly improves simulation efficiency, while providing relatively accurate results in this example.

3.6 Concluding Remarks

Leap methods are promising for accelerating stochastic simulation, while providing acceptable simulation accuracy. The τ-leap method proposed by Gillespie [38, 39]

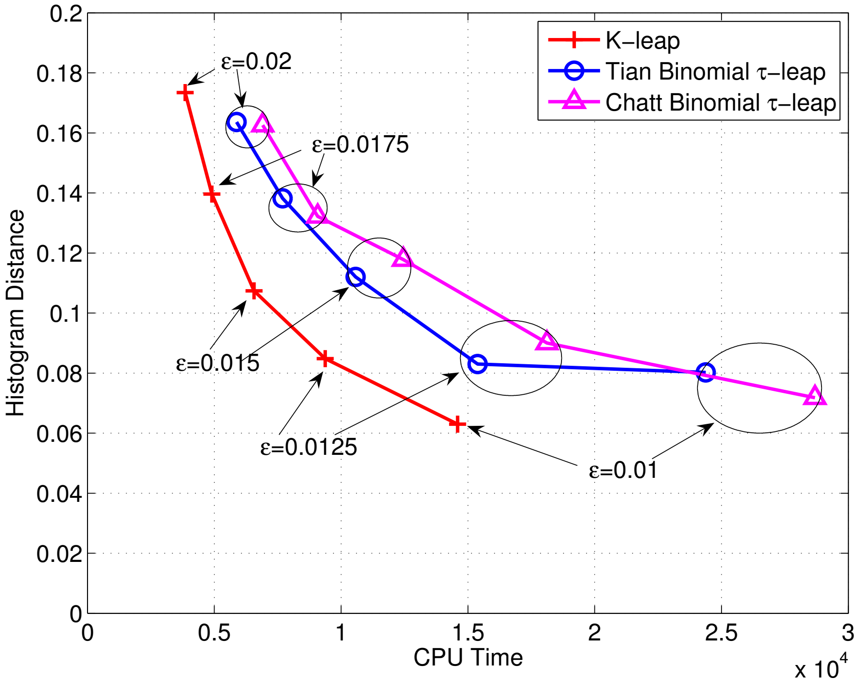


Figure 3.3: Histogram distance of the molecular number of the product at $t = 601$ versus CPU time for the example of LacZ/LacY. The histogram is obtained after 10^4 simulation runs. The CPU time is the total time (in seconds) of 10^4 runs.

has received much attention and been improved in many aspects [40–42, 54, 65]. In the τ-leap method, the number of firings of each reaction channel during a leap is a Poisson random variable, whose value is unbounded, although the probability that it takes very large value can be small. This runs the risk of violating the leap condition, thereby causing large simulation inaccuracy. In this chapter, we developed an alternative leap method, K-leap method, in which we constrain the total number of reactions in during leap to be a predetermined number K. As the number of firings of each reaction channel during a leap is bounded, our K-leap method can better satisfy the leap condition, thereby improving simulation accuracy. Moreover, since our K-leap method becomes the exact SSA when $K = 1$, it can naturally fold back to the exact SSA, if leaping is inappropriate due to the sensitivity of the propensity functions to population changes.

Our K-leap method is different from the k_α-leap method proposed by Gillespie [38]. In the k_α-leap method, only the number of times that a reaction channel α fires during a leap is constrained to be a constant k_α, while the other channels can fire an unbounded number of times, Therefore, the k_α-leap method has the same problem of the τ-leap method. In contrast, our K-leap method is designed to avoid the unbounded number of firings of each reaction channel, which can improve simulation accuracy.

Further improvement on our K-leap method is possible. For example, the estimated-midpoint leap [38] and implicit leap method [65], that are employed to improve the accuracy of the τ-leap method, can also be used to improve our K-leap method. One issue that remains unexplored in our K-leap method is the effect of the constraint that we impose on the number of reactions occurred during each leap. While this constraint can enforce the leap condition, it also limits the granularity of the total number of reactions occurred during certain time period. Although the number of reactions occurred for each channel during a leap is a random variable which can take any integer values from 0 to K, constraining the total number of reactions to be a constant K may cause certain states of the reaction system to be unreachable. The impact of such a constraint on simulation accuracy may be worth further investigation.

CHAPTER 4

Modified K-Leap Methods for Accelerated Stochastic Simulation of Gene Networks

4.1 Motivation

While Gillespie's exact SSA allows one to simulate the occurrence of every reacting event in a gene network or a more general chemically reacting system [33], it requires huge computation to produce a realization of the system's evolution, when the total molecular population of the system is relatively large. Recently, significant effort has been made to speed up stochastic simulation. Particularly, the leap method allows each reaction channel to fire more than once during each simulation step or leap, thereby accelerating simulation. In the τ-leap method [38, 39], the number of times that a reaction channel fires in a time interval of duration τ is approximated by a Poisson random variable. The deterministic number τ is determined according to the leap condition, which says that the state change in any leap should be small enough that no propensity function will experience a macroscopically significant change in its value. However, as the realization of a Poisson random variable can be any nonnegative integer, there always is a certain probability that the leap condition is violated, which will cause large simulation inaccuracy. In the extreme case, the τ-leap method can produce negative numbers of molecules [40–42]. The binomial τ-leap method attempt to avoid negative numbers of molecules by approximating Poisson random variables with binomial random variables [41, 42]. The modified τ-leap method can avoid negative numbers of molecules and is particularly efficient for simulating systems where certain molecular species have small number of molecules while the other species have large number of molecules [40].

Recently, we developed a K-leap method for stochastic simulation of genetic networks or chemically reacting systems [57, 58]. Unlike the τ-leap method where the time τ that is leaped over in a step is calculated from the leap condition, the K-leap method constrains the total number of reactions occurred during a leap to be deterministic number K that is calculated from the leap condition. Since the number of times that each channel fires during a leap is upper bounded by K, the leap condition can be better satisfied, thereby improving simulation accuracy. Moreover, the K-leap method can exploit the fact that some reaction channels may not fire during a leap to improve simulation speed.

In this chapter, we explore the ideas behind the modified τ-leap method [40] and the K-leap [58] to improve simulation efficiency. In the modified τ-leap method, all reaction channels are partitioned into two sets: critical reactions where the number of reactant molecules is small and noncritical reactions where the number of reactant molecules is relatively large. Then, the exact SSA is applied to the critical reactions and the τ-leap method is applied to the noncritical reactions. Since the exact SSA is a special case of the K-leap method when $K = 1$, if we replace the exact SSA with the K-leap method in the modified τ-leap method, we can improve simulation speed. This leads to a hybrid of K- and τ-leap methods that we will present in this chapter. We also develop a modified K-leap method where the K-leap method is applied to both the critical and noncritical reactions with different values of K. Using numerical examples, we will demonstrate that compared to the modified τ-leap method and the K-leap method, the hybrid leap method and the modified K-leap method offer better performance in terms of simulation speed and accuracy.

The remaining part of this chapter is organized as follows: In Section 4.2 we develop the hybrid leap method and the modified K-leap method. In Section 4.3 numerical examples are used to demonstrate the performance of the new leap methods, and conclusions are drawn in Section 4.4.

4.2 The Hybrid τ/K-leap Method and Modified K-leap Method

Let us consider the modified τ-leap method. Although the molecular number of each individual species involved in critical reactions is small, the total molecular number of all species involved in the critical reactions can be relatively large, which implies that the propensity $a_0^c(\mathbf{x})$ can be relatively large. Since the expected value of τ'' is $E[\tau''] = 1/a_0^c(\mathbf{x})$, τ'' can be relatively small with large probability. Therefore, the step size of a leap, τ, is often limited by τ'', because τ' can be large relative to τ'' with large probability. This implies that simulation speed of the modified τ-leap method is often limited by simulating the occurrence of critical reactions.

If we denote the total number of critical reactions that occur during a leap as K_c, we have $K_c = 1$ if $\tau'' \leq \tau'$ and $K_c = 0$ if $\tau'' > \tau'$ in the modified τ-leap method. If we allow K_c to be a positive integer that is greater than one but still small relative to $\min_{m \in \mathcal{R}_c}\{k_{m,max}\}$ with a large probability, we would expect that simulation speed can be significantly improved while making a minor sacrifices in simulation accuracy. Essentially, although the modified τ-leap method can avoid negative number of molecules for those species involved in critical reactions, it may be overly restrictive to apply the exact SSA to critical reactions, thereby slowing simulation. If we combine the ideas behind the modified τ-leap method and the K-leap method, we can choose K_c more flexibly and improve simulation speed. This leads to the two new leap methods that we will present next.

4.2.1 The Hybrid τ/K-leap Method

We partition the reactions into critical reactions and noncritical reactions using the criterion defined in the modified τ-leap method. We then employ the K-leap method instead of the exact SSA that is used in the modified τ-leap method to simulate the occurrence of critical reactions, while using the τ-leap method to simulate the occurrence of noncritical reactions. Because two different types of leap methods are used in this new simulation method, we refer to this leap method as the hybrid τ/K-leap method.

As in the modified τ-leap method, we calculate a tentative step size τ' using (2.11) or (2.16) with the reaction index running over $\mathcal{R}_{nc}$ for noncritical reactions. We then choose a small number K_c for the total number of critical reactions that occur during a leap. While K_c can be any integers between one and $\min_{m\in\mathcal{R}_c}\{k_{m,\max}\}$, we typically choose K_c to be between one and five. The K-leap method then generates another tentative step size τ'' from the Gamma PDF with $a_0(\mathbf{x})$ replaced by $a_0^c(\mathbf{x})$ and K replaced by K_c. Then the actual step size is chosen as $\tau = \min\{\tau', \tau''\}$.

For noncritical reactions, we generate $K_m, m \in \mathcal{R}_{nc}$ from a Poisson random variable with mean $a_m(\mathbf{x})\tau$. Let us use $\{K_m, m \in \mathcal{R}_c\}$ to represent the set of all K_m's, $m \in \mathcal{R}_c$. For critical reactions, if $\tau'' \leq \tau'$, then we generate $\{K_m, m \in \mathcal{R}_c\}$ from the multinomial PDF (3.2) with K replaced by K_c and θ_m calculated as $\theta_m = a_m(\mathbf{x})/a_0^c(\mathbf{x})$, $m \in \mathcal{R}_c$. It is a little bit more involved to generate $\{K_m, m \in \mathcal{R}_c\}$ when $\tau'' > \tau'$. In the modified τ-leap method, $\tau'' > \tau'$ implies that no critical reaction occurs during a leap, and thus, we have $K_m = 0$, for all $m \in \mathcal{R}_c$. However, in our hybrid τ/K-leap method, since $K_c \geq 1$, $\tau'' > \tau'$ does not imply that no critical reaction occurs during a leap, but the total number of critical reactions occurring during a leap is less than K_c.

In order to generate $\{K_m, m \in \mathcal{R}_c\}$ when $\tau'' > \tau'$, we need to first obtain the joint distribution of $\{K_m, m \in \mathcal{R}_c\}$ given τ and $\sum_{m\in\mathcal{R}_c} K_m < K_c$ which is denoted as $p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau)$. Defining a random variable $K_c' = \sum_{m\in\mathcal{R}_c} K_m$, we prove in the Appendix B that $p(K_m, m \in \mathcal{R}_c|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau)$ is given as follows

$$\begin{aligned} &p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau) \\ =&p(K_c'|K_c' < K_c, \tau)p(K_m, m \in \mathcal{R}_c|\sum_{m\in\mathcal{R}_c} K_m = K_c', \tau), \end{aligned} \tag{4.1}$$

where $p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m = K_c', \tau)$ is a multinomial PDF with parameters K_c' and $\theta_m = a_m(\mathbf{x})/a_0^c(\mathbf{x})$, $m \in \mathcal{R}_c$, and $p(K_c'|K_c' < K_c, \tau)$ is given by

$$p(K_c'|K_c' < K_c, \tau) = \begin{cases} \frac{[a_0^c(\mathbf{x})\tau]^{K_c'}}{K_c'!\sum_{j=0}^{K_c-1}[a_0^c(\mathbf{x})\tau]^j/j!}, & K_c' = 0, \cdots, K_c - 1 \\ 0, & \text{otherwise.} \end{cases} \tag{4.2}$$

Based on $p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau)$ given in (4.1), we can generate $K_m, m \in \mathcal{R}_c$ as follows. We first generate K_c' using the PDF $p(K_c'|K_c' < K_c, \tau)$ in (4.2) and then generate $\{K_m, m \in \mathcal{R}_c\}$ from the multinomial PDF $p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m = K_c', \tau)$. From the derivations of $p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau)$ in the Appendix B, we know that $p(K_c'|K_c' < K_c, \tau)$ is a truncated Poisson PDF. Therefore, it is not difficult to generate K_c'.

Finally, after generating τ, K_m, $m = 1, \cdots, M$, we can update the state vector using (2.15) and update the time as $t = t + \tau$. The hybrid τ/K-leap method is summarized in the following algorithm:

Algorithm 4 (The Hybrid τ/K-Leap Method)

1. *Initialization (set the initial number of molecules and initial time t, set n_c and K_c)*

2. *Calculate the propensity function, $a_m(\mathbf{x})$,$m = 1, \cdots, M$ and their sum $a_0(\mathbf{x}) = \sum_{m=1}^{M} a_m(\mathbf{x})$.*

3. *Partition all reactions into critical reactions and noncritical reactions: If a reaction R_m has $a_m(\mathbf{x}) > 0$ and $k_{m,\max} \leq n_c$, where $k_{m,\max}$ is defined in the* (2.18)*, then it is a critical reaction; otherwise, it is a noncritical reaction.*

4. *Calculate a tentative step size τ' using* (2.11) *or* (2.16) *with the index running over only the noncritical reactions.*

5. *Compute $a_0^c(\mathbf{x})$, the sum of the propensity functions of the critical reactions. Generate τ'' according to the Gamma PDF with parameters K_c and $a_0^c(\mathbf{x})$.*

6. *Set $\tau = \min(\tau', \tau'')$. For all noncritical reactions, generate K_m, $m \in \mathcal{R}_{nc}$, according to the Poisson random variable with mean $a_m(\mathbf{x})\tau$. For all critical reactions, if $\tau' < \tau''$, generate K_m, $m \in \mathcal{R}_c$, according to the PDFs in* (4.1) *and* (4.2)*; otherwise, generate K_m, $m \in \mathcal{R}_c$, according to the multinomial PDF in* (3.2) *with parameters K_c and $\theta_m = a_m(\mathbf{x})/a_c^0(\mathbf{x})$, $m \in \mathcal{R}_c$.*

7. *Update $t \leftarrow t + \tau$ and update the state vector $\mathbf{x} \leftarrow \mathbf{x} + \sum_{m=1}^{M} \boldsymbol{\nu}_m K_m$.*

8. *Record $(t, \mathbf{x})$ if desired. Go to step 2, or else stop.*

4.2.2 The Modified K-Leap Method

Considering several nice features of the K-leap method that we describe in Section 3.2, we would expect that simulation speed and accuracy could be improved, if we employ the K-leap method instead of the τ-leap method to simulate the occurrence of noncritical reactions. This leads to the modified K-leap method that we depict in the following. Again, we partition the reactions into critical reactions and noncritical reactions using the criterion defined in the modified τ-leap method. We then apply the K-leap method to both critical and noncritical reactions but constrain the total number of critical and noncritical reactions occurring during a leap to two different numbers, K_c and K_{nc}, respectively.

For noncritical reactions, we first calculate K_{nc} using (3.7) or (3.13) or (3.15) with the index running over noncritical reactions. We then generate a tentative time step τ' from the Gamma PDF with parameters K_{nc} and $a_0^{nc}(\mathbf{x})$. For critical reactions, we first select a small value for K_c as we describe in the hybrid τ/K-leap method, and then generate a tentative time step τ'' from the Gamma PDF with parameters K_c and $a_0^c(\mathbf{x})$. The actual time that is leaped over is chosen as $\tau = \min\{\tau', \tau''\}$.

Let us use $\{K_m, m \in \mathcal{R}_{nc}\}$ to represent the set of all K_m's, $m \in \mathcal{R}_{nc}$. If $\tau' < \tau''$, for noncritical reactions, we generate $\{K_m, m \in \mathcal{R}_{nc}\}$ according to the multinomial PDF (3.2) with parameters K_{nc} and $\theta_m = a_m(\mathbf{x})/a_0^{nc}(\mathbf{x})$, $m \in \mathcal{R}_{nc}$ where $a_0^{nc}(\mathbf{x}) = \sum_{m\in\mathcal{R}_{nc}} a_m(\mathbf{x})$; for critical reactions, we generate $\{K_m, m \in \mathcal{R}_c\}$ according to the PDFs in (4.1) and (4.2) using the method described in Section 4.2.1. If $\tau' = \tau''$, we generate $\{K_m, m \in \mathcal{R}_{nc}\}$ according to the multinomial PDF (3.2) with parameters K_{nc} and $\theta_m = a_m(\mathbf{x})/a_0^{nc}(\mathbf{x})$, $m \in \mathcal{R}_{nc}$, and generate $\{K_m, m \in \mathcal{R}_c\}$ according to the multinomial PDF (3.2) with parameters K_c and $\theta_m = a_m(\mathbf{x})/a_0^c(\mathbf{x})$, $m \in \mathcal{R}_c$. If $\tau' > \tau''$, for critical reactions, we generate $\{K_m, m \in \mathcal{R}_c\}$ according to the multinomial PDF (3.2) with parameters K_c and $\theta_m = a_m(\mathbf{x})/a_0^c(\mathbf{x})$, $m \in \mathcal{R}_c$; for noncritical reactions, we need to generate $\{K_m, m \in \mathcal{R}_{nc}\}$ according to their joint conditional PDF $p(\{K_m, m \in \mathcal{R}_{nc}\}| \sum_{m\in\mathcal{R}_{nc}} K_m < K_{nc}, \tau)$. Similar to the derivation of (4.1), we can obtain $p(\{K_m, m \in \mathcal{R}_{nc}\}|\sum_{m\in\mathcal{R}_{nc}} K_m < K_{nc}, \tau)$ as

$$\begin{aligned} & p(\{K_m, m \in \mathcal{R}_{nc}\}| \sum_{m\in\mathcal{R}_{nc}} K_m < K_{nc}, \tau) \\ & = p(K'_{nc}|K'_{nc} < K_{nc}, \tau) p(\{K_m, m \in \mathcal{R}_{nc}\}| \sum_{m\in\mathcal{R}_{nc}} K_m = K'_{nc}, \tau), \end{aligned} \tag{4.3}$$

where K'_{nc} is defined as $K'_{nc} \overset{\triangle}{=} \sum_{m\in\mathcal{R}_{nc}} K_m$, $p(K'_{nc}|K'_{nc} < K_{nc}, \tau)$ can be obtained from (4.2) by replacing K'_c with K'_{nc}, K_c with K_{nc} and $a_0^c(\mathbf{x})$ with $a_0^{nc}(\mathbf{x})$, and $p(\{K_m, m \in \mathcal{R}_{nc}\}|\sum_{m\in\mathcal{R}_{nc}} K_m = K'_{nc}, \tau)$ is a multinomial PDF with parameters K'_{nc} and $\theta_m = a_m(\mathbf{x})/a_0^{nc}(\mathbf{x})$, $m \in \mathcal{R}_{nc}$. Similar to the generation of $\{K_m, m \in \mathcal{R}_c\}$ from (4.1) and (4.2), we can first generate K'_{nc} from $p(K'_{nc}|K'_{nc} < K_{nc}, \tau)$ and then generate $\{K_m, m \in \mathcal{R}_{nc}\}$ from the multinomial PDF $p(\{K_m, m \in \mathcal{R}_{nc}\}|\sum_{m\in\mathcal{R}_{nc}} K_m = K'_{nc}, \tau)$.

Finally, after generating τ, K_m, $m = 1, \cdots, M$, we can update the state vector using (2.15) and update the time as $t = t + \tau$. The modified K-leap method is summarized in the following algorithm:

Algorithm 5 (The Modified K-Leap Method)

1. *Initialization (set the initial number of molecules and initial time t, set n_c and K_c)*

2. *Calculate the propensity function, $a_m(\mathbf{x})$,$m = 1, \cdots, M$ and their sum $a_0(\mathbf{x}) = \sum_{m=1}^{M} a_m(\mathbf{x})$.*

3. *Partition all reactions into critical reactions and noncritical reactions: If a reaction R_m has $a_m(\mathbf{x}) > 0$ and $k_{m,\max} \le n_c$, where $k_{m,\max}$ is defined in the* (2.18)*, then it is a critical reaction; otherwise, it is a noncritical reaction.*

4. *Calculate K_{nc} using* (3.7) *or* (3.13) *or* (3.15) *with the index running over noncritical reactions. Generate a tentative time step τ' from the Gamma PDF with parameters K_{nc} and $a_0^{nc}(\mathbf{x})$.*

5. *Generate another tentative time step τ'' according to the Gamma PDF with parameters K_c and $a_0^c(\mathbf{x})$.*

6. *Set $\tau = \min(\tau', \tau'')$. If $\tau' < \tau''$, generate $\{K_m, m \in \mathcal{R}_{nc}\}$ according to the multinomial PDF* (3.2) *with parameters K_{nc} and $\theta_m = a_m(\mathbf{x})/a_0^{nc}(\mathbf{x})$, $m \in \mathcal{R}_{nc}$, and generate $\{K_m, m \in \mathcal{R}_c\}$ according to the PDFs* (4.1) *and* (4.2)*. If $\tau' = \tau''$, generate $\{K_m, m \in \mathcal{R}_{nc}\}$ according to the multinomial PDF* (3.2) *with parameters K_{nc} and $\theta_m = a_m(\mathbf{x})/a_0^{nc}(\mathbf{x})$, $m \in \mathcal{R}_{nc}$, and generate $\{K_m, m \in \mathcal{R}_c\}$ according to the multinomial PDF* (3.2) *with parameters K_c and $\theta_m = a_m(\mathbf{x})/a_0^c(\mathbf{x})$, $m \in \mathcal{R}_c$. If $\tau' > \tau''$, generate $\{K_m, m \in \mathcal{R}_{nc}\}$ according to the PDF* (4.3)*, and generate $\{K_m, m \in \mathcal{R}_c\}$ according to the multinomial PDF* (3.2) *with parameters K_c and $\theta_m = a_m(\mathbf{x})/a_0^c(\mathbf{x})$, $m \in \mathcal{R}_c$.*

7. *Update $t \leftarrow t + \tau$ and update the state vector $\mathbf{x} \leftarrow \mathbf{x} + \sum_{m=1}^{M} \boldsymbol{\nu}_m K_m$.*

8. *Record $(t, \mathbf{x})$ if desired. Go to step 2, or else stop.*

The readers can also find some practical implementation methods in [58] for the K-leap method that include an efficient method for generating multinomial random variables and a method for handling negative number of molecules.

4.3 Numerical Examples

In order to demonstrate the performance of our new leap methods, we simulated expression of LacZ/LacY genes based on a model in [37, 41] and a chemical reaction system. To assess the accuracies of different simulation methods, we first obtain the histograms of the populations of each molecular species at the end of simulation from a series of repeated exact SSA runs. We then simulate the same chemical reaction system over the same time interval by the same number of runs, using the modified τ-leap, hybrid τ/K-leap and modified K-leap methods, and obtain the histograms of the simulation results. Finally, we employ the histogram distances between the results of the exact SSA and those of a leap method, as used in [54], to measure the simulation accuracy: a small histogram distance implies high simulation accuracy. In modified τ-leap and hybrid τ/K-leap methods, we used (2.16) to calculate τ for noncritical reactions. In modified K-leap method, we used (3.13) to calculate K_{nc}.

An important issue for the efficiency of stochastic simulations is the generation of random numbers. We use the Poisson random number generator and the Gamma random number generator in [62]. We employ $M - 1$ binomial random number generators to generate M multinomial random numbers [58]. In generating a binomial random variable $\mathcal{B}(n, p)$, we use the BTPE algorithm of [61] for $np > 10$, and the BG algorithm of [60] for $np \leq 10$. To generate a random variable that follows a truncated Poisson distribution in (4.2), we modify the Poisson random number generator in [62] as follows. The rejection method is used by the Poisson random number generator in [62]. We further reject any number that is greater than K_c, so that the random variable generated follows the distribution in (4.2). All simulations are run in Matlab on a PC with a 3.20 GHz CPU and 2G-byte memory running Windows XP.

4.3.1 A System with Several Critical Reaction Channels

As we discussed earlier, if we can choose $K_c > 1$, it is expected that we can significantly improve simulation speed. To verify this, we simulated the following reacting system that consists of 12 reaction channels:

$$\begin{aligned}
&S_1 \xrightarrow{c_1} S_2 + S_3,\ S_2 \xrightarrow{c_2} S_1,\ S_3 \xrightarrow{c_3} \emptyset,\ S_4 \xrightarrow{c_4} \emptyset,\ S_4 + S_5 \xrightarrow{c_5} S_6,\\
&S_5 + S_5 \xrightarrow{c_6} S_7,\ S_5 \xrightarrow{c_7} \emptyset,\ S_6 + S_6 \xrightarrow{c_8} S_{10},\ S_6 \xrightarrow{c_9} \emptyset,\\
&S_7 + S_8 \xrightarrow{c_{10}} S_9 + S_{10},\ S_9 + S_{11} \xrightarrow{c_{11}} S_{12} + S_{13},\ S_{12} + S_{13} \xrightarrow{c_{12}} S_9 + S_{11}.
\end{aligned} \tag{4.4}$$

In our simulations, the probability rate constants are chosen to be:

$$\begin{aligned}
&c_1 = 10,\ c_2 = 10,\ c_3 = 1 \times 10^{-6},\ c_4 = 1 \times 10^{-6},\ c_5 = 1.5 \times 10^{-6},\\
&c_6 = 1 \times 10^{-8},\ c_7 = 1 \times 10^{-6},\ c_8 = 3.42 \times 10^{-8},\ c_9 = 1 \times 10^{-6},\\
&c_{10} = 1 \times 10^{-6},\ c_{11} = 1.28 \times 10^{-7},\ c_{12} = 3.2 \times 10^{-7}.
\end{aligned} \tag{4.5}$$

and the initial state is set to be

$$\begin{aligned}
&X_1(0) = 10,\ X_2(0) = 10,\ X_3(0) = 5 \times 10^3,\ X_4(0) = 2 \times 10^4,\ X_5(0) = 5 \times 10^4,\\
&X_6(0) = 8 \times 10^2,\ X_7(0) = 6 \times 10^4,\ X_8(0) = 5 \times 10^4,\ X_9(0) = 1 \times 10^3,\\
&X_{10}(0) = 1 \times 10^3,\ X_{11}(0) = 1.5 \times 10^3,\ X_{12}(0) = 1.2 \times 10^3,\ X_{13}(0) = 1.4 \times 10^3.
\end{aligned} \tag{4.6}$$

We set the parameter n_c=20. From the initial state given in (4.6) and n_c, we find that there are two critical channels R_1 and R_2. We run simulation 5×10^4 times, and each time starts at $t = 0$ and ends at $t = 2$. We then calculate the histogram distance between the results of a leap method and those of the exact SSA.

Figure 4.1 depicts the histogram distance of $X_4(2)$ versus CPU time for different leap methods. For the modified τ-leap method, we plot the simulation results for several different values of the parameter ϵ. It is seen that when $\epsilon > 0.002$, we cannot reduce simulation time by increasing ϵ. Essentially, although we can increase the step size of the τ-leap method, τ', by increasing ϵ, when $\epsilon > 0.002$, the actual step size τ is limited by the step size τ'' generated from the exact SSA, because $\tau = \min(\tau', \tau'')$. This confirms our claim about the modified τ-leap method in Section

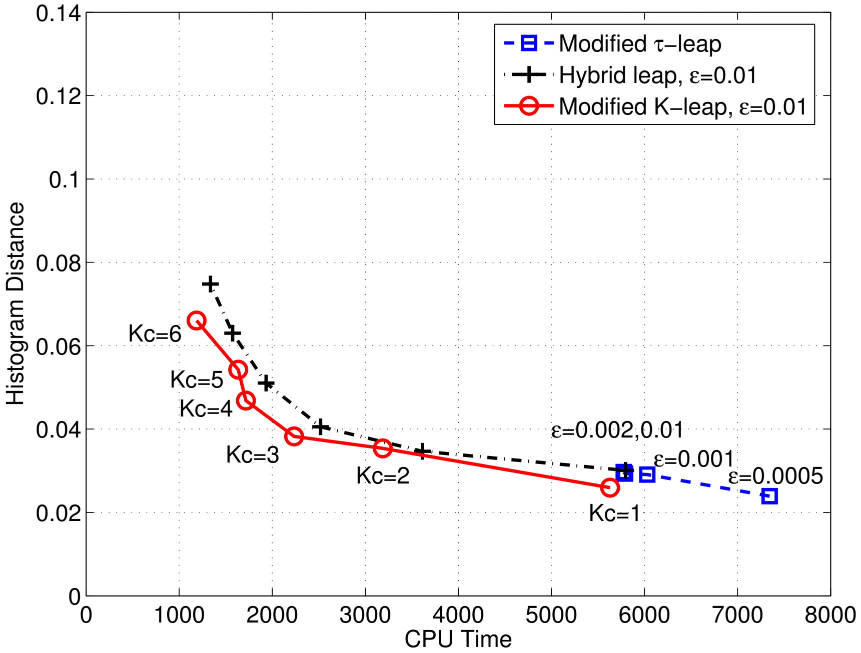


Figure 4.1: Histogram distance of modified K-leap, hybrid τ/K-leap and modified τ-leap method of $X_4(t)$ at $t = 2$ versus CPU time for the reacting system given in (4.4) with rate constants in (4.5) and initial state in (4.6). The histogram is obtained after 5×10^4 simulation runs and the CPU time is the total time (in seconds) of 5×10^4 runs.

4.2 that motivates us to develop our two new leap methods. It is seen from Fig. 4.1 that when $K_c = 2$, the modified K-leap method only needs almost half of the simulation time of the modified τ-leap method, while providing almost the same histogram distance. The hybrid τ/K-leap method requires a little more simulation time than the modified K-leap method, but the simulation time is still much less than that of the modified τ-leap method, while providing almost the same histogram distance. We can further increase simulation speed of the modified K-leap method and the hybrid τ/K-leap method by increasing K_c. When $K_c = 6$, the simulation time of the modified K-leap method and hybrid τ/K-leap method is less than 1/5 of the simulation time of the modified τ-leap method, while the histogram distance of the modified K-leap method and the hybrid τ/K-leap method is slightly larger than

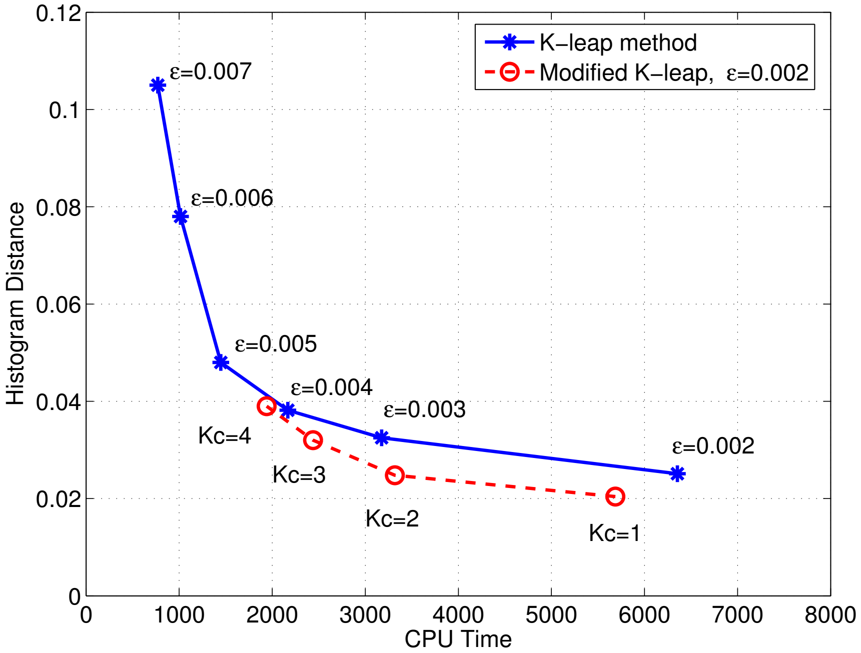


Figure 4.2: Histogram distance K-leap and modified K-leap method of $X_4(t)$ at $t = 2$ versus CPU time for the reacting system given in (4.4) with rate constants in (4.5) and initial state in (4.6). The histogram is obtained after 5×10^4 simulation runs and the CPU time is the total time (in seconds) of 5×10^4 runs.

that of the modified τ-leap method. These observations demonstrate that compared to the modified τ-leap method, the modified K-leap method and the hybrid τ/K-leap method can reduce simulation time without sacrificing simulation accuracy and that they also provide more flexibility of trading off simulation accuracy for speed by changing the value of K_c.

Figure 4.2 depicts the histogram distance of $X_4(2)$ versus CPU time for the K-leap and modified K-leap methods. The K-leap method uses (3.13) to calculate K and the partition method in Section IV.B of [58] to avoid negative numbers of molecules. It is seen that the modified K-leap outperforms the K-leap method when $\epsilon = 0.002$ in this particular example. Table 4.1 lists the average number of steps, CPU time and histogram distance error for different leap methods. If we compare the second column of Table 4.1(a), (b) and (c), four leap methods yield almost the

Table 4.1: Average number of steps of one simulation run, CPU time of 5×10^4 runs (in seconds), and histogram distance of $X_4(2)$ for the reacting system given in (4.4)

(a) Hybrid τ/K-leap and modified K-leap method with $\epsilon = 0.01$

	$K_c = 1$			$K_c = 2$			$K_c = 4$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
Hybrid leap	401.05	5796.5	0.029	200.78	3614.7	0.035	100.77	1933.4	0.051
MK-leap	400.85	5630.3	0.026	200.86	3187.4	0.035	100.79	1716.4	0.047

(b) modified τ-leap

	$\epsilon = 0.0005$			$\epsilon = 0.001$			$\epsilon = 0.01$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
$M\tau$-leap	523.93	7344.7	0.024	424.44	6030.5	0.028	401.04	5776.8	0.028

(c) K-leap method

	$\epsilon = 0.002$			$\epsilon = 0.003$			$\epsilon = 0.005$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
K-leap	571.52	6354.1	0.025	249.18	3176.2	0.033	90.145	1447.1	0.048

same histogram distance, but the modified K-leap and the hybrid τ/K-leap method need the smallest number of steps, while the modified τ-leap method needs the largest number of steps. Therefore, the modified K-leap and the hybrid τ/K-leap method consume considerable less CPU time than the modified τ-leap method. If we compare the different columns of Table 4.1(a), we see that for a fixed ϵ, we can increase simulation speed by increasing K_c while slightly sacrificing simulation accuracy.

4.3.2 Expression of LacZ/LacY Genes and Activities of the Proteins

This example was first simulated by Kierzek [37] and 22 reactions are described in detail in 3.5.3 and also in [37, 41, 63, 64]. It characterizes the expression of the LacZ and LacY genes in bacterial organisms and the corresponding protein production and degradation over generation cycle time span.

As in [40], we run simulation using the exact SSA, starting at $t = 0$ with initial condition given by Kierzek [37], and ending at $t = 1000$. We then use the result of exact SSA as the initial condition, and run simulation 10^4 times using the modified τ-leap method of [40] and our modified K-leap method. Each run starts at $t = 1000$ and ends at $t = 1001$. In the modified τ- and K-leap methods, we use $n_c = 20$ and thus $\{R_i\}_{i=1}^{9}$, R_{18} and R_{19} are critical reactions. Because the molecule number of some critical species is either zero or one, we choose $K_c = 1$ to avoid negative number of molecules. In each step, the number of RNAP molecules is generated from a Gaussian random variable with mean 35 and standard deviation 3.5, while the number of Ribosome molecules is generated from a Gaussian random variable with mean 350 and standard deviation 35, as depicted in [41]. We then calculate the histogram distance between the results of a leap method and those of the exact SSA.

Figure 4.3 depicts the histogram distance of *LacZ* at $t = 1001$ versus CPU time for the modified K-leap method, modified τ-leap method and K-leap method. Since $K_c = 1$, the hybrid leap method is the same as the modified τ-leap method in this example. It is seen that the modified K-leap method outperforms the other two leap methods, since the modified K-leap method requires considerably less CPU time than other two methods, while providing the same simulation accuracy. Table 4.2 lists the number of steps, CPU time and the histogram distance for these leap methods. If we compare three leap methods at a given ϵ, we see that the modified K-leap method needs almost the same number of steps as the modified τ-leap method and a slightly larger number of steps than the K-leap method, while it offers the fastest speed. The reason for this observation is as follows. First, we have $K_c = 1$ for the modified K-leap method, and thus the modified K-leap method needs almost the same number of steps

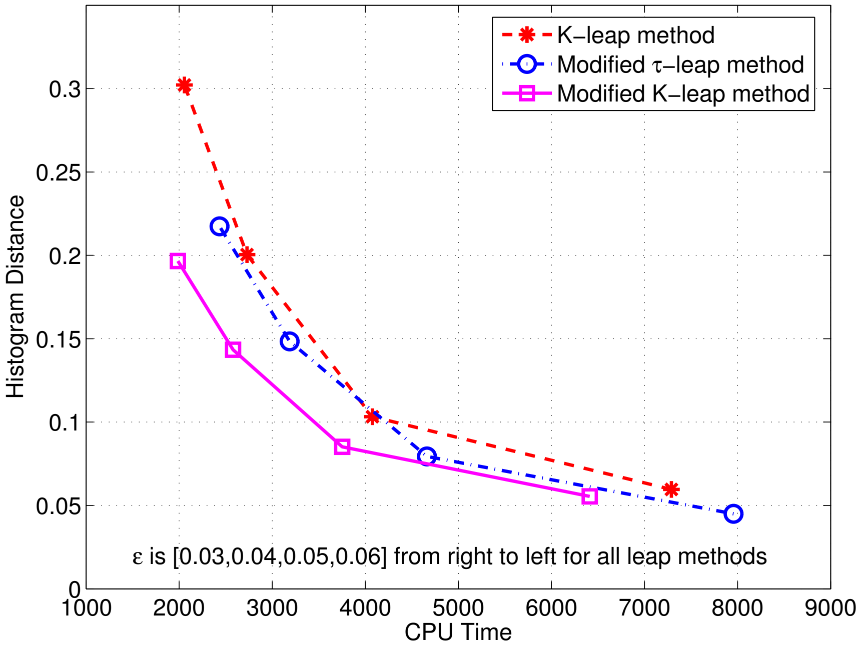


Figure 4.3: Histogram distance of K-leap, modified K-leap and modified τ-leap method of $LacZ(t)$ at $t = 1001$ versus CPU time for the LacZ/LacY expression system. The histogram is obtained after 10^4 simulation runs and the CPU time is the total time (in seconds) of 10^4 runs.

as the modified τ-leap method. However, the K-leap method does not impose such restriction, and thus it requires less number of steps. Let us denote the numbers of non-critical and critical channels as M_{nc} and M_c, respectively. Since some non-critical channels have a very small propensity function, these channels do not fire during a leap with a high probability. Exploiting this property, the modified K-leap method needs to generate $M_k < M_{nc}$ random variables for the number of reactions of M_{nc} noncritical channels with a high probability. On the other hand, the modified τ-leap method always needs to generate M_{nc} Poisson random variables. For this reason, the modified K-leap method is faster than the modified τ-leap method. Recall that when $\tau'' > \tau'$, the modified K-leap method does not need to generate any random variable for critical reactions. Since $a_0^c(\mathbf{x})$ is very small relative to $a_0^{nc}(\mathbf{x})$, the probability for $\tau'' > \tau'$ is high. On the other hand, although the K-leap method employs the

Table 4.2: Average number of steps of one simulation run, CPU time of 10^4 runs (in seconds) and histogram distance of *LacZ* for the LacZ/LacY Model

	$\epsilon = 0.03$			$\epsilon = 0.04$			$\epsilon = 0.06$		
	Steps	Time	HD	Steps	Time	HD	Steps	Time	HD
K-leap	1875.3	7288.9	0.060	1064.1	4077.2	0.103	517.15	2057.1	0.302
MK-leap	1923.8	6410.8	0.055	1112.9	3852.4	0.085	566.30	1986.5	0.197
$M\tau$-leap	1921.7	7957.4	0.048	1112.2	4661.8	0.086	566.32	2433.9	0.217

efficient method of generating multinomial random variable in Section IV.A of [58], it still needs to generate slightly more random variables than the modified K-leap method. Therefore, the modified K-leap method is faster than the K-leap method in this example.

4.4 Conclusion

Compared to Gillespie's exact stochastic simulation algorithm, Gillespie's τ-leap method can significantly increase simulation speed while making small sacrifice in simulation accuracy in many cases. However, in the reacting systems where certain molecular species have a small number of molecules, the τ-leap method can yield negative number of molecules, thereby causing large simulation inaccuracy. The modified τ-leap method attempts to avoid the negative number of molecules by partitioning all the reaction channels into two sets of reactions: critical and noncritical reactions, and then applying the exact SSA to the critical reactions and the τ-leap method to the noncritical reactions. Another leap method named K-leap method can bound the total number of reactions that occur during a leap according to the leap condition, thereby improving simulation accuracy.

In this chapter, we have explored the ideas behind the modified τ-leap method and the K-leap method and developed two new leaps method named the hybrid leap method and the modified K-leap method. In the hybrid leap method, we apply the K-leap method to the critical reactions and the τ-leap method to the noncritical reactions. In the modified K-leap method, we applied the K-leap method to critical and noncritical reaction separately with different total numbers of reactions for each set of reactions. Our numerical results have demonstrated that compared to the modified τ-leap method, our two new leap methods can significantly increase simulation speed while providing almost the same simulation accuracy. Our two new leap methods also offer more flexibility for trading off simulation accuracy for speed. We also demonstrated that the modified K-leap provides slightly better performance than the hybrid leap method in terms of simulation speed and accuracy.

CHAPTER 5

Unbiased Tau-leap Method for Stochastic Simulation of Chemically Reacting Systems

As we discussed earlier, approximate SSAs allow a number of reactions to occur in each simulation step. Therefore, there are always certain changes in propensity functions in all existing leap methods. To reduce the simulation errors caused by such changes in propensity functions, Gillespie proposed a midpoint τ-leap method, where the mean of the Poisson random variables is calculated from the propensity functions at an estimated midpoint during a leap [38]. The midpoint τ-leap approach was also applied to the binomial τ-leap method [41]. However, the midpoint Poisson τ-leap and binomial τ-leap are still not accurate since these values are estimated.

Due to the changes in propensity functions during a leap, the mean of K_m used in all the τ-leap methods mentioned earlier is not equal to the true mean, as we will demonstrate in section 5.1 and 5.2. Hence, all existing τ-leap methods produce bias in simulation results. Although such bias can be small when the leap step size is small, it limits simulation speed and accuracy, because the step size can be otherwise increased if such bias is absent. This motivates us to develop several unbiased τ-leap methods that will be presented in this chapter. Specifically, we first analyze the true mean of K_m based on the chemical master equation (CME). After getting the true mean, we modify the Poisson and binomial τ-leap methods so that the simulation results are unbiased. Moreover, we also analyze the variance of K_m and develop an unbiased Poisson/Gaussian/Binomial τ-leap method to remove bias and correct errors in the variance of K_m.

5.1 Motivating Examples

In the Poisson, binomial, and multinomial τ-leap methods discussed earlier, the mean of K_m generated during a leap from t to $t+\tau$ is $a_m(\mathbf{x})\tau$, where $a_m(\mathbf{x})$ is calculated from the system state at time t: $\mathbf{X}(t) = \mathbf{x}$. Clearly, $a_m(\mathbf{x})\tau$ is not equal to the true mean of K_m, because there always are certain changes in propensity functions in the time interval $[t, t+\tau]$. Therefore, the numbers of reactions generated during each leap by these τ-leap methods are biased. The midpoint τ-leap method uses the state at an estimated midpoint to calculate the propensity functions, which are then used to determine the mean of K_m, $m = 1, \cdots, M$. This can reduce but not completely eliminate the bias. To see the bias caused by the (midpoint) τ-leap methods, let us consider the following three simple reacting systems.

We first consider a reacting system that consists of a single monomolecular reaction channel:

$$S_1 \xrightarrow{c_1} \emptyset. \tag{5.1}$$

Let us consider the time interval $[0, \tau]$, denote $X_1(0) = x_1$ and define $\mu_1(\tau) = E[K_1]$, where $E[\cdot]$ denotes the expected value of the random variable in the brackets. It is easy to show that $\mu_1(\tau)$ obeys the following ordinary differential equation (ODE):

$$\frac{d\mu_1(\tau)}{d\tau} = -c_1\mu_1(\tau) + c_1x_1, \tag{5.2}$$

Table 5.1: Expected number of reactions occurring during one leap for three elementary reactions

		$S_1 \rightarrow \emptyset$	$S_1 + S_1 \rightarrow S_2$	$S_1 + S_2 \rightarrow S_3$
$\epsilon = 0.01$	Poisson-τ	615.00	153.75	287.53
	Poisson mid-τ	611.93	152.98	286.09
	True Mean	611.94	153.12	286.31
$\epsilon = 0.05$	Poisson-τ	3075.00	768.74	1437.66
	Poisson mid-τ	2998.15	749.64	1401.99
	True Mean	2999.39	750.15	1402.70
$\epsilon = 0.10$	Poisson-τ	6150.00	1537.49	2875.32
	Poisson mid-τ	5842.50	1461.60	2733.41
	True Mean	5852.50	1464.50	2738.31

*The leap step size τ is calculated from (2.11) for a given ϵ.
*The true mean is calculated from (5.3) for $S_1 \rightarrow \emptyset$ and obtained from 1×10^5 simulation runs using Gillespie's exact SSA for $S_1 + S_1 \rightarrow S_2$ and $S_1 + S_2 \rightarrow S_3$.

whose solution is given by

$$\mu_1(\tau) = x_1(1 - e^{-c_1\tau}). \tag{5.3}$$

The τ-leap method and the midpoint τ-leap method generate K_1 using a mean of $c_1 x_1 \tau$ and $c_1\lceil x_1 - c_1 x_1 \tau/2 \rceil \tau$, respectively, which is clearly different from the true mean of K_1 given in (5.3). In Table 5.1, we compare the mean of K_1 used in the (midpoint) τ-leap method with the true mean, when $x_1 = 61500$ and $c_1 = 0.5$. The leap step size τ is calculated from (2.11) for a given ϵ. It is seen that the bias can be considerably large, particularly when ϵ, or equivalently τ, is relatively large.

We next consider the reacting system consisting of a single bimolecular reaction with one reactant species:

$$S_1 + S_1 \xrightarrow{c_1} S_2. \tag{5.4}$$

The ODE of $\mu_1(\tau)$ in this example involves the second moment of K_1, and thus, we cannot find $\mu_1(\tau)$ in closed-form. However, we can run a large number of simulations using Gillespie's exact SSA and then obtain a very accurate estimate of $\mu_1(\tau)$. The τ-

leap method and the midpoint τ-leap method generate K_1 using a mean of $c_1x_1(x_1 - 1)\tau/2$ and $c_1\tilde{x}_1(\tilde{x}_1 - 1)\tau$, where $\tilde{x}_1 = \lceil x_1 - c_1x_1(x_1 - 1)\tau/2\rceil$, respectively. In Table 5.1, we compare the mean of K_1 used in the (midpoint) τ-leap method with the mean estimated from exact stochastic simulations, when $x_1 = 61500$ and $c_1 = 0.0001$. Again, it is seen that the bias can be considerably large.

The third example is the bimolecular reaction with two different reactant species:

$$S_1 + S_2 \xrightarrow{c_1} S_3. \tag{5.5}$$

Again, $\mu_1(\tau)$ cannot be obtained in closed-form but, can be accurately estimated from simulations using Gillespie's exact SSA. The τ-leap method and the midpoint τ-leap method generate K_1 using a mean of $c_1x_1x_2\tau$ and $c_1\tilde{x}_1\tilde{x}_2\tau$, where $\tilde{x}_1 = \lceil x_1 - c_1x_1x_2\tau/2\rceil$ and $\tilde{x}_2 = \lceil x_2 - c_1x_1x_2\tau/2\rceil$, respectively. In Table 5.1, we compare the mean of K_1 used in the (midpoint) τ-leap method with the mean estimated from exact stochastic simulations, when $x_1 = 61500$, $x_2 = 54000$ and $c_1 = 0.0001$. Again, it is seen that the bias can be considerably large, when ϵ is relatively large.

Note that Table 5.1 gives the bias of the (midpoint) τ-leap method in one leap. To simulate the evolution of a reacting system during a given period of time, we typically need many leaps. The total bias can be larger than that in a single leap. If such bias can be removed, one will expect that simulation accuracy can be improved for a given ϵ, or, simulation speed can be increased by increasing ϵ without sacrificing simulation accuracy. This motivates us to develop unbiased τ-leap methods that we will present in the following.

5.2 The Unbiased τ-leap Methods

As the existing τ-leap methods has severe biased during the simulation, we proposed our new unbiased τ-leap methods [66], including unbiased Poisson τ-leap, unbiased binomial τ-leap and unbiased Poisson/Gaussian/Binomial τ-leap methods.

5.2.1 The Unbiased Poisson τ-leap Method

To develop an unbiased τ-leap method, we need to find the mean of $K_1, \cdots, K_M$ in each leap, and then generate $K_1, \cdots, K_M$ from their probability distributions with the true mean. The mean of $K_1, \cdots, K_M$ can be derived from the CME of the probability mass function (PMF) of $\mathbf{K}$, $P(\mathbf{K};\tau)$, which is given by [67, 68]

$$\begin{aligned}\frac{\partial P(\mathbf{K};\tau)}{\partial \tau} =& \sum_{m=1}^{M}\Big\{a_m(\mathbf{K}-\mathbf{e}_m)P(\mathbf{K}-\mathbf{e}_m;\tau) \\ & -a_m(\mathbf{K})P(\mathbf{K};\tau)\Big\};\end{aligned} \tag{5.6}$$

with initial condition P($\mathbf{0}$;0)=1, where $\mathbf{e}_m$ is the mth column of the $M \times M$ identity matrix, and

$$a_m(\mathbf{K}) \stackrel{\triangle}{=} c_m h_m(\mathbf{X}(t) + \boldsymbol{\nu}\mathbf{K}). \tag{5.7}$$

Let us define $\boldsymbol{\mu}(\tau) \stackrel{\triangle}{=} E[\mathbf{K}]$. Multiplying both sides of (5.6) by $\mathbf{K}$ and then summing them over all possible values of $\mathbf{K}$, we get

$$\begin{aligned}\frac{d\boldsymbol{\mu}(\tau)}{d\tau} =& \sum_{m=1}^{M}\sum_{\mathbf{K}} \mathbf{e}_m a_m(\mathbf{K})P(\mathbf{K};\tau) \\ =& E[\mathbf{a}(\mathbf{K})],\end{aligned} \tag{5.8}$$

where $\mathbf{a}(\mathbf{K}) \stackrel{\triangle}{=} [a_1(\mathbf{K}), \cdots, a_M(\mathbf{K})]^T$. If $a_m(\mathbf{K})$, $m = 1, \cdots, M$ are linear functions of $\mathbf{K}$, which is true if all reactions are of the zeroth and/or first order, then $E[\mathbf{a}(\mathbf{K})]$ can be written as a linear function of $\boldsymbol{\mu}(\tau)$. In this case, we obtain a first order linear

ODE for $\boldsymbol{\mu}(\tau)$, which is ready to be solved analytically or using an efficient numerical method. However, it is often that $a_m(\mathbf{K})$, $m = 1, \cdots, M$ are nonlinear functions of $\mathbf{K}$ due to higher order reactions involved. In this case, $E[\mathbf{a}(\mathbf{K})]$ involves not only $\boldsymbol{\mu}(\tau)$ but also the second and possibly higher order moments of $\mathbf{K}$, and thus, it is difficult to obtain $\boldsymbol{\mu}(\tau)$ by solving (5.8). To overcome this problem, we approximate $\mathbf{a}(\mathbf{K})$ by its first order Taylor expansion, which can be found from (5.7) as follows:

$$\mathbf{a}(\mathbf{K}) \approx \mathbf{a}(0) + \mathbf{F}\mathbf{K}, \tag{5.9}$$

where $\mathbf{F}$ is an $M \times M$ matrix whose entry on the mth row and m'th column is $[\mathbf{F}]_{m,m'} = f_{mm'}$ given in (3.4) and $\mathbf{a}(0) = [a_1(0), \cdots, a_M(0)]^T$ containing the propensity functions at time t. Substituting (5.9) into (5.8), we can approximate (5.8) by the following first order linear ODE:

$$\frac{d\boldsymbol{\mu}(\tau)}{d\tau} = \mathbf{F}\boldsymbol{\mu}(\tau) + \mathbf{a}(0). \tag{5.10}$$

The initial condition of the ODE is $\boldsymbol{\mu}(0) = \mathbf{0}$. It is easy to solve the ODE (5.10) analytically or using an efficient numerical method to get $\boldsymbol{\mu}(\tau)$.

After we obtain $\boldsymbol{\mu}(\tau)$, we can generate $K_1, \cdots, K_M$ using their distributions with a mean equal to $\boldsymbol{\mu}(\tau)$. Applying this idea to the Poisson τ-leap method, we keep steps 1, 2, 3, 5 and 6 in Algorithm 2 unchanged, but modify step 4 as follows: find $\boldsymbol{\mu}(\tau)$ by solving ODE (5.10) and then generate K_m, $1 \cdots, K_M$, according to the Poisson distribution with mean $\boldsymbol{\mu}(\tau)$. We refer to our new τ-leap method as unbiased Poisson τ-leap method. Strictly speaking, our unbiased Poisson τ-leap method does not completely eliminate the bias, because the ODE (5.10), that is used to obtain $\boldsymbol{\mu}(\tau)$, is an approximation of (5.8), and if a numerical method is employed to solve (5.10), it can also produce small errors. However, it is expected that for a τ determined by the leap condition, the approximation error introduced by (5.10) is small, although a further investigation on the approximation error may be needed. Moreover, very accurate and efficient numerical methods for solving (5.10) are available, as we will discuss in section 5.3. Our numerical examples in section 5.4 will demonstrate that our unbiased τ-leap method can make the bias negligible, while the (midpoint) Poisson τ-leap method causes significantly large bias.

5.2.2 The Unbiased Binomial τ-leap Method

As we discussed in section 2.4.3, the mean of K_m, $m = 1, \cdots, M$, used in the binomial τ-leap method is the same as that used in the Poisson τ-leap method. Therefore, the binomial τ-leap method also is biased. As we pointed out in section 2.4.4, the midpoint binomial τ-leap method generates K_m, $m = 1, \cdots, M$, using a mean identical to that used in the midpoint Poisson τ-leap method, which implies that the midpoint binomial τ-leap method is also biased.

Applying the same idea of the unbiased Poisson τ-leap method, we can also remove the bias in the binomial τ-leap method. Specifically, we can get the mean of K_m, $m = 1, \cdots, M$, $\mu_m(\tau)$, from (5.10), and then generate K_m, $m = 1, \cdots, M$, from a binomial random variable $\mathcal{B}(k_{m,\max}, p_m)$, where $k_{m,\max}$ is obtained in the same way as in [41] and [42] and $p_m = \mu_m(\tau)/k_{m,\max}$. Since the binomial τ-leap method of Tian and Burrage [41] cannot handle the cases where more than two reaction channels share certain reactants, we now modify the binomial τ-leap method of Chatterjee *et al.* [42] to obtain the unbiased binomial τ-leap method. More specifically, the unbiased binomial τ-leap algorithm keeps the steps 1, 2, 3, 5 and 6 in Algorithm 2, but changes step 4 as follows: find the mean of K_m, $m = 1, \cdots, M$, $\mu_m(\tau)$, from (5.10), set $\tilde{x}_n = X_n(t)$, and for $m = 1$ to M reaction channels, do the following:

(a) Find $k_{m,\max} = \min_{\nu_{im}<0,\ i\in[1,N]} \lfloor \widetilde{x}_i/|\nu_{im}| \rfloor$, where $\lfloor x \rfloor$ denotes the largest integer less than x.

(b) calculate $p = \mu_m(\tau)/k_{m,\max}$ and generate K_m from the binomial distribution with parameter $k_{m,\max}$ and p.

(c) Set $\tilde{x}_n = \tilde{x}_n + \nu_{nm} K_m$ for $n = 1, \cdots, N$, if $\nu_{nm} < 0$.

5.2.3 The Unbiased Poisson/Gaussian/Binomial τ-leap Method

The variance of a Poisson random variable is equal to the mean. Although the unbiased Poisson τ-leap method can remove the bias, it may not be able to remove the errors in variance, if the variance of K_m is significantly different from the mean, which is possibly the case when changes in propensity functions are relatively large. In the binomial τ-leap method, once the mean of K_m and $k_{m,\max}$ are given, the variance of K_m is determined. Therefore, the variance of K_m in the binomial τ-leap method may also be significantly different from the true variance. If we can remove the errors in both the mean and variance of K_m, $m = 1, \cdots, M$, we can further improve simulation accuracy. Towards this end, we need to find the variance of K_m, $m = 1, \cdots, M$.

Let us define the covariance matrix of $\mathbf{K}$ as $\mathbf{C}(\tau) \overset{\triangle}{=} E[(\mathbf{K} - \boldsymbol{\mu}(\tau))(\mathbf{K} - \boldsymbol{\mu}(\tau))^T]$. Multiplying both sides of (5.6) by $(\mathbf{K} - \boldsymbol{\mu}(\tau))(\mathbf{K} - \boldsymbol{\mu}(\tau))^T$ and then summing them over all possible values of $\mathbf{K}$, we obtain

$$\begin{aligned}
\frac{d\mathbf{C}(\tau)}{d\tau} &= \sum_{\mathbf{k}} \sum_{m=1}^{M} \{(\mathbf{e}_m(\mathbf{K} - \boldsymbol{\mu}(\tau))^T + (\mathbf{K} - \boldsymbol{\mu}(\tau))\mathbf{e}_m^T \\
&\quad + \mathbf{e}_m \mathbf{e}_m^T) a_m(\mathbf{K}) P(\mathbf{K}\ t)\} \\
&= E[\mathbf{a}(\mathbf{K})(\mathbf{K} - \boldsymbol{\mu}(\tau))^T] + E[(\mathbf{K} - \boldsymbol{\mu}(\tau))\mathbf{a}(\mathbf{K})^T] \\
&\quad + E[\mathrm{diag}(\mathbf{a}(\mathbf{K}))],
\end{aligned} \tag{5.11}$$

where diag($\mathbf{x}$) represents a diagonal matrix whose ith diagonal entry is the ith entry of vector $\mathbf{x}$. If $a_m(\mathbf{K})$, $m = 1, \cdots, M$ are linear functions of $\mathbf{K}$, then we can simplify (5.11) to a first order linear ODE which is ready to be solved. However, as we mentioned earlier, it is often that $a_m(\mathbf{K})$, $m = 1, \cdots, M$ are nonlinear functions of $\mathbf{K}$ due to higher order reactions involved. In this case, the right hand side of (5.11) involves not only $\mathbf{C}(\tau)$ but also the third and possibly higher order moments of $\mathbf{K}$, and thus, it is difficult to obtain $\mathbf{C}(\tau)$ by solving (5.11). To overcome this problem, we again approximate $\mathbf{a}(\mathbf{K})$ by its first order Taylor expansion. Using this approximation, we can approximate (5.11) by

$$\frac{d\mathbf{C}(\tau)}{d\tau} = \mathbf{F}\mathbf{C}(\tau) + \mathbf{C}(\tau)\mathbf{F}^T + \text{diag}\big(\mathbf{F}\boldsymbol{\mu}(\tau) + \mathbf{a}(0)\big). \tag{5.12}$$

Using (5.10), we can further simplify (5.12) to

$$\frac{d\mathbf{C}(\tau)}{d\tau} = \mathbf{F}\mathbf{C}(\tau) + \mathbf{C}(\tau)\mathbf{F}^T + \text{diag}\Big(\frac{d\boldsymbol{\mu}(\tau)}{d\tau}\Big), \tag{5.13}$$

where the initial conditions are $\mathbf{C}(0) = \mathbf{0}$ and $\boldsymbol{\mu}(0) = \mathbf{0}$.

The matrix ODE (5.13) consists of M^2 scalar ODEs. It may require relatively large computation to solve (5.13). To reduce computation burden, we assume that K_m, $m = 1, \cdots, M$, are independent, which implies that $[\mathbf{C}(\tau)]_{mm'} = 0$ if $m \neq m'$. Note that in the Poisson τ-leap method, [38, 39] it was assumed that K_m, $m = 1, \cdots, M$, are independent. Under the independent assumption, we can reduce (5.13) to the following M ODEs:

$$\frac{d[\mathbf{C}(\tau)]_{mm}}{d\tau} = 2f_{mm}[\mathbf{C}(\tau)]_{mm} + \frac{d\mu_m(t)}{d\tau}, \ m = 1, \cdots, M. \tag{5.14}$$

We can solve (5.14) and (5.10) jointly to get $\mu_m(\tau)$ and $[\mathbf{C}(\tau)]_{mm}$, $m = 1, \cdots, M$.

Depending on the values of $\mu_m(\tau)$ and $[\mathbf{C}(\tau)]_{mm}$, $m = 1, \cdots, M$, we can generate K_m from a truncated Gaussian distribution with mean $\mu_m(\tau)$ and variance

$[\mathbf{C}(\tau)]_{mm}$, a Poisson distribution or a binomial distribution with properly chosen parameters. In fact, if $\mu_m(\tau) \geq 10$, a Poisson distribution can be well approximated by a Gaussian distribution [69]. And, it is more efficient to generate a Gaussian random variable than a Poisson random variable. When $\mu_m(\tau) < 10$, we can generate K_m from a binomial distribution, if $[\mathbf{C}(\tau)]_{mm}$ is significantly different from $\mu_m(\tau)$, say $|[\mathbf{C}(\tau)]_{mm} - \mu_m(\tau)|/\mu_m(\tau) > 0.1$. This leads to our unbiased Poisson/Gaussian/Binomial τ-leap method, which is described as follows: we keep steps 1, 2, 3, 5 and 6 in Algorithm 2 unchanged, but modify step 4 as follows: Find $\mu_m(\tau)$ and $[\mathbf{C}(\tau)]_{mm}$, $m = 1, \cdots, M$ by solving ODEs in (5.14) and (5.10). When $\mu_m(\tau) \geq 10$, we first generate a Gaussian random variable, G, with mean $\mu_m(\tau)$ and variance $[\mathbf{C}(\tau)]_{mm}$. If $G < 0$, we regenerate G until it is nonnegative; we then round G to the nearest integer, G_I, and let $K_m = G_I$. When $\mu_m(\tau) < 10$, if $|[\mathbf{C}(\tau)]_{mm} - \mu_m(\tau)|/\mu_m(\tau) < 0.1$, we generate K_m from a Poisson distribution; otherwise, we first solve equations $k_m p_m = \mu_m(\tau)$ and $k_m p_m(1 - p_m) = [\mathbf{C}(\tau)]_{mm}$ to find k_m and p_m. And then, after rounding k_m to the nearest integer, if $k_m \leq k_{m,\max}$, we generate K_m from a binomial random variable $\mathcal{B}(k_m, p_m)$; otherwise, we generate K_m from a binomial random variable $\mathcal{B}(k_{m,\max}, \mu_m(\tau)/k_{m,\max})$.

5.3 Implementation Issues for Solving ODEs

To implement the unbiased τ-leap methods, we need to solve the ODE (5.10) and (5.14). Since these ODEs are first order linear ODEs, their solutions can be found in closed form. Note that both $\mathbf{F}$ and $\mathbf{a}(0)$ in (5.10) and (5.14) are dependent on the initial state $\mathbf{X}(t)$. For relatively small reacting system, we can express the solution of these ODEs as a function of $\mathbf{X}(t)$ and then calculate $\boldsymbol{\mu}(\tau)$ and $[\mathbf{C}]_{mm}$,

$m = 1, \cdots, M$ in each leap using the specific value of $\mathbf{X}(t)$. Since we do not need to directly solve (5.10) and (5.14) in each leap, the computation required to obtain $\boldsymbol{\mu}(\tau)$ and $[\mathbf{C}]_{mm}$, $m = 1, \cdots, M$, is very small in this case. However, for relatively large systems, we may not be able to write the solution of (5.10) and (5.14) as a function of $\mathbf{X}(t)$. In this case, we need to directly solve (5.10) and (5.14) in each leap using a specific initial condition. Since we need to do eigen-decomposition on $\mathbf{F}$ to obtain the analytical solution to (5.10) and (5.14), relatively large computation is needed to obtain $\boldsymbol{\mu}(\tau)$ and $[\mathbf{C}]_{mm}$, $m = 1, \cdots, M$, analytically. However, we can employ an efficient numerical method, such as Runge-Kutta method and Bulirsch-Stoer Method [62], to solve (5.10) and (5.14). Since the leap step size τ is typically not large, such numerical methods can obtain an accurate solution with a small amount of computation.

5.4 Numerical Examples

In order to demonstrate the accuracy and efficiency of our unbiased τ-leap methods, we simulated several chemical reaction systems. To assess the accuracies of different simulation methods, we first obtain the histograms of the populations of each molecular species at the end of simulation from a series of repeated exact SSA runs. We then simulate the same chemical reaction system over the same time interval by the same number of runs, using different leap methods. We then plot the estimated PDF of exact SSA and leap methods so that we can see the performance of each method. As in section 2.5, we also employ the histogram distances between the results of the exact SSA and those of a leap method to measure the simulation accuracy. In all leap methods, we use equation (2.11) and a specific ϵ to calculate τ.

An important issue for the efficiency of stochastic simulations is the generation of random numbers. As we mentioned earlier, we used the Poisson random number generator in [62] in our simulations. In generating a binomial random variable $\mathcal{B}(n, p)$, we employed the BTPE algorithm [61] for $np > 10$, and the BG algorithm [60] for $np \leq 10$. We used the fourth-order Runge-Kutta method [62] to solve the ODE (5.10) and (5.14). All simulations are run in Matlab on a PC with a 3.20 GHz CPU and 2G-byte memory running Windows XP.

5.4.1 Three Elementary Reactions

In section 5.1, we used three elementary reactions to illustrate the bias of existing τ-leap methods. We now present the simulation results for these three reactions using the same parameters and initial conditions described in section 5.1. We ran simulation 2×10^4 times and each time starts at $t = 0$ and end at $t = 2$.

Table 5.2: Mean of $X_1(2)$ in three elementary reactions

		$S_1 \rightarrow \emptyset$	$S_1 + S_1$ $\rightarrow \emptyset$	$S_1 + S_2$ $\rightarrow \emptyset$
Exact SSA		22624	4622.4	10196
$\epsilon = 0.01$	Poisson-τ	22512	4602.5	10180
	Poisson mid-τ	22627	4624.4	10197
	Unbiased Poisson- τ	22624	4624.0	10196
$\epsilon = 0.03$	Poisson-τ	22283	4560.2	10146
	Poisson mid-τ	22629	4624.8	10197
	Unbiased Poisson-τ	22624	4624.0	10196
$\epsilon = 0.10$	Poisson-τ	21444	4495.8	10031
	Poisson mid-τ	22665	4632.6	10201
	Unbiased Poisson-τ	22624	4624.5	10196

Figures 5.1, 5.2 and 5.3 depict the estimated PDF of $X_1(2)$ for three reactions, respectively. It is seen that the Poisson τ-leap method yields significant bias. The estimated PDF from the midpoint Poisson τ-leap method matches that from the exact SSA when $\epsilon = 0.03$, but exhibits considerable bias when $\epsilon = 0.1$. In contrast,

the estimated PDF from our unbiased Poisson τ-leap method matches that from the exact SSA for both $\epsilon = 0.03$ and $\epsilon = 0.1$. The same observation can be seen from Table 5.2, where the mean of $X_1(2)$ calculated from the simulation results of different simulation methods is listed.

5.4.2 Decaying-Dimerizing Reactions

This example was originally used by Gillespie [38, 39] to test the Poisson τ-leap method. As we have used this example to test the K-leap method, it includes three molecular species, with four reactions (3.18), rate constant (3.19) and initial conditions (3.20). We run simulation 5×10^4 times, and each time starts at $t = 0$ and ends at $t = 10$.

Figure 5.4 and 5.5 depict the estimated PDF of $X_1(10)$ and $X_2(10)$ from the simulation results of the exact SSA, Poisson, midpoint Poisson and unbiased Poisson/Gaussian/Binomial τ-leap methods. We used $\epsilon = 0.03$ to determine the leap step size in all three τ-leap methods. For $X_1(10)$, all three τ-leap methods do not exhibit clear bias, but our unbiased Poisson/Gaussian/Binomial τ-leap method offers a variance closest to the true variance. For $X_2(10)$, the Poisson τ-leap method suffers significant bias; the midpoint τ-leap method yields very small but noticeable bias, whereas our unbiased Poisson/Gaussian/Binomial τ-leap method does not produce noticeable bias.

Figure 5.6 and 5.7 depict the histogram distances of the Poisson, midpoint Poisson, unbiased Poisson and unbiased Poisson/Gaussian/Binomial τ-leap methods. It is seen that our unbiased Poisson/Gaussian/Binomial τ-leap method has the best performance, since it offers the smallest histogram distance for a given simulation time, or equivalently, it needs the shortest simulation time for a given histogram distance. It is interesting to note that the histogram distances of the midpoint Poisson and unbiased Poisson τ-leap methods for $X_1(10)$ are worse than that of the Poisson τ-leap method. This is due to the fact that all three τ-leap methods do not produce noticeable bias, but the variance of the Poisson τ-leap method is better than the midpoint Poisson and unbiased Poisson τ-leap methods, as partly shown in Figure 5.4. Since our unbiased Poisson/Gaussian/Binomial τ-leap method can reduce errors in both mean and variance, it is not surprising to see that it offers the best performance.

5.4.3 Epidermal Growth Factor Receptor Signaling Pathway

Each cell in a multicellular organism has been programmed during development to respond to a specific set of extracellular signals. Such extracellular signals are transduced into the cell through cell signaling pathways. Signalling pathways through the receptor tyrosine kinase (RTK) family of receptors regulates a wide range of biological phenomena, including cell proliferation and differentiation, and the epidermal growth factor receptor (EGFR) is an important member of the RTK family. A number of computational models have been employed to investigate the dynamical behavior of the EGFR pathway.

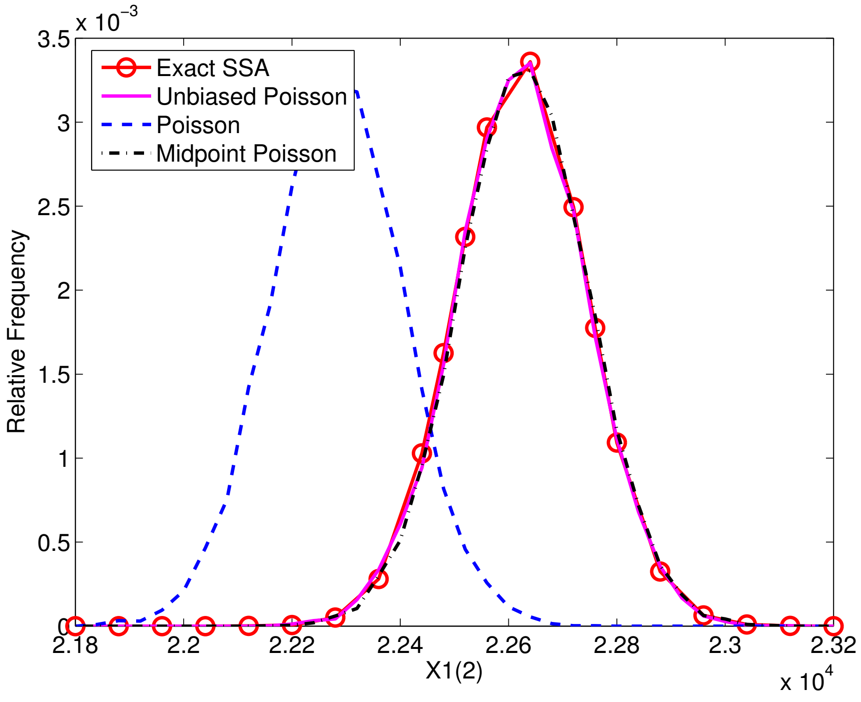
Here we simulate the EGFR signaling pathway using a computational model described in [43], [70] and [71]. This model consists of 23 molecular species and 47 reaction channels, which are listed in Table I of [43]. In our simulations, we used all rate constants and the initial condition listed in the table I of [43], except that the initial concentration of the epidermal growth factor (EGF) was chosen to be 1 nM. From the initial concentration of EGF, the initial population of EGF can be found as 1.152×10^6. The initial populations of other species are the same as those in [43]. We ran simulations 10^4 times, and each time starts at $t = 0$ and ends at $t = 8$ using exact SSA, the binomial τ-leap method of Chatterjee *et al.* [42], the midpoint binomial τ-leap method and our unbiased binomial τ-leap method.

Table 5.3 lists the mean of the number of molecules for several species at $t = 8$. It is seen that our unbiased binomial τ-leap method produces almost the same mean as the exact SSA, while the binomial τ-leap method and the midpoint binomial τ-leap method produce considerable bias. For other species that are not listed in Table 5.3, all three leap methods yield almost the same mean as the exact SSA. Figure 5.8 depicts the PDF of the number of Grb molecules at $t = 8$ estimated from the results of 10^4 simulation runs. It is observed that the PDF obtained from our unbiased binomial τ-leap method matches that obtained from the exact SSA, while the PDFs obtained from the (midpoint) binomial τ-leap method exhibits bias. Figure 5.9 depicts the histogram distance of *Grb* versus CPU time. It is seen that our unbiased binomial τ-leap method yields much smaller histogram distance than the (midpoint) binomial τ-leap method, while requiring almost the same CPU time for a given ϵ. For other species, our unbiased binomial τ-leap method offers smaller or almost the same histogram distance as the (midpoint) binomial τ-leap method (these results are not shown here).

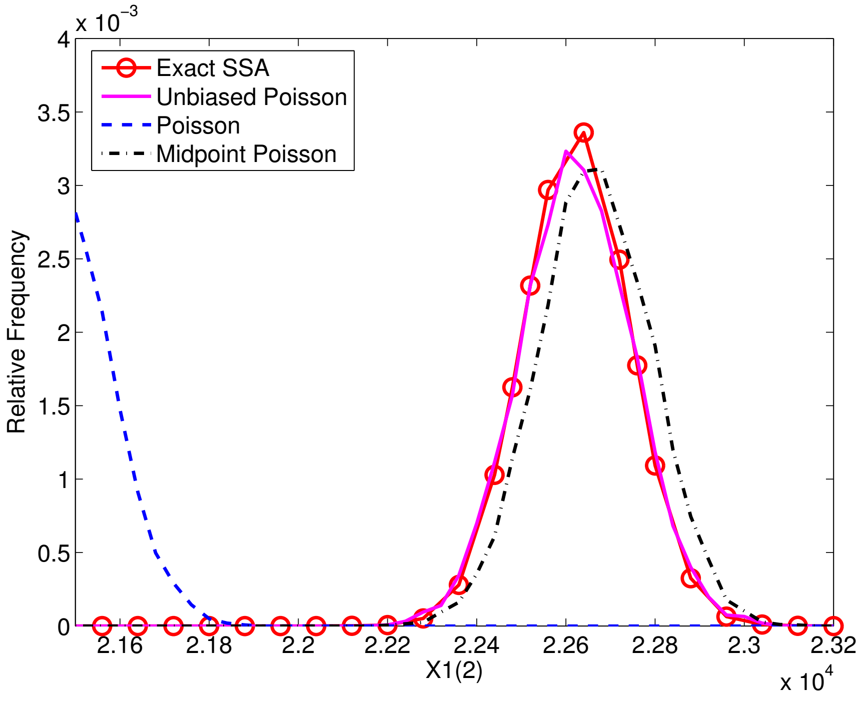
5.5 Concluding Remarks

The τ-leap method speeds up simulation by allowing a number of reactions to occur during a time interval. This inevitably causes the propensity functions to change during a leap. Without knowing the exact mean of the number of times that each channel fires during a leap, the τ-leap method uses the propensity functions at the beginning of the leap to estimate the mean, while the midpoint τ-leap method uses the propensity functions at an estimated midpoint to calculate the mean, and then uses this estimated mean in generating the number of reactions occurring during a leap for each reaction channel. In this chapter, we have demonstrated that the mean used in the (midpoint) τ-leap method is not equal to the true mean. Therefore, the (midpoint) τ-leap method produces biased simulation results, which can cause large simulation errors and limit simulation speed.

To remove the bias in simulation results, we have analyzed the mean of the number of times that each channel fires during a leap based on the chemical master equation. Using the mean obtained from the analysis, we developed unbiased τ-leap methods that can almost completely remove the bias in simulation results. Moreover, we have also derived the variance of the number of times that each channel fires during a leap again based the chemical master equation, which was further exploited to devise an unbiased Poisson/Gaussian/Binomial τ-leap method. Since the unbiased Poisson/Gaussian/Binomial τ-leap method eliminates almost all the bias and reduces the errors in the variance, it can significantly improve simulation accuracy.

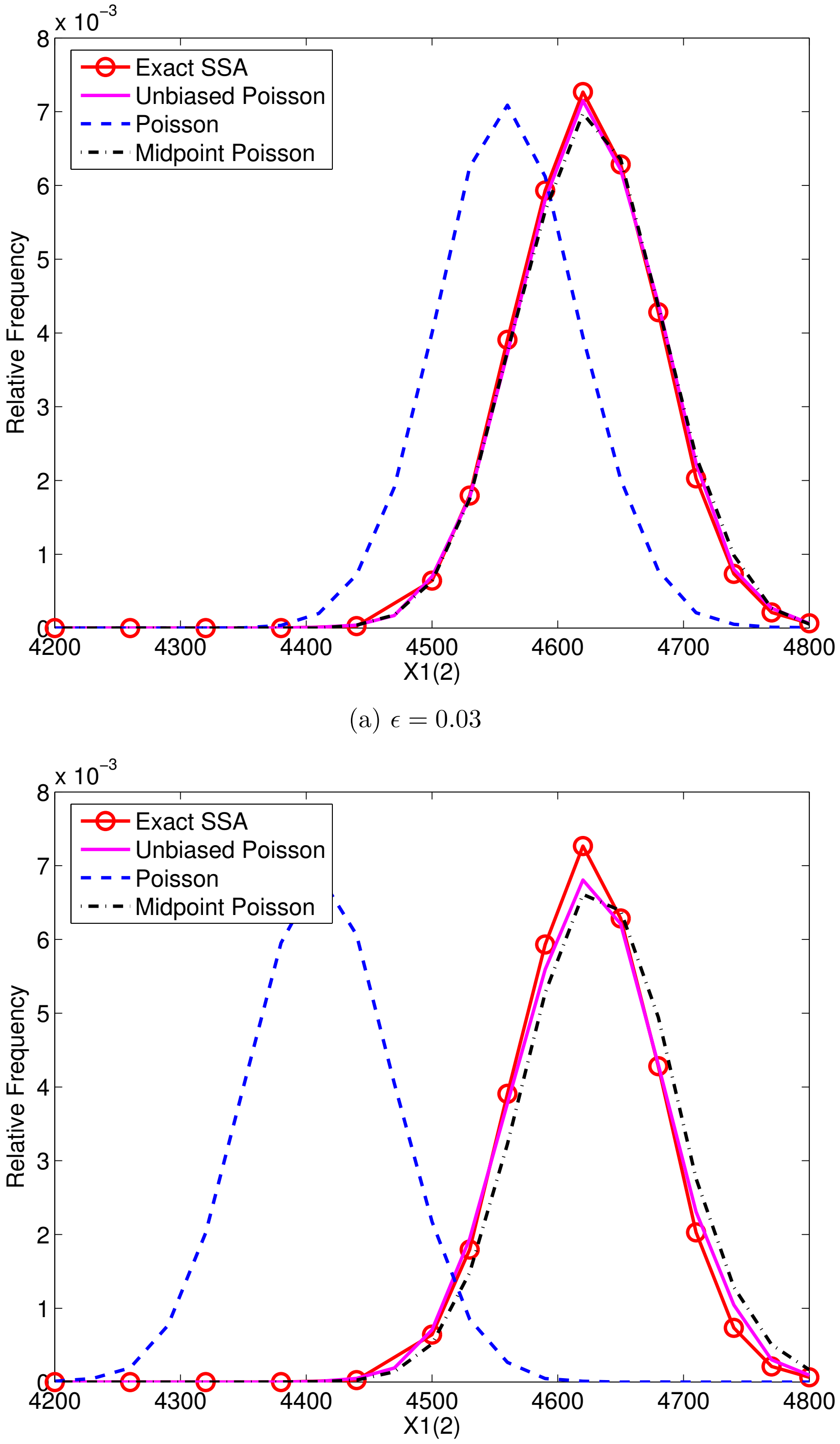


(a) $\epsilon = 0.03$



(b) $\epsilon = 0.1$

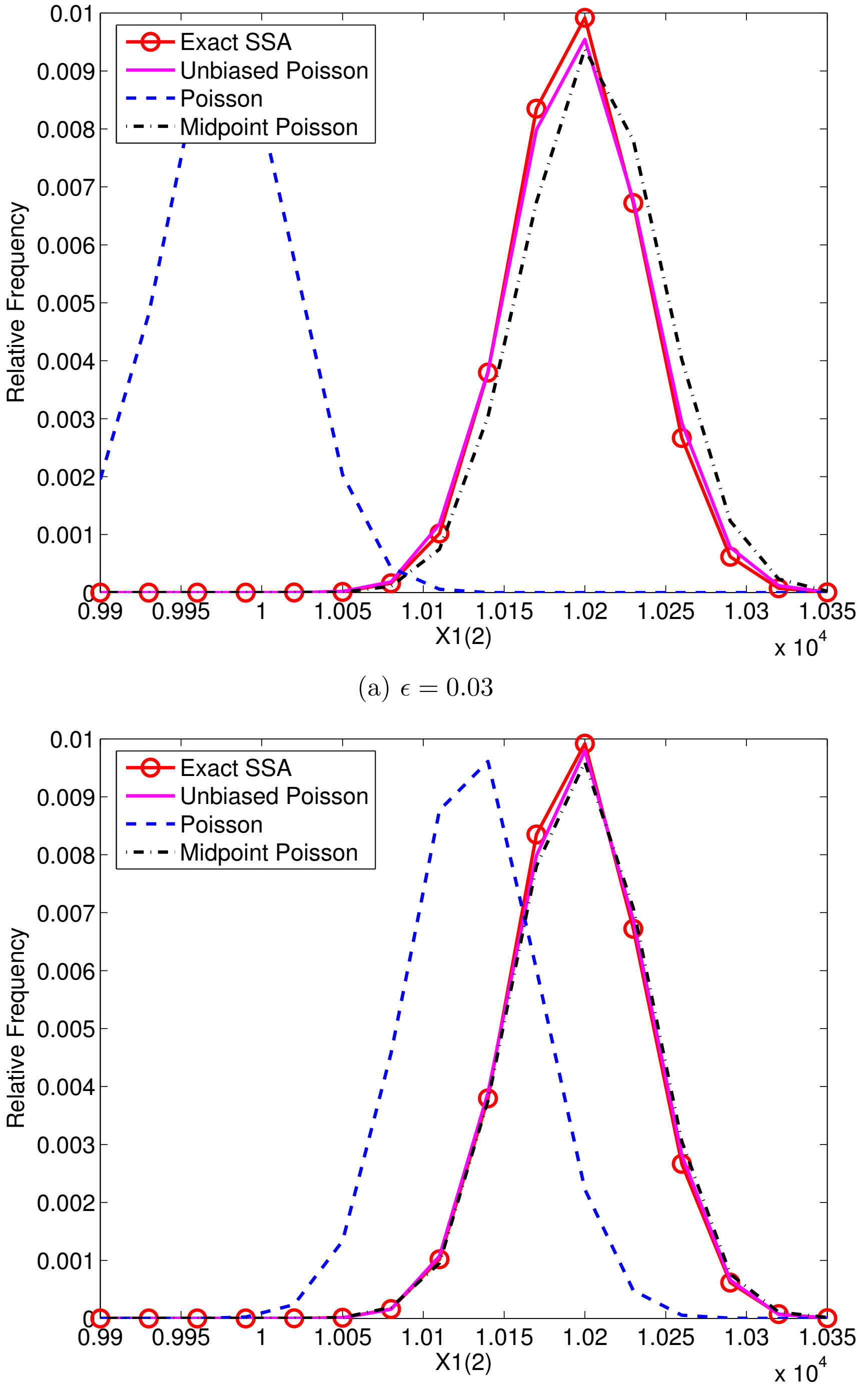
Figure 5.1: The estimated PDF of $X_1(2)$ from 2×10^4 simulation runs for reaction (5.1) with $c_1 = 0.5$ and $X_1(0) = 61500$.



(a) $\epsilon = 0.03$

(b) $\epsilon = 0.1$

Figure 5.2: The estimated PDF of $X_1(2)$ from 2×10^4 simulation runs for reaction (5.4) with $c_1 = 0.0001$ and $X_1(0) = 61500$.



(a) $\epsilon = 0.03$

(b) $\epsilon = 0.1$

Figure 5.3: The estimated PDF of $X_1(2)$ from 2×10^4 simulation runs for reaction (5.5) with $c_1 = 0.00008$, $X_1(0) = 61500$ and $X_2(0) = 54000$.

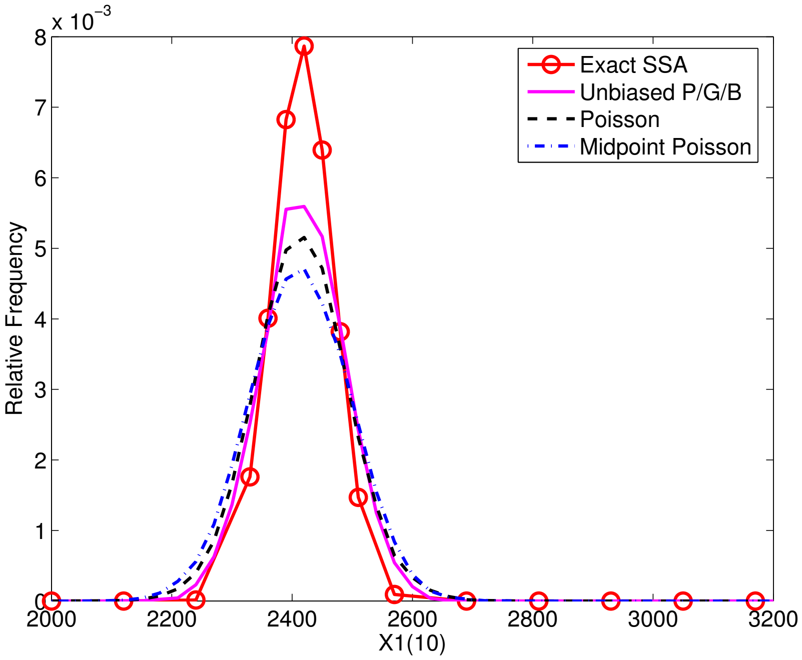


Figure 5.4: The estimated PDF of $X_1(10)$ from 5×10^4 simulation runs for the decay-dimerizing reactions (3.18) with rate constant (3.19) and initial condition (3.20). The leap methods use $\epsilon = 0.03$ to calculate τ.

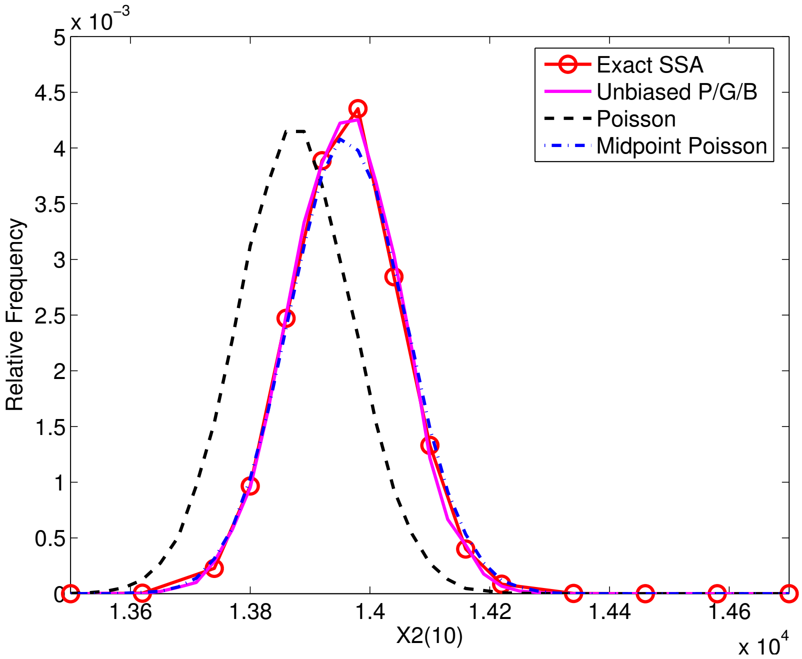


Figure 5.5: The estimated PDF of $X_2(10)$ from 5×10^4 simulation runs for the decay-dimerizing reactions (3.18) with rate constant (3.19) and initial condition (3.20). The leap methods use $\epsilon = 0.03$ to calculate τ.

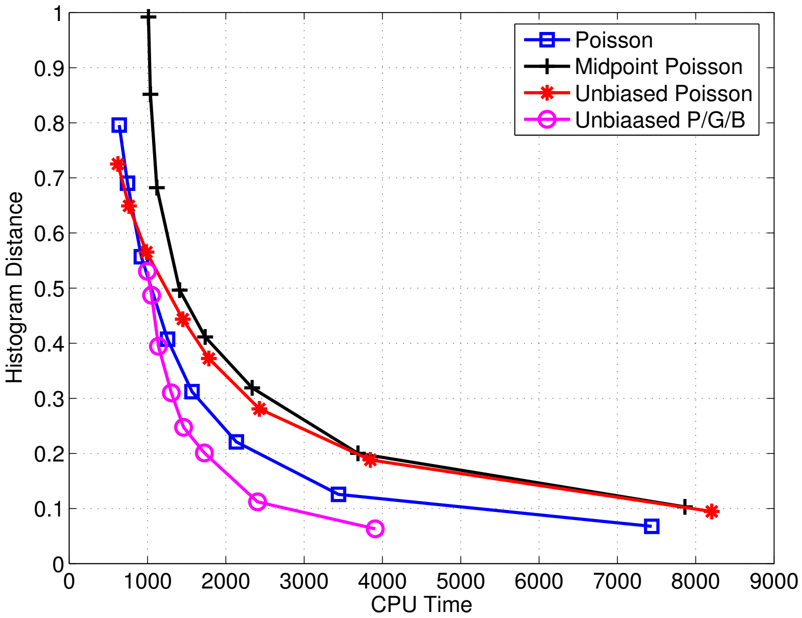


Figure 5.6: Histogram distance of $X_1(10)$ versus CPU time for the decaying-dimerizing reactions (3.18) with rate constants (3.19) and the initial condition (3.20). The histogram is obtained after 5×10^4 simulation runs and the CPU time is the total time (in seconds) of 5×10^4 runs.

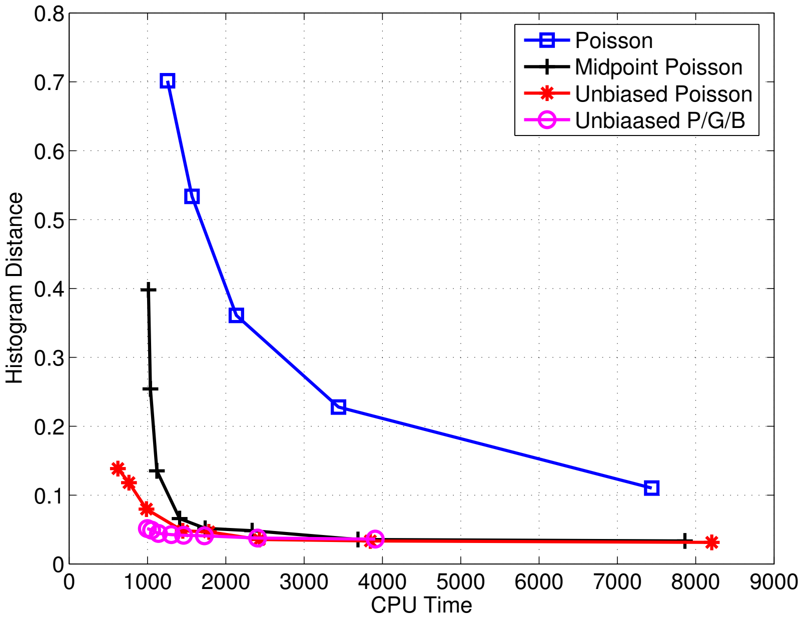


Figure 5.7: Histogram distance of $X_2(10)$ versus CPU time for the decaying-dimerizing reactions (3.18) with rate constants (3.19) and the initial condition (3.20). The histogram is obtained after 5×10^4 simulation runs and the CPU time is the total time (in seconds) of 5×10^4 runs.

Table 5.3: Mean of the number of molecules for several species in the EGF receptor signal pathway

Species	Exact SSA	Unbiased binomial τ-leap	Binomial τ-leap	Binomial mid-τ leap
Grb	26007	26006	25986	25985
Sh-G	61516	61517	61537	61537
Shc	5810.8	5810.7	5818.2	5817.5
Ra	12595	12595	12601	12599

*Leap methods use $\epsilon = 0.01$.

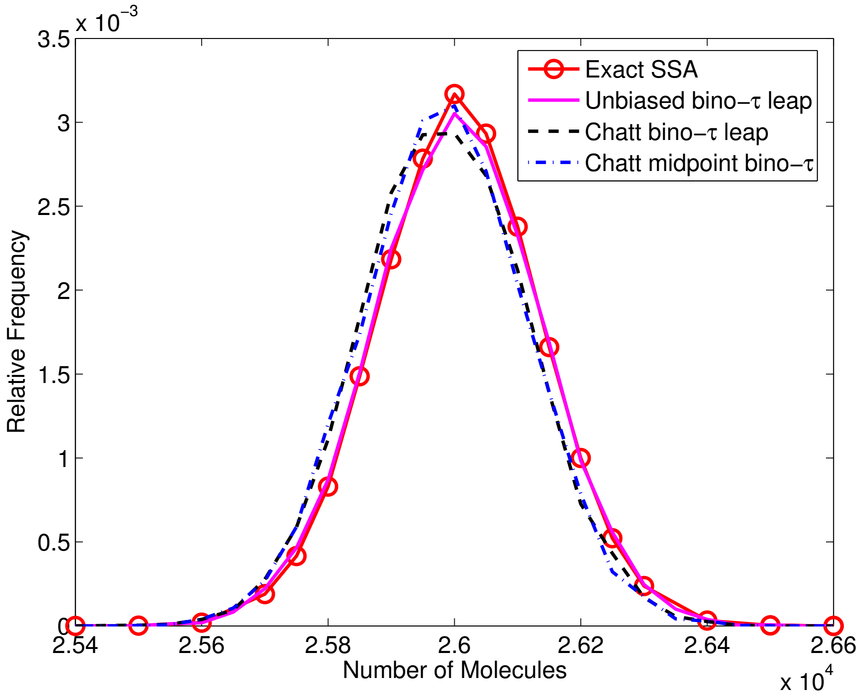


Figure 5.8: The estimated PDF of Grb at $t = 8$ in the EGF receptor signaling pathway. The PDF is estimated from the results of 10^4 simulation runs. Leap methods use $\epsilon = 0.01$ to calculate τ.

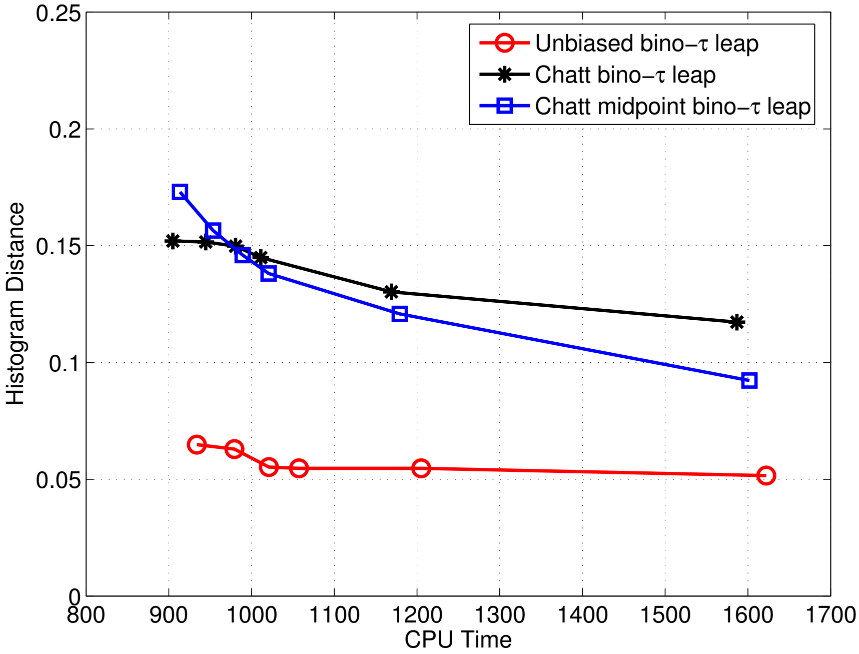


Figure 5.9: Histogram distance of Grb at $t = 8$ versus CPU time in the EGF receptor signaling pathway. The histogram is obtained from the results of 10^4 simulation runs and the CPU time is the total time (in seconds) of 10^4 runs.

CHAPTER 6

Improving the Weighted Stochastic Simulation Algorithm

6.1 Motivation

Although Gillespie's exact stochastic simulation algorithm (SSA) [32] can be used to simulate the stochastic dynamics of such systems, it often requires prohibitive computation to estimate the probability of a rare event which occurs in the system with an extremely small probability within a specified limited time. As some rare events in the cells of living organisms can have devastating effects, [72, 73] it is very important that computational simulation and analysis of systems with critical rare events can efficiently capture such rare events.

The weighted SSA (wSSA) recently developed by Kuwahara and Mura [44] based on the importance sampling technique enables one to efficiently estimate the probability of a rare event. However, the wSSA does not provide any method for selecting optimal values for importance sampling parameters. More recently, Gillespie *et al.* [45] analyzed the accuracy of the results yielded from the wSSA and proposed a refined wSSA (rwSSA) that employed a try-and-test method for selecting optimal values for importance sampling parameters. It was shown that the rwSSA could further improve the performance of wSSA. However, the try-and-test method requires some initial guessing for the sets of values from which the parameters can take. If the guessed values do not include the optimal value, then one cannot get appropriate values for the parameters. Moreover, if the number of parameters is greater than one, a very large set of values need to be guessed and tested, which may increase the likelihood of missing the optimal values and also increase computational overhead.

In this chapter, we first apply the importance sampling technique to the next reaction method (NRM) of the SSA [49] and develop a weighted NRM (wNRM) to improve the simulation efficiency, since the NRM only requires to generate one random variable per step while the SSA requires two random variables per step. We then develop a systematic method for selecting optimal values for importance sampling parameters, that can be incorporated into the wSSA or the wNRM resulting in an improved wSSA (iwSSA) or an improved wNRM (iwNRM). Our method does not need initial guessing and thus can guarantee near optimal values for the parameters. Our numerical results demonstrate that the variance of the estimated probability of the rare event provided by our iwSSA or iwNRM can be more than one order magnitude lower than that provided by the wSSA or the rwSSA for a given number of simulation runs. Moreover, our wSSA and wNRM require less simulation time than the rwSSA for the same number of simulations runs.

The remaining part of this chapter is organized as follows. In Section 6.2, we first describe the system setup and then briefly review Gillespie's exact SSA, [32], [33] the wSSA [44] and the rwSSA. [45] In Section 6.3, we develop the wNRM. In Section 6.4, we develop a systematic method for selecting optimal values for importance sampling methods and describe the iwSSA and iwNRM. In Section 6.5, we give some numerical examples that illustrate the benefits our iwSSA and the iwNRM. Finally in Section 6.6, we draw several conclusions.

6.2 Weighted Stochastic Simulation Algorithms

In order to capture a rare event which occurs with an extremely low probability in a given time period, Gillespie's SSA may require huge computation. Recently, a weighted SSA (wSSA) [44] and a refined wSSA (rwSSA) [45] were developed to estimate the probability of a rare event with a substantial reduction of computation. Following Kuwahara and Mura, [44] and Gillespie *et al.* [45], we define the rare event E_R that we want to capture in simulation as follows:

$$\begin{aligned}&E_R \text{ is an event that starting at time 0 in state } \mathbf{x}_0, \text{ the system will first reach}\\&\text{ any state in a specific set } \Omega \text{ at some time} \leq T, \text{ and the probability of}\\&E_R \text{ is very small, i.e., } P(E_R) \ll 1\end{aligned} \tag{6.1}$$

If we employ Gillespie's exact SSA to estimate $P(E_R)$, we would have to make a large number n of simulation runs, with each starting at time 0 in state $\mathbf{x}_0$ and terminating either when some state $\mathbf{x} \in \Omega$ is first reached or when the system time reaches T. If k is the number of those n runs that terminate for the first reason, then $P(E_R)$ is estimated as $\hat{P}(E_R) = k/n$. Since $P(E_R) \ll 1$, n should be very large to get a reasonably accurate estimate of $P(E_R)$. The wSSA and the rwSSA employ the importance sampling technique to reduce the number of runs needed to estimate $P(E_R)$.

Specifically, wSSA generates τ from its PDF (2.6) in the same way as used in Gillespie's exact SSA, but generates the reaction index μ from the following PMF:

$$q_\mu = b_\mu(\mathbf{x})/b_0(\mathbf{x}),\ \mu = 1, \cdots, M, \tag{6.2}$$

where $b_\mu(\mathbf{x}) = \gamma_\mu a_\mu(\mathbf{x}),\ \mu = 1, \ldots, M,\ b_0(\mathbf{x}) = \sum_{\mu=1}^{M} b_\mu(\mathbf{x})$ and $\gamma_\mu, \mu = 1, \cdots, M$ are positive constants that need to be chosen carefully before simulations are run.

Suppose a trajectory J generated in a simulation run contains h reactions, then the wSSA changes the probability of the trajectory from $P_J = \prod_{i=0}^{h-1} a_{\mu_i}(\mathbf{x}_i)/a_0(\mathbf{x}_i)$ to $Q_J = \prod_{i=0}^{h-1} b_{\mu_i}(\mathbf{x}_i)/b_0(\mathbf{x}_i)$. By choosing appropriate $\gamma_\mu, \mu = 1, \cdots, M$, one can increase the probability of the trajectories that lead to the rare event. If k trajectories out of n simulation runs lead to the rare event, then the importance sampling technique tells us that an unbiased estimate of $P(E_R)$ is given by

$$\begin{aligned} \hat{P}(E_R) &= \frac{1}{n}\sum_{j=1}^{k} \frac{P_J^j}{Q_J^j} \\ &= \frac{1}{n}\sum_{j=1}^{k} \frac{\prod_{i=0}^{h-1} a_{\mu_i}^j(\mathbf{x}_i)/a_0^j(\mathbf{x}_i)}{\prod_{i=0}^{h-1} b_{\mu_i}^j(\mathbf{x}_i)/b_0^j(\mathbf{x}_i)} \\ &= \frac{1}{n}\sum_{j=1}^{k}\prod_{i=0}^{h-1} w_i^j, \end{aligned} \tag{6.3}$$

where j and i are indices of the trajectories and reactions in a trajectory, respectively, $b_{\mu_i}^j(\mathbf{x}_i) = \gamma_\mu a_{\mu_i}^j(\mathbf{x}_i)$, and

$$w_i^j = \frac{p_{\mu_i}^j}{q_{\mu_i}^j} = \frac{a_{\mu_i}^j(\mathbf{x}_i)/a_0^j(\mathbf{x}_i)}{b_{\mu_i}^j(\mathbf{x}_i)/b_0^j(\mathbf{x}_i)}, \tag{6.4}$$

which can be obtained in each simulation step.

Kuwahara and Mura [44] did not provide any method for choosing γ_μ, although their numerical results with some pre-specified γ_μ for several reaction systems demonstrated that the wSSA could reduce computation substantially. Gillespie *et al.* [45] analyzed the variance of $\hat{P}(E_R)$ obtained from the wSSA and proposed the rwSSA that used a try-and-test method for choosing γ_μ. In the try-and-test method, several sets of values are pre-specified for γ_μ, $\mu = 1, \cdots, M$. A relatively small number of simulation runs are made for each set of the values to obtain an estimate of the variance of $\hat{P}(E_R)$, and then the set of values that yielded the smallest variance is chosen. Although the try-and-test method provides a way of choosing γ_μ, it requires some guessing to get several sets of pre-specified values for all γ_μ and also some computational overhead to estimate the variance of $\hat{P}(E_R)$ for each set of values. To avoid these problems, we will develop a more systematic method for choosing γ_μ, $\mu = 1, \cdots, M$ in Section 6.4, after we develop a wNRM to improve simulation speed in the next section.

6.3 Weighted Next Reaction Method for Stochastic Simulation

Both the wSSA and the rwSSA are based on the direct method of Gillespie's exact SSA, which needs to generate two random variables in each simulation step. However, the next reaction method (NRM) of Gibson and Bruck [49] requires only one random variable in each simulation step, which reduces computation. In this section, we apply the importance sampling technique to the NRM and develop the wNRM to reduce computation.

The key to making the wSSA more efficient than Gillespie's SSA is to change the probability of each reaction appropriately but without changing the distribution of the time τ between any two consecutive reactions. Since the NRM determines the reaction occurring in a simulation step by choosing the reaction that requires the smallest waiting time, it seems difficult to change the probability of each reaction without changing the distribution of τ. However, we notice that the PDF of τ in (2.6) only depends on $a_0(\mathbf{x})$ not individual $a_\mu(\mathbf{x})$. Hence, we can change the probability of each reaction by changing the corresponding propensity function but without changing the distribution of τ so long as we keep the sum of the propensity functions equal to $a_0(\mathbf{x})$. To this end, we define

$$d_m(\mathbf{x}) = \frac{b_m(\mathbf{x})a_0(\mathbf{x})}{b_0(\mathbf{x})}, \; m = 1, \cdots, M, \tag{6.5}$$

where $b_m(\mathbf{x}) = \gamma_m a_m(\mathbf{x})$ is defined in the same way as in the wSSA. It is easy to verify that $d_0(\mathbf{x}) = \sum_{m=1}^{M} d_m(\mathbf{x}) = a_0(\mathbf{x})$. If we generate τ_m from an exponential distribution $p(\tau_m) = d_m(\mathbf{x})\exp(-d_m(\mathbf{x})\tau_m)$, $\tau_m > 0$, as the waiting time of reaction channel m, and choose $\mu = \arg_m \min\{\tau_m, m = 1, \cdots, M\}$ as the index of the channel that fires, then it can be easily shown that the PDF of $\tau = \min\{\tau_m, m = 1, \cdots, M\}$ follows the exponential distribution in (2.6) and that the probability of reaction μ is $q_\mu = d_\mu(\mathbf{x})/d_0(\mathbf{x}) = b_\mu(\mathbf{x})/b_0(\mathbf{x})$. If we repeat this procedure in each simulation step, we would have modified the first reaction method (FRM) [33] and got a weighted FRM (wFRM). Clearly, the wFRM is not efficient since it generates M random variables in each step. However, following Gibson and Bruck, [49] we can convert the wFRM into a more efficient wNRM by reusing τ_ms.

Specifically, suppose that the μth reaction channel fires in the current step. After updating the state vector and propensity functions, we calculate new $d_m(\mathbf{x})$, $m = 1, \cdots, M$, which we denote as $d_m^{new}(\mathbf{x})$. Then, we generate a random variable τ_μ from an exponential distribution with parameter $d_\mu^{new}(\mathbf{x})$. For other channels with an index $m \neq \mu$, we update τ_m as follows: $\tau_m \leftarrow d_m(\mathbf{x})/d_m^{new}(\mathbf{x})(\tau_m - \tau)$. Gibson and Bruck [49] have shown that the new τ_m, $m = 1, \cdots, M$, are independent exponential random variables with parameters $d_m^{new}(\mathbf{x})$, $m = 1, \cdots, M$, respectively. Therefore, in the next step, we can again choose $\mu = \arg_m \min\{\tau_m, m = 1, \cdots, M\}$ as the index of the channel that fires, and set $\tau = \min\{\tau_m, m = 1, \cdots, M\}$. Essentially, our wNRM runs simulation in the same way as the NRM except that the wNRM generates τ_m using a parameter $d_m(\mathbf{x})$ instead of $a_m(\mathbf{x})$. To estimate the probability of the rare event $\hat{P}(E_R)$, we calculate a weight $w_\mu = \frac{p_\mu}{q_\mu} = \frac{a_\mu(\mathbf{x})/a_0(\mathbf{x})}{d_\mu(\mathbf{x})/d_0(\mathbf{x})} = a_\mu(\mathbf{x})/d_\mu(\mathbf{x})$ in each step and get $\hat{P}(E_R)$ using (6.3).

Therefore, based on the analysis above, the wNRM is summarized in the following algorithm:

Algorithm 6 (wNRM)

```
$k_1 \leftarrow 0$, $k_2 \leftarrow 0$, set values for all $\gamma_m$.
for i=1 to n, do
    $t \leftarrow 0$, $\mathbf{x} \leftarrow \mathbf{x}_0$, $w \leftarrow 1$.
    evaluate all $a_m(\mathbf{x})$ and $b_m(\mathbf{x})$; calculate all $d_m(\mathbf{x})$.
    for each m, generate a unit interval uniform random variable $r_m$; $\tau_m = \ln(1/r_m)/d_m(\mathbf{x})$.
    while $t \leq T$, do
        if $\mathbf{x} \in \Omega$, then
            $k_1 \leftarrow k_1 + w$, $k_2 \leftarrow k_2 + w^2$
            break out the while loop
        end if
        $\mu = \arg_m \min\{\tau_m, m = 1, \cdots, M\}$, $\tau = \min\{\tau_m, m = 1, \cdots, M\}$.
        $w \leftarrow w \times a_\mu(\mathbf{x})/d_\mu(\mathbf{x})$.
        $\mathbf{x} \leftarrow \mathbf{x} + \boldsymbol{\nu}_\mu$, $t \leftarrow t + \tau$.
        evaluate all $a_m(\mathbf{x})$ and $b_m(\mathbf{x})$; calculate all $d_m^{new}(\mathbf{x})$;
        for all $m \neq \mu$, $\tau_m \leftarrow d_m(\mathbf{x})/d_m^{new}(\mathbf{x})(\tau_m - \tau)$
```

```
            generate a unit interval uniform random variable $r_\mu$; $\tau_\mu \leftarrow \ln(1/r_\mu)/d_\mu^{new}(\mathbf{x})$.
            $d_m(\mathbf{x}) \leftarrow d_m^{new}(\mathbf{x})$.
        end while
    end for
    $\sigma^2 = k_2 - k_1^2$
    calculate $\hat{P}(E_R) = k_1/n$, with a 68% uncertainty of $\pm\sigma/\sqrt{n}$.
```

As in the wSSA, Algorithm 6 does not provide a method for choosing parameters γ_m, $m = 1, \cdots, M$. Although we could incorporate the try-and-test method in rwSSA into Algorithm 6, we will develop a more systematic method for choosing parameters in the next section. This parameter selection method will be applicable to both the wSSA and the wNRM and will significantly improve the performance of both algorithms as will be demonstrated in Section 6.5.

6.4 Parameter Selection for wSSA and wNRM

Let us denote the set of all possible state trajectories in the time interval $[0\ T]$ as $\mathcal{J}$ and the set of trajectories that first reach any state in Ω during $[0\ T]$ as $\mathcal{J}_E$. Let the probability of a trajectory J be P_J. Then we have $P(E_R) = \sum_{J\in\mathcal{J}_E} P_J = \sum_{J\in\mathcal{J}} P_J 1(J \in \mathcal{J}_E)$, where the indicator function $1(J \in \mathcal{J}_E) = 1$ if $J \in \mathcal{J}_E$ or 0 if $J \notin \mathcal{J}_E$. Importance sampling used in the wSSA and the wNRM arises from the factor that we can write $P(E_R)$ as

$$P(E_R) = \sum_{J\in\mathcal{J}} \frac{P_J 1(J \in \mathcal{J}_E)}{Q_J} Q_J, \tag{6.6}$$

where Q_J is the probability used in simulation to generate trajectory J, which is different from the true probability P_J if the system evolves naturally. If we make n simulation runs with altered trajectory probabilities, (6.6) implies that we can estimate $P(E_R)$ as $\hat{P}(E_R) = \frac{1}{n}\sum_{J\in\mathcal{J}} \frac{P_J 1(J\in\mathcal{J}_E)}{Q_J}$ which is essentially (6.3). The variance of $\hat{P}(E_R)$ depends on Q_Js. Appropriate Q_Js yield small variance, thereby improving the accuracy of the estimate or equivalently reducing the number of runs for a given variance. The "rule of thumb" for choosing good Q_Js is that Q_J should be roughly proportional to $P_J 1(J \in \mathcal{J}_E)$. We next rely on this rule to develop a method of choosing parameters for the wSSA and the wNRM.

However, at least two difficulties arise if we consider (6.6). First, the number of all possible trajectories is very large and we do not know the trajectories that lead to the rare event and their probabilities. Second, since we can only adjust the probability of each reaction in each step, it is not clear how this adjustment can affect the probability of a trajectory. To overcome these difficulties, we next get an approximate expression for $P(E_R)$ based on which we apply importance sampling.

The number of reactions K_T occurring in the time interval $[0\ T]$ is typically large, since the average time between two consecutive reactions is $1/a_0(\mathbf{x})$ which is typically much smaller than T. The standard deviation of K_T is typically much smaller than K_T. Therefore, we can approximate K_T by its mean value $\overline{K}_T$. Now, let us denote the rare event occurs at the Kth reaction with $K \leq \overline{K}_T$ as E_K and its probability as P_K. Then we have

$$P(E_R) \approx \sum_{K=1}^{\overline{K}_T} P_K. \tag{6.7}$$

Since E_K with $K < \overline{K}_T$ occurs at $t < T$ and the mean first passage time of the rare event is much larger than T, [45] it is reasonable to expect that $P_{\overline{K}_T} > P_K$ for all

$K < \overline{K}_T$. Therefore, based on the 'rule of thumb' described earlier, it is reasonable to maximize the probability of $E_{\overline{K}_T}$ in simulations with importance sampling.

Before proceeding with our derivations, we need to specify Ω. Let us denote the state when the rare event occurs at t as $\mathbf{X}^e(t)$. In the rest of the chapter, we assume that Ω contains one single state defined as $X_i^e(t) = X_i(0) + \eta$, where η is a constant and $i \in \{1, 2, \cdots, N\}$. Let us denote the number of the mth reaction occurring in the trajectory leading to the rare event as K_m. Then we have

$$\eta = \sum_{m=1}^{M} \nu_{im} K_m. \tag{6.8}$$

We divide all reactions into three groups: G_1 group consists of reactions with $\nu_{im}\eta > 0$, G_2 group consists of reactions with $\nu_{im}\eta < 0$, and G_3 group consists of reactions with $\nu_{im} = 0$. Note that the reactions in G_1 (G_2) group increase (decrease) the probability of the rare event and that the reactions in G_3 group do not affect $X_i(t)$ directly.

We typically only need to consider elementary reactions including bimolecular and monomolecular reactions. [47] Hence the possible values for all ν_{im} are $0, \pm 1, \pm 2$. For the simplicity of derivations, we now only consider the case where $\nu_{im} = 0, \pm 1$, i.e., we assume that the system dose not have any bimolecular reactions with two identical reactant molecules or dimerization reactions. We will later generalize our method to the system with dimerization reactions. Let us define $K_{G_1} = \sum_{m \in G_1} K_m$ and $K_{G_2} = \sum_{m \in G_2} K_m$, then (6.8) becomes

$$\eta = K_{G_1} - K_{G_2}. \tag{6.9}$$

We next consider systems with only G_1 and G_2 reaction groups and then consider more general systems with all three reaction groups.

6.4.1 Systems with G_1 and G_2 reaction groups

Let us consider event $E_{\overline{K}_T}$ and let $Q_{\overline{K}_T}$ be the probability of $E_{\overline{K}_T}$ used in simulation which is different from its true probability $P_{\overline{K}_T}$ when the system evolves naturally. As we discussed earlier, our goal is to use the maximum value of $Q_{\overline{K}_T}$. The last reaction in $E_{\overline{K}_T}$ should be a reaction from G_1 group. Otherwise, the rare event has already occurred before the $\overline{K}_T$th reaction occurs. Suppose that in simulation the total probability of the occurrence of reactions in G_1 group is a constant Q_{G_1} and similarly the total probability of the occurrence of reactions in G_2 group is $Q_{G_2} = 1 - Q_{G_1}$. Since we have

$$K_{G_1} + K_{G_2} = \overline{K}_E, \tag{6.10}$$

K_{G_1} and K_{G_2} follow a binomial distribution and we have

$$Q_{\overline{K}_K} = \frac{(\overline{K}_T - 1)!}{(K_{G_1} - 1)! K_{G_2}!} Q_{G_1}^{K_{G_1}} (1 - Q_{G_1})^{K_{G_2}}. \tag{6.11}$$

From (6.9) and (6.10), we get $K_{G_1} = (\overline{K}_T + \eta)/2$ and $K_{G_2} = (\overline{K}_T - \eta)/2$. Substituting K_{G_1} and K_{G_2} into (6.11), we can find Q_{G_1} and Q_{G_2} that maximize $Q_{\overline{K}_K}$ as follows:

$$\begin{aligned} Q_{G_1} &= \frac{(\overline{K}_T + \eta)}{2\overline{K}_T} \\ Q_{G_2} &= \frac{(\overline{K}_T - \eta)}{2\overline{K}_T}. \end{aligned} \tag{6.12}$$

To ensure that reactions in G_1 (G_2) group occur with probability Q_{G_1} (Q_{G_1}), at each step of simulation, we adjust the probability of each reaction as follows

$$q_m = \begin{cases} \frac{Q_{G_1} a_m(\mathbf{x})}{a_{G_1}(\mathbf{x})}, & m \in G_1 \\ \frac{Q_{G_2} a_m(\mathbf{x})}{a_{G_2}(\mathbf{x})}, & m \in G_2, \end{cases} \tag{6.13}$$

where $a_{G_1}(\mathbf{x}) = \sum_{m \in G_1} a_m(\mathbf{x})$ and $a_{G_2}(\mathbf{x}) = \sum_{m \in G_2} a_m(\mathbf{x})$. It is easy to verify that $\sum_{m \in G_1} q_m = Q_{G_1}$ and $\sum_{m \in G_2} q_m = Q_{G_2}$. As we discussed earlier, the weight for estimating the probability of the rare event is $w_\mu = p_\mu / q_\mu$ if the μth reaction channel fires.

6.4.2 Systems with G_1, G_2 and G_3 reaction groups

Again, we consider event $E_{\overline{K}_T}$. Let us define $K_{G_3} = \sum_{m \in G_3} K_m$. Then we have

$$K_{G_1} + K_{G_2} + K_{G_3} = \overline{K}_T. \tag{6.14}$$

Similar to Q_{G_1} and Q_{G_2}, we denote the total probability of the occurrence of reactions in G_3 group as Q_{G_3}. Clearly, K_{G_1}, K_{G_2} and K_{G_3} follow a multinomial distribution and we have

$$Q_{\overline{K}_T} = \sum_{K_{G_2}=0}^{\overline{K}_T} \frac{(\overline{K}_T - 1)!}{(K_{G_1} - 1)! K_{G_2}! K_{G_3}!} Q_{G_1}^{K_{G_1}} Q_{G_2}^{K_{G_2}} Q_{G_3}^{K_{G_3}}. \tag{6.15}$$

From (6.9) and (6.14), we get $K_{G_1} = K_{G_2} + \eta$ and $K_{G_3} = \overline{K}_E - \eta - 2K_{G_2}$. Since $K_{G_3} \geq 0$, we have $K_{G_2} \leq (\overline{K}_E - \eta)/2$. Then we can write (6.15) as

$$Q_{\overline{K}_T} = \sum_{K_{G_2}=0}^{(\overline{K}_T-\eta)/2} \frac{(\overline{K}_T - 1)!}{(K_{G_2} + \eta - 1)! K_{G_2}! (\overline{K}_T - \eta - 2K_{G_2})!} Q_{G_1}^{K_{G_2}+\eta} Q_{G_2}^{K_{G_2}} Q_{G_3}^{\overline{K}_T-\eta-2K_{G_2}}. \tag{6.16}$$

Since there are $(\overline{K}_T - \eta)/2 + 1$ terms in the summation in (6.16), it is difficult to find Q_{G_1}, Q_{G_2} and Q_{G_3} that maximize $Q_{\overline{K}_T}$.

Let $\overline{K}_{G_1}$, $\overline{K}_{G_2}$ and $\overline{K}_{G_3}$ be the average number of reactions from G_1, G_2 and G_3 that occur in the time interval $[0\ T]$ when the system evolves naturally. Since we have $\overline{K}_{G_1} + \overline{K}_{G_2} + \overline{K}_{G_3} = \overline{K}_T$, we define $P_{G_1} = \overline{K}_{G_1}/\overline{K}_T$, $P_{G_2} = \overline{K}_{G_2}/\overline{K}_T$ and $P_{G_3} = \overline{K}_{G_3}/\overline{K}_T$. Then we can approximate $P_{\overline{K}_T}$ using the right hand side of (6.16) but with Q_{G_i}, $i = 1, 2, 3$, replaced by P_{G_i}, $i = 1, 2, 3$, respectively, i.e.,

$$P_{\overline{K}_T} \approx \sum_{K_{G_2}=0}^{(\overline{K}_T-\eta)/2} \frac{(\overline{K}_T - 1)!}{(K_{G_2} + \eta - 1)! K_{G_2}! (\overline{K}_T - \eta - 2K_{G_2})!} P_{G_1}^{K_{G_2}+\eta} P_{G_2}^{K_{G_2}} P_{G_3}^{\overline{K}_T-\eta-2K_{G_2}}. \tag{6.17}$$

Suppose that the $(K_{\max} + 1)$th term in the summation in (6.17) is the largest, then based on the "rule of thumb" we described earlier, we can maximize the $(K_{\max} + 1)$th term in the summation in (6.16) instead of $Q_{\overline{K}_T}$ to find Q_{G_1}, Q_{G_2} and Q_{G_3}.

It is not difficult to find the $(K_{\max}+1)$th term in the summation in (6.17). Let us denote the $(K_{G_2}+1)$th term in the summation in (6.17) as $f(K_{G_2})$. We can exhaustively search over all $f(K_{G_2})$, $K_{G_2}=0,\cdots,(\overline{K}_T-\eta)/2$ to find $K_{\max}$. However, this may require relatively large computation because the factorials involved in $f(K_{G_2})$. We can reduce computation by searching over $g(K_{G_2})=f(K_{G_2}+1)/f(K_{G_2})$, $K_{G_2}=1,\cdots,(\overline{K}_T-\eta)/2-1$, which are given by

$$g(K_{G_2})=\frac{(\overline{K}_T-\eta-2K_{G_2})(\overline{K}_T-\eta-2K_{G_2}-1)P_{G_1}P_{G_2}}{(K_{G_2}+\eta)(K_{G_2}+1)P_{G_3}^2}. \tag{6.18}$$

Specifically, we calculate all $g(K_{G_2})$ from (6.18). If $g(K_{G_2})>1$ but $g(K_{G_2}+1)<1$, then $f(K_{G_2})$ is a local maximum. After obtaining all local maximums, we can find the global maximum $f(K_{\max})$ from the local maximums.

Let us define $\widetilde{K}_{G_2}=K_{\max}$, $\widetilde{K}_{G_1}=\widetilde{K}_{G_2}+\eta$ and $\widetilde{K}_{G_3}=\overline{K}_T-\eta-2\widetilde{K}_{G_2}$. Then we can find Q_{G_1}, Q_{G_2} and Q_{G_3} that maximize the $(\widetilde{K}_{G_2}+1)$th term in the summation in (6.16) as follows

$$\begin{aligned}Q_{G_1}&=\frac{\widetilde{K}_{G_1}}{\overline{K}_T}=\frac{\widetilde{K}_{G_2}+\eta}{\overline{K}_T}\\Q_{G_2}&=\frac{\widetilde{K}_{G_2}}{\overline{K}_T}\\Q_{G_3}&=\frac{\widetilde{K}_{G_3}}{\overline{K}_T}=\frac{\overline{K}_T-\eta-2\widetilde{K}_{G_2}}{\overline{K}_T}.\end{aligned} \tag{6.19}$$

Substituting Q_{G_1} and Q_{G_2} in (6.19) into (6.13), we get the probability q_m, $m\in G_1$ or G_2 that is used to generate the mth reaction in each step of simulation. For G_3 group, we get the probability of each reaction as follows

$$q_m=\frac{Q_{G_3}a_m(\mathbf{x})}{a_{G_3}(\mathbf{x})},\ m\in G_3, \tag{6.20}$$

where $a_{G_3}(\mathbf{x})=\sum_{m\in G_3}a_m(\mathbf{x})$.

While we can use q_m in (6.20) to generate reactions in G_3 group, we next develop a method for fine-tuning q_m, $m\in G_3$, which can further reduce the variance of

$\hat{P}(E_R)$. We divide G_3 group into three subgroups: G_{31}, G_{32} and G_{33}. Occurrence of reactions in G_{31} group increases probability of occurrence of reactions in Q_{G_1} group or reduces the probability of the occurrence of the reactions in Q_{G_2} group, which in turn increases the probability of the rare event. Occurrence of reactions in G_{32} group reduces probability of occurrence of reactions in Q_{G_1} group or increases the probability of the occurrence of reactions in Q_{G_2} group, which reduces the probability of the rare event. Occurrence of reactions in G_{33} group does not change the probability of occurrence of reactions in Q_{G_1} and Q_{G_2} groups, which does not change the probability of the rare event.

Let $\overline{K}_{G_{31}}$, $\overline{K}_{G_{32}}$ and $\overline{K}_{G_{33}}$ be the average number of reactions from G_{31}, G_{32} and G_{33} that occur in the time interval $[0\ T]$ when the system evolve naturally. we define $P_{G_{31}} = \overline{K}_{G_{31}}/\overline{K}_T$, $P_{G_{32}} = \overline{K}_{G_{32}}/\overline{K}_T$ and $P_{G_{33}} = \overline{K}_{G_{33}}/\overline{K}_T$. Our goal is to make Q_{31} to be greater than $P_{G_{31}}$ and Q_{32} to be less than $P_{G_{32}}$ to increase the probability of the rare event. To this end, we propose the following formula to determine Q_{31}, Q_{32} and Q_{33}:

$$\begin{aligned} Q_{G_{31}} &= P_{G_{31}} + Q_{G_3}\alpha - P_{G_3}\beta \\ Q_{G_{32}} &= P_{G_{32}} + Q_{G_3}(1-\alpha) - P_{G_3}(1-\beta) \\ Q_{G_{33}} &= P_{G_{33}} \end{aligned} \tag{6.21}$$

where $\alpha, \beta \in (0\ 1)$ are two pre-specified constants. It is not difficult to verify from (6.21) that $Q_{G_{31}} + Q_{G_{32}} + Q_{G_{33}} = Q_{G_3}$. To ensure that $Q_{G_{31}} \geq P_{G_{31}}$ and $Q_{G_{32}} \leq P_{G_{32}}$, we choose α and β satisfying $0 \leq \beta < 1$, $\frac{P_{G_3}}{Q_{G_3}}\beta \leq \alpha < 1$ if $Q_{G_3} \geq P_{G_3}$, or satisfying $0 \leq \beta < 1$, $\max\Big\{0, 1 - \frac{P_{G_3}}{Q_{G_3}}(1-\beta)\Big\} \leq \alpha < 1$ if $Q_{G_3} < P_{G_3}$. Finally, we obtain q_m for $m \in G_3$ as follows

$$q_m = \begin{cases} \frac{Q_{G_{31}} a_m(\mathbf{x})}{a_{G_{31}}(\mathbf{x})}, & m \in G_{31} \\ \frac{Q_{G_{32}} a_m(\mathbf{x})}{a_{G_{32}}(\mathbf{x})}, & m \in G_{32} \\ \frac{Q_{G_{33}} a_m(\mathbf{x})}{a_{G_{33}}(\mathbf{x})}, & m \in G_{33}, \end{cases} \tag{6.22}$$

where $a_{G_{3i}}(\mathbf{x}) = \sum_{m \in G_{3i}} a_m(\mathbf{x})$, $i = 1, 2, 3$.

6.4.3 Systems with dimerization reactions

So far we assumed that the system did not have any dimerization reactions, i.e. the system consisted of reactions with $|\nu_{im}| = 0$ or 1. We now generalize our methods developed earlier to the system with dimerization reactions. If there are dimerization reactions in G_1 and G_2 groups, we further divide G_1 group into G_{11} and G_{12} subgroups and G_2 group into G_{21} and G_{22} subgroups. The G_{11} group contains reactions with $\nu_{im}\text{sign}(\eta) = 1$, where $\text{sign}(\eta) = 1$ when $\eta > 0$ and $\text{sign}(\eta) = -1$ when $\eta < 0$. The G_{12} group contains reactions with $\nu_{im}\text{sign}(\eta) = 2$. The G_{21} group contains reactions with $\nu_{im}\text{sign}(\eta) = -1$, while the G_{12} group contains reactions with $\nu_{im}\text{sign}(\eta) = -2$.

Let us define $K_{G_{11}} = \sum_{m \in G_{11}} K_m$; $K_{G_{12}} = \sum_{m \in G_{12}} K_m$; $K_{G_{21}} = \sum_{m \in G_{21}} K_m$; $K_{G_{22}} = \sum_{m \in G_{22}} K_m$. Clearly, we have $K_{G_1} = K_{G_{11}} + K_{G_{12}}$ and $K_{G_2} = K_{G_{21}} + K_{G_{22}}$. Then (6.8) becomes

$$K_{G_{11}} + 2K_{G_{12}} - K_{G_{21}} - 2K_{G_{22}} = \eta. \tag{6.23}$$

Let us consider systems with G_1 and G_2 groups but without G_3 group. Although we still have $K_{G_1} + K_{G_2} = \overline{K}_T$ or equivalently $K_{G_{11}} + K_{G_{12}} + K_{G_{21}} + K_{G_{22}} = \overline{K}_T$, we cannot obtain four unknowns $K_{G_{11}}$, $K_{G_{12}}$, $K_{G_{21}}$ and $K_{G_{22}}$ from only two equations.

Suppose that $\overline{K}_{G_{11}}$, $\overline{K}_{G_{12}}$, $\overline{K}_{G_{21}}$ and $\overline{K}_{G_{22}}$ are average number of reactions from G_{11}, G_{12}, G_{21} and G_{22} groups that occur in the time interval $[0\ T]$ if the system evolves naturally. We notice from (6.13) that we do not change the ratio of the probabilities of two reactions in the same group, i.e., $q_{m_1}/q_{m_2} = p_{m_1}/p_{m_2}$ if m_1 and m_2 are in the same group. Therefore, we would expect that $K_{G_{12}}/K_{G_{11}} = \overline{K}_{G_{12}}/\overline{K}_{G_{11}}$ and $K_{G_{22}}/K_{G_{21}} = \overline{K}_{G_{22}}/\overline{K}_{G_{21}}$. Using these two relationships, we can write (6.23) as

$$\lambda_1 K_{G_1} - \lambda_2 K_{G_2} = \eta \tag{6.24}$$

where $\lambda_1 = \frac{(\overline{K}_{G_{11}}+2\overline{K}_{G_{12}})}{(\overline{K}_{G_{11}}+\overline{K}_{G_{12}})}$ and $\lambda_2 = \frac{(\overline{K}_{G_{21}}+2\overline{K}_{G_{22}})}{(\overline{K}_{G_{21}}+\overline{K}_{G_{22}})}$.

From (6.10) and (6.24), we obtain $K_{G_1} = (\lambda_1\overline{K}_T + \eta)/(\lambda_1 + \lambda_2)$ and $K_{G_2} = (\lambda_2\overline{K}_T - \eta)/(\lambda_1 + \lambda_2)$. Substituting K_{G_1} and K_{G_2} into (6.11) and maximizing $Q_{\overline{K}_K}$, we obtain

$$\begin{aligned} Q_{G_1} &= \frac{\lambda_1\overline{K}_T + \eta}{(\lambda_1 + \lambda_2)\overline{K}_T} \\ Q_{G_2} &= \frac{\lambda_2\overline{K}_T - \eta}{(\lambda_1 + \lambda_2)\overline{K}_T}. \end{aligned} \tag{6.25}$$

We then substitute Q_{G_1} and Q_{G_2} into (6.13) to get q_m.

Now let us consider the systems with G_1, G_2 and G_3 reactions. From (6.24) we have $K_{G_1} = (\lambda_2 K_{G_2} + \eta)/\lambda_1$, and from (6.14) and (6.24) we obtain $K_{G_3} = \overline{K}_T - [(\lambda_1 + \lambda_2)K_{G_2} + \eta]/\lambda_1$. Since $K_{G_3} \geq 0$, we have $K_{G_2} \leq (\lambda_1\overline{K}_T - \eta)/(\lambda_1 + \lambda_2)$. Following the derivations in Section 6.4.2, we can get q_m for any reaction. More specifically, substituting K_{G_1}, K_{G_3} and the upper limit of K_{G_2} into (6.15), we obtain $Q_{\overline{K}_T}$. We can also get $P_{\overline{K}_T}$ similar to (6.17) by replacing Q_{G_i} in $Q_{\overline{K}_T}$ with P_{G_i}. Then we determine the maximum term in the summation of P_{G_i} and denote the value of K_{G_2} corresponding to the maximum term as $K_{\max}$. We find Q_{G_1}, Q_{G_2} and Q_{G_3} by maximizing the $(K_{\max} + 1)$th term in the summation in $Q_{\overline{K}_T}$. Finally, we substitute Q_{G_1} and Q_{G_2} into (6.13) to get q_m, $m \in G_1$ or G_2. For the reactions in G_3 group, we can either substitute Q_{G_3} into (6.20) to obtain q_m, or if we want to fine-tune q_m, we use (6.21) and (6.22) to get q_m.

6.4.4 The iwSSA Algorithm

The key to determining probability of each reaction q_m is to find the total probability of each group, Q_{G_1}, Q_{G_2}, Q_{G_3}, $Q_{G_{31}}$, $Q_{G_{32}}$ and $Q_{G_{33}}$. This requires the average number of reactions of each group occurring during the interval [0 T], $\overline{K}_T$, $\overline{K}_{11}$, $\overline{K}_{12}$, $\overline{K}_{21}$, $\overline{K}_{22}$, $\overline{K}_{31}$, $\overline{K}_{32}$, $\overline{K}_{33}$, if the system evolves naturally. If the system is relatively simple, we may get these numbers analytically. If we cannot obtain them analytically, we can estimate them by running Gillespie's exact SSA. Since the number of runs needed to estimates these numbers is much smaller than the number of runs needed to estimate the probability of the rare event, the computational overhead is negligible.

We now incorporate our probability selection method into the wSSA and summarize the improved wSSA in the following algorithm.

Algorithm 7 (iwSSA)

```
run Gillespie's exact SSA $10^3$-$10^4$ times to get estimates of $\overline{K}_T$, $\overline{K}_{11}$, $\overline{K}_{12}$, $\overline{K}_{21}$, $\overline{K}_{22}$, $\overline{K}_{31}$, $\overline{K}_{32}$, $\overline{K}_{33}$.
calculate $Q_{G_1}$, $Q_{G_2}$, $Q_{G_3}$, $Q_{G_{31}}$, $Q_{G_{32}}$ and $Q_{G_{33}}$.
$k_1 \leftarrow 0$, $k_2 \leftarrow 0$.
for $i$=1 to $n$, do
    $t \leftarrow 0$, $\mathbf{x} \leftarrow \mathbf{x}_0$, $w \leftarrow 1$.
    while $t \leq T$, do
        if $\mathbf{x} \in \Omega$, then
            $k_1 \leftarrow k_1 + w$, $k_2 \leftarrow k_2 + w^2$
            break out the while loop
        end if
        evaluate all $a_m(\mathbf{x})$; calculate $a_0(\mathbf{x})$.
        generate two unit-interval uniform random variables $r_1$ and $r_2$.
        $\tau \leftarrow \ln(1/r)1)/a_0(\mathbf{x})$
        calculate all $q_m$ from (6.13), (6.20) or (6.22).
        $\mu \leftarrow$ smallest integer satisfying $\sum_{m=1}^{\mu} q_m > r_2 q_0$.
        $w \leftarrow w \times \big(a_\mu(\mathbf{x})/a_0(\mathbf{x})\big)/\big(q_\mu(\mathbf{x})/q_0(\mathbf{x})\big)$.
        $\mathbf{x} \leftarrow \mathbf{x} + \boldsymbol{\nu}_\mu$, $t \leftarrow t + \tau$.
    end while
end for
$\sigma^2 = k_2 - k_1^2$
estimate $\hat{P}(E_R) = k_1/n$, with a 68% uncertainty of $\pm\sigma/\sqrt{n}$.
```

Comparing with the rwSSA, [45] our iwSSA does not need to make some guessing about the parameters for adjusting the probability of each reactions q_m, but directly calculate q_m using a systematically developed method. This has two main advantages. First, our iwSSA will always adjust q_m appropriately to reduce the variance of $\hat{P}(E_R)$, whereas the rwSSA may not adjust q_m as well as our iwSSA, especially if the initial guessed values are far away from the optimal values. Second, as we mentioned earlier, the computational overhead of our iwSSA is negligible, whereas rwSSA requires non-negligible computational overhead for determining parameters. Indeed, as we will show in Section 6.5, the variance of $\hat{P}(E_R)$ provided by our iwSSA can be more than one order of magnitude lower than that provide by the rwSSA for given number of n. Moreover, our iwSSA is faster than the rwSSA, since the iwSSA requires less computational overhead to adjust q_m.

We can also incorporate our probability selection method into wNRM to develop an improved wNRM. To this end, we replace the first line of Algorithm 6 with the first three lines of Algorithm 7, change the fourth line of Algorithm 6 to the following: evaluate all $a_m(\mathbf{x})$, find all q_m from (6.13), (6.20) or (6.22), and calculate $d_m(\mathbf{x})$ as $d_m(\mathbf{x}) = q_m a_0(\mathbf{x})$. Finally, we change the fourteenth line of Algorithm 6 to the following: evaluate all $a_m(\mathbf{x})$ and q_m; calculate all $d_m^{new}(\mathbf{x})$.

6.5 Numerical Examples

In this section we present simulation results for several chemical reaction systems to demonstrate the accuracy and efficiency of our iwSSA and iwNRM. All simulations were run in Matlab on a PC with an Intel dual Core 2.67 GHz CPU and 3G-byte memory running Windows XP.

6.5.1 Single species production-degradation model

This simple system was originally used by Kuwahara and Mura [44] and then Gillespie *et al.* [45] to test the wSSA and the rwSSA. It includes the following two chemical reactions:

$$R_1 : S_1 \xrightarrow{c_1} S_1 + S_2,\ R_2 : S_2 \xrightarrow{c_2} \emptyset. \tag{6.26}$$

In reaction R_1 species S_1 synthesizes species S_2 with a probability rate constant c_1, while in reaction R_2 species S_2 is degraded with a probability rate constant c_2. We used the same initial state and probability rate constants as used in Refs. [44] and [45]: $X_1(0) = 1$, $X_2(0) = 40$, $c_1 = 1$ and $c_2 = 0.025$.

It is observed that the system is at equilibrium, since $a_1(\mathbf{x}_0) = c_1 \times X_1(0) = c_2 \times X_2(0) = a_2(\mathbf{x}_0)$. It can be shown [44] that $X_2(t)$ is a Poisson random variable with mean equal to 40. Refs. [44] and [45] sought to estimate $P(E_R)$=$P_{t\leq 100}(X_2 \rightarrow \theta|\mathbf{x}_0)$, the probability of $X_2(t) = \theta$ for $t \leq 100$ and several values of θ between 65 and 80. Since θ is about four to six standard deviations above the mean value 40, $P_{t\leq 100}(X_2 \rightarrow \theta|\mathbf{x}_0)$ is very small.

Kuwahara and Mura [44] employed the wSSA to estimate $P(E_R)$ and used $b_1(\mathbf{x}) = \delta a_1(\mathbf{x})$ and $b_2(\mathbf{x}) = 1/\delta a_2(\mathbf{x})$ with $\delta = 1.2$ for four different values of θ: 65, 70, 75 and 80. Gillespie *et al.* [45] applied the rwSSA to estimate $P(E_R)$ and used the same way to determine $b_1(\mathbf{x})$ and $b_2(\mathbf{x})$ but found that $\delta = 1.2$ is near optimal for $\theta = 65$ and that $\delta = 1.3$ is near optimal for $\theta = 80$. We repeated the simulation of Gillespie *et al.* [45] for $\theta = 65, 70, 75$ and 80 with $\delta = 1.2, 1.25, 1.25$ and 1.3, respectively. We then applied our iwSSA and iwNRM to estimate $P(E_R)$ for $\theta = 65, 70, 75$ and 80. This system has only two types of reaction: R_1 is a G_1 reaction and R_2 is a G_2 reaction. Since the system is at equilibrium with $a_0(\mathbf{x}_0) = 2$, $\overline{K}_T$ with $T = 100$ is estimated to be 200. Using (6.12), we get $q_1 = Q_{G_1} = (\overline{K}_T + \theta)/2/\overline{K}_T$ and $q_2 = 1 - q_1$.

Table 6.1 gives the estimated probability $\hat{P}(E_R)$ and its sample variance σ^2 for the iwNRM, the iwSSA and the rwSSA, obtained from 10^7 simulation runs with $\theta = 65, 70, 75$ and 80. It is seen that $\hat{P}(E_R)$ is almost identical for all three methods. However, our iwNRM and iwSSA provide variance almost two order of magnitude lower than the rwSSA for $\theta = 80$, or less than or almost one order of magnitude lower than the rwSSA for $\theta = 75, 70$ and 65. Moreover, our iwNRM and iwSSA need about 60% and 70% CPU time of the rwSSA, respectively. Note that the CPU time for the

Table 6.1: Estimated probability of rare event and sample variance as well as CPU time with 10^7 runs of iwNRM, iwSSA and rwSSA methods for the example of single species production-degradation model

(a) $\theta = 65$

	$\hat{P}(E_R)$	σ^2	TIME
iwNRM	2.29×10^{-3}	5.09×10^{-6}	14472
iwSSA	2.29×10^{-3}	5.10×10^{-6}	16737
rwSSA	2.29×10^{-3}	3.39×10^{-5}	24340

(b) $\theta = 70$

	$\hat{P}(E_R)$	σ^2	TIME
iwNRM	1.68×10^{-4}	3.40×10^{-8}	16140
iwSSA	1.68×10^{-4}	3.40×10^{-8}	18555
rwSSA	1.68×10^{-4}	4.29×10^{-7}	25492

(c) $\theta = 75$

	$\hat{P}(E_R)$	σ^2	TIME
iwNRM	8.42×10^{-6}	1.10×10^{-10}	15640
iwSSA	8.42×10^{-6}	1.10×10^{-10}	18582
rwSSA	8.43×10^{-6}	3.58×10^{-9}	26314

(d) $\theta = 80$

	$\hat{P}(E_R)$	σ^2	TIME
iwNRM	2.99×10^{-7}	1.82×10^{-13}	16260
iwSSA	2.99×10^{-7}	1.82×10^{-13}	18960
rwSSA	2.99×10^{-7}	1.29×10^{-11}	26987

rwSSA in Table 6.1 does not include the time needed for searching for the optimal value of δ for each θ. The less CPU time used by the iwNRM is expected since it only requires to generate one random variable in each step, whereas the iwSSA and rwSSA needs to generate two random variables. It is also reasonable that the iwSSA requires less CPU time than rwSSA, because the iwSSA needs less computation to calculate the probability of each reaction in each step. Figure 6.1 compares the standard deviation ($\sigma/\sqrt{n}$) of $\hat{P}(E_R)$ for the iwSSA and the rwSSA with different

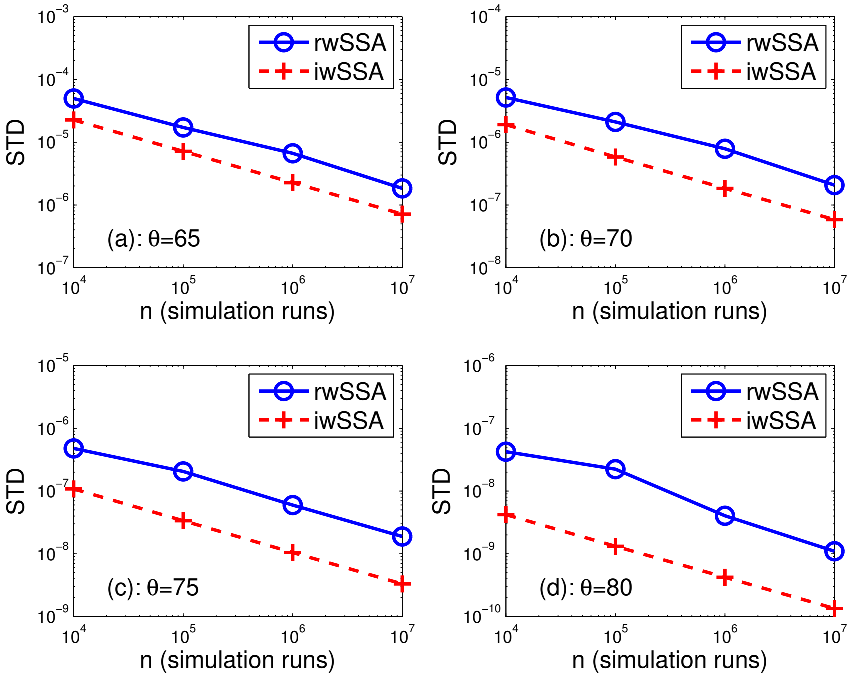


Figure 6.1: The standard deviation (STD) versus number of simulation runs for single species production-degradation model (6.26) with $c_1 = 1$, $c_2 = 0.025$, $X_1(0) = 1$ and $X_2(0) = 40$ with $\theta = 65, 70, 75$ and 80.

number of runs, n. Since the iwNRM provides almost the same standard deviation as the iwSSA, we do not plot it in the figure. It is seen that our iwSSA consistently yields much smaller standard deviation than rwSSA for all values of n.

6.5.2 A reaction system with G_1, G_2 and G_3 reactions

The previous system only contains a G_1 reaction and a G_2 reaction. We also used the following system with G_1, G_2 and G_3 reactions to test our iwNRM and iwSSA:

$$R_1: \; S_1 \xrightarrow{c_1} S_2, \;\; R_2: \; S_2 \xrightarrow{c_2} \emptyset, \;\; R_3: \; S_3 \xrightarrow{c_3} S_1 + S_3, \;\; R_4: \; S_1 \xrightarrow{c_4} \emptyset. \qquad (6.27)$$

In this system, a monomer S_1 converts to S_2 with a probability rate constant c_1 while S_2 is degraded with a probability rate constant c_2. Meanwhile, another species S_3 synthesizes S_1 with a probability rate constant c_3 and S_1 degrades with a probability rate constant c_4. In our simulations, we used the following values for the probability rate constants and the initial state:

$$c_1 = 0.1,\ c_2 = 0.1,\ c_3 = 8,\ c_4 = 0.1, \tag{6.28}$$

and

$$X_1(0) = 40,\ X_2(0) = 40,\ X_3(0) = 1. \tag{6.29}$$

This system is at equilibrium and the mean value of $X_2(t)$ is 40. We are interested in $P(E_R) = P_{t\leq 10}(X_2 \rightarrow \theta|\mathbf{x}(0))$, the probability of $X_2(t) = \theta$ for $t \leq 10$. We chose $\theta = 65$ and 68 in our simulations. To apply our iwSSA and iwNRM to estimate $P(E_R)$, we divide the system into three groups. A G_1 group contains reaction R_1; a G_2 group includes reaction R_2; a G_3 group consists of reactions R_3 and R_4. More precisely, the G_3 group is further divided into a G_{31} group which contains reaction R_3 and a G_{32} group which contains reaction R_4. Since the system is at equilibrium and we have $a_0(\mathbf{x}_0) = 20$, $a_1(\mathbf{x}_0) = 4$, $a_2(\mathbf{x}_0) = 4$, $a_3(\mathbf{x}_0) = 8$ and $a_4(\mathbf{x}_0) = 4$, we get $\overline{K}_T = 200$, $\overline{K}_1 = 40$, $\overline{K}_2 = 40$, $\overline{K}_3 = 80$ and $\overline{K}_4 = 40$. Therefore, we get the following probabilities: $P_{G_1} = 0.2$, $P_{G_2} = 0.2$ and $P_{G_3} = 0.6$.

If $\theta = 65$, we have $\eta = 25$. Using (6.18), we obtained $\widetilde{K}_{G_2} = 29$, and then got $\widetilde{K}_{G_1} = 54$ and $\widetilde{K}_{G_3} = 117$. Substituting $\widetilde{K}_{G_1}$, $\widetilde{K}_{G_2}$ and $\widetilde{K}_{G_3}$ into (6.19), we got $Q_{G_1} = 0.27$, $Q_{G_2} = 0.145$ and $Q_{G_3} = 0.585$. We then chose $\alpha = 0.85$ and $\beta = 0.80$, and calculated $Q_{G_{31}}$ and $Q_{G_{32}}$ from (6.21) as $Q_{G_{31}} = 0.4173$ and $Q_{G_{32}} = 0.1678$. Similarly, if $\theta = 68$, we got $\widetilde{K}_{G_1} = 54$, $\widetilde{K}_{G_2} = 26$ and $\widetilde{K}_{G_3} = 120$, which resulted in $Q_{G_1} = 0.27$ and $Q_{G_2} = 0.13$. Again, selecting $\alpha = 0.85$ and $\beta = 0.80$, we got $Q_{G_{31}} = 0.430$ and $Q_{G_{32}} = 0.170$. To test if our iwNRM and iwSSA are sensitive to parameters α and β, we also used another set of parameters $\alpha = 0.80$ and $\beta = 0.75$.

Table 6.2: Estimated probability of the rare event $\hat{P}(E_R)$ and the sample variance σ^2 as well as the CPU TIME (in seconds) with 10^7 runs of iwNRM, iwSSA and rwSSA for the system given in (6.27).

(a) $\theta = 65$

	$\hat{P}(E_R)$	σ^2	TIME
iwNRM without G_3 fine-tuning	1.14×10^{-4}	2.77×10^{-7}	13381
iwSSA without G_3 fine-tuning	1.14×10^{-4}	2.74×10^{-7}	17484
iwNRM with $\alpha = 0.85$, $\beta = 0.80$	1.14×10^{-4}	1.27×10^{-7}	13504
iwSSA with $\alpha = 0.85$, $\beta = 0.80$	1.14×10^{-4}	1.28×10^{-7}	16649
iwNRM with $\alpha = 0.80$, $\beta = 0.75$	1.14×10^{-4}	1.29×10^{-7}	13540
iwSSA with $\alpha = 0.80$, $\beta = 0.75$	1.14×10^{-4}	1.29×10^{-7}	17243
rwSSA	1.14×10^{-3}	1.54×10^{-6}	24499

(b) $\theta = 68$

	$\hat{P}(E_R)$	σ^2	TIME
iwNRM without G_3 fine-tuning	1.49×10^{-5}	1.14×10^{-8}	14087
iwSSA without G_3 fine-tuning	1.49×10^{-5}	1.09×10^{-8}	17285
iwNRM with $\alpha = 0.85$, $\beta = 0.80$	1.49×10^{-5}	3.28×10^{-9}	13920
iwSSA with $\alpha = 0.85$, $\beta = 0.80$	1.49×10^{-5}	3.29×10^{-9}	17862
iwNRM with $\alpha = 0.80$, $\beta = 0.75$	1.49×10^{-5}	3.32×10^{-9}	14018
iwSSA with $\alpha = 0.80$, $\beta = 0.75$	1.49×10^{-5}	3.30×10^{-9}	17858
rwSSA	1.49×10^{-5}	7.93×10^{-8}	24739

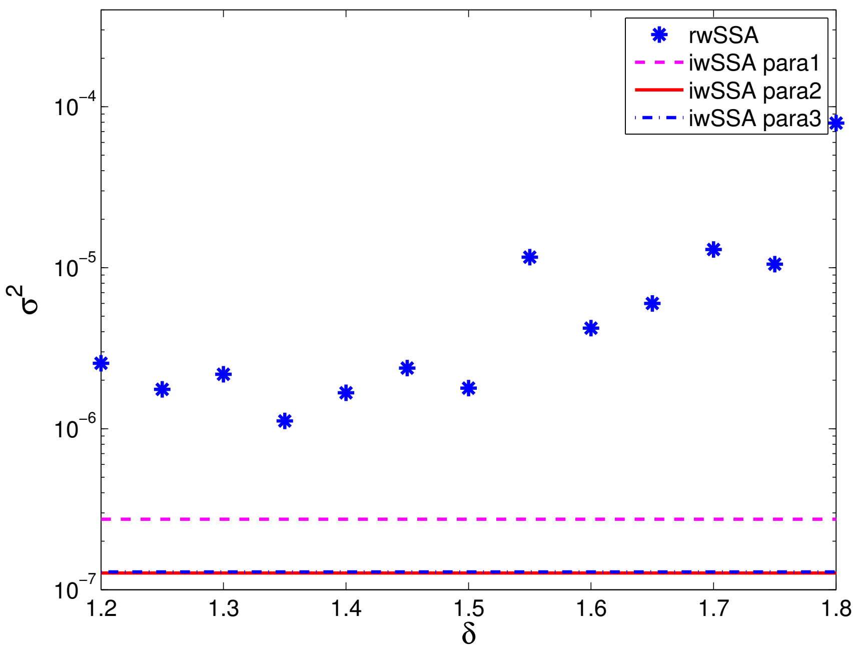
*The probability of the rare event estimated from 10^{11} runs of exact SSA method is 1.14×10^{-4} for $\theta = 65$ and 1.49×10^{-5} for $\theta = 68$.

In order to compare the performance of our iwNRM and iwSSA with that of the rwSSA, we also ran simulations with the rwSSA. In the rwSSA, we chose the following parameters $\gamma_1 = \delta$, $\gamma_2 = 1/\delta$ and $\gamma_m = 1$, $m = 3, 4$ to adjust propensity functions. Since the optimal value of α is unknown, we ran the rwSSA for $\delta = 1.2$, 1.25, 1.3, 1.35, 1.40, 1.45, 1.50, 1.55, 1.60, 1.65, 1.70, 1.75 and 1.80 to determine the best δ. Figure 6.2 shows the variance of $\hat{P}(E_R)$ obtained from the simulations with the rwSSA and our iwSSA. Since the iwNRM yielded almost the same variance as our iwSSA, we only plotted the variance obtained from the iwSSA. It is seen that our iwSSA provides variance more than one order of magnitude lower than that provided by rwSSA with the best δ. Is is also observed that our iwSSA is not very sensitive to the parameters α and β, since the variance obtained with two different sets of values for α and β is almost the same.

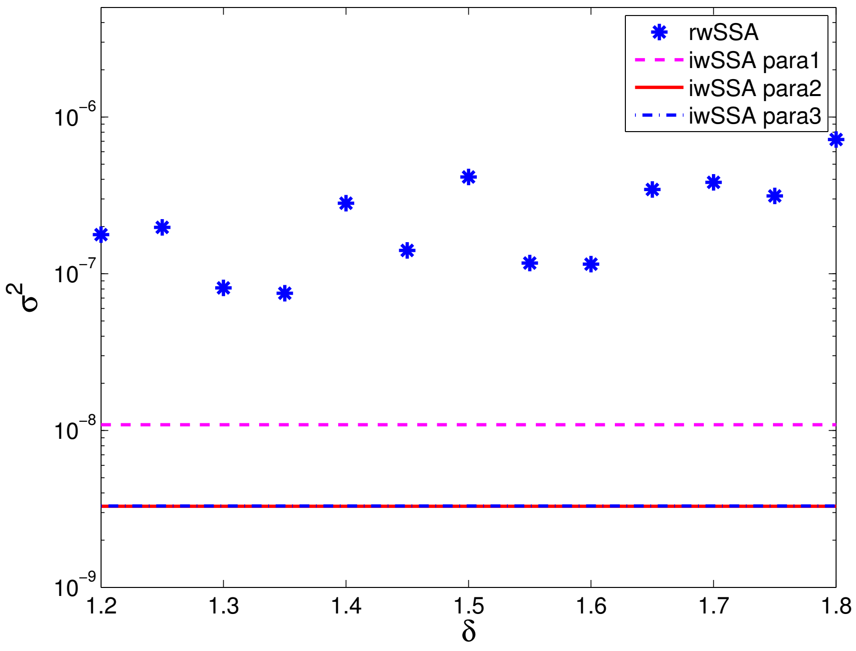
Table 6.2 lists $\hat{P}(E_R)$ and its variance obtained from $n = 10^7$ runs of the rwSSA, the iwNRM and the iwSSA. We first ran the iwNRM and the iwSSA without fine-tuning the probability of reactions in G_3 group and calculated q_m using (6.20). We then ran iwNRM and iwSSA with fine-tuning the probability of reactions in G_3 group and used two sets of parameters ($\alpha = 0.85$, $\beta = 0.80$; $\alpha = 0.80$, $\beta = 0.75$) and (6.21) to calculate q_m for the reactions in G_3 group. We also made 10^{11} runs of the exact SSA to estimate $\hat{P}(E_R)$. It is seen that the iwNRM, the iwSSA and the rwSSA all yield the same $\hat{P}(E_R)$ as the exact SSA. However, the iwNRM and iwSSA with fine-tuning the probabilities of G_3 reactions offer variance more than one order of magnitude lower than that provided by the rwSSA. Without fine-tuning the probabilities of G_3 reactions, the iwNRM and the iwSSA provided a little bit larger variance but still almost one order of magnitude lower than that provided by the rwSSA. Table 6.2 also shows that the iwNRM and the iwSSA needed only 60%-70% CPU time needed by the rwSSA. Again, the CPU time of the rwSSA in Table 6.2 does not include the time needed for searching for the optimal value of δ for each θ. If we include this time, the CPU time of the rwSSA will be almost doubled.

6.6 Conclusion

The wSSA and the rwSSA are innovative variation of Gillespie's standard SSA. They provide an efficient way of estimating the probability of a rare event that occurs in chemical reaction systems with an extremely low probability in a given time period. In this chapter we developed two methods for improving the performance of the wSSA and the rwSSA. In the first method, we applied the importance sampling technique used in the wSSA to the next reaction method of the SSA and developed the wNRM which is more efficient than the wSSA and the rwSSA. In the second method, we introduced a systematic method for selecting the values of importance sampling parameters, which is lack in the wSSA and the rwSSA. Incorporating this parameter selection method into the wNRM and the wSSA, we obtained an improved version of wNRM and wSSA: iwNRM and iwSSA. The numerical examples showed that comparing with the rwSSA, our iwNRM and iwSSA could substantially reduce the variance of the estimated probability of the rare event and speed up simulation for a given number of simulation runs.



(a) $\theta = 65$



(b) $\theta = 68$

Figure 6.2: Variance σ^2 obtained from 10^7 runs of the iwSSA and the rwSSA for the system in (6.27) with $c_1 = 0.1$, $c_2 = 0.1$, $c_3 = 8$, $c_4 = 0.1$, $X_1(0) = 40$, $X_2(0) = 40$ and $X_3(0) = 1$. iwSSA para 1 represents the iwSSA without fine-tuning the probability of reactions in G_3 group; iwSSA para 2 and 3 represent the iwSSA with fine-tuning the probability of reactions in G_3 group using two sets of parameters: $\alpha = 0.85$, $\beta = 0.8$ and $\alpha = 0.80$, $\beta = 0.75$. Since the variance of the iwSSA does not depend on δ used in the rwSSA, it appears as a horizontal line.

CHAPTER 7

Stochastic Simulation of Delay-Induced Circadian Rhythms in Drosophila

7.1 Motivation

Almost all living organisms, including animals, plants, fungi and cyanobacteria, exhibit daily periodic oscillations in their biochemical or physiological behavior, which are known as circadian rhythms [74–80]. The mechanism of circadian oscillation has been an extensive research topic in the last three decades. It has been found that circadian rhythms in fact are determined by oscillatory expression of certain genes [81, 82]. Specifically, circadian clocks consist of a network of interlocked transcriptional-translational feedback loops formed by a number of genes [75]. In *Drosophila*, transcription of *per* and *tim* genes is activated by a heterodimer consisting of two transcriptional activators dCLOCK and CYCLE [83–86]. The PER protein in turn binds to the dCLOCK-CYCLE heterodimer, which inhibits the DNA binding activity of the dimer, thereby repressing the transcription of *per* and *tim* [84–87]. While this forms a negative feedback loop, there is also a positive feedback loop, in which PER and TIM activate dCLOCK synthesis by binding dCLOCK and relieving dCLOCK's repression of dclock transcription [88, 89].

Several mathematical models have been proposed for circadian rhythms in *Drosophila* [85,87,90–95]. The models of Smolen *et al.* [85,87] introduce time delays in the expression of *dclock* and *per* genes, while other models do not have such delays. Numerical simulations using ordinary differential equations (ODE) show that all these models can produce circadian oscillations. In particular, times delays were found to be essential for simulation of circadian oscillations with the model of Smolen *et al.* [85,87].

Since there is significant stochasticity in gene expression arising from fluctuations in transcription and translation [19–21], it is desirable to simulate circadian oscillations in the presence of noise. Toward this end, several stochastic models were proposed [77, 96–99], and Gillespie's stochastic simulation algorithm (SSA) [32, 33] were employed to simulate circadian oscillations. All these stochastic models [77, 96–99] do not include time delays. In order to reflect the noise in gene expression, Smolen *et al.* used two approximate stochastic simulation methods to simulate circadian oscillation based on their models with delays [85, 87]. However, their models lumped transcription and translation into one single process and did not model the process that dCLOCK binds to or dissociates with *dclock* and *per* genes to activate or inhibit transcription. Since transcription is a major source of intrinsic noise [20, 21], the approximate stochastic simulation of Smolen *et al.* may underestimate the effect of noise. Li and Lang [100] used similar approximate stochastic simulation methods to simulate reduced model of Smolen *et al.* [87], but with an emphasis on the noise-sustained oscillation in the region of parameter values where the deterministic model predicted no oscillation. Currently, no exact stochastic simulation has been done for circadian rhythm models with random delays, partially due to the fact that Gillespie's SSA cannot handle delays in certain reactions.

Recently, we developed an exact SSA algorithm for systems of chemical reactions with delays [101]. The goal of this chapter is to apply this exact SSA to simulate circadian oscillations in *Drosophila* using a model with time delays and to investigate the effects of noise and random time delays on circadian oscillations. We first develop two stochastic models with random delays for circadian oscillations in *Drosophila* based on the two deterministic models of Smolen *et al.* [85, 87]. Using our exact SSA, we then simulate free-running circadian oscillation under constant darkness. Our simulations demonstrate that both models can produce sustained oscillations. The variability in oscillation period is very small although the variability in oscillation peaks is considerably large. In particular, although time delays are essential to oscillation, random fluctuations in time delays do not cause significant changes in oscillation period as long as the average delays are fixed. Our simulations also showed that circadian oscillations of both models are robust to parameter variations. The entrainment by light was also simulated for both models, yielding results consistent with experimental observations. To see the effect of transcription noise, we also run simulations with different values for the rate that dCLOCK binds or unbinds to *per* and *dclock* genes.

7.2 Methods

7.2.1 The detailed model of circadian oscillation with time delays

Model Description

We develop a stochastic model for the *Drosophila* circadian oscillator based on the deterministic model of Smolen *et al.* [85], which is depicted in Figure 7.1. In Smolen's model, transcription of *dclock* gene is repressed by dCLOCK protein after a time delay

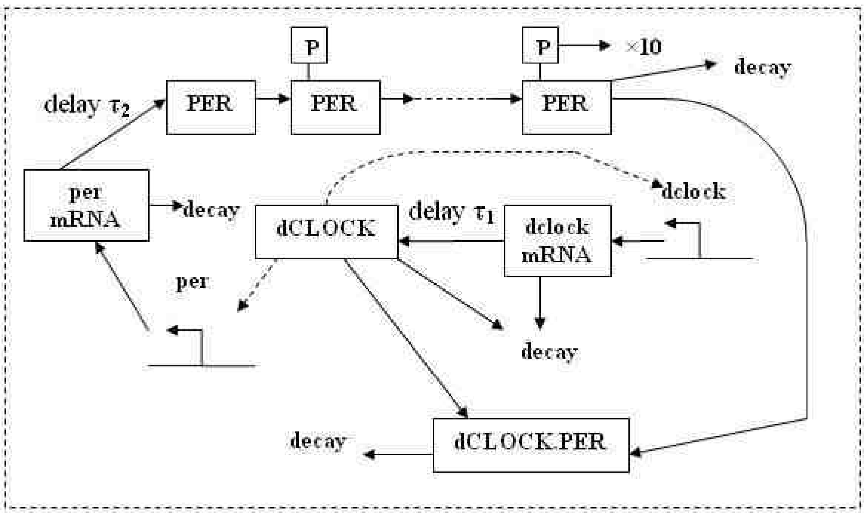


Figure 7.1: Schematic of the detailed model for circadian oscillators in *Drosophila*.

of τ_1 [84,86]. dCLOCK activates the synthesis of PER protein with a time delay of τ_2. PER is then phosphorylated [102], and unphosphorylated and phosphorylated PER can bind to dCLOCK thereby relieving dCLOCK's repression of *dclock* transcription. It was reported that PER undergos multiple and sequential phosphorylation [34], but exact times of phosphorylation are unknown. Following Smolen *et al* [85], we assumed that PER can be phosphorylated up to 10 times. Although the TIM gene also plays an important role in circadian rhythm, Smolen *et al.* [85] used a single "lumped" variable, PER, to represent both PER and TIM, since the time courses of PER and TIM proteins are similar in shape and largely overlap. Smolen *et al.* [85] characterized the circadian oscillator in *Drosophila* using 23 ordinary differential equations (ODE). We first convert these 23 ODEs into 46 chemical reactions. Smolen *et al.* [85] lumped transcription and translation of *dclock* and *per* into one single step. They did not model the process that dCLOCK binds to and dissociates with *dclock* gene and *per* gene. Since this binding and unbinding processes, transcription and translation are major sources of intrinsic noise [8–10, 14], we model these processes explicitly. Our stochastic model, containing 29 molecular species in Table 7.1, is featured with 54 reactions in Table 7.2 which include 44 reactions converted from Smolen's ODE and 10 new reactions.

Table 7.1: Detailed stochastic model for the *Drosophila* circadian oscillator.

Rid	Reaction	Rate (h^{-1})
1	dclockg →dclockg + dclockm	$c_1 = 17$
2	dclockm →∅	$c_2 = 0.64$
3	dclockm →dclockm + dCLOCK (delay τ_1)	$c_3 = 16$
4	dCLOCK →∅	$c_4 = V_{dc}/(K_{dc} + [CLK_{tot}])^*$
5	dclockg+dCLOCK→dclockg.dCLOCK	$c_5 = 0.144$
6	dclockg.dCLOCK→dclockg + dCLOCK	$c_6 = 72$
7	dCLOCK+perg→perg.dCLOCK	$c_7 = 0.144$
8	perg.dCLOCK→dCLOCK+perg	$c_8 = 72$
9	perg.dCLOCK→perg.dCLOCK+perm	$c_9 = 20$
10	perm →∅	$c_{10} = 0.35$
11	perm →perm + PER_0 (delay τ_2)	$c_{11} = 30.625$
12∼21	PER_i→PER_{i+1},$i = 0, \cdots, 9$	$c_{12} \sim c_{21} = V_{ph}/(K_{ph} + [TOT_{unph}])^*$
22	PER_{10}→∅	$c_{22} = V_{dp}/(K_{dp} + [PER_{10}])^*$
23∼33	dCLOCK+PER_i→dCLOCK.PER_i	$c_{23}{\sim}c_{33} = 0.06$
34∼43	dCLOCK.PER_i→dCLOCK.PER_{i+1}, $i = 0, \cdots, 9$	$c_{34} \sim c_{43} = c_{12}$
44∼53	dCLOCK.PER_i→∅, $i = 0, \cdots, 9$	$c_{44} \sim c_{53} = c_4$
54	dCLOCK.PER_{10}→∅	$c_{54} = c_4$ $+V_{dp}/(K_{dp} + [dCLOCK.PER_{10}])^*$

* See text for detailed descriptions.

Reaction 1 ∼ 4 represent transcription of *dclock* gene, degradation of *dclock* mRNA, translation of *dclock* mRNA and degradation of dCLOCK protein, respectively. Reaction 5 models the process that dCLOCK protein binds to *dclock* gene and reaction 6 represents dissociation of dCLOCK with *dclock*. Reaction 7 and 8 specify the event that dCLOCK binds to and dissociates with *per* gene. Reaction 9, 10 and 11 represent transcription of *per* gene after it is activated by dCLOCK, degradation of *per* mRNA and translation of *per* mRNA, respectively. Reaction 12 ∼ 21 represent the phosphorylation of PER and reaction 22 represents the degradation of PER. Reaction 23 ∼ 33 represent the association of dCLOCK with PER at different levels

Table 7.2: Molecular species

Species	Description
dclockg	*dclock* gene
dclockm	*dclock* mRNA
dCLOCK	dCLOCK protein
dclock.dCLOCK	*dclock* gene bounded by dCLOCK protein
perg	*per* gene
per.dCLOCK	*per* gene bounded by dCLOCK protein
perm	*per* mRNA
PER$_i$,$i = 0, \cdots, 10$	PER protein with i phosphorylations
dCLOCK.PER$_i$,$i = 0, \cdots, 10$	Complex of dCLOCK and PER with i phosphorylations

of phosphorylation. Reaction 34 $\sim$ 43 describe the phosphorylation of dCLOCK and PER$_i$ ($i = 0, \cdots, 9$) heterodimer. Reaction 44 $\sim$ 54 represent the degradation of dCLOCK and PER$_i$($i = 0, \cdots, 9$) heterodimer.

Parameter Estimation

Each reaction is associated with a reaction probability rate constant, c, which determines the probability that a specific reaction occurs in an infinitesimal time interval. The probability rate constant c of a specific reaction can be calculated from conventional rate constant k as follows: $c = k$ for a monomolecular reaction, $c = k/\Omega$ for a bimolecular reaction with two different reactants and $c = 2k/\Omega$ for a bimolecular reaction with one reactant [46], where $\Omega = AV$, and $A = 6.022 \times 10^{23}$ is the Avogadro constant and V is the system volume. We assume that a lateral neuron in *Drosophila* is a sphere of a radius around $6\mu m$ [87, 103], which results in a volume $V = 8.3 \times 10^{-13}$ L. As many other existing models [85, 87, 104], we do not separate nuclear and cytoplasmic compartments. We retain most parameter values from Smolen's *et al.* [85] including c_4 and $c_{12}, \cdots, c_{54}$. The remaining 10 parameters, c_1, c_2, c_3 and $c_5, \cdots, c_{11}$, are determined in our simulation. In the following, we describe 54 reactions and how each probability rate constant was determined.

We assume that there are two copies of *dclock* genes and thus the initial value for the number of molecules of *dclockg* in Table 1 is 2. Since no experimental reports are available for transcription rate c_1, we choose $c_1 = 17$ h^{-1}, which is close to the value used in a previous computational model [104]. The degradation rate of *dclock* mRNA c_2 is calculated as $\ln(2)/T_{1/2}$, where $T_{1/2}$ is the half-life of *dclock* mRNA. Hardin *et al.* [105] shows that mRNA of periodic genes in *Drosophila* has short half-life, varying from order of minutes to tens of minutes. Lin *et al.* [106] shows that *Drosophila* mRNAs vary considerably in half-life from tens of minutes to more than 10 hours. Here we assume that the average half-life of *dclock* mRNA is 65 minutes, which is 1.08 h and thus c_2 is 0.64 h^{-1}. The synthesis rate of dCLOCK protein was chosen to be $v_{sc} = 1.7$ nM h^{-1} in the model of Smolen *et al.* [85], which equivalently is $1.7 \times 10^{-9} \times \Omega = 850$ molecules per hour. In our model, the average dCLOCK synthesis rate is $c_1c_3/c_2 \times 2$ molecules per hour since we assume the number of molecules of *dclockg* in Table 1 is 2. Letting $c_1c_3/c_2 \times 2 = 850$, we get $c_3 = 16$ h^{-1}. Since transcription rate has a significant impact on the noise [20, 21], we tested the sensitivity of simulation results to c_1. Increasing or decreasing c_1 two times while fixing the ratio of c_1c_3/c_2 only causes negligible change in the mean and standard error (SE) of period and peaks (data not shown). The rate c_4 is calculated as $c_4 = V_{dc}/(K_{dc} + [CLK_{tot}])$ [85], where $V_{dc} = 7.0$ nM h^{-1}, $K_{dc} = 10$ nM and $[CLK_{tot}]$ is the concentration of total dCLOCK given by

$$[CLK_{tot}] \triangleq \sum_{i=0}^{10} [dCLOCK.PER_i] + [dCLOCK] + [dclkg.dCLOCK] + [per.dCLOCK]. \tag{7.1}$$

Here $[\cdot]$ represents the concentration of the species in the bracket. A time delay τ_1

is included in reaction 3 accounting for time needed for transcription, translation and other potential mechanism for activating the transcription of *dclock*. Smolen *et al.* chose τ_1 to be a deterministic number equal to 5 h [85]. Taking into account uncertainty in this delay, we choose τ_1 as a random variable uniformly distributed in the interval [4h, 6h].

In reaction 5, dCLOCK binds to the E-box of *dclock* [84, 86], but there is no experimental report on the values of c_5 and the dissociation rate c_6. However, the dissociation rate of myogenin protein with the E-box of E12 gene was reported to be 0.0205 s^{-1} [107]. Therefore, we choose c_6 = 0.02 s^{-1}=72 h^{-1}. The equilibrium constant k_6/k_5 of reaction 5 and 6 is equal to the Michaelis constant K_2 in Ref. 12 that describes the regulation of dCLOCK synthesis by dCLOCK and was chosen to be 1 nM [85]. Using this equilibrium constant, we calculate c_5 to be 0.144 h^{-1}. Reaction 7 and 8 specify the event that dCLOCK binds to and dissociates with *per* gene. The equilibrium constant k_8/k_7 of reaction 7 and 8 is equal to the Michaelis constant K_1 in Ref. 12, which was 1 nM. This Michaelis constant reflects the regulation of PER synthesis by dCLOCK. After choosing $c_8 = c_6$, c_7 is calculated from the equilibrium constant k_8/k_7 as $c_7 = k_7/k_8/\Omega \times c_8 = 0.144$ h $^{-1}$.

The transcription rate of *per* gene c_9 is chosen to be 20 h^{-1}, and the degradation rate of *per* mRNA c_{10} is calculated as $c_{10} = 0.35$ h^{-1} from the half-life of *per* mRNA which was estimated to be 2 h [105,106]. Also, we assume that there are two copies of *per* gene, and thus, the initial value for the number of molecules of *perg* in Table 1 is 2. The synthesis rate of PER protein was chosen to be $v_{sp} = 7$ nM h^{-1} in the model of Smolen *et al.* [85], which equivalently is $7 \times 10^{-9} \times \Omega = 3500$ molecules per hour. In our model, the average PER synthesis rate is $c_9 c_{11}/c_{10} \times 2$ molecules per hour.

Letting $c_9c_{11}/c_{10} \times 2 = 3500$, we got $c_{11} = 30.625$ h^{-1}. Similar to c_1, we also tested the sensitivity of simulation results to c_9. Increasing or decreasing c_9 two times while fixing the ratio of c_9c_{11}/c_{10} only causes negligible change in the mean and standard error (SE) of period and peaks (data not shown).

A delay τ_2 is introduced in reaction 11. This time delay accounts for the time needed for the transcription and translation of *per* gene. Smolen *et al.* [85] chose τ_2 to be 8 hours. However, the total time needed for transcription and translation maybe less than 8 hours [108] and also there may be some fluctuations in τ_2. Therefore, we chosen τ_2 as a random variable with mean 6 h, uniformly distributed in [4.8h, 7.2h].

Reaction 12 ∼ 21 represent the phosphorylation of PER, whose probability rate constants are all equal to $V_{ph}/(K_{ph} + [TOT_{unph}])$ [85], where $V_{ph} = 32$ nM h^{-1}, $K_{ph} =$ 8 nM and $[TOT_{unph}]$ is the concentration of all forms of PER with less than 10 phosphorylations given by

$$[TOT_{unph}] \triangleq \sum_{i=0}^{9}([PER_i] + [dCLOCK.PER_i]). \tag{7.2}$$

Reaction 22 represents the degradation of PER and c_{22} is equal to $V_{dp}/(K_{dp} + [PER_{10}])$, where $V_{dp} = 22$ nM h^{-1} and $K_{dp} = 3$ nM [85].

Reaction 23 ∼ 33 represent the association of dCLOCK with PER at different levels of phosphorylation. The deterministic rate for all these reactions are 30 nM^{-1}h^{-1} and thus the probability rate constants are $c_i = 0.06$ h$^{-1}, i = 23, \cdots, 33$. Reaction 34 ∼ 43 describe the phosphorylation of dCLOCK and PER$_i$ ($i = 0, \cdots, 9$) heterodimer and we have $c_i = c_{12},\ i = 34, \cdots, 43$. Reaction 44 ∼ 54 represent the degradation of dCLOCK and PER$_i$($i = 0, \cdots, 9$) heterodimer. We have $c_i = c_4,\ i = 44, \cdots, 53$ and $c_{54} = c_4 + V_{dp}/(K_{dp} + [dCLOCK.PER_{10}])$, where V_{dp} and K_{dp} are given earlier.

7.2.2 The reduced model of circadian oscillation with time delays

Model Description

Smolen *et al.* [87] also simplified their detailed model described earlier by removing the phosphorylation of PER. This reduced model was characterized by 2 ODEs. We first convert these 2 ODEs into 4 chemical reactions. Again, we explicitly model the binding and unbinding of dCLOCK to *dclock* and *per* genes, as well as the transcription and translation of *dclock* and *per* genes. Our reduced stochastic model consists of 9 molecular species and 14 reactions specified in Table 7.3. Comparing with reactions in Table 7.1 and 7.3, we see that reduced model is obtained by removing reactions related to phosphorylation of PER and phosphorylated PER. Similarly to the detailed model, we retained most parameter values from Smolen *et al* [87], including c_4 and c_{12}. The parameters not presented in Smolen's reduced model are determined and explained in the following subsection.

Table 7.3: Reduced stochastic model for the *Drosophila* circadian oscillator.

Rid	Reaction	Rate (h^{-1})
1	dclockg →dclockg + dclockm	$c_1 = 10$
2	dclockm →∅	$c_2 = 0.64$
3	dclockm →dclockm + dCLOCK (delay τ_1)	$c_3 = 4$
4	dCLOCK →∅	$c_4 = 0.5$
5	dclockg+dCLOCK→dclockg.dCLOCK	$c_5 = 1.44$
6	dclockg.dCLOCK→dclockg + dCLOCK	$c_6 = 72$
7	dCLOCK+perg→perg.dCLOCK	$c_7 = 0.48$
8	perg.dCLOCK→dCLOCK+perg	$c_8 = 72$
9	perg.dCLOCK→perg.dCLOCK+perm	$c_9 = 10$
10	perm →∅	$c_{10} = 0.35$
11	perm →perm + PER (delay τ_2)	$c_{11} = 4.375$
12	PER→∅	$c_{12} = 0.5$
13	dCLOCK+ PER →dCLOCK.PER	$c_{13} = 0.06$
14	dCLOCK.PER→∅	$c_{14} = 1$

Parameter Estimation

The rate c_1 is chosen to be 10 h^{-1} which is slightly lower than that in the detailed model. This is because the synthesis rate of dCLOCK protein in the reduced model of Smolen *et al.* [87] was $v_{sc} = 0.25$ nM h^{-1}, which is smaller than v_{sc} in the detailed model. The rate $c_2 = 0.64$ h^{-1} is the same as that in the detailed model. Letting c_1c_3/c_2 equal to v_{sc}, we calculate $c_3 = 4$ h^{-1}. The rate c_4 is the same as that in the reduced model of Smolen *et al.* [87], equal to 0.5 h^{-1}.

The unbinding rate of dCLOCK to *dclock* gene, c_6, is chosen identical to that in the detailed model. The equilibrium constant k_6/k_5, which is equal to the Michaelis constant K_2 in Ref. 14 that describes the regulation of dCLOCK synthesis by dCLOCK, was chosen to be 0.1 nM [87]. Using this equilibrium constant, we calculate c_5 to be 1.44 h^{-1}. Similarly, c_8 is the same as that in the detailed model. The equilibrium constant k_8/k_7, which is equal to the Michaelis constant K_1 in Ref. 14 that describes the regulation of PER synthesis by the transcriptional activators dCLOCK, is chosen to be 0.3 nM [87]. Then we calculate c_7 to be 0.48 h^{-1}.

The transcription rate of *per* gene c_9 is chosen to be 10 h^{-1}, which is lower than that in the detailed model, because the synthesis rate of PER protein in the reduced model of Smolen *et al.* [87] was $v_{sp} = 0.5$ nM h^{-1} which is smaller than that in the detailed model. The degradation rate of *per* mRNA is again $c_{10} = 0.35$ h^{-1}. Letting c_9c_{11}/c_{10} equal to v_{sp}, we calculate c_{11} as $c_{11} = 4.375$ h^{-1}. The degradation rate of PER c_{12} is the same as that in the reduced model of Smolen *et al.* [87], equal to 0.5 h^{-1}. The degradation rate of dCLOCK and PER complex is $c_{13} = 0.06$ h^{-1}, identical to that in the detailed model and we have $c_{14} = c_4 + c_{12}$.

Time delays τ_1 and τ_2 are chosen as follows. As the effective delay contributed by PER phosphorylation is incorporated into τ_1 and τ_2, τ_1 and τ_2 should be longer than those in the detailed model. Therefore we chose τ_1 and τ_2 uniformly distributed in the time interval [5h 9h] and [7h 11h], respectively.

7.2.3 Stochastic simulation

Gillespie's SSA [33] is often employed to simulate the stochastic dynamics of genetic networks [21, 35]. However, Gillespie's SSA cannot deal with delays in certain reactions. Recently, we developed an exact SSA for systems of chemical reactions with delays [101], which can handle both deterministic and random delays. We use this exact SSA to simulate the dynamics of the systems described in Table 7.1 and 7.3.

7.2.4 Data analysis

Customized Matlab Software (Mathworks Inc.) was written to analyze data generated from stochastic simulations, e.g., to calculate the mean and SE of protein levels, to identify the peaks of dCLOCK and PER during oscillation, and to calculate the peak amplitudes. Oscillation periods were calculated using the short-time Fourier transform (STFT) method [109]. Specifically, Fourier transform was applied to protein levels of dCLOCK and PER within a time window of 70 hours, after the mean level was subtracted. The largest peak at a non-zero frequency was identified as the oscillation frequency within the time window and the period of the oscillation is the inverse of the oscillation frequency. Note that the maximum period that can be identified by the STFT is 35 hours since a time window of 70 hours was used.

7.3 Results

7.3.1 Simulation of oscillation in the presence of noise

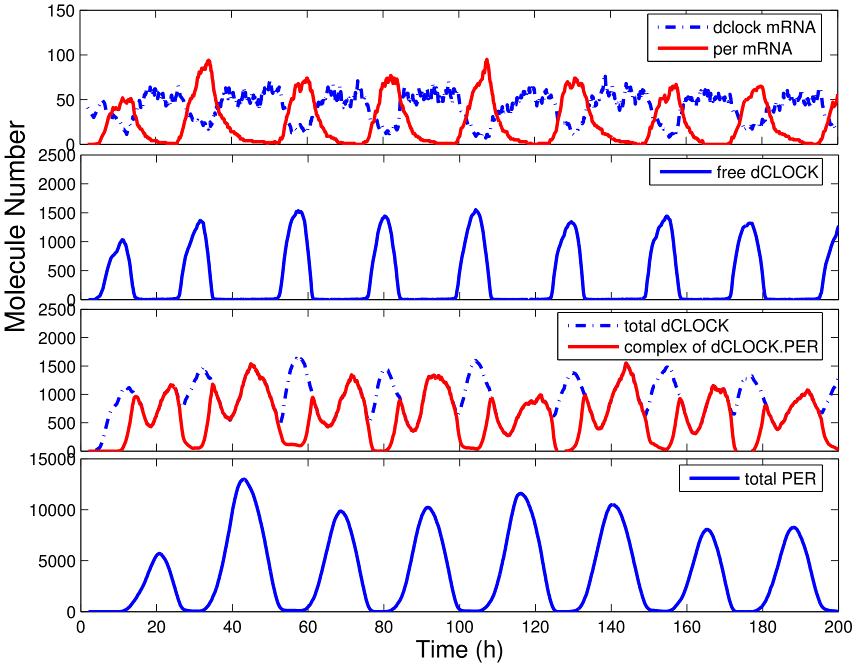


Figure 7.2: One trajectory of *dclock* and *per* mRNA, free dCLOCK, total dCLOCK, dCLOCK.PER complex and total PER for the detailed stochastic model in constant darkness.

We first ran simulations using the detailed model. Figure 7.2 depicts one trajectory of the number of molecules of *dclock* and *per* mRNA, free dCLOCK protein, the total number of molecules of dCLOCK that includes dCLOCK, dclock.dCLOCK, per.dCLOCK and dCLOCK.PER$_i$, $i = 0, \cdots, 10$, in Table 7.2, the total number of molecules of dCLOCK.PER which includes dCLOCK.PER$_i$, $i = 0, \cdots, 10$, in Table 7.2, and the total number of PER protein that includes PER$_i$ and dCLOCK.PER$_i$, $i = 0, \cdots, 10$, in Table 7.2. We here simulated free-running rhythms in constant darkness. Figure 7.2 clearly shows oscillations of the levels of mRNA and protein despite some random fluctuations. It is seen that *per* and *dclock* oscillations are almost in antiphase

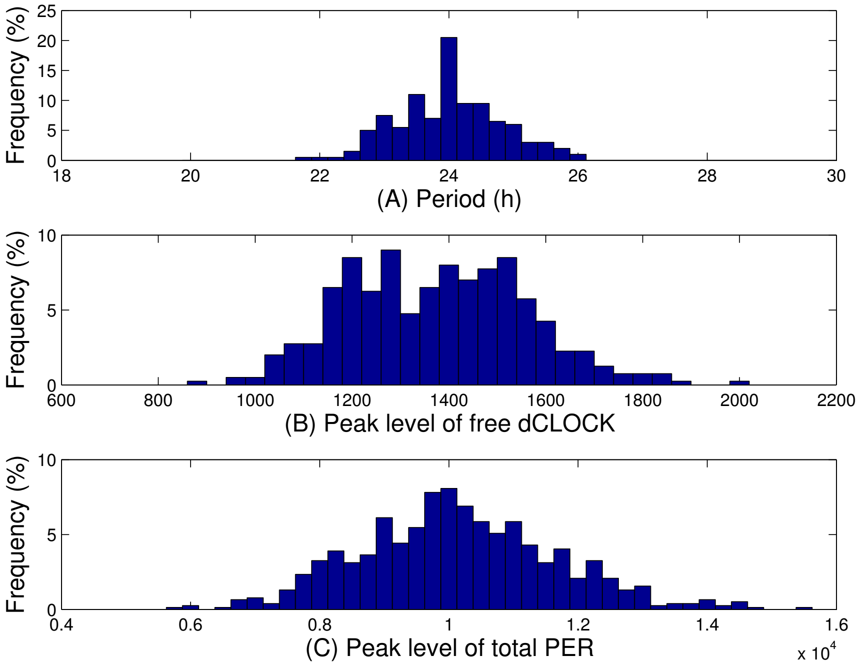


Figure 7.3: The histogram of periods and peaks of free dCLOCK and total PER for the detailed stochastic model in constant darkness.

with each other, which is consistent with the experiment observations [108, 110, 111]. It appears that there are more fluctuations in mRNA levels than the corresponding protein levels. This is due to the fact that the number of mRNA molecules is much lower than those of proteins. Even though the shape of free dCLOCK and total PER looks smooth, the peaks of free dCLOCK and total PER vary significantly, due to the transcription and translation noise. In the third panel of Figure 2, dCLOCK.PER complex shows two peaks in one circadian cycle, because peaks of dCLOCK.PER are determined by peaks of both free dCLOCK and PER. Whether such dynamics reflect the level of dCLOCK.PER in real systems is still unknown experimentally [85].

We also simulated 100 runs to get the statistics of oscillation. Figure 7.3A depicts the histogram of oscillation periods. It is seen that most periods are within the range between 23 and 25 hours. Figure 7.3B and C show the histogram of the number of molecules of free dCLOCK and total PER at oscillation peaks, respectively. As listed

Table 7.4: Statistics of oscillations for the detailed stochastic model

	Mean	SE	CV
Period (h)	23.93	0.78	3.26%
Peak value of Total PER	10149	1530.1	15.08%
Peak value of free dCLOCK	1377.1	184.62	13.41%
Peak value of total dCLOCK	1437.1	156.24	10.87%
Peak-to-through Amplitude of total dCLOCK	1016.5	191.51	18.84%

in Table 7.4, the mean of the period is 23.93 hours, which is very close to 24 hours, and the SE of the period is 0.78 hours. The coefficient of variation (CV, SE divided by mean) is therefore 3.26%, which is very low. Since CV is a normalized measure of dispersion of a probability distribution, a small CV for period implies that the periods lie in a small interval around its mean value with a large probability. Table 7.4 also contains the mean, SE and CV of the peak levels of free dCLOCK, total PER and total dCLOCK, as well as the peak-to-through amplitude of total dCLOCK. Since the through amplitude of free dCLOCK and PER is zero, their peak-to-through amplitude is equal to their peak levels. It is seen that the CVs of the peak levels of free dCLOCK, total PER and total dCLOCK are 13.41%, 15.08%, 10.87%, respectively, and that the CV of the peak-to-through amplitude of total dCLOCK is 18.84%. Taken together, we see that the oscillation period is very robust in the presence of intrinsic noise, although there are significant fluctuations in oscillation peaks.

We now discuss simulation results from the reduced model. Figure 7.4 shows one trajectory of *dclock* and *per* mRNA, free dCLOCK, total dCLOCK, dCLOCK.PER and total PER. Again, consistent with the experiment observations, *per* and *dclock* oscillate in antiphase. Compared with the trajectories produced by the detailed model, the trajectories here appear to have more random fluctuations, which is due to the fact that the number of molecules of each species in the reduced model is much

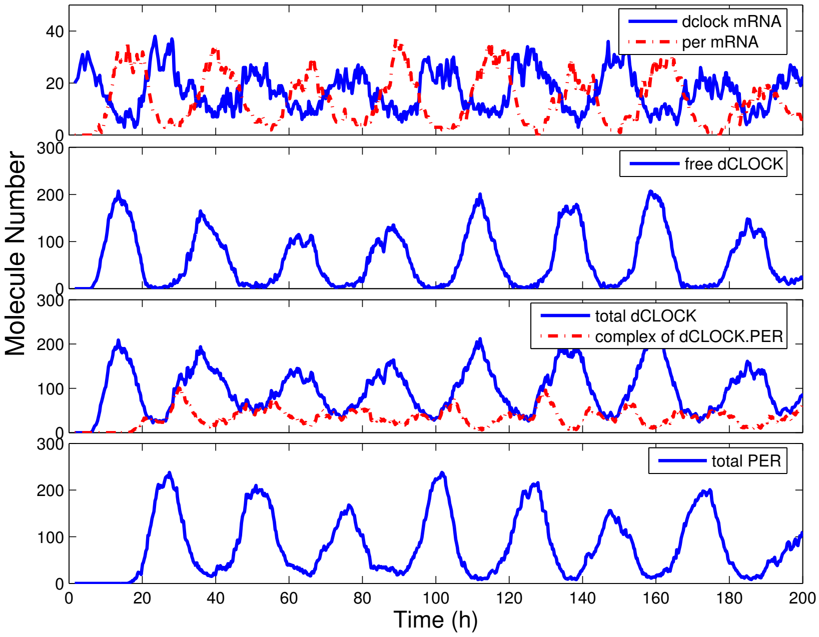


Figure 7.4: One trajectory of *dclock* and *per* mRNA, free dCLOCK, total dCLOCK, dCLOCK.PER complex and total PER for the reduced stochastic model in constant darkness.

smaller than those of the detailed model. The histograms of periods and peaks of free dCLOCK and total PER peak obtained from 100 simulation runs are depicted in Figure 7.5. Table 7.5 lists the mean, SE and CV of the period, peaks of total PER, free dCLOCK, and total dCLOCK, as well as the peak-to-through amplitude of total dCLOCK. It is seen that the CV of period is almost the same as that of the detailed model, but the CVs of peaks and peak-to-through amplitude are slightly larger than those of the detailed model. Therefore, both the detailed and reduced models can produce robust oscillation period in the presence of intrinsic noise despite significant fluctuations in oscillation peaks. Note that levels of dCLOCK and PER are very different in two models. Therefore, our simulation results for two models demonstrate that oscillation is robust across a wide range of molecular levels or under quite different levels of intrinsic noise.

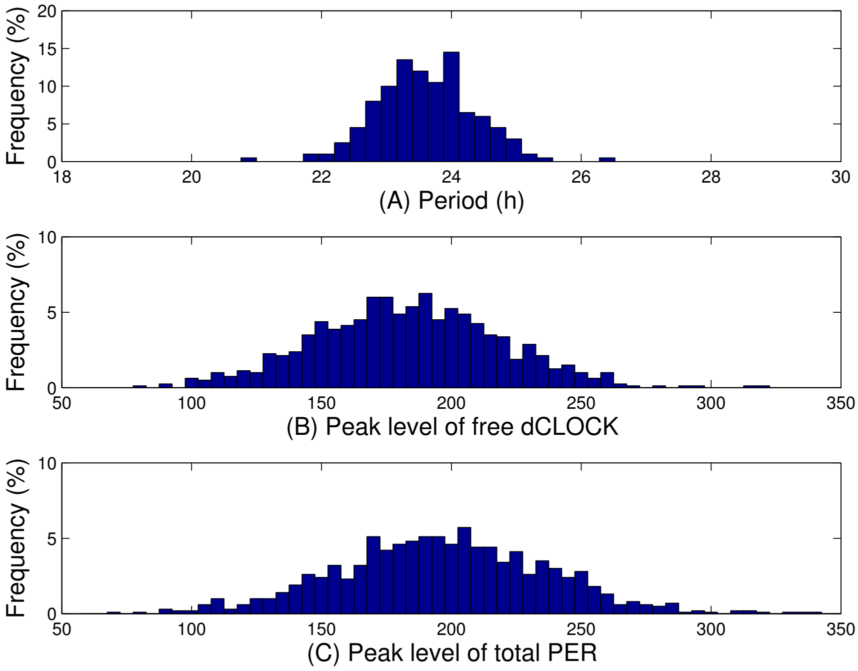


Figure 7.5: The histogram of periods and peaks of free dCLOCK and total PER for the reduced stochastic model under constant darkness.

We investigate the effect of the random time delays with fixed average time delays. Since both detailed and reduced models produced similar results, we here only present results for detailed model. Note that time delays τ_1 and τ_2 in our simulations are random variables uniformly distributed in $[\tau_{min}, \tau_{max}]$, where the standard value of $\tau_{min} = 0.8\tau_{mean}$ and standard value of $\tau_{max} = 1.2\tau_{mean}$, with τ_{mean} denoting the average time delay. To test the sensitivity of the range of random delays, we run

Table 7.5: Statistics of oscillations for the reduced stochastic model

	Mean	SE	CV
Period (h)	23.60	0.80	3.39%
Peak value of Total PER	196.47	40.49	20.61%
Peak value of free dCLOCK	183.09	35.49	19.38%
Peak value of total dCLOCK	201.51	31.05	15.41%
Peak-to-through Amplitude of total dCLOCK	172.47	36.10	20.93%

more simulations using different τ_{min} and τ_{max} but with a fixed τ_{mean}. Specifically, when we fix τ_{mean} to be 5 h and 6 h for τ_1 and τ_2, respectively, if τ_1 and τ_2 are uniformly distributed in $[0.7\tau_{mean}, 1.3\tau_{mean}]$, the mean period is 23.89 h and the SE is 0.76 h; if τ_1 and τ_2 are uniformly distributed in $[0.6\tau_{mean}, 1.4\tau_{mean}]$, the mean period is 23.81 h and the SE is 0.79 h. In both cases, the mean period and SE are very close to the results from standard value of τ_1 and τ_2. Therefore, our simulations show that the random changes in the delays do not cause significant variations in the oscillation period as long as the average delays are fixed.

Smolen *et al.* [85, 87] also investigated the effects of noise using stochastic simulation. There are three major differences between our stochastic simulation and that of Smolen *et al.*: 1) we employed exact SSA, whereas they used approximate SSAs, 2) two delays critical to circadian oscillation are random in our simulation but deterministic in the simulation of Smolen *et al.*, and 3) we explicitly simulated the transcription process and the binding/unbinding events between dCOLCK and per and dclock promoters, whereas Smolen *et al.* lumped transcription and translation of dclock and per into a one-step process.

To convert concentration into number of molecules, we used the volume of typical lateral neuron cells, whereas Smolen *et al.* determined a scale factor by trial. For the detailed model, this resulted in different scale factors and protein levels in our simulation as shown in Figure 2 are approximate 10 times of those in the simulation of Smolen *et al.* as depicted in Figure 3 of Ref. 12. To make fair comparison, we ran simulations using the same scale factor as Smolen *et al.* [85]. Our simulation results showed that the mean peak values of PER, free dCLOCK and total dCLOCK are 1205, 176 and 183, respectively, which are comparable to the results of Smolen *et al.* [85]. The mean period in our simulation is 24 h and the CV of periods is 3.33%.

These results are also comparable to the results of Smolen *et al.*: a mean period of 23.5 h and a CV of 5%. The CVs of the peaks of PER, free dCLOCK and dCLOCK in our simulation are 15.47%, 15.20% and 12.26%, respectively, which are greater than the CV of PER (9%) in the simulation of Smolen *et al.* [85]. For the reduced model, it turns out that protein levels in our simulation are similar to those in the simulation of Smolen *et al.* [87]. The CV of periods in our simulation (3.39%) is slightly smaller than that obtained in simulation of Smolen *et al.* (4.78%). Since no result about the CV of peak protein levels was reported by Smolen *et al.*, we cannot compare the CV of peak protein levels.

In summary, although the noise in our models may be stronger than that in the models of Smolen *et al.* due to the random delays, transcription process, and random activation and repression of the promoters of per and dclock, the CV of periods in our simulation is slightly smaller than that in the simulation of Smolen *et al.* [87]. This result indicates that approximate simulation may have yielded non-negligible errors. It is difficult to evaluate the effect of such possible errors in the approximate method of Smolen *et al.* [87], but our simulation method is exact and can correctly capture the stochastic dynamics of the circadian rhythm. It seems that strong noise in our model is reflected in the peak protein levels because the CVs of peak protein levels in our detailed model are larger than those in the detailed model of Smolen *et al.* [85].

7.3.2 Robustness test in the presence of noise

In living cells, biochemical parameters often vary significantly from cell to cell due to stochastic effects, even if the cells are genetically identical [112]. But circadian oscillations with close period are still withstood in *Drosophila* or mammals. Therefore, a model of circadian rhythm should be robust in the sense that small parameter variations should not lead to large period variations. For the deterministic models, Smolen *et al.* [85, 87] have shown that circadian rhythm is robust when a parameter changes its value by 15% ∼ 20%. Here, we test if circadian rhythm is robust with respect to parameter changes in the presence of intrinsic noise. To test robustness, each parameter is decreased or increased by 20% from the standard value, with all other parameters fixed at the standard values, and then the mean and SE of oscillation periods and peaks are determined from simulation results. Since τ_1 and τ_2 are random variables, we decrease or increase their mean values by 20%.

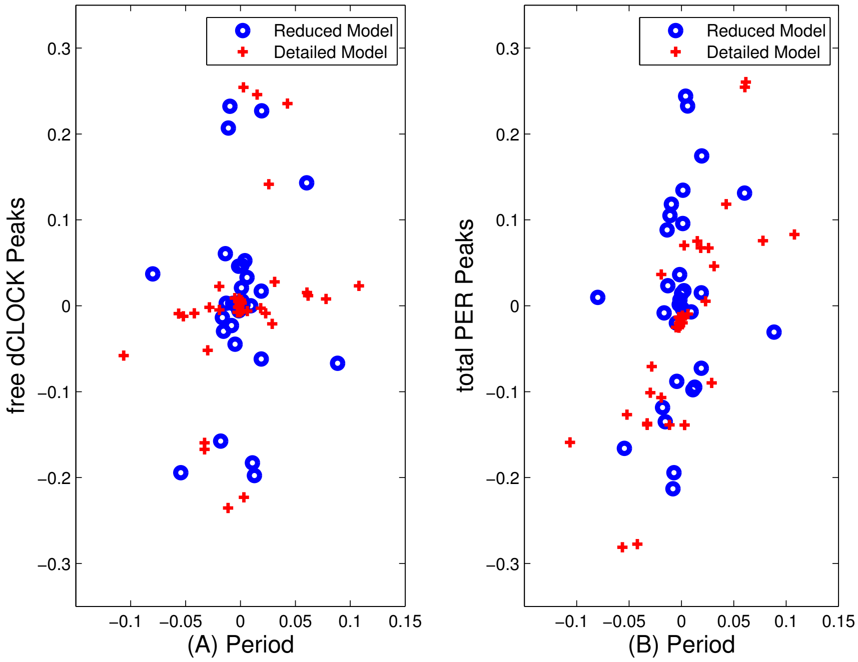


Figure 7.6: Relative change of the mean values of periods and peaks of free dCLOCK (left) and total PER (right) after the value of one parameter increases or decreases by 20% of the standard value while other parameters are fixed. The relative change of the period is defined as (T_1-T_0)/T_0, where T_0 is the mean of the period for the standard value of the parameter and T_1 is for the new value of the parameter. The relative change of the peaks is defined similarly.

We first tested the robustness of oscillations for the detailed model. There are 17 different probability rate constants and 2 time delays. Therefore, 39 set of simulations including the set with standard parameter values were run. Figure 7.6 plots the relative change of the mean values of periods and peaks between the results obtained using standard parameters and those obtained using a changed parameter. It is seen that most changes in the period are in the interval [-0.05, 0.05] and that the changes in peaks are relatively large. Figure 7.7 plots the CV of the period and peaks for all parameter sets. It is seen that CV of the periods are very small, in the interval [0.02, 0.05]. When we changed each individual parameter by 20% of its standard value, the mean of the period was never changed more than 11%. The period is most sensitive to τ_2, the time delay needed for *per* translation. When the mean value of τ_2 was decreased (increased) by 20% of its standard value, the mean period was decreased (increased) by 10.66% (10.78%) and the CV of the period was 3.23% (3.32%), which is almost the same as the CV for the standard parameters. The peak of the free dCLOCK is most sensitive to c_3, the probability rate constant of translation of *dclock* mRNA to dCLOCK protein. Decreasing (increasing) c_3 by 20% decreased (increased) the mean peak of free dCLOCK by 23.54% (25.44%), and the corresponding CV was 13.04% (14.14%). The peak of the total PER is most sensitive to c_{11}, the probability constant rate of translation of *per* mRNA to PER protein. Decreasing (increasing) c_{11} by 20% decreased (increased) the mean peak of total PER by 28.12% (25.45%), and the corresponding CV was 15.37% (14.28%). Therefore, the system appears to have small variation in the period but relatively large variation in the peaks when a parameter changes. This is very reasonable from the biological point of view since circadian rhythm is endogenous, which requires very small variation in the period even when some parameters are changed due to the change of external cues. The relatively large variation in the peaks is due to the stochastic fluctuation of gene transcription and translation.

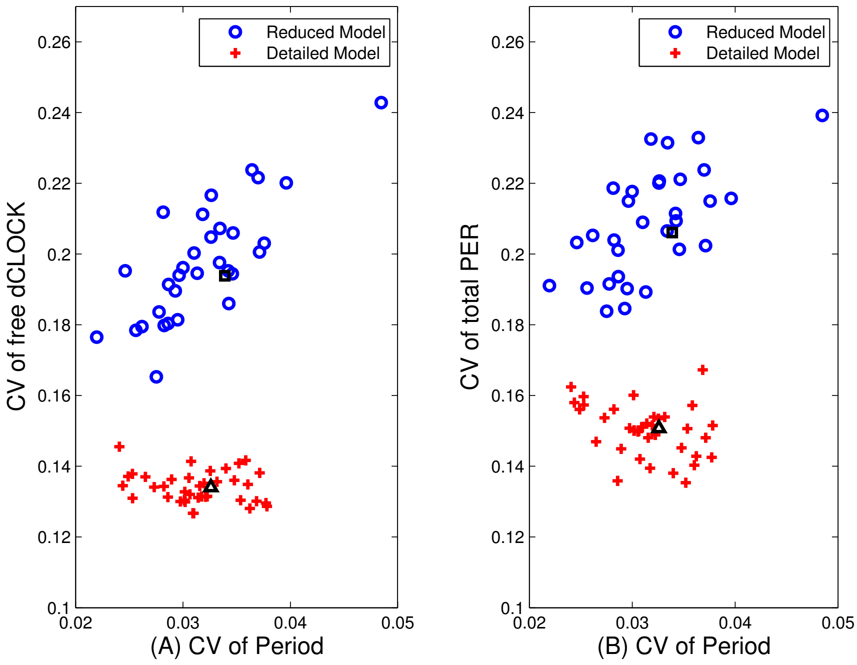


Figure 7.7: CVs of periods and peaks of free dCLOCK (left) and total PER (right) after the value of one parameter increases or decreases by 20% of the standard value while other parameters are fixed. CVs of periods and peaks of free dCLOCK and total PER for the standard parameter set are also shown as □ for reduced model and △ for detailed model.

We also tested the robustness of oscillation for the reduced model. The reduced model has 14 probability rate constants and 2 time delays. Therefore, 33 sets of simulations were run including the set with standard parameter values. Figure 7.6 plots relative change of the mean value of the period and peaks for the parameter sets with one changed parameter comparing with the standard parameter set and Figure 7.7 plots the CV of the period and peaks for all parameter sets. It is seen that the change is small in period but relatively large in peaks when a parameter changes. It is also seen that CV of the periods are very small for both models, in the interval [0.02, 0.05]. Therefore, the system is very robust to the parameter variation in oscillation period. Note that the CV of the peaks of the reduced model is larger than that of the detailed model. This is due to the fact that the reduced model has lower number of molecules in the system so that there is larger internal noise.

As in the detailed model, the period, the peak of the free dCLOCK and the peak of the total PER in the reduced model are most sensitive to τ_2, c_3 and c_{11}, respectively. Specifically, decreasing (increasing) the mean value of τ_2 by 20% decreased (increased) the mean period by 8.01% (8.81%) and the corresponding CV was 3.42% (3.36%). Decreasing (increasing) c_3 by 20% decreased (increased) the mean peak of free dCLOCK by 19.76% (22.69%) and the corresponding CV was 20.60% (17.85%). Decreasing (increasing) c_{11} by 20% decreased (increased) the mean peak of total PER by 21.31% (23.36%) and the corresponding CV was 21.49% (20.10%). Comparing the results of two models, it appears that both models have small changes in the mean period and relatively large changes in the mean peaks and that the reduced model has slightly larger CVs.

7.3.3 Light entrainment of oscillation in the presence of noise

Models of circadian rhythms must be able to maintain synchrony with environmental cycles to drive behavioral, physiological and metabolic outputs at appropriate time of day [80]. Circadian rhythms can be entrained by external cues, such as daily environmental cycles of light, temperature, etc. But light is generally considered as the strongest and most pervasive factor. Therefore, the responses of the rhythm are often simulated by light pulses or light/dark (L/D) cycles [85,91,113–115]. We first consider the detailed model. In *Drosophila*, light induces to enhance the degradation of phosphorylated TIM [85,116,117]. Since there is no separate variable for TIM in our model, the degradation of phosphorylated PER was induced to simulate the effect of light,

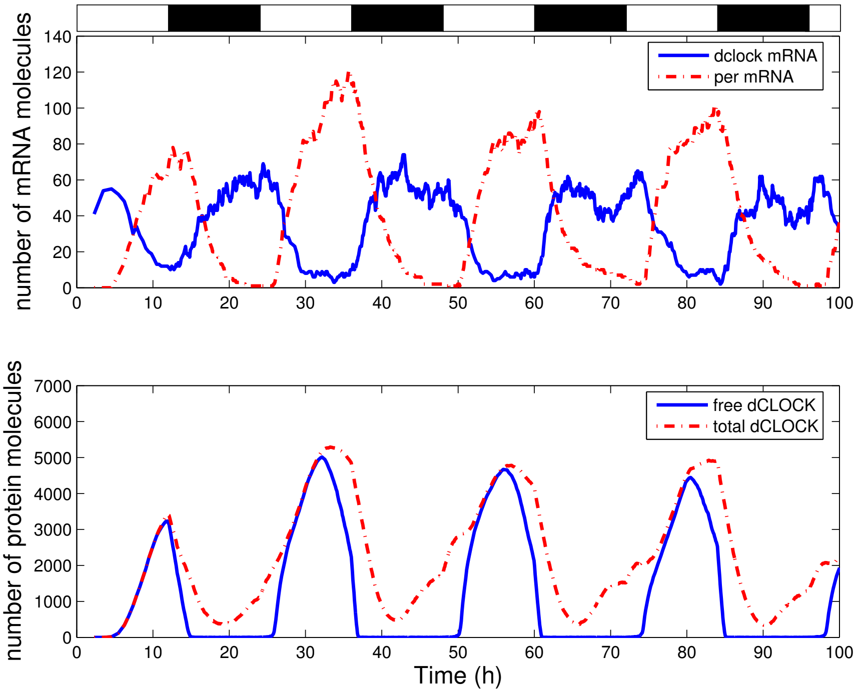


Figure 7.8: One trajectory of *dclock* and *per* mRNA, free dCLOCK and total dCLOCK protein for the detailed stochastic model with light response under L/D cycle.

as done by Smolen *et al* [85]. Here the phosphorylated PER includes all unbounded and bounded PERs (PER$_1$ $\sim$ PER$_{10}$ and dCLOCK.PER$_1$ $\sim$ dCLOCK.PER$_{10}$). The dCLOCK is released after the PER complex with dCLOCK is degraded by light and the degradation rate of all phosphorylated PER is 0.9 h^{-1} [85]. In addition, to keep the oscillation period, the maximum degradation rate of dCLOCK, V_{dc}, was reduced to 1.5 nM h^{-1} [85] and the probability rate constants c_4 and $c_i, i = 44, \cdots, 54$ in our stochastic model were reduced correspondingly.

Figures 7.8 and 7.9 plot one trajectory of *dclock* mRNA, *per* mRNA, free dCLOCK, total dCLOCK and total PER, which demonstrates the entrainment of simulated circadian oscillations under the L/D cycle. The L/D cycle uses 12 hours light first and then 12 hours dark every 24 hours. It is seen that the peak of free dCLOCK is enhanced under new condition. Figure 7.9 also shows that the shape of the time

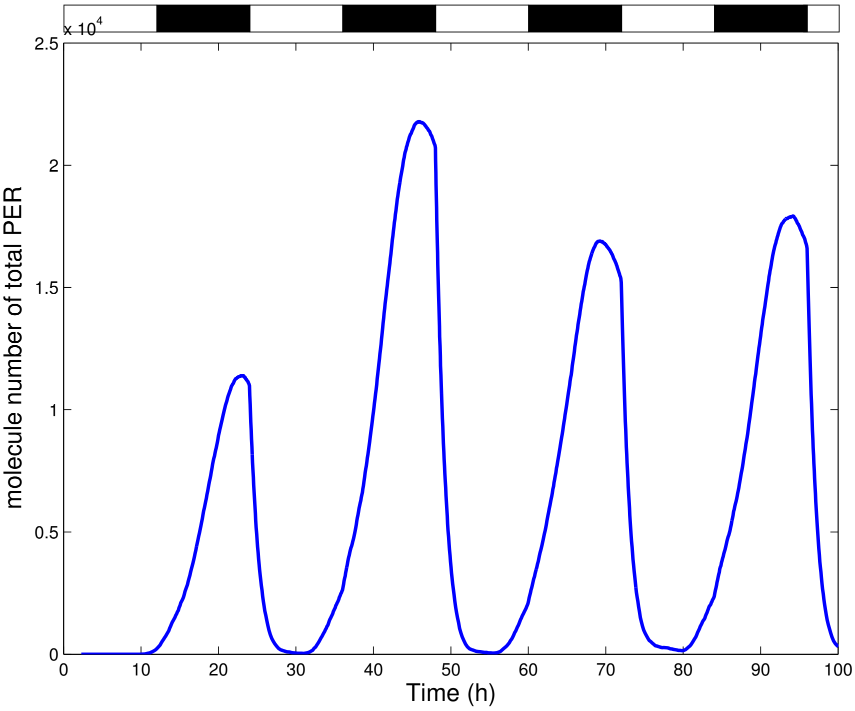


Figure 7.9: One trajectory of total PER protein for the detailed stochastic model with light response under L/D cycle.

trajectory of total PER under L/D circle differs significantly from that of the constant darkness as shown in Figure 7.2. The number of molecules of total PER drops off much more quickly when switching from dark to light than that in Figure 7.2, which is consistent with the experimental results [86, 88] and Smolen's simulation results [85]. Moreover, the mean and CV of oscillation period obtained from 100 runs of simulation under the L/D cycle are 24.03 h and 3.0%, respectively. The mean and CV of peak values of free dCLOCK are 4329.5 and 18.30%, respectively. The mean and CV of peak value of total PER are 17785 and 13.47%, respectively. Therefore, the model not only runs well under the L/D cycle, but also shows stable period but with considerable fluctuations in oscillation peaks.

L/D cycle was also applied to the reduced model to test the light entrainment. Since the light exposure was simulated by enhancing PER degradation [87], the prob-

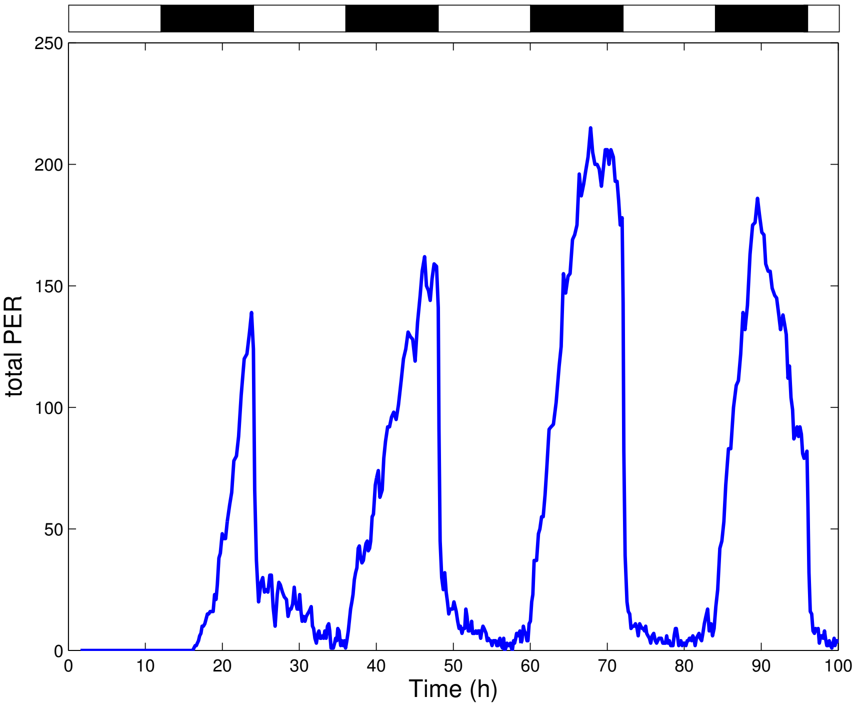


Figure 7.10: One trajectory of total PER protein for the reduced stochastic model with light response under L/D cycle.

ability rate constants for the degradation of unbounded PER and bounded PER, c_{12} and c_{14}, both are increased by 4.5 h^{-1} [87]. Figure 7.10 shows one trajectory of total PER under L/D circle. Observations similar to those for the detailed model were seen: the number of molecules of total PER falls more quickly between dark-to-light switch than that under constant darkness; the oscillation appears to have a stable period but significant fluctuations in peak values of total PER as well as the peaks of free and total dCLOCK proteins (data not shown).

7.3.4 Impact of transcription activation rate

As we mentioned earlier, the rate that dCLOCK binds to *per* and *dclock* genes is unknown but was estimated in our simulation. The rate that dCLOCK dissociates with *per* and *dclock* genes is chosen to be equal to the experimentally reported dis-

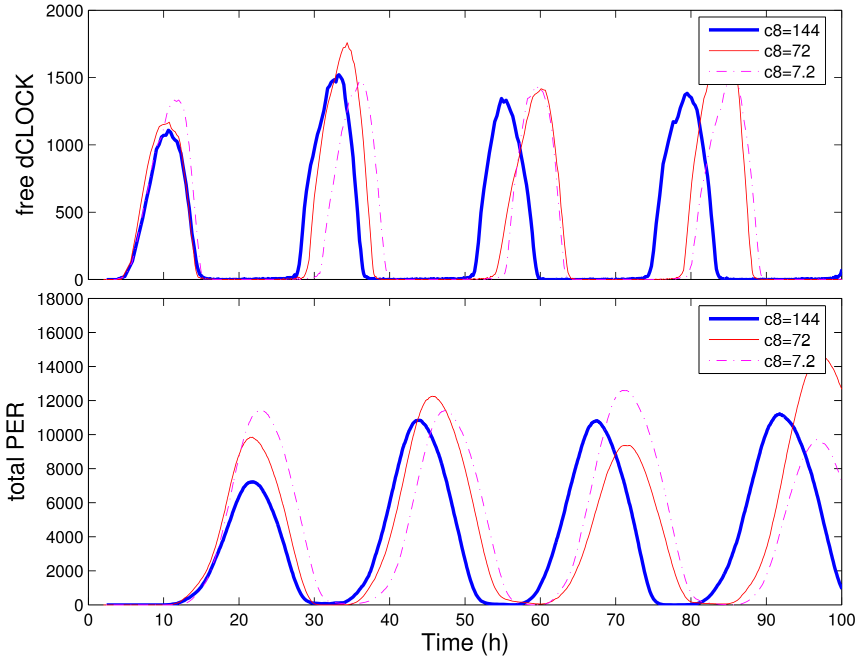


Figure 7.11: One trajectory of free dCLOCK and total PER protein for the detailed stochastic model with $c_8 = 144$ h^{-1}, 72 h^{-1} and 7.2 h^{-1}.

sociation rate of myogenin protein with the E-box of E12 gene. In a deterministic model, these rates generally do not affect oscillation as long as their ratio is fixed. However, these rates may have significant effects on transcription noise even when their ratio is fixed [20, 21, 118]. In the following, we change the value of c_8 while keep the ratio c_8/c_7 fixed to see whether the oscillation period changes. Since both detailed model and reduced models yield similar results, we only give results for the detailed model.

The standard value of c_8, chosen from experimental result, was 72 h^{-1} as described earlier and we ran simulations using two other values for c_8: 144 h^{-1} and 7.2 h^{-1}. We found that if we further increase c_8 beyond 144 h^{-1}, it would not affect simulation results. Therefore, we only compare the simulation results under these three values. Figure 7.11 shows one trajectory of free dCLOCK and total PER under three different values of c_8. It is seen that the period of oscillations for the higher unbinding rate is slightly smaller than that for the lower unbinding rate. The mean values of the period obtained from 100 runs of simulation for $c_8 = 144, 72, 7.2$ h^{-1} were 23.84, 23.93 and 24.21 h, respectively, and the correspondingly CVs are 2.89%, 3.26%, 3.30%. The mean peaks of free dCLOCK, total dCLOCK and total PER for three values of c_8, as well as the corresponding CVs, are almost the same since the ratio of c_8/c_7 is fixed. Therefore, under the three values tested, the rates that dCLOCK binds/unbinds to *per* and *dclock* genes do not have significant effect on oscillations as long as their ratio is fixed. However, if we further decrease the binding/unbinding rates by a factor of 100 and 1000, the mean of oscillation period changes to 26.70 h and 37.40 h, respectively. Note that this is consistent with the results of Forger and Peskin [118], as well as Gonze *et al.* [96], where oscillation period is changed significantly [118] or oscillations become irregular [96], when the binding and unbinding rates are decreased by at least two orders of magnitude. Since the rate change by a factor of 10 is significant, the oscillation period is relatively robust to the binding/unbinding rate within a reasonable range around the experimental reported rate.

7.4 Discussion

We have presented a detailed and a reduced stochastic model for delay-induced circadian rhythm in *Drosophila* based on the deterministic models of Smolen *et al.* [85, 87], and employed our recently developed exact stochastic simulation algorithm [101] to simulate the circadian rhythm. This work is unique since no exact stochastic simulation has been carried out for circadian rhythms based on a model with random time delays. As discussed in [101], several SSAs have been developed for reaction systems with delays [119, 120]. However, the algorithm in [119] and two algorithms in [120] are not exact. Ref. 33 also proved that another heuristic algorithm in [120] is exact but requires more computation than the exact SSA in [101]. Since both algorithms are exact, they should produce the same statistical results. Another SSA for systems with delays was proposed in [121], but an approach similar to that in [119] was used, and thus, it is not exact either. Smolen *et al.* [85,87], as well as Li and Lang [100], also simulated delay-induced circadian oscillation, but they used approximate stochastic simulation methods.

Our simulation results demonstrated that the intrinsic noise cause large fluctuations in oscillation peaks but very small fluctuations in oscillation period. This observation is seen in all simulations under different conditions, such as constant darkness and L/D cycles. Deterministic simulation cannot reveal this phenomena, since both period and peaks are constant. Our stochastic simulations also showed that circadian oscillation is robust in the presence of noise in the sense that noise has little effect on oscillation period although it can change oscillation peaks significantly. We also showed that random delays within certain range do not cause significant variations in the oscillation period as long as the average delays are fixed. To best of our knowledge, these two results have not been observed in previous stochastic simulation of circadian rhythms. These two observations imply that circadian oscillation is robust in the presence of noise and random delays and that the randomness inherent to the oscillation circuit may not have much biological impact on the organism. As discussed in [112], when a protein regulates its targets, it often operates on a Hill curve. Once the level of the regulating protein is higher or lower than certain value, the protein operates at the top or bottom of the curve and the fluctuation of its level to certain extend does not affect much the regulating effect on its targets.

Therefore, the relatively large variations in the peak values of PER and dCLOCK proteins observed in our simulation may not have a strong biological impact.

Similar to previous deterministic simulations and approximate stochastic simulations [85,87], our stochastic simulation show that both detailed and reduced stochastic models can provide sustained oscillations under darkness and L/D cycles. Our results also show right phase of all the components in the system, correct phase and anti-phase relationship of mRNAs and proteins, and also the appropriate lags between mRNAs and proteins. Our stochastic simulation further demonstrated that circadian rhythm is robust to parameter variations in the presence of noise. Increasing or decreasing each parameter by 20% of its standard value changes the mean period by less than 11% and causes negligible changes in the CV of oscillation periods. The model is not sensitive to the time delay during the *dclock* mRNA translation, but it is most sensitive to the average time delay during *per* mRNA translation, which shows that time delay is essential to circadian oscillation in the two models. However, random fluctuations in these two time delays have little effect on the oscillation period as long as the average delays are fixed. We also found that the binding and unbinding rates of dCLOCK to *dclock* and *per* genes within a reasonable range have little impact on the circadian oscillation. Increasing or decreasing the binding and unbinding rates by 10 times relative to an experimentally reported rate while keeping their ratio fixed does not cause significant changes in the period and peaks of oscillation.

We have compared our exact simulations with approximate simulations of Smolen *et al.* [85, 87] in Section 7.3.1. Another work by Li and Lang [100] also employed approximate SSAs to simulate the reduced model of Smolen *et al.* [87]. Like Smolen *et al.* [85, 87], Li and Lang [100] used deterministic delays, whereas we employed random delays which are more appropriate to reflect the delays in transcription, translation and other chemical process. Li and Lang emphasized on the noise induced oscillation and showed that noise can sustain oscillation in the parameter region where no oscillation is predicted by the deterministic model, whereas we here focused on the robustness of oscillation in the presence of intrinsic noise and the effect of random delays. We showed that the oscillation is robust in the presence of noise since there is very little variability in oscillation period in spite of large random variability in peaks, and that random changes in delays within a large interval around the fixed average delay cause little variability in the oscillation period.

CHAPTER 8

Summary and Future Work

8.1 Summary

In this thesis, we first reviewed existing stochastic simulation algorithms including the exact SSA and several approximate SSAs, and then developed several novel SSAs including the K-leap method, the hybrid τ/K-leap method, the modified K-leap method, the unbiased τ-leap method, the iwNRM and the iwSSA, to increase simulation speed and reduce simulation errors. We also proposed two stochastic models for the circadian rhythm of *Drosophila* and simulated the dynamics of the circadian system.

The K-leap method constrained the total number of reactions occurring during each leap step, therefore, it can better satisfy the leap condition and improve simulation accuracy. Moreover, since our K-leap method becomes the exact SSA when $K = 1$, it can naturally fold back to the exact SSA, if leaping is inappropriate due to the sensitivity of the propensity functions to population changes. In certain biochemical systems and gene networks there often have some reactions involving in a small number of molecules and other reactions involving in a large number of molecules. To efficiently deal with such cases, we developed a hybrid τ/K-leap method and a modified K-leap method to accelerate simulation speed without losing simulation accuracy.

We analyzed the bias of the results yielded from all existing τ-leap methods. To remove such bias, we developed several unbiased τ-leap methods including unbiased Poisson τ-leap method, unbiased binomial τ-leap mehtod and unbiased Poisson/Gaussian/Binomial τ-leap method. These unbiased leap methods incorporate a semi-analytical method for calculating the mean and variance of the number of reactions occuring in a specific time period into simualtion. Therefore, they can significantly improve simulation accuracy.

One difficult problem in stochastic simulation of chemical reacting system is to estimate the probability of rare events that occur with an extremely low probability within a specific time period. To deal with rare events, we applied the importance sampling technique to the next reaction method of the SSA and developed the wNRM which is more efficient than the wSSA and the rwSSA. We also introduced a systematic method for selecting importance sampling parameters. Incorporated this method into the wSSA and the wNRM, we got the iwSSA and he iwNRM. Simulation results showed that comparing with the rwSSA, our iwNRM and iwSSA could substantially reduce the variance of the estimated probability of the rare event and speed up simulation for a given number of simulation runs.

As an application of stochastic simulation, we developed two stochastic models for circadian clock in *Drosophila* and then used stochastic simulation to investigate the dynamic of the circadian rhythm. Our simulation results showed that in the presence of intrinsic noise, the period of circadian oscillation is highly robust although the oscillation peaks undergo significant random fluctuations. Moreover although average time delays are essential to simulation of oscillation, random changes in time delays within certain range around fixed average time delay cause little variability in the oscillation period. We also found that circadian oscillation is robust to the changes in model parameters, and that oscillation can be entrained by light/dark circles.

8.2 Future Work

Although we have made a good progress in the development of efficient stochastic simulation algorithms and in the application of these algorithms in modeling and simulating gene networks, there are several research topics that worth further investigation, as listed in the following.

8.2.1 Unbiased K-Leap Method for Stochastic Simulation of Chemically Reacting Systems

In Chapter 3, we have developed the K-leap method that constrains the total number of reactions occuring during each leap step so that it can better satisfy the leap condition and improve simulation accuracy. In Chapter 5, we have proposed an unbiased τ-leap method, which significantly improves accuracy of the τ-leap. Since current K-leap method is also biased, if the true mean of $K_1, \cdots, K_M$ in each leap based on CME can be found, then we can also develop an unbiased K-leap method to further improve the performance of the K-leap method.

We expect that we can derive an ODE for the mean value of $K_1, \cdots, K_M$ under the constraint $\sum_{m=1}^{M} K_m = K$, using an approach similar to that used in deriving (5.10). Once we get these mean values by solving the ODE, we can use them in the K-leap method to remove the bias. We expect that the unbiased K-leap method can outperform the K-leap method and the unbiased τ-leap method.

8.2.2 Error Control and CME-based Leap Method

Several issues related to leap method worth further investigation. The first issue is error control in leap methods. The parameter, ϵ, in the leap condition is critical to the control of simulation errors and speed. However, the quantitative relationship between ϵ and simulation errors needs to be elucidated. If we can quantify such relationship, we will be able to develop better leap methods. The second issue is to explore the CME to improve simulation speed and accuracy. Existing stochastic simulation algorithms were developed based on the fundamental premise of stochastic kinetics, but not the CME. Since the CME precisely describes the time evolution of the system, it is expected that we can develop better SSAs if information from the CME can be explored. Therefore, perhaps we can combine the analytical form of CME and numerical simulation into a unified framework. If this is successful, we expect to develop extremely efficient and accurate SSAs.

8.2.3 Circadian Rhythm and Computational Modeling

Circadian rhythms are based on a molecular mechanism regulated at the transcriptional, translational and post-translational levels [122]. In Chapter 7, we have developed two stochastic models and apply stochastic simulation with delays to investigated the circadian rhythms in *Drosophila*. Our stochastic models were developed from Smolen's deterministic models [85, 87], which used a single "lumped" variable, PER, to represent both PER and TIM proteins, since the time courses of PER and TIM proteins are similar in shape and largely overlap. As the TIM gene also plays an important role in circadian rhythm, a stochastic model that includes both PER and TIM may worth further study.

Stochastic modeling not only has become a powerful and useful tool in investigating the regulatory mechanism of circadian rhythm, but also can provide testable predictions or reveal unexpected results. We have only considered the circadian system of *Drosophila*, the circadian rhythm of other living organisms also worth investigation using the stochastic modeling and simulation techniques developed in this thesis.

APPENDIX A

Derivation of Equation (3.1) and (3.2)

We can write the joint PDF $p(K_1, \cdots, K_M, \tau | \sum_{m=1}^{M} K_m = K)$ as

$$p(K_1, \cdots, K_M, \tau | \sum_{m=1}^{M} K_m = K) = p(\tau | \sum_{m=1}^{M} K_m = K) p(K_1, \cdots, K_M | \tau, \sum_{m=1}^{M} K_m = K). \tag{A.1}$$

Since K reactions occur in the interval $[t,\ t+\tau]$, and occurrence of each reaction is an independent event, we have $\tau = \sum_{k=1}^{K} \tau_k$, where τ_k, $k = 1, \cdots, K$, are the time between two consecutive occurrences of a reaction, and are independent random variable following an exponential PDF with parameter $a_0(\mathbf{x})$. As proved by Gillespie [38], the sum of K independent, and identically distributed exponential random variables is a Gamma random variable. Hence, we obtain the PDF $p(\tau | \sum_{m=1}^{M} K_m = K)$ in (3.1).

We can express $p(K_1, \cdots, K_M | \tau, \sum_{m=1}^{M} K_m = K)$ in (A.1) as

$$\begin{aligned} p(K_1, \cdots, K_M | \tau, \sum_{m=1}^{M} K_m = K) =& p(K_1 | \tau, \sum_{m=1}^{M} K_m = K) \\ & \times \prod_{m=2}^{M} p(K_m | K_1, \cdots, K_{m-1}, \tau, \sum_{j=1}^{M} K_j = K). \end{aligned} \tag{A.2}$$

We can write $p(K_1 | \tau, \sum_{m=1}^{M} K_m = K)$ in (A.2) as

$$\begin{aligned} p(K_1 | \tau, \sum_{m=1}^{M} K_m = K) &= \frac{p(K_1, \sum_{m=1}^{M} K_m = K | \tau)}{p(\sum_{m=1}^{M} K_m = K | \tau)} \\ &= \frac{p(K_1 | \tau) p(\sum_{m=1}^{M} K_m = K | \tau, K_1)}{p(\sum_{m=1}^{M} K_m = K | \tau)} \end{aligned} \tag{A.3}$$

Let us define $\tilde{a}_m \triangleq \sum_{j=m}^{M} a_j(\mathbf{x})$. Notice that $\tilde{a}_1 = a_0(\mathbf{x})$. The PDF $p(K_1|\tau)$ is a Poisson PDF with mean $a_1(\mathbf{x})\tau$, and $p(\sum_{j=m}^{M} K_j|\tau)$ is a Poisson PDF with mean $\tilde{a}_m\tau$. Therefore, Eq. (A.3) becomes

$$p(K_1 = k_1|\tau, \sum_{m=1}^{M} K_m = K) = \begin{cases} \frac{a_1^{k_1}\tilde{a}_2^{K-k_1}}{k_1!(K-k_1)!}\frac{K!}{\tilde{a}_1^{K}}, & k_1 = 0, \cdots, K \\ 0 & \text{otherwise.} \end{cases} \tag{A.4}$$

Given $K_j = k_j$, $j = 1, \cdots, m$, let us define $\tilde{K}_m \triangleq K - \sum_{j=1}^{m-1} k_j$. Similar to (A.3), we have

$$\begin{aligned} p(K_m|K_1, \cdots, K_{m-1}, \tau, \sum_{j=1}^{M} K_j = K) &= p(K_m|\tau, \sum_{j=m}^{M} K_j = \tilde{K}_m) \\ &= \frac{p(K_m|\tau)p(\sum_{j=m}^{M} K_j = \tilde{K}_m|\tau, K_m)}{p(\sum_{j=m}^{M} K_j = \tilde{K}_m|\tau)}, \end{aligned} \tag{A.5}$$

for $m \geq 2$. When $m = M$, it is clear from (A.5) that we have

$$p(K_M = k_M|K_1, \cdots, K_{M-1}, \tau, \sum_{j=1}^{M} K_j = K) = \begin{cases} 1, & k_M = \tilde{K}_M \\ 0 & \text{otherwise} \end{cases} \tag{A.6}$$

When $2 \leq m < M$, similar to deriving (A.4), we can obtain the following from (A.5)

$$p(K_m = k_m|K_1, \cdots, K_{m-1}, \tau, \sum_{j=1}^{M} K_j = K) = \begin{cases} \frac{a_m^{k_m}\tilde{a}_{m+1}^{\tilde{K}_m-k_m}}{k_m!(\tilde{K}_m-k_m)!}\frac{\tilde{K}_m!}{\tilde{a}_m^{\tilde{K}_m}}, & k_m = 0, \cdots, \tilde{K}_m \\ 0 & \text{otherwise} \end{cases} \tag{A.7}$$

Substituting (A.4), (A.6), and (A.7) into (A.2), we obtain $p(K_1, \cdots, K_M|\tau, \sum_{m=1}^{M} K_m = K)$ as given at the right hand side of (3.2). Since $p(K_1, \cdots, K_M|\tau, \sum_{m=1}^{M} K_m = K)$ does not depend on τ, we write it as $p(K_1, \cdots, K_M| \sum_{m=1}^{M} K_m = K)$ in (3.2). Notice that $p(\tau| \sum_{m=1}^{M} K_m = K)$ is independent of K_m, $m = 1, \cdots, M$. Therefore, given the constraint $\sum_{m=1}^{M} K_m = K$, τ is independent of K_m, $m = 1, \cdots, M$.

APPENDIX B

Derivation of Equation (4.1)

We define a random variable $K_c' = \sum_{m\in\mathcal{R}_c} K_m$, then $p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau)$ can be written as

$$p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau) = p(\{K_m, m \in \mathcal{R}_c\}|K_c' < K_c, \tau). \qquad \text{(B.1)}$$

Due to the definition of K_c', K_c' is dependent on $\{K_m, m \in \mathcal{R}_c\}$, and thus we can express (B.1) as

$$p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau) = p(\{K_m, m \in \mathcal{R}_c\}, K_c'|K_c' < K_c, \tau), \qquad \text{(B.2)}$$

which can be further written as

$$p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau) = p(K_c'|K_c' < K_c, \tau)p(\{K_m, m \in \mathcal{R}_c\}|K_c', K_c' < K_c, \tau). \qquad \text{(B.3)}$$

Given K_c', the condition $K_c' < K_c$ is redundant, because if we generate K_c' according to the PDF $p(K_c'|K_c' < K_c, \tau)$, we always have $K_c' < K_c$. Considering this fact and the definition of K_c', we can write the second term at the right hand side of (B.3) as

$$p(\{K_m, m \in \mathcal{R}_c\}|K_c', K_c' < K_c, \tau) = p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m = K_c', \tau). \qquad \text{(B.4)}$$

Combining (B.3) and (B.4), we obtain $p(\{K_m, m \in \mathcal{R}_c\}|\sum_{m\in\mathcal{R}_c} K_m < K_c, \tau)$ given in (4.1).

Notice that the PDF $p(K_c'|\tau)$ follows a Poisson distribution. Therefore, $p(K_c'|K_c' < K_f, \tau)$ is a truncated Poisson distribution, and it is given as follows:

$$\begin{aligned} p\big(K_c'|K_c' < K_c, \tau\big) &= \begin{cases} \frac{p(K_c'|\tau)}{p(K_c' < K_c|\tau)}, & K_c' = 0, \cdots, K_c - 1 \\ 0, & \text{otherwise}, \end{cases} \\ &= \begin{cases} \frac{\exp[-a_0^c(\mathbf{x})\tau](a_0^c(\mathbf{x})\tau)^{K_c'}/K_c'!}{\sum_{j=0}^{K_c-1}(\exp(-a_0^c(\mathbf{x})\tau)(a_0^c(\mathbf{x})\tau)^j/j!)}, & K_c' = 0, \cdots, K_c - 1 \\ 0, & \text{otherwise}, \end{cases} \end{aligned} \tag{B.5}$$

which can be simplified to (4.2).

Bibliography

[1] Zupan, B., Demsar, J., Bratko, I., Juvan, P., Halter, J. A., Kuspa, A., and Shaulsky, G., "GenePath: a system for automated construction of genetic networks from mutant data," *Bioinformatics*, Vol. 19, 2003, pp. 383.

[2] Cai, X. and Wang, X., "Stochastic modeling and simulation of gene networks," *IEEE Signal Processing Magazine*, Vol. 24, 2007, pp. 27–36.

[3] Smolen, P., Baxter, D. A., and Byrne, J. H., "Modeling transcriptional control in gene networks-methods, recent results, and future directions," *Bull. Math. Biol.*, Vol. 62, 2000, pp. 247–292.

[4] Hasty, J., McMillen, D., Isaacs, F., and Collins, J. J., "Computational studies of gene regulatory networks: in numero molecular biology," *Nat. Rev. Genet.*, Vol. 2, 2001, pp. 268–279.

[5] de Jong, H., "Modeling and simulation of genetic regulatory systems: a literature review," *J. Comput. Biol.*, Vol. 9, 2002, pp. 67–103.

[6] Kauffman, S. A., "Metabolic stability and epigenesis in randomly constructed genetic nets," *J. Theor. Biol.*, Vol. 22, 1969, pp. 437–467.

[7] Shmulevich, I., Dougherty, E. R., and Zhang, W., "From boolean to probabilistic boolean networks as models of genetic regulatory networks," *Proc. IEEE*, Vol. 90, 2002, pp. 1778–1792.

[8] McAdams, H. H. and Arkin, A., "It's a noisy business! Genetic regulation at the nanomolar scale," *Trends in Genetics*, Vol. 15, 1999, pp. 65–69.

[9] Elowitz, M. B. and Leibler, S., "A synthetic oscillatory network of transcriptional regulators," *Nature*, Vol. 403, 2000, pp. 335–338.

[10] Fraser, H. B., Hirsh, A. E., Giaever, G., Kumm, J., and Eisen, M. B., "Noise minimization in eukaryotic gene expression," *PLoS Biol.*, Vol. 2, 2004, pp. e137.

[11] Fedoroff, N. and Fontana, W., "Genetic networks: Small numbers of big molecules," *Science*, Vol. 297, 2002, pp. 1129–1131.

[12] Ozbudak, E. M., Thattai, M., Kurtsser, I., Grossman, A. D., and van Oudenaarden, A., “Regulation of noise in the expression of a single genes,” *Nat. Genet.*, Vol. 31, 2002, pp. 69–73.

[13] Elowitz, M. B., Levine, A. J., Siggia, E. D., and Swain, P. S., “Stochastic gene expression in a single cell,” *Science*, Vol. 297, 2002, pp. 1183–1186.

[14] Blake, W. J., Kaern, M., Cantor, C. R., and Collins, J. J., “Noise in eukaryotic gene expression,” *Nature*, Vol. 422, 2003, pp. 633–637.

[15] Novick, A. and Weiner, M., “Enzyme induction as an all-or-none phenomenon,” *Proc. Nat. Academy Science*, Vol. 43, 1957, pp. 553–566.

[16] Rigney, D. R. and Schieve, W. C., “Stochastic model of linear, continuous protein synthesis in bacterial populations,” *J. Theor. Biol.*, Vol. 69, 1977, pp. 761–766.

[17] Berg, O. G., “A model for the statistical fluctuations of protein numbers in a microbial population,” *J. Theor. Biol.*, Vol. 71, 1978, pp. 587–603.

[18] Ko, M. S., “A stochastic model for gene induction,” *J. Theor. Biol.*, Vol. 153, 1991, pp. 181–194.

[19] Rao, C. V., Wolf, D. M., and Arkin, A. P., “Control, exploitation and tolerance of intracellular noise,” *Nature*, Vol. 420, 2002, pp. 231–237.

[20] Raser, J. M. and O’Shea, E. K., “Noise in gene expression: origins, consequences, and control,” *Science*, Vol. 309, 2005, pp. 2010–2013.

[21] Kærn, M., Elston, T. C., Blake, W. J., and Collins, J. J., “Stochasticity in gene expression: from theories to phenotypes,” *Nature Review Genetics*, Vol. 6, 2005, pp. 451–464.

[22] Cai, L., Friedman, N., and Xie, X. S., “Stochastic protein expression in individual cells at the single molecule level,” *Nature*, Vol. 440, Mar. 2006, pp. 358–362.

[23] Goutsias, J. and Kim, S., “Stochastic Transcriptional Regulatory Systems with Time Delays: A Mean-Field Approximation,” *Journal of Computational Biology*, Vol. 13, No. 5, 2006, pp. 1049–1076.

[24] Goutsias, J., “A Hidden Markov Model for Transcriptional Regulation in Single Cells,” *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 3, No. 1, 2006, pp. 57–71.

[25] Shahrezaei, V. and Swain, P. S., “The stochastic nature of biological networks,” *Curr. Opin. Biotechnol.*, Vol. 19, 2008, pp. 369–374.

[26] Barkai, N. and Leibler, S., “Robustness in simple biochemical networks,” *Nature*, Vol. 387, 1997, pp. 913–917.

[27] Barkai, N. and Leibler, S., "Circadian clocks limited by noise," *Nature*, Vol. 403, 1999, pp. 267–268.

[28] van Dassow, G., Meir, E., Munro, E. M., and Odell, G. M., "The segment polarity network is a robust developmental module," *Nature*, Vol. 406, 2000, pp. 188–192.

[29] Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P., *Molecular Biology of the Cell*, Garland Science, London, 4th ed., 2002.

[30] Alberts, B., Bray, D., Hopkin, K., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P., *Essential Cell Biology*, Garland Science, London, 2nd ed., 2004.

[31] Kozak, M., "Pushing the limits of the scanning mechanism for initiation of translation," *Gene*, Vol. 299, 2002, pp. 1–34.

[32] Gillespie, D. T., "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions," *J. Comput. Phys.*, Vol. 22, 1976, pp. 403–434.

[33] Gillespie, D. T., "Exact stochastic simulation of coupled chemical reaction," *J. Phys. Chem.*, Vol. 81, 1977, pp. 2340–2361.

[34] Gillespie, D. T., "A rigorous derivation of the chemical master equation," *Physica A*, Vol. 188, 1992, pp. 402–425.

[35] McAdams, H. H. and Arkin, A., "Stochastic mechanisms in gene expression," *Proc. Natl. Acad. Sci. USA*, Vol. 94, 1997, pp. 814–819.

[36] Arkin, A., Ross, J., and McAdams, H. H., "Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected Escherichia coli cells," *Genetics*, Vol. 149, 1998, pp. 1633–1648.

[37] Kierzek, A. M., "STOCKS: STOChastic kinetic simulations of biochemical systems with Gillespie algorithm," *Bioinformatics*, Vol. 18, No. 3, 2002, pp. 470–481.

[38] Gillespie, D. T., "Approximate accelerated stochastic simulation of chemically reacting systems," *J. Chem. Phys.*, Vol. 115, 2001, pp. 1716–1733.

[39] Gillespie, D. T. and Petzold, L. R., "Improved leap-size selection for accelerated stochastic simulation," *J. Chem. Phys.*, Vol. 119, No. 6, 2003, pp. 8229–8234.

[40] Cao, Y., Gillespie, D. T., and Petzold, L. R., "Avoid negative population in explicit poisson tau-leaping," *J. Chem. Phys.*, Vol. 123, art. no. 54104, 2005.

[41] Tian, T. and Burrage, K., "Binomial leap methods for simulating stochastic chemical kinetics," *J. Chem. Phys.*, Vol. 121, 2004, pp. 10356–10364.

[42] Chatterjee, A., Vlachos, D. G., and Katsoulakis, M. A., "Binomial distribution based τ-leap accelerated stochastic simulation," *J. Chem. Phys.*, Vol. 122, art. no. 024112, 2005.

[43] Pettigrew, M. F. and Resat, H., "Multinomial tau-leaping method for stochastic kinetic simulations," *J. Chem. Phys.*, Vol. 126, No. 8, 2007.

[44] Kuwahara, H. and Mura, I., "An efficient and exact stochastic simulation method to analyze rare events in biochemical systems," *J. Chem. Phys.*, Vol. 129, 2008, pp. 165101.

[45] Gillespie, D. T., Roh, M., and Petzold, L. R., "Refining the weighted stochastic simulation algorithm," *J. Chem. Phys.*, Vol. 130, 2009, pp. 174103.

[46] Gillespie, D. T., "Stochastic simulation of chemical kinetics," *Annu. Rev. Phys. Chem.*, Vol. 58, 2007, pp. 35–55.

[47] Gillespie, D. T., "The chemical Langevin equation," *J. Chem. Phys.*, Vol. 113, No. 1, 2000, pp. 297–306.

[48] McQuarrie, D. A., "Stochastic approach to chemical kinetics," *J. Appl. Prob.*, Vol. 4, 1967, pp. 413–478.

[49] Gibson, M. A. and Bruck, J., "Exact stochastic simulation of chemical systems with many species and many channels," *J. Phys. Chem. A*, Vol. 105, 2000, pp. 1876–1889.

[50] Cao, Y., Li, H., and Petzold, L. R., "Efficient formulation of the stochastic simulation algorithm for chemically reacting systems," *J. Chem. Phys.*, Vol. 121, No. 9, 2004, pp. 4059–4067.

[51] Lok, L. and Brent, R., "Automatic generation of cellular reaction networks with Moleculizer 1.0," *Nat. Biotech.*, Vol. 23, 2005, pp. 131–136.

[52] McCollum, J. M., Peterson, G. D., Cox, C. D., Simpson, M. L., and Samatova, N. F., "The sorting direct method for stochastic simulation of biochemical systems with varyinng reaction execution behavior," *Comput. Biol. Chem.*, Vol. 30, 2006, pp. 39–49.

[53] Li, H. and Petzold, L. R., "Logarithmic direct method for discrete stochastic simulation of chemically reacting systems," Technical Report,2006.

[54] Cao, Y., Gillespie, D. T., and Petzold, L. R., "Efficient stepsize selection for the tau-leap simulation method," *J. Chem. Phys.*, Vol. 124, No. 4, art. no. 044109, 2006.

[55] Papoulis, A., *Probability, Random Variables and Stochastic Processes*, McGraw-Hill, New York, 1984.

[56] Cao, Y. and Petzold, L. R., "Accuracy limitations and the measurement of errors in the stochastic simulation of chemically reacting systems," *J. Comput. Phys.*, Vol. 212, 2005, pp. 6–24.

[57] Cai, X. and Xu, Z., "K-leap methods for accelerating stochastic simulation of gene expression," *IEEE Conf. Genomic Signal Processing and Statistics*, College Station, Texas, 2006.

[58] Cai, X. and Xu, Z., "K-Leap methods for accelerating stochastic simulation of chemically reacting systems," *J. Chem. Phys.*, Vol. 126, No. 7, 2007, pp. 074102.

[59] Devroye, L., *Non-uniform Random Variate Generation*, Springer-Verlag, Berlin, 1986.

[60] Devroye, L., "Generating the maximum of independent identically distributed random variables," *Computers and Mathematics with Application*, Vol. 6, 1980, pp. 305–315.

[61] Kachitvichyanukul, V. and Schmeiser, B. W., "Bionmial random variate gneration," *Communications of the ACM*, Vol. 31, No. 2, 1988, pp. 216–222.

[62] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Numerical Recipes in C*, Cambridge University Press, Cambridge, 1995.

[63] Kennell, D. and Riezman, H., "Transcription and translation initiation frequencies of the *Escherichiacoli* lac operon," *J. Mol. Biol.*, Vol. 114, 1977, pp. 1–21.

[64] Kierzek, A. M., Zaim, J., and Zielenkiewicz, P., "The effect of transcription and translation initiation frequencies on the stochastic fluctuations in prokaryotic gene expression," *J. Biol. Chem.*, Vol. 276, 2001, pp. 8165–8172.

[65] Rathinam, M., Petzold, L. R., Cao, Y., and Gillespie, D. T., "Stiffness in stochastic chemically reacting system: the implicit tau-leaping method," *J. Chem. Phys.*, Vol. 119, No. 24, 2003, pp. 12784–12794.

[66] Xu, Z. and Cai, X., "Unbiased tau-leap methods for stochastic simulation of chemically reacting systems." *J. Chem. Phys.*, Vol. 128, 2008, pp. 154112.

[67] Goutsias, J., "Quasiequlibrium approximation of fast reaction kinetics in stochastic biochemical systems," *J. Chem. Phys.*, Vol. 122, art. no. 184102, 2005.

[68] Haseltine, E. L. and Rawlings, J. B., "Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics," *J. Chem. Phys.*, Vol. 117, No. 15, 2002, pp. 6959–6969.

[69] Le, C. T., *Introductory Biostatistics*, John Wiley & Sons, Inc., 2003.

[70] Kholodenko, B. N., Demin, O. V., Moehren, G., and Hoek, J. B., "Quantification of short term signaling by the epidermal growth factor receptor," *J. Biol. Chem.*, Vol. 274, 1999, pp. 30169–30181.

[71] Resat, H., Ewald, J. A., Dixon, D. A., and Wiley, H. S., "An integrated model of epidermal growth factor receptor trafficking and signal transduction," *Biophys J.*, Vol. 85, 2003, pp. 730–743.

[72] Csete, M. and Doyle, J., "Bow ties, metabolism and disease," *Trends Biotechnol*, Vol. 22, 2004, pp. 446–450.

[73] Egger, G., Liang, G., Aparicio, A., and Jones, P. A., "Epigenetics in human disease and prospects for epigenetic therapy," *Nature*, Vol. 429, 2004, pp. 457.

[74] Moore-Ede, M., Sulzman, F. M., and Fuller, C. A., *The Clocks that Time Us Physiology of the Circadian Timing System*, Harvard Univ. Press, Cambridge, MA, 1982.

[75] Young, M. W. and Kay, S. A., "Time zones: a comparative genetics of circadian clocks," *Nat. Rev. Genet.*, Vol. 2, 2001, pp. 702–715.

[76] Yu, W. and Hardin, P. E., "Circadian oscillators of *Drosophila* and mammals," *J. Cell Sci.*, Vol. 119, 2006, pp. 4793–4795.

[77] Gonze, D., Halloy, J., and Goldbeter, A., "Deterministic versus stochastic models for circadian rhythms," *J. Biol. Phys.*, Vol. 28, 2002, pp. 637–653.

[78] Williams, J. A. and Sehgal, A., "Molecular components of the circadian system in *Drosophila*," *Annu. Rev. Physiol.*, Vol. 63, 2001, pp. 729–755.

[79] Reppert, S. M. and Weaver, D. R., "Molecular analysis of mammalian circadian rhythms," *Annu. Rev. Physiol.*, Vol. 63, 2001, pp. 647–676.

[80] Hardin, P. E., "The circadian timekeeping system of *Drosophila*," *Curr. Biol.*, Vol. 15, 2005, pp. R714–722.

[81] Dunlap, J. C., "Molecular bases for circadian clocks," *Cell.*, Vol. 96, 1999, pp. 271–290.

[82] Edmunds, L. N., *Cellular and Molecular Bases of Biological Clocks*, Springer Verlag, New York, 1988.

[83] Darlington, T. K., Wager-Smith, K., Ceriani, M. F., Staknis, D., Gekakis, N., Steeves, T. D., Weitz, C. J., Takahashi, J. S., and Kay, S. A., "Closing the circadian loop: CLOCK-induced transcription of its own inhibitors PER and TIM," *Science*, Vol. 280, 1998, pp. 1599–1603.

[84] Lee, C., Bae, K., and Edery, I., "PER and TIM inhibit the DNA binding activity of a *Drosophila* CLOCK-CYC/dBMAL1 heterodimer without disrupting formation of the heterodimer: a basis for circadian transcription," *Mol. Cell Biol.*, Vol. 19, 1999, pp. 5316–5323.

[85] Smolen, P., Baxter, D. A., and Byrne., J. H., "Modeling circadian oscillations with interlocking positive and negative feedback loops," *J. Neurosci.*, Vol. 21, 2001, pp. 6644–6656.

[86] Bae, K., Lee, C., Hardin, P. E., and Edery, I., "dCLOCK is present in limiting amounts and likely mediates daily interactions between the dCLOCK-CYC transcription factor and the PER-TIM complex," *J. Neurosci.*, Vol. 20, 2000, pp. 1746–1753.

[87] Smolen, P., Baxter, D. A., and Byrne., J. H., "A reduced model clarifies the role of feedback loops and time delays in the *Drosophila* circadian oscillator," *Biophys. J.*, Vol. 83, 2002, pp. 2349–2359.

[88] Lee, C., Bae, K., and Edery, I., "The *Drosophila* CLOCK protein undergoes daily rhythms in abundance, phosphorylation, and interactions with the PER-TIM complex," *Neuron.*, Vol. 21, 1998, pp. 857–867.

[89] Glossop, N., Lyons, L. C., and Hardin, P. E., "Interlocked feedback loops within the *Drosophila* circadian oscillator," *Science*, Vol. 286, 1999, pp. 766–768.

[90] Goldbeter, A., "A model for circadian oscillation in the *Drosophila* period protein (PER)," *Proc. R. Soc. B*, Vol. 261, 1995, pp. 319–324.

[91] Scheper, T. O., Klinkenberg, D., van Pelt, J., and Pennartz, C., "A model of molecular circadian clocks: multiple mechanisms for phase shifting and a requirement for strong nonlinear interactions." *J. Biol. Rhythms.*, Vol. 14, 1999, pp. 213–220.

[92] Tyson, J. J., Hong, C. I., Thron, C. D., and Novak, B., "A simple model of circadian rhythms based on dimerization and proteolysis of PER and TIM," *Biophys. J.*, Vol. 77, 1999, pp. 2411–2417.

[93] Leloup, J. C. and Goldbeter, A., "Chaos and birhythmicity in a model for circadian oscillations of the PER and TIM in *Drosophila*," *J. Theo. Biol.*, Vol. 198, 1999, pp. 445–459.

[94] Petri, B. and Stengl, M., "Phase response curves of a molecular model oscillator: implications for mutual coupling of paired oscillators," *J. Biol. Rhythms*, Vol. 16, 2001, pp. 125–141.

[95] Ueda, H. R., Hagiwara, M., and Kitano, H., "Robust oscillations within the interlocked feedback model of *Drosophila* circadian rhythm," *J. Theo. Biol.*, Vol. 210, 2001, pp. 401–406.

[96] Gonze, D., Halloy, J., and Goldbeter, A., "Robustness of circadian rhythms with respect to molecular noise," *Proc. Natl. Acad. Sci. USA*, Vol. 99, 2002, pp. 673–678.

[97] Gonze, D., Halloy, J., and Goldbeter, A., "Stochastic models for circadian oscillations: emergence of a biological rhythm," *International J. Quantum Chem.*, Vol. 98, 2004, pp. 228–238.

[98] Calander, N., "Propensity of a circadian clock to adjust to the 24h day-night light cycle and its sensitivity to molecular noise," *J. Theor. Biol.*, Vol. 241, 2006, pp. 716–724.

[99] Miura, S., Shimokawa, T., and Nomura, T., "Stochastic simulation on a model of circadian rhythm generation," *Biosystems.*, Vol. 93, 2008, pp. 133–140.

[100] Li, Q. and Lang, X., "Internal noise-sustained circadian rhythms in a *Drosophila* model," *Biophys. J.*, Vol. 94, 2008, pp. 1983–1994.

[101] Cai, X., "Exact stochastic simulation of coupled chemical reactions with delays." *J. Chem. Phys.*, Vol. 126, 2007, pp. 124108.

[102] Edery, I., Zwiebel, L. J., Dembinska, M. E., and Rosbash, M., "Temporal phosphorylation of the *Drosophila* period protein." *Proc. Natl. Acad. Sci. USA*, Vol. 91, 1994, pp. 2260–2264.

[103] Ewer, J., Frisch, B., Hamblen-Coyle, M. J., Rosbash, M., and Hall, J. C., "Expression of the period clock gene within different cell types in the brain of *Drosophila* adults and mosaic analysis of these cells' influence on circadian behavioral rhythms," *J. Neurosci.*, Vol. 12, 1992, pp. 3321–3349.

[104] Xie, Z. and Kulasiri, D., "Modelling of circadian rhythms in *Drosophila* incorporating the interlocked PER/TIM and VRI/PDP1 feedback loops." *J. Theo. Biol.*, Vol. 245, 2006, pp. 290–304.

[105] Hardin, P. E., Hall, J. C., and Rosbash, M., "Circadian oscillations in period gene mRNA levels are transcriptionally regulated." *Proc. Natl. Acad. Sci. USA*, Vol. 89, 1992, pp. 11711–11715.

[106] Lin, Y., Han, M., Shimada, B., Wang, L., Gibler, T. M., Amarakone, A., Awad, T. A., Stormo, G. D., Gelder, R. N. V., and Taghert., P. H., "Influence of the period-dependent circadian clock on diurnal, circadian, and aperiodic gene expression in *Drosophila* melanogaster." *Proc. Natl. Acad. Sci. USA*, Vol. 99, 2002, pp. 9562–9567.

[107] Spinner, D. S., Liu, S., Wang, S., and Schmidt, J., "Interaction of the myogenic determination factor myogenin with E12 and a DNA target: mechanism and kinetics." *J. Mol. Biol.*, Vol. 317, 2002, pp. 431–445.

[108] So, W. V. and Rosbash, M., "Post-transcriptional regulation contributes to *Drosophila* clock gene mRNA cycling." *EMBO J.*, Vol. 16, 1997, pp. 7146–7155.

[109] Vetterli, M. and Kovacevic, J., *Wavelets and Subband Coding*, Prentice Hall, 1995.

[110] Sehgal, A., Rothenfluh-Hilfiker, A., Hunter-Ensor, M., Chen, Y., Myers, M. P., and Young, M. W., "Rhythmic expression of timeless: a basis for promoting circadian cycles in period gene autoregulation." *Science*, Vol. 270, 1995, pp. 808–810.

[111] Bae, K., Lee, C., Sidote, D., Chuang, K. Y., and Edery, I., "Circadian regulation of a *Drosophila* homolog of the mammalian Clock gene: PER and TIM function as positive regulators." *Mol. Cell. Biol.*, Vol. 18, 1998, pp. 6142–6151.

[112] Alon, U., *An Introduction to Systems Biology: Design Principles of Biological Circuits*, Chapman & Hall/Crc Press, 2006.

[113] Leloup, J. C. and Goldbeter, A., "A model for circadian rhythms in *Drosophila* incorporating the formation of a complex between the PER and TIM proteins," *J. Biol. Rhythms.*, Vol. 13, 1998, pp. 70–87.

[114] Leloup, J. C. and Goldbeter, A., "Modeling the molecular regulatory mechanism of circadian rhythms in *Drosophila*," *Bioessays.*, Vol. 22, 2000, pp. 84–93.

[115] Gonze, D., Leloup, J. C., and Goldbeter, A., "Theoretical models for circadian rhythms in *Neurospora* and *Drosophila*," *C. R. Acad. Sci. III.*, Vol. 323, 2000, pp. 57–67.

[116] Myers, M. P., Wager-Smith, K., Rothenfluh-Hilfiker, A., and Young, M. W., "Light-induced degradation of TIMELESS and entrainment of the *Drosophila* circadian clock." *Science*, Vol. 271, 1996, pp. 1736–1740.

[117] Zeng, H., Qian, Z., Myers, M. P., and Rosbash, M., "A light-entrainment mechanism for the *Drosophila* circadian clock." *Nature*, Vol. 380, 1996, pp. 129–135.

[118] Forger, D. B. and Peskin, C. S., "Stochastic simulation of the mammalian circadian clock," *Proc. Natl. Acad. Sci. USA*, Vol. 102, 2005, pp. 321–324.

[119] Bratsun, D., Volfson, D., Tsimring, L. S., and Hasty, J., "Delay-induced stochastic oscillations in gene regulation," *Proc. Natl. Acad. Sci. USA*, Vol. 102, 2005, pp. 14593–14598.

[120] Barrio, M., Burrage, K., Leier, A., and Tian, T., "Oscillatory regulation of Hes1: Discrete stochastic delay modelling and simulation," *PLoS Comput Biol.*, Vol. 2, 2006, pp. e117.

[121] Ribeiro, A., Zhu, R., and Kauffman, S. A., "A general modeling strategy for gene regulatory networks with stochastic dynamics," *J. Comput Biol.*, Vol. 13, 2006, pp. 1630–1639.

[122] Leloup, J.-C., "Circadian clocks and phosphorylation: Insights from computational modeling," *Cent. Eur. J. Biol.*, Vol. 4, No. 3, 2009, pp. 290–303.