STOCHASTIC MODELING OF BIOCHEMICAL SYSTEMS WITH FILTERING AND SMOOTHING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MERVE HAKSEVER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
SCIENTIFIC COMPUTING

SEPTEMBER 2019

Approval of the thesis:

STOCHASTIC MODELING OF BIOCHEMICAL SYSTEMS WITH FILTERING AND SMOOTHING

submitted by **MERVE HAKSEVER** in partial fulfillment of the requirements for the degree of **Master of Science in Scientific Computing Department, Middle East Technical University** by,

Prof. Dr. Ömür Uğur _____
Director, Graduate School of **Applied Mathematics**

Assist. Prof. Dr. Hamdullah Yücel _____
Head of Department, **Scientific Computing**

Prof. Dr. Ömür Uğur _____
Supervisor, **Scientific Computing, METU**

Assoc. Prof. Dr. Derya Altıntan _____
Co-supervisor, **Department of Mathematics, Selçuk University**

Examining Committee Members:

Prof. Dr. Vilda Purutçuoğlu
Statistics Department, METU _____

Prof. Dr. Ömür Uğur
Institute of Applied Mathematics, METU _____

Prof. Dr. Gerhard-Wilhelm Weber
Faculty of Engineering Management,
Poznan University of Technology _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: MERVE HAKSEVER

Signature :

ABSTRACT

STOCHASTIC MODELING OF BIOCHEMICAL SYSTEMS WITH FILTERING AND SMOOTHING

Haksever, Merve
M.S., Department of Scientific Computing
Supervisor : Prof. Dr. Ömür Uğur
Co-Supervisor : Assoc. Prof. Dr. Derya Altıntan

September 2019, 80 pages

Deterministic modeling approach is the traditional way of analyzing the dynamical behavior of a reaction network. However, this approach ignores the discrete and stochastic nature of biochemical processes. In this study, modeling approaches, stochastic simulation algorithms and their relationships to each other are investigated. Then, stochastic and deterministic modeling approaches are applied to biological systems, Lotka-Volterra prey-predator model, Michaelis-Menten enzyme kinetics and JACK-STAT signaling pathway. Also, numerical solutions for ODE system and realizations obtained through stochastic simulation algorithms are compared.

In general, it is not possible to assess all elements of the state vector of biochemical systems. Hence, some statistical models are used to obtain the best estimation. Filtering and smoothing distributions can be obtained via Bayes' rule. However, as an alternative to approximate these distributions Monte Carlo methods might be used. In the second part, bootstrap particle filter algorithm is derived and applied to birth-death process. Estimated probability distribution functions are compared according to number of particles used.

Keywords: mathematical modeling, simulation algorithms, filtering, smoothing

ÖZ

BİYOKİMYASAL SİSTEMLERİN STOKASTİK MODELLEMESİ İLE FİLTRELEME VE YUMUŞATMA ALGORİTMALARI

Haksever, Merve
Yüksek Lisans, Bilimsel Hesaplama Bölümü
Tez Yöneticisi : Prof. Dr. Ömür Uğur
Ortak Tez Yöneticisi : Doç. Dr. Derya Altıntan

Eylül 2019, 80 sayfa

Deterministik modelleme yaklaşımı reaksiyon ağlarının dinamiklerini analiz etmede geleneksel yöntemidir. Buna rağmen bu yaklaşım biyokimyasal süreçlerin ayrık ve stokastik doğasını görmezden gelir. Bu çalışmada modelleme yaklaşımları, stokastik modelleme algoritmaları ve bunların birbirleri ile olan ilişkisi incelendi. Stokastik ve deterministik modelleme yaklaşımları Lotka-Volterra av-avcı modeli, Michaelis-Menten enzim kinetiği ve JAK-STAT sinyal yoluna uygulandı. Ayrıca ODE sistemin numerik çözümleri ve stokastik simulasyon algoritması ile elde edilen gerçekleştirmeler karşılaştırıldı.

Biyokimyasal süreçlerin durum vektörünün bütün elemanlarının belirlenmesi genellikle mümkün değildir. Bu yüzden bazı istatiksel modeller en iyi tahmini elde etmek için kullanılır. Filtreleme ve yumuşatma dağılımları Bayes' kuralı ile bulunabilir. Buna rağmen alternatif olarak Monte Carlo yöntemleri bu dağılımları yaklaşık olarak bulmakta kullanılabilir. İkinci kısımda önyüklemeli parçacık filtreleme algoritması türetildi ve doğum-ölüm süreçleri modeline uygulandı. Tahmin edilen olasılık dağılım fonksiyonları kullanılan parçacık sayısına göre karşılaştırıldı.

Anahtar Kelimeler: matematiksel modelleme, stokastik simulasyon algoritmaları, filt-

releme, yumuşatma

To My Family

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor Prof. Dr. Ömür Uğur for patiently guiding, motivating and encouraging me throughout this work. I also would like to thank him for introducing me to my co-advisor Assoc. Prof. Dr. Derya Altıntan, who has always been there for me whenever I needed, helped me open-heartedly and pushed me to do my best. I thank her from the bottom of my heart.

I would like to thank my committee members Prof. Dr. Vilda Purutçuoğlu and Prof. Dr. Gerhard-Wilhelm Weber for their valuable insights and comments. I also would like to thank Assist. Prof. Dr. Hamdullah Yücel for his valuable insights during my graduate studies.

I would like to express my special gratitude to Fatma Cankara and Fatma İstanbullu for never reading my thesis and depriving me of their valuable comments. Nevertheless, I am very grateful for their support and friendship along the way.

Last but not least, I owe my deepest gratitude to my dear parents, Ferhat and Hiroko Haksever and, my brother Serhat Can Haksever for their unconditional love, encouragement, and support.

I would like to acknowledge the support of the Scientific and Technological Research Council of Turkey (TÜBİTAK) to project with program number 3501 and grant number 11E252.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

$\mathbb{R}$	Set of Real Numbers
$\mathbb{N}$	Set of Natural Numbers
$\mathbb{Z}_0^+$	Set of Positive Integer Numbers
$\mathbb{N}^*$	Set of Natural Numbers without 0
BSPS	Backward Simulation Particle Smoothing
CFPE	Chemical Fokker-Planck Equation
CLE	Chemical Langevin Equation
CME	Chemical Master Equation
CTCM	Continuous Time Markov Chain
EKF	Extended Kalman Filter
FFBS	Forward Filtering-Backward Smoothing
i.i.d	independent and identically distributed
IS	Importance Sampling
KF	Kalman Filter
LV	Lotka-Volterra
ODE	Ordinary Differential Equations
PF	Particle Filter
RRE	Reaction Rate Equations
RTCM	Random Time Change Model
SIS	Sequential Importance Sampling
SMC	Sequential Monte Carlo
SSA	Stochastic Simulation Algorithm
TFS	Two-Filter Smoothing

CHAPTER 1

MODELING OF BIOCHEMICAL SYSTEMS

Modeling can be defined basically as a simplified representation of a real problem. Experimental techniques are usually insufficient or time consuming to analyze complex systems such as analyzing time evolution of a population of organisms. Hence, modeling might be used to predict future behavior of these complex dynamic systems. Nowadays, modeling approaches are a crucial part of scientific fields since they clarify the knowledge of a particular system. Another benefit of modeling is that it allows researchers to validate whether their understanding of a system is correct or not.

There are three types of models which are analogue models, scale models and mathematical models. The analogue models represents the system by another, more comprehensible system. Flow of water in pipes to simulate flow of electricity in wires is an example for the analogue models [32]. The scale models depict a system with another system which is larger or smaller than the real system. For instance, using real cars to understand effect of car accidents might be expensive. Instead, small cars that are minimized according to some rules can be an alternative [18]. The mathematical models describe a real system in terms of mathematical concepts. It establishes relationships between variables and parameters through a set of equations. For example, Lotka-Volterra equations which describe dynamics of a biological system involving a prey and a predator [46].

Mathematical models can be classified based on the solution obtained: (i) Analytical models, (ii) Numerical models. The former reach the exact solution but some simplifications on the model are required. The latter reach an approximate solution by

imposing numerical algorithms on computers. Mathematical models can also be analyzed in two different classes which are deterministic and stochastic. In deterministic models the state variables are determined by non-random parameters and previous states. Therefore, for deterministic model the same initial state gives the same solution. However, deterministic models are usually unstable. Indeed, any small changes in the initial state may lead to large perturbations in the solution. On the other hand, in stochastic models, parameters and states are described by random variables or probability distributions instead of constant values. Hence, at the end many equally likely solutions may be obtained. By this way, it is possible to include uncertainty due to the lack of knowledge of the process or an inaccuracy in measurement [36, 41, 46].

Biochemical processes can also be modeled by one of the ways described above. However, in this thesis main focus will be mathematical models. Biochemical processes are usually represented in terms of chemical reactions. Thus, this chapter starts with the general description of the chemical reaction channels. Then, different mathematical models depending on the representation of the abundance of species to analyze the time evolution of the state vector in a reaction network will be introduced.

1.1 Chemical Reactions

A biochemical reaction network is a composition of different types of species interacting with each other via distinct reaction channels. These networks can place within a container or a cell. Molecules of species move in the container or the cell and collide with each other due to their gained energy. These collisions may end with a chemical reaction depending on the orientation and the kinetic energy of each molecules of species. Chemical reactions may change the amount of molecules of the species or transform species into other species [24, 39].

For a formal description of a chemical reaction system, we will consider a system consisting of n_s different types of chemical species, $\{S_1, S_2, \ldots, S_{n_s}\}$, and n_r distinct chemical reactions, $\{R_1, R_2, \ldots, R_{n_r}\}$. Each chemical reaction is associated with a parameter called the *deterministic reaction rate constant,* $k_j \in \mathbb{R}^+, j = 1, 2, \ldots, n_r$. The reaction rate constant is specific for each reaction and depends on the tempera-

ture of the environment that the reaction takes place and presence of catalyst [39]. Moreover, it will be assumed that the chemical reaction system satisfies two special conditions: (i) The system is well-stirred in a container with constant volume V. (ii) The system is thermally equilibrated with a constant temperature T. The first condition guarantees that the position of a randomly selected molecule of each species is uniformly distributed through the container. The second condition guarantees that the molecules of each species have a Maxwell-Boltzman velocity distribution [24]. To sum up, a reaction channel R_j, $j = 1, 2, \ldots, n_r$, with reaction rate constant k_j of a chemical reaction network has the following form:

$$R_j : \alpha_1^j S_1 + \alpha_2^j S_2 + \ldots + \alpha_{n_s}^j S_{n_s} \xrightarrow{k_j} \bar{\alpha}_1^j S_1 + \bar{\alpha}_2^j S_2 + \ldots + \bar{\alpha}_{n_s}^j S_{n_s}, \tag{1.1}$$

where $\alpha_i^j \in \mathbb{Z}_0^+$, $\bar{\alpha}_i^j \in \mathbb{Z}_0^+$ are *reactant, product coefficients* of ith species S_i, $i = 1, 2, \ldots, n_s$, respectively. The reactant coefficient denotes the number of reactant molecules consumed with a single occurrence of the reaction channel R_j. The product coefficient denotes the number of molecules of species produced at a single occurrence of the reaction channel R_j. In biochemical reaction network, the *stoichiometric element* v_{ij}:

$$v_{ij} \equiv \bar{\alpha}_i^j - \alpha_i^j, \tag{1.2}$$

represents the net change in the number of molecules of S_i species with a single occurrence of reaction channel R_j. Also, *the state vector* of the system at time $t \geq 0$ will be shown as $X(t) = (X_1(t), X_2(t), \ldots, X_{n_s}(t))$, where $X_i(t)$ represents the number of molecules of S_i species, $i = 1, 2, \ldots, n_s$, at time $t \geq 0$. If we are interested in concentration of species instead of the number of molecules of species, then $Z_i(t) = X_i(t)/V$ represents the concentration of each species S_i, $i = 1, \ldots, n_s$.

The four reactions given below are known as *elementary reactions*. These chemical reactions provide representations of complex biochemical processes in a simple way [37].

1. A reaction channel which transforms a molecule of S_1 species into a molecule of S_2 species is represented as follows:

$$R_1 : S_1 \xrightarrow{k_1} S_2.$$

This reaction channel is called an *unimolecular reaction channel* with reaction rate constant k_1.

2. A special kind of unimolecular reactions is *synthesis reaction* which has the form:

$$R_2 : * \xrightarrow{k_2} S_1,$$

with reaction rate constant k_2.

3. A reaction channel which produces a molecule of S_3 species by using a molecule of S_1 species and a molecule of S_2 species is represented as follows :

$$R_3 : S_1 + S_2 \xrightarrow{k_3} S_3.$$

This reaction channel is called a *bimolecular reaction cahnnel* with reaction rate constant k_3.

4. A special kind of bimolecular reactions is *dimerization reaction* which has the form:

$$R_4 : 2\, S_1 \xrightarrow{k_4} S_2,$$

with reaction rate constant k_4.

A chemical reaction which can proceed in both forward and backward direction is called a *reversible reaction*. Reversible reactions can be represented shortly in a single line, i.e.

$$R_5 : S_1 + S_2 \underset{k_5}{\overset{k_3}{\rightleftharpoons}} S_3.$$

Reactions involving more than two species can be written as a combination of these elementary reactions. For instance, trimerisation reaction of species A which has the form:

$$3A \rightarrow A_3,$$

where A_3 is a new species whose each molecule involve three A molecules. This trimerization reaction can be written as a pair of bimolecular reactions as follows:

$$2A \rightarrow A_2, \quad A_2 + A \rightarrow A_3.$$

The above chemical reactions are mostly used to represent chemical reaction channels of chemical reaction network under study.

Until now, reactions are categorized depending on the number of reactant species and elements of a chemical reaction are explained. Another characteristic of a chemical reaction is the rate of reaction. The measure of change in concentration of the reactant/product species in a unit time interval is called the *rate of reaction.* The *rate law* is an equation which relates the rate of reaction and concentration of reactants. Understanding the rate of reaction is the subject of *chemical kinetics* which is the branch of physical chemistry [39].

Chemical kinetics also deals with construction of a mathematical model for a reaction network [46]. It is based on mass-action law which states that the rate of reaction is proportional to the concentrations of reactants [36]. The mathematical model, called reaction rate equation (RRE), is of concern of the deterministic approach. RREs are the set of ordinary differential equations (ODE) in which each equation is set through the rate laws of reactions.

1.2 Deterministic Approach

Traditionally, dynamical behavior of a reaction network is analyzed by deterministic modeling approach. In this approach, the abundance of each species S_i, for some $i = 1, 2, \ldots, n_s$, at time $t \geq 0$ is represented by real-valued concentrations $Z_i(t)$. Time evolution of concentration of all species in the system of interest is described

by a set of ODE in the following form:

$$\begin{aligned}
&\frac{d}{dt}Z_1(t) = f_1(Z_1, \ldots, Z_{n_s}),\\
&\frac{d}{dt}Z_2(t) = f_2(Z_1, \ldots, Z_{n_s}),\\
&\vdots\\
&\frac{d}{dt}Z_{n_s}(t) = f_{n_s}(Z_1, \ldots, Z_{n_s}),
\end{aligned} \tag{1.3}$$

subject to the initial conditions

$$Z_1(0) = z_1, \quad Z_2(0) = z_2, \quad \ldots, \quad Z_{n_s}(0) = z_{n_s},$$

where the functions f_i, $i = 1, 2, \ldots, n_s$, depend on the concentrations of reactants and reaction rate constants of reaction channels involving species S_i. The ODE system given in (1.3) is RRE [28]. If the system of interest involves many species and/or the f_i are nonlinear, then it can be difficult to obtain analytical solutions. Therefore, numerical solution methods, such as Euler method, Runge-Kutta methods, that iteratively compute the approximate solution, starting from the initial value, may be used [30].

Despite the extensive usage of deterministic models to analyze dynamical behavior of a reaction network, it has some drawbacks. For example, the abundance of molecules of species in this approach is represented by real-valued concentrations, however, using this kind of representation can be inappropriate when the number of molecules of species are so low. Also, in this approach, it is not possible to trace exact time of occurrence of one reaction. Moreover, biochemical reactions cannot be mechanically isolated which means that some external noise may affect the environment of the chemical reaction network. Deterministic modeling approach ignores aforementioned discrete and stochastic nature of a chemical reaction network. Therefore, the deterministic models need to be extended to stochastic models [28].

1.3 Stochastic Approach

A *stochastic process* is a system such that the state of the system evolves with time randomly. It can be simply represented as a set of random variables [36]. A *Markov*

process is a stochastic process with the property that the future state of the system only depends on the current state of it [46]. In this section, it will be shown that how we can model the dynamical behavior of a reaction channel by using Markov processes.

There are two necessary quantities to characterize a reaction channel R_j, $j = 1, 2, \ldots, n_r$, in stochastic approach.

The first quantity is the stoichiometric vector $v_j = (v_{1j}, \ldots, v_{n_s j})$, each component of which is stoichiometric element, v_{ij}, defined in (1.2). Accordingly, if $X(t) = x$ is the current state of the system at time t, then $x + v_j$ is the updated state immediately after a single occurrence of the reaction R_j .

The second quantity is the propensity function $a_j(x)$. Propensity function is a quantity to express the probability of occurrence of a specific reaction for the given state of the system. According to the mass-action kinetics, the propensity function for jth reaction channel R_j is proportional to the reaction rate constant and the number of the number of distinct combination of reactants [37]. It can be written as as follows:

$$a_j(x) = c_j h_j(x), \tag{1.4}$$

where c_j is the *stochastic reaction rate constant* for the reaction R_j. Stochastic reaction rate constant c_j is not the same as the reaction rate constant k_j which is defined in the deterministic approach. The reason for this is that in deterministic approach, the amount of species are represented in terms of concentration, mole per liter. On the other hand, in stochastic approach it is an integer which represents the number of molecules of each species. Table 1.1 summarizes the relationship between the deterministic reaction rate constant and the stochastic reaction rate constant for some elementary reactions. Here, n_A is the Avagadro's constant. (Details of conversion from deterministic reaction rate constant to stochastic reaction rate constant we refer to [37, 46].)

By using stochastic reaction rate constant the following probability is defined [24, 28]:

$c_j dt \equiv$ the probability that a randomly selected pair of reactant molecules of species will react in accordance with R_j reaction during an infinitesimal time interval $[t, t + dt)$.

Table 1.1: Rate Constant Conversion

Reaction	Stochastic Rate Constant
$R_1 : S_1 \xrightarrow{k_1} S_2$	$c_1 = k_1$
$R_2 : * \xrightarrow{k_2} S_1$	$c_2 = n_A V k_2$
$R_3 : S_1 + S_2 \xrightarrow{k_3} S_3$	$c_3 = k_3/n_A V$
$R_4 : 2S_1 \xrightarrow{k_4} S_2$	$c_4 = 2k_4/n_A V$

Further, in (1.4), $h_j(x)$ gives the current number of the distinct combination of molecules of reactant species which can be found as

$$h_j(x) = \prod_{i=1}^{n_s} \binom{x_i}{\alpha_i^j},$$

where α_i^j is the reactant coefficient of species S_i as in (1.1). As a result, the following definition for $a_j(x)dt$ can be inferred:

$a_j(x)dt \equiv$ the probability that for given $X(t) = x$, one R_j reaction will occur in the container with volume V during an infinitesimal time interval $[t, t + dt)$.

This is also known as the *fundamental premise of the stochastic formulation of chemical kinetics* [22].

Table 1.2 shows the propensity functions based on mass-action kinetics and stoichiometric vectors for elementary reactions.

Table 1.2: Propensity functions and stoichiometric vectors for given reactions

Reaction	Propensity	Stoichiometric Vector
$R_1 : S_1 \xrightarrow{c_1} S_2$	$a_1(x) = c_1 x_1$	$v_1 = (-1, 1)^T$
$R_2 : * \xrightarrow{c_2} S_1$	$a_2(x) = c_2$	$v_2 = (1, 0)^T$
$R_3 : S_1 + S_2 \xrightarrow{c_3} S_3$	$a_3(x) = c_3 x_1 x_2$	$v_3 = (-1, -1, 1)^T$
$R_4 : 2S_1 \xrightarrow{c_4} S_2$	$a_4(x) = c_4 \frac{1}{2} x_1 (x_1 - 1)$	$v_4 = (-2, 1)^T$

Apart form mass-action kinetics, there are complex reaction kinetics approaches to express the propensity function of a reaction channel. For example, methods based on Michaelis-Menten kinetics [42] and Hill kinetics [44] can be used to approximate propensity functions in stochastic modeling. However, in these representations, propensity function $a_j(x)$, $j = 1, 2, \ldots, n_r$, has nonlinear dependency on the number

of molecules of chemical species and it contains more than one reaction rate constants. Hence, these methods are not efficient as mass-action kinetics.

In conclusion, let the probability of occurrence of R_j reaction in the time interval $[t, t+dt)$ for given initial state be denoted by $p(X(t+dt) = x + v_j, t+dt | X(t) = x, t)$. Then, definition of $a_j(x)dt$ will lead

$$p(x + v_j, t + dt|x, t) = a_j(x)dt + o(dt),$$

where $o(dt)$ satisfies the condition that $\lim_{dt\to 0} \frac{o(dt)}{dt} = 0$ [22]. In other words, probability of occurrence of more than one reaction in the time interval $[t, t+dt)$ can be ignored when $dt \to 0$.

It can be observed that the propensity function is defined only by the current state x. This means that the propensity function satisfies the condition for being Markov process. Besides, chemical and biochemical processes cannot be considered as discrete processes. Hence, it can be inferred that biochemical reaction networks evolves continuous in time and discrete in space. As a result, the continuous time Markov chains (CTMC) can be used to analyze the dynamics of biochemical system of interest [1]. In this approach the state vector of the stochastic process X satisfies random time change model (RTCM).

1.3.1 Random Time Change Model

In [2], Anderson and Kurtz state that the state of the system can simply be represented as

$$X(t) = X(0) + \sum_{j=1}^{n_r} C_j(t) v_j, \tag{1.5}$$

where $C_j(t)$, $j = 1, 2, \ldots, n_r$ is the number of occurrence of reaction R_j up to time t and $X(0)$ is the initial state of the system.

In this study, it is also shown that counting process $C_j(t)$, $j = 1, 2, \ldots, n_r$, is a Poisson process with the rate $a_j(X(t))$. Therefore, $C_j(t)$ can be written as

$$C_j(t) = \xi_j\left(\int_0^t a_j(X(s))ds \right), \tag{1.6}$$

where $\xi_j(\lambda)$ represents the Poisson process with parameter λ. So, the state vector given in (1.5) can be rewritten in the following form:

$$X(t) = X(0) + \sum_{j=1}^{n_r} \xi_j\bigg(\int_0^t a_j(X(s))ds \bigg) v_j. \tag{1.7}$$

This representation of the state vector is known as RTCM [47].

In addition to the state vector representation, it is also possible to obtain the time derivative of the probability mass function of the reaction system in stochastic modeling. In the following section details of this derivation will be given.

1.3.2 Chemical Master Equation

Based on the stochasticity of biochemical reaction network, the probability mass function $p(x, t|x_0, t_0)$ for the stochastic process X is defined as follows:

$$\begin{aligned} p(x, t|x_0, t_0) \equiv &\text{ the probability that the state of the system at given time } t \\ &\text{is } X(t) = x \text{ for the given initial state } X(t_0) = x_0 \end{aligned} \tag{1.8}$$

This conditional probability mass function (1.8) is called as the *grand probability function* [24]. To represent the time evolution of this grand probability function, we will find an expression for $p(x, t + dt|x_0, t_0)$ as follows:

$$\begin{aligned} p(x, t + dt|x_0, t_0) = &\, p(x, t|x_0, t_0)\bigg[1 - \sum_{j=1}^{n_r} a_j(x)dt + o(dt)\bigg] \\ &+ \sum_{j=1}^{n_r} p(x - v_j, t|x_0, t_0)\bigg[a_j(x - v_j)dt + o(dt)\bigg] \\ &+ o(dt). \end{aligned} \tag{1.9}$$

In [24], it is proved that the first term in the right handside of (1.9) corresponds to the probability of no reaction will occur in the time interval $[t, t + dt)$, while the second term corresponds to the probability of occurrence of exactly one reaction in the time interval $[t, t + dt)$. The last term represents the probability of occurrence of more than one reactions in the time interval $[t, t + dt)$. Since these three cases are mutually exclusive events, they can be all summed up by the addition law of probability. If

$p(x,t|x_0,t_0)$ is subtracted from both side of (1.9), and limit is taken for $dt \to 0$, the following Chemical Master Equation (CME) is obtained:

$$\frac{\partial}{\partial t}p(x,t|x_0,t_0) = \sum_{j=1}^{n_r}\Big[a_j(x-v_j)p(x-v_j,t|x_0,t_0) - a_j(x)p(x,t|x_0,t_0)\Big], \quad (1.10)$$

with the initial conditions

$$p(x,t=t_0|x_0,t_0) = \begin{cases} 1 & \text{if } x = x_0, \\ 0 & \text{if } x \neq x_0. \end{cases}$$

Theoretically, analytical solution of CME given in (1.10) gives exact description of Markov process X. However, it is difficult to solve the CME even with the numerical methods except for some particular cases. The reason of this difficulty is called "the curse of dimensionality". It means that the computational cost grows exponentially with the number of reacting species or the number of molecules of each species in the chemical reaction network [48]. So, stochastic simulation algorithms (SSA), in which the realizations of the biochemical system satisfying the CME of interest are obtained, as an alternative [1, 26, 28].

Although deterministic and stochastic approaches are based on different foundation, there is relationship between them [46]. In fact, they are exactly the same in case of unimolecular reactions. The rate of change of the expected value of state vectors at time $t \geq 0$ can be written as follows:

$$\frac{d}{dt}E[X(t)] = \frac{\partial}{\partial t}\sum_x xp(x,t|x_0,t_0) = \sum_x x\frac{\partial}{\partial t}p(x,t|x_0,t_0), \quad (1.11)$$

where $E[\cdot]$ denotes mean of X at time t. Also, x represents the realization, support of the random variable X.

Inserting (1.10) into (1.11) leads

$$\frac{d}{dt}E[X(t)] = \sum_x x\sum_{j=1}^{n_r}\Big[a_j(x-v_j)p(x-v_j,t|x_0,t_0) - a_j(x)p(x,t|x_0,t_0)\Big]. \quad (1.12)$$

Since changing the order of summation does not affect the final result, (1.12) can be written as:

$$\frac{d}{dt}E[X(t)] = \sum_{j=1}^{n_r}\Big[\sum_x xa_j(x-v_j)p(x-v_j,t|x_0,t_0) - \sum_x xa_j(x)p(x,t|x_0,t_0)\Big]. \quad (1.13)$$

By shifting x to $x + v_j$ we obtain

$$\sum_x x a_j(x - v_j) p(x - v_j, t|x_0, t_0) = \sum_x (x + v_j) a_j(x) p(x, t|x_0, t_0).$$

Here, it must be noted that $x + v_j \in X$. As a result, (1.13) can be written in the following form:

$$\begin{aligned}
\frac{d}{dt}E[X(t)] &= \sum_{j=1}^{n_r} \left[\sum_x (x + v_j) a_j(x) p(x, t|x_0, t_0) - \sum_x x a_j(x) p(x, t|x_0, t_0) \right] \\
&= \sum_{j=1}^{n_r} \Big[E[(X(t) + v_j) a_j(X(t))] - E[X(t) a_j(X(t))] \Big] \\
&= \sum_{j=1}^{n_r} E[v_j a_j(X(t))] \\
&= \sum_{j=1}^{n_r} v_j E[a_j(X(t))].
\end{aligned} \tag{1.14}$$

If all the reactions of the system under consideration are unimolecular then, corresponding propensity functions will be linear which in turn will give us $E[a_j(X(t))] = a_j(E[X(t)])$. This will give us the possibility to write the last line of (1.14) as follows:

$$\frac{d}{dt}E[X(t)] = \sum_{j=1}^{n_r} v_j a_j(E[X(t)]). \tag{1.15}$$

Let $Y(t) = E[X(t)]$. Then, (1.15) will take the form [28]:

$$\frac{d}{dt}Y(t) = \sum_{j=1}^{n_r} v_j a_j(Y(t)), \tag{1.16}$$

which is the same as the ODE system for deterministic model. However, it should be noted that in (1.16), RRE uses the number of molecules of species and stochastic rate constant instead of concentrations and deterministic rate constant, respectively. To be brief, rate constant will remain the same if all chemical reaction channels have similar form with R_1 given in Table 1.1. Hence, the deterministic formulation and the expected value of stochastic formulation will be exactly the same. Otherwise, rate constant conversions are required according to Table 1.1.

Up to now, dynamics of biochemical reaction networks analyzed through RTCM and CME provided that the abundance of species is low and represented by integer values.

On the other hand, in case of tremendous amount of species, RRE in terms of concentrations used to analyze dynamics of biochemical reaction systems. Yet, stochastic approach is not restricted to low copy number of species. In case of stochasticity, Langevin equation provides an alternative model to biochemical reaction networks with large amount of species.

1.3.3 Chemical Langevin Equation

Diffusion process is a Markov process which evolves continuously in time with continuous states. *Langevin equation* is a particular type of stochastic differential equations which is used to express dynamics of diffusion processes [46]. If the number of molecules of species in a biochemical system is excessively high, then representation of the amount of species in terms of integer-valued molecule numbers will be inappropriate. Hence, the amount of molecules of species is represented by real valued concentration. Accordingly, the process X can be defined by diffusion process. Let $Z_i(t) = X_i(t)/V$ denote the concentration of species S_i, $i = 1, 2, \ldots, n_s$ at time $t \geq 0$. Dividing both sides of (1.7) by the volume V yields

$$Z(t) = Z(0) + \frac{1}{V}\sum_{j=1}^{n_r} \xi_j \bigg(\int_0^t a_j(X(s))ds \bigg) v_j. \tag{1.17}$$

For unimolecular and bimolecular reactions involving species with high number of molecules, the propensity functions can be written as $a_j(X(t)) = V\tilde{a}_j(Z(t))$. Here, $\tilde{a}_j(Z(t))$ represents the propensity function computed by using the corresponding deterministic rate constant k_j and the concentration of species [1, 2]. Hence,(1.17) can be written as

$$Z(t) = Z(0) + \frac{1}{V}\sum_{j=1}^{n_r} \xi_j \bigg(V \int_0^t \tilde{a}_j(Z(s))ds \bigg) v_j. \tag{1.18}$$

Adding and subtracting the term $\bigg(V \int_0^t \tilde{a}_j(Z(s))ds \bigg) v_j$ to the right hand side of (1.18) yields

$$\begin{aligned} Z(t) = Z(0) + \frac{1}{V}\sum_{j=1}^{n_r} \bigg[\xi_j \bigg(V \int_0^t \tilde{a}_j(Z(s))ds \bigg) - V \int_0^t \tilde{a}_j(Z(s))ds \bigg] v_j \\ + \sum_{j=1}^{n_r} \bigg(\int_0^t \tilde{a}_j(Z(s))ds \bigg) v_j. \end{aligned} \tag{1.19}$$

By central limit theorem [33], which states that $\frac{\xi_j(V\lambda) - V\lambda}{\sqrt{V}}$ converges to $W_j(\lambda)$ for for large V, (1.19) can be rewritten as follows:

$$Z(t) = Z(0) + \frac{1}{\sqrt{V}} \sum_{j=1}^{n_r} W_j\bigg(\int_0^t \tilde{a}_j(Z(s))ds \bigg) v_j + \sum_{j=1}^{n_r} \bigg(\int_0^t \tilde{a}_j(Z(s))ds \bigg) v_j, \tag{1.20}$$

where $W_j(\cdot)$ represents the standard Brownian motion.

In fact, (1.20) is the solution of the following stochastic differential equation [5]:

$$dZ_i(t) = \frac{1}{\sqrt{V}} \sum_{j=1}^{n_r} v_{ji} \sqrt{\tilde{a}_j(Z(t))} dW_j(t) + \sum_{j=1}^{n_r} v_{ji} \tilde{a}_j(Z(t))dt, \tag{1.21}$$

which is called the Chemical Langevin Equation (CLE) [27, 28]. The first term on the right handside of the equation (1.21) is *the stochastic component* or *the system noise*, and the second term is *the deterministic component* of the stochastic differential equation [14].

It can be seen that the only difference between RRE given in (1.3) and CLE given in (1.21) is the stochastic component. If the number of molecules of species and the volume of the container approach to infinity while the concentration of species remain constant, in other words, when thermodynamic limit condition [28] is satisfied, then the stochastic term approaches to zero. As a result, CLE can be transformed to the following RRE:

$$\frac{d}{dt} Z(t) = \sum_{j=1}^{n_r} v_{ji} \tilde{a}_j(Z(t)). \tag{1.22}$$

In summary, if the reaction system is modeled by using the stochastic approach which assumes that the dynamics of the reaction system is modeled by discrete state and continuous time, then the state vector of the system satisfies RTCM given in (1.5). Further, time evolution of corresponding probability mass function is represented by CME given in (1.10). If the dynamics of the reaction system is modeled by diffusion process, then the state vector of the reaction system satisfies CLE given in (1.21). Besides, the time evolution of corresponding probability density function satisfies chemical Fokker-Planck equation (FPE) which will be examined in the next section.

1.3.4 Chemical Fokker-Planck Equation

It is possible to obtain chemical Fokker-Planck equation (CFPE) from CME given in (1.10). However, the following two conditions must be satisfied [27]: (i) The abundance of molecules of each species is defined as real-valued concentrations instead of integer valued molecule numbers. (ii) The functions $g_j(x) \equiv a_j(x)p(x,t|x,t_0)$ for $j = 1, 2 \ldots, n_r$, are smooth functions of x. Then, Taylor's theorem for multivariate functions gives the following:

$$\begin{aligned} g_j(x - v_j) = g_j(x) + \sum_{n=1}^{\infty} \sum_{m_1,m_2,\ldots,m_{n_s}=0}^{n} \frac{1}{m_1!m_2!\ldots m_{n_s}!} \\ \times (-v_{j1})^{m_1} \ldots (-v_{jn_s})^{m_{n_s}} \frac{\partial^n g_j(x)}{\partial x_1^{m_1} \ldots \partial x_{n_s}^{m_{n_s}}}, \end{aligned} \tag{1.23}$$

where $m_1 + m_2 + \ldots + m_{n_s} = n$.

If (1.23) is substituted into the CME given in (1.10), then we obtain

$$\begin{aligned} \frac{\partial}{\partial t}p(x,t|x_0,t_0) &= \sum_{j=1}^{n_r} \sum_{n=1}^{\infty} \sum_{m_1,m_2,\ldots,m_{n_s}=0}^{n} \frac{1}{m_1!m_2!\ldots m_{n_s}} \\ &\quad \times (-v_{j1})^{m_1} \ldots (-v_{jn_s})^{m_{n_s}} \frac{\partial g_j(x)}{\partial x_1^{m_1} \ldots \partial x_{n_s}^{m_{n_s}}} \\ &= \sum_{n=1}^{\infty} (-1)^n \sum_{m_1,m_2,\ldots,m_{n_s}=0}^{n} \frac{1}{m_1!m_2!\ldots m_{n_s}!} \frac{\partial^n}{\partial x_1^{m_1} \ldots \partial x_{n_s}^{m_{n_s}}} \\ &\quad \times \left(\left[\sum_{j=1}^{M} (v_{j1}^{m_1}, \ldots, v_{jn_s}^{m_{n_s}} a_j(x)) \right] p(x,t|x_0,t_0) \right), \end{aligned} \tag{1.24}$$

This is the chemical Kramer-Moyal equation. If the right hand side of (1.24) is truncated at $n = 2$, then the following FPE is obtained [27]:

$$\begin{aligned} \frac{\partial}{\partial t}P(x,t|x_0,t_0) = &- \sum_{i=1}^{n_s} \frac{\partial}{\partial x_i} \left[\left(\sum_{j=1}^{n_r} v_{ji} a_j(x) \right) P(x,t|x_0,t_0) \right] \\ &+ \frac{1}{2} \sum_{i=1}^{n_s} \frac{\partial^2}{\partial x_i^2} \left[\left(\sum_{j=1}^{n_r} v_{ji}^2 a_j(x) \right) P(x,t|x_0,t_0) \right] \\ &+ \sum_{\substack{i,i'=1 \\ i<i'}}^{n_s} \frac{\partial^2}{\partial x_i \partial x_i'} \left[\left(\sum_{j=1}^{n_r} v_{ji} v_{ji'} a_j(x) \right) P(x,t|x_0,t_0) \right]. \end{aligned}$$

So far, mathematical equations modeling biochemical reaction systems are introduced. Since obtaining analytical solutions of these equations are usually not possi-

ble, simulation methods are used to chase the time evolution of the stochastic process under consideration.

1.3.5 Stochastic Simulation Algorithms

SSA can be roughly classified as exact, approximate and hybrid algorithms based on whether being approximation or not to the exact solution, or the combination of different pure approaches.

1.3.5.1 Exact Simulation Methods

Exact simulation algorithms simulate every reaction event and generate independent realizations of the biochemical system. However, this exactness increases the computation time, which is the most crucial disadvantage of the exact methods. The SSAs are based on the *reaction probability function* $p(\tau, j|x, t)$, where τ, j are the random numbers for the time to next firing reaction, the index of corresponding fired reaction, respectively [22, 28]. It is defined as follows:

$$p(\tau, j|x, t)d\tau \equiv \text{given } X(t) = x, \text{ the probability of having next fired reaction } R_j \\ \text{in the time interval } [t + \tau, t + \tau + d\tau).$$

The reaction probability function is a joint probability distribution function of the discrete random variable $j \in \{1, 2, \ldots, n_r\}$ and continuous random variable $0 \leq \tau \leq \infty$. Derivation of a formula for $p(\tau, j|x, t)$ follows probability laws (for details of derivation we refer to [22]):

$$p(\tau, j|x, t) = a_j(x) \exp\big(-a_0(x)\tau\big), \quad a_0(x) = \sum_{j'=1}^{n_r} a_{j'}(x). \tag{1.25}$$

Based on (1.25) two simple stochastic simulation algorithms are proposed by Gillespie [22, 23]. These algorithms are the direct method and the first reaction method. They differ from each other by Monte Carlo step that generates random numbers τ and j .

Direct Method

The direct method is based on the idea that any joint probability density functions can be written as the product of one marginal and one conditional probability density function [22]. The algorithm proceeds as follows:

0. Specify the initial time $t = t_0$ and the initial state of the system $X(t_0) = x$. Also, specify the final time $t_{final} > t$ of the simulation.

1. Calculate the propensity functions $a_j(x)$, $j = 1, 2, \ldots, n_r$ for each reaction channels and the total propensity $a_0(x)$ of the system.

2. Generate the random numbers (τ, j) according to $P(\tau, j|x, t)$ as follows:

   Draw two random numbers r_1 and r_2 from uniform distribution in the unit interval such that

   $$\tau = -\frac{1}{a_0}\ln(r_1),$$

   $$j = \text{the smallest integer satisfying } \sum_{j'=1}^{n_r} a_{j'}(x) > r_2 a_0(x).$$

3. Update the time as $t \to t + \tau$ and state as $x \to x + v_j$.

4. Keep (x, t). If $t < t_{final}$ and the number of molecules of species higher than zero, then return to the step 1.

In direct method, time to the next firing reaction is decided through a uniform random variable. The other approach, the first-reaction method, is theoretically equivalent to the direct method. Yet, in this method, tentative reaction time for each reaction is obtained through uniform random variables. Then, the shortest time, that is, the reaction which occurs first, is chosen.

First-Reaction Method

First-reaction method differs from direct method in Monte Carlo step that generates random variables for τ and j. In the Monte Carlo step of the first-reaction method, n_r random numbers are generated for each reaction channel to compute tentative times to the next reaction. Then, reaction channel which has the earliest firing time is chosen as the next reaction [22]. The algorithm proceeds as follows:

0. Specify the initial time $t = t_0$ and the initial state of the system $X(t_0) = x$. Also, specify the final time $t_{final} > t$ of the simulation.

1. Calculate the propensity functions $a_j(x)$ for each reaction, $j = 1, 2, \ldots, n_r$, and the total propensity $a_0(x)$ of the system.

2. Generate (τ, j) according to $P(\tau, j|x, t)$ as follows:

   Draw n_r random numbers $r_1, r_2, \ldots, r_{n_r}$ from the uniform distribution in the unit interval and

$$\tau_v = -\frac{1}{a_0}\ln(r_v), v = 1, 2, \ldots, n_r,$$
$$\tau = \text{ the smallest } \tau_v,$$
$$j = v \text{ for the smallest } \tau_v.$$

3. Update the time as $t \to t + \tau$ and state as $x \to x + v_j$.

4. Keep (x, t). If $t < t_{final}$ the number of molecules of species higher than zero then return to the step 1.

The first-reaction method generates random variables as many as the number of reaction channels. Hence, it will be less efficient than the direct method when the number of reaction channels in the reaction network is higher than three [22].

Apart from these two methods, there are several improvements to the exact simulation methods. In [21], Gibbson & Bruck proposed the next reaction method to reduce computational complexity of the first reaction method. In [10], Cao et al. proposed modified direct method to reduce computational time of direct method by using dependency graph and sorting propensity functions of chemical reaction.

1.3.5.2 Approximate Simulation Methods

The computational cost of exact methods is very high. Therefore, approximate methods which sacrifice some accuracy to decrease computational cost of exact methods are proposed as an alternative [46]. Approximate methods are based on the idea that

time of the process is divided into small subintervals such that in each time interval all propensity functions of reaction channels are constant. At the end, if the subintervals are short, then more accurate realizations will be obtained. On the contrary, if subintervals are long, then the simulation will proceed faster. In this section, some approximate simulation methods will be explained.

Poisson Timestep Method

In Poisson timestep method, it is assumed that the number of occurrence of reactions in the short time interval has Poisson distribution. The time step Δt is fixed such that propensity functions of each reaction channels are constant during the time interval with length Δt . The algorithm proceeds as follows [46]:

0. Specify the initial time $t = t_0$ and the initial state of the system $X(t_0) = x$ and stoichiometric matrix $S^{n_s \times n_r}$. Here, columns of the stoichiometric matrix are stoichiometric vectors v_j, $j = 1, 2, \ldots, n_r$. Also, specify the final time $t_{final} > t$ of the simulation.
1. Calculate the propensity functions $a_j(x)$, $j = 1, 2, \ldots, n_r$. Generate n_r dimensional reaction vector r such that j-th entry is the Poisson random variable with the parameter $a_j(x)\Delta t$, denoted by $Po(a_j(x)\Delta t)$.
2. Update the time as $t \to t + \Delta t$ and state as $x \to S^{n_s \times n_r} \times r^{n_r \times 1}$
3. Keep (x, t). If $t < t_{final}$ the number of molecules of species higher than zero then return to the step 1.

In Poisson timestep method selecting time increment is not specified. Conditions for the time increment is proposed in Gillespie's tau-leaping method.

Tau-leaping Method

Tau-leaping method is introduced by Gillespie [25].This method is an improvement of the Poisson timestep method. In tau-leaping method, selecting a time increment τ provides a balance between speed and accuracy as follows [36]:

- To increase speed, τ is chosen large enough so that more than one reactions

occurs in time interval
$[t+\tau, t+\tau+d\tau)$.

- To increase accuracy, τ is chosen small enough that changes in the propensity functions of each reaction channels in time interval $[t+\tau, t+\tau+d\tau)$ is not noticeable.

The second assumption is known as *leap condition* [25]. Leap condition ensures that the probability of occurrence of R_j reaction during the time interval dt somewhere inside $[t, t+\tau)$, which is $a_j(x)dt$ as in Definition (1.3), is not affected by other reaction channels. Besides, the number of occurrence of events during specific time interval is defined by *Poisson random variable*. Based on leap condition and the definition of Poisson random variable, number of firing of reaction channel R_j is Poisson random variable with parameter $a_j(x)\tau$.

There are two controversial problems on selecting τ properly: (i) How we can be sure that τ is large enough. (ii)How we can know the selected τ will not cause negative population. For the first problem two refinements are made by Gillespie [9, 29]. Also, to avoid negative populations several strategies have been proposed. For instance, Poisson random numbers are replaced by binomial ones in [7] or reactions are separated as critical and non-critical reactions in [12].

1.3.5.3 Hybrid Methods

Hybrid simulation methods combine exact and approximate simulation methods to decrease computational cost of exact algorithms while including probabilistic nature of biochemical reaction systems [34]. To achieve this goal, chemical species are partitioned into different classes depending on their abundance. As a result, reactions are also partitioned depending on species involved. Different simulation methods are used to generate realizations of the state vector for each group simultaneously. In general, *fast reactions* involve chemical species with high copy number. These reactions are simulated by approximate or deterministic methods. On the other hand, *slow reactions* involve species with low copy number of molecules. These reactions are simulated by exact methods [46]. In this section fundamentals of some hybrid

models will be explained.

Discrete/ODE Methods

In discrete/ODE methods, fast reactions are simulated by ODE system as in Section 1.2. At the same time, slow reactions are simulated by one of the exact methods given above. Moreover, these methods are based on the assumption that step size for the ODE solver is chosen such that any slow reactions do not occur in this time interval [46]. As a result, a generalized algorithm proceeds as follows:

0. Specify the initial time $t = t_0$ and the initial state of the system $X(t_0) = x$. Also, specify the final time $t_{final} > t$ of the simulation.
1. Calculate propensity functions of the slow reactions.
2. Decide an appropriate time step dt to solve ODE system.
3. Solve the ODE system in the time interval $[t, t + dt)$ to find state vector of species with high copy numbers of molecules.
4. Calculate again propensity functions of slow reactions at time $t + dt$. Compare these propensity functions with ones calculated at step 1. Check whether any slow reaction is occurred or not.
5. If there is no slow reaction, then update states of species with high copy number of molecules according to the solution of ODE at time step $t + dt$.
6. If any slow reaction is occurred, then find the time t_s of the earliest one. Update the time as $t \to t + t_s$. Solve the ODE system over the time interval $[t, t + t_s)$. Update the state vector of both group of species to time $t + t_s$.
7. If $t < t_{final}$ then return to step 1. If not, finalize the algorithm.

In [34], Kiehl et al. proposed a hybrid simulation method that uses Gillespie's direct method to update slow reactions and Runge-Kutta method to numerical integration of ODE system that corresponds to fast reactions. As another example, in [38], next-reaction method and 4^{th}- order Runge-Kutta method are combined for slow and fast reactions, respectively.

Maximal Timestep Method

The maximal timestep method is the combination of exact simulation methods mentioned above for slow reactions and the tau-leaping method for the fast reactions. The general algorithm is as follows:

0. Initialize the system time as $t = t_0$ and decide t_{final}.

1. Calculate propensity functions of fast reactions and decide a convenient time step τ.

2. Propensity function need to be constant during the firing of a slow reaction. If τ is an appropriate time step for tau-leaping method, then there should not be any slow reaction during the time interval $[t, t + \tau]$.

3. If no slow reaction occurred then update the system according to decided time step τ.

4. If any slow reaction is occurred, then find the time t_s of the earliest one. Update the time as $t \rightarrow t + t_s$. Update the state of both group of species according to their respective methods.

5. $t < t_{final}$ then return to step 1. If not, finalize the algorithm.

In [40], Puchalka and Kierzek proposed a simulation method that combines tau-leaping and next reaction method. Also, in [8], reaction system is partitioned into three groups and direct method, tau-leaping method and a numerical solution method (Euler Murayama) for stochastic differential equation are used as simulation methods.

1.4 Application

In this section mathematical models that described in previous sections will be applied to Lotka-Volterra prey-predator model, Michaelis-Menten enzyme kinetics and JAK-STAT Pathway.

1.4.1 Lotka-Volterra Prey-Predator Model

Deterministic Model

Lotka-Volterra (LV) model represents interaction between prey and predator in a given environment [46]. The reaction system consists two species, prey and predator represented by X_1 and X_2, respectively. These species interact through the following reaction channels:

$$\begin{aligned} X_1 &\xrightarrow{k_1} 2X_1, \\ X_1 + X_2 &\xrightarrow{k_2} 2X_2, \\ X_2 &\xrightarrow{k_3} \emptyset. \end{aligned}$$

The first reaction channel represents prey reproduction with deterministic reaction rate constant k_1. The second one shows hunted preys by predator with (deterministic) reaction rate constant k_2. The last one is the extinction of predator due to natural causes with deterministic reaction rate constant k_3.

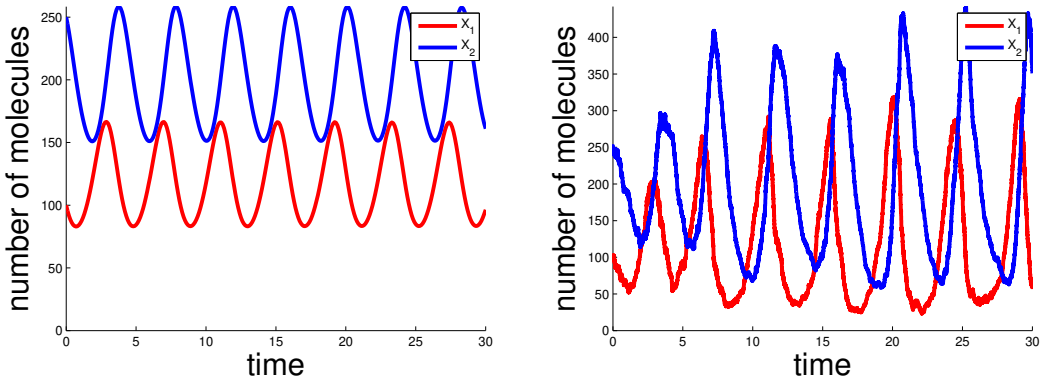
In deterministic modeling of biochemical reaction systems, the abundance of species is usually represented in terms of concentrations. However, in this section molecular approach and stochastic reaction rate constants will be used to see better comparison between deterministic and stochastic approaches for the modeling of biochemical reaction systems.

Let, state vector of LV system at time t is $X(t) = (X_1(t), X_2(t)) = (x_1, x_2) = x$. Initial state vector is chosen as $X(0) = (100, 250)^T$ and stochastic reaction rate constants are chosen as a vector $c = (2, 0.01, 1.2)$. Based on the mass-action kinetics, rate law of the first, third unimolecular reaction channels at time $t \geq 0$ are $c_1X_1(t)$ and $c_3X_2(t)$, respectively. On the other hand, rate law of the second bimolecular reaction $t \geq 0$ channel is $c_2X_1(t)X_2(t)$. According to rate laws, the abundance of species X_1 will increase with rate $c_1X_1(t)$ and decrease with rate $c_2X_1(t)X_2(t)$. Similarly, the abundance of species X_2 will increase with rate $c_2X_1(t)X_2(t)$ and decrease with rate $c_3X_2(t)$. These results will give the following ODE system, namely RRE, which

shows time evolution of state of LV system:

$$\begin{aligned}\frac{d}{dt}X_1(t) &= c_1X_1(t) - c_2X_1(t)X_2(t)\\ \frac{d}{dt}X_2(t) &= c_2X_1(t)X_2(t) - c_3X_2(t).\end{aligned} \tag{1.26}$$

For given initial state of species and rate constants of reaction channels the solution of RRE describes entire dynamics of the system. Figure 1.1a shows the dynamics of Lotka-Volterra model (1.26) with MATLAB function `ode45`.



(a) *Deterministic solution of Lotka-Volterra model by MATLAB function ode45*

(b) *Single realization of SSA with direct method for Lotka-Volterra model.*

Figure 1.1: Dynamics of Lotka Volterra Prey-Predator Model.

Stochastic Model

To characterize LV system in stochastic approach, propensity functions and stoichiometric vectors of each reaction is given in Table 1.3.

Table 1.3: Propensity functions and stoichiometric vectors of Lotka-Volterra system

Reaction	Propensity	Stoichiometric Vector
$X_1 \xrightarrow{c_1} 2X_1$	$a_1(x) = c_1x_1$	$\upsilon_1 = (1,0)^T$
$X_1 + X_2 \xrightarrow{c_2} 2X_2$	$a_2(x) = c_2x_1x_2$	$\upsilon_2 = (-1,1)^T$
$X_2 \xrightarrow{c_3} \emptyset$	$a_3(x) = c_3x_2$	$\upsilon_3 = (0,-1)^T$

The RTCM representation of LV model is given as follows:

$$\begin{aligned}
X_1(t) &= X_1(0) + \sum_{j=1}^{3} \xi_j \bigg(\int_0^t a_j(X(s))ds \bigg) v_{ji} \\
&= 100 + \xi_1 \bigg(\int_0^t c_1 X_1(s) ds \bigg) - \xi_2 \bigg(\int_0^t c_2 X_1(s) X_2(s) ds \bigg), \\
X_2(t) &= X_2(0) + \sum_{j=1}^{3} \xi_j \bigg(\int_0^t a_j(X(s))ds \bigg) v_{ji} \\
&= 250 + \xi_2 \bigg(\int_0^t c_2 X_1(s) X_2(s) ds \bigg) - \xi_3 \bigg(\int_0^t c_3 X_2(s) ds \bigg).
\end{aligned}$$

Then, CME of the LV system can be written as follows:

$$\begin{aligned}
\frac{\partial}{\partial t} p(x_1, x_2; t|x_0; t_0) &= c_1(x_1 - 1)p(x_1 - 1, x_2; t|x_0; t_0) - c_1 x_1 p(x_1, x_2; t|x_0; t_0) \\
&\quad + c_2 p(x_1 + 1, x_2 - 1; t|x_0; t_0) - c_2 x_1 x_2 p(x_1, x_2; t|x_0; t_0) \\
&\quad + c_3 x_1 (x_2 + 1) p(x_1, x_2 + 1; t|x_0; t_0) - c_3 x_3 p(x_1, x_2; t|x_0; t_0).
\end{aligned}$$

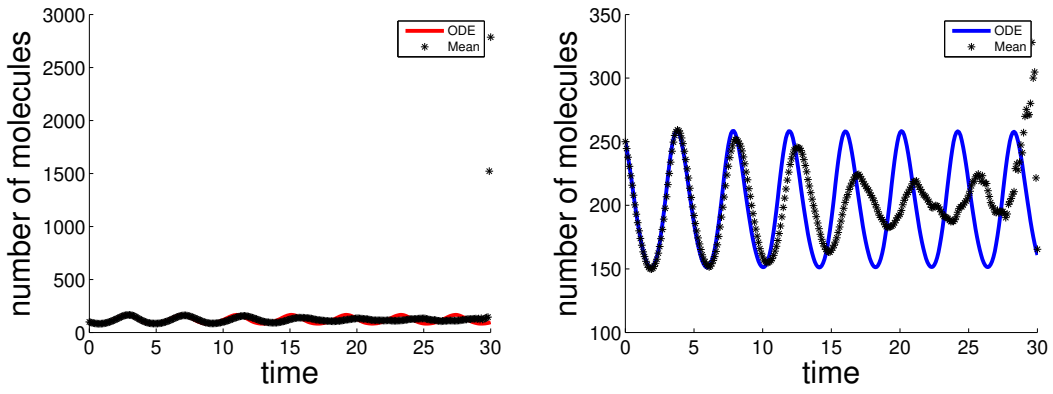
To obtain realizations of the LV system for stochastic modeling, we will use SSA algorithms. In Figure 1.1b, one iteration of Gillespie's stochastic simulation algorithm with direct method can be seen. Further, Figure 1.2 indicates the relationship between expected value of stochastic model and deterministic model. It can be seen that they are not coincide with each other because of existence of bimolecular reaction in Lotka-Volterra system.

Using (1.21), we can obtain CLE for LV system as follows:

$$\begin{aligned}
dZ_1(t) &= \frac{1}{\sqrt{V}} (\sqrt{k_1 X_1(t)} dW_1(t) - \sqrt{k_2 X_1(t) X_2(t)} dW_2(t)) + \sqrt{k_1 X_1(t)} dt - \sqrt{k_2 X_1(t) X_2(t)} dt \\
dZ_2(t) &= \frac{1}{\sqrt{V}} (\sqrt{k_2 X_1(t) X_2(t)} dW_2(t) - \sqrt{k_3 X_2(t)} dW_3(t)) + \sqrt{k_2 X_1(t) X_2(t)} dt - \sqrt{k_3 X_2(t)} dt
\end{aligned}$$

Finally, CFPE representation of Lotka Volterra system becomes

$$\begin{aligned}
&\frac{\partial}{\partial t} p(x_1, x_2; t|x_0; t_0) = -\frac{\partial}{\partial x_1} [(c_1 x_1 - c_2 x_1 x_2) p(x_1, x_2; t|x_0; t_0)] \\
&- \frac{\partial}{\partial x_2} [(c_2 x_1 x_2 - c_3 x_2) p(x_1, x_2; t|x_0; t_0)] + \frac{1}{2} \frac{\partial}{\partial x_1^2} [(c_1 x_1 + c_2 x_1 x_2) p(x_1, x_2; t|x_0; t_0)] \\
&- \frac{\partial}{\partial x_2^2} [(c_2 x_1 x_2 + c_3 x_2) p(x_1, x_2; t|x_0; t_0)] + \frac{\partial^2}{\partial x_1 \partial x_2} [(-c_2 x_1 x_2) p(x_1, x_2; t|x_0; t_0)]
\end{aligned}$$



(a) (b)

Figure 1.2: (a) Red line indicates ODE solution of prey (X_1). Black dots are mean of state vectors for X_1 after 500 iterations of SSA (b) Blue line indicates ODE solution of predator (X_2). Black dots are mean of state vectors for X_2 after 500 iterations of SSA

1.4.2 Michaelis-Menten Enzyme Kinetics

Deterministic Model

Michaelis-Menten enzyme kinetic system consists four chemical species substrate (S), product (P), enzyme (E) and enzyme-substrate (ES) complex. An enzyme is a biological catalyst which speeds up the chemical reaction [36]. These chemical species interact through the following three chemical reaction channels:

$$S + E \xrightarrow{k_1} ES$$
$$ES \xrightarrow{k_2} S + E$$
$$ES \xrightarrow{k_3} P + E$$

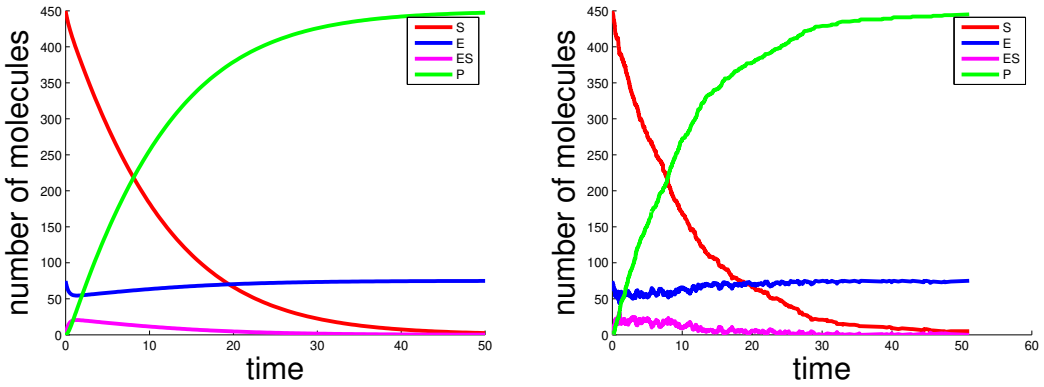
The first reaction represents that substrate S and enzyme E constitute enzyme-substrate complex with deterministic reaction rate constant k_1. In the second reaction ES dissociates to S and E with deterministic reaction rate constant k_2 . In the third reaction ES dissociates to S and P with deterministic reaction rate constant k_3.

As the same reason stated LV model, molecular approach will be used to examine dynamics of MM enzyme kinetics. Let the state vector of MM enzyme kinetic at time t is $X(t) = (S(t), E(t), ES(t), P(t)) = (x_1, x_2, x_3, x_4) = x$. Initial state vector is chosen as $X(0) = (450, 75, 0, 0)^T$ and stochastic reaction rate constants are chosen as

$c = (0.020.51.6)$. According to mass-action kinetics, rate law of the first reaction is $c_1 S(t)E(t)$. Similarly, for unimolecular second and third reactions we get rate laws as $c_2 ES(t)$ and $c_3 ES(t)$, respectively. As a result, for describing the dynamics of Michaelis-Menten enzyme kinetic, we write:

$$\begin{aligned}
\frac{d}{dt}S(t) &= c_1 S(t)E(t) - c_2 ES(t) \\
\frac{d}{dt}E(t) &= -c_1 S(t)E(t) + (c_2 + c_3)ES(t) \\
\frac{d}{dt}ES(t) &= c_1 S(t)E(t) - (c_2 + c_3)ES(t) \\
\frac{d}{dt}P(t) &= c_3 ES(t)
\end{aligned} \tag{1.27}$$

Figure 1.3a depicts the solution of RREs (1.27) of Michaelis Menten enzyme kinetic with MATLAB function `ode45`.



(a) *Deterministic solution of Michaelis-Menten model by MATLAB function ode45*

(b) *Single realization of SSA with direct method for Michaelis-Menten model.*

Figure 1.3: Dynamics of Michaelis-Menten enzyme kinetics.

Stochastic Model

In Table 1.4, propensity functions and stoichiometric vectors of Michaelis-Menten model can be seen.

Table 1.4: Propensity functions and stoichiometric vectors of Michaelis Menten enzyme kinetics

Reaction	Propensity	Stoichiometric Vector
$S + E \xrightarrow{k_1} ES$	$a_1(x) = c_1 x_1 x_2$	$v_1 = (-1, -1, 1, 0)^T$
$ES \xrightarrow{k_2} S + E$	$a_2(x) = c_2 x_3$	$v_2 = (1, 1, -1, 0)^T$
$ES \xrightarrow{k_3} P + E$	$a_3(x) = c_3 x_3$	$v_3 = (0, 1, -1, 1)^T$

RTCM representation of Michaelis-Menten enzyme kinetics is

$$
\begin{aligned}
S(t) &= 450 - \xi_1\bigg(\int_0^t c_1 S(s)E(s)ds\bigg) + \xi_2\bigg(\int_0^t c_2 ES(s)ds\bigg) \\
E(t) &= 75 - \xi_1\bigg(\int_0^t c_1 S(s)E(s)ds\bigg) + \xi_2\bigg(\int_0^t c_2 ES(s)ds\bigg) + \xi_3\bigg(\int_0^t c_2 ES(s)ds\bigg) \\
ES(t) &= \xi_1\bigg(\int_0^t c_1 S(s)E(s)ds\bigg) - \xi_2\bigg(\int_0^t c_2 ES(s)ds\bigg) - \xi_3\bigg(\int_0^t c_2 ES(s)ds\bigg) \\
P(t) &= \xi_3\bigg(\int_0^t c_2 ES(s)ds\bigg)
\end{aligned}
$$

Then, CME of Michaelis-Menten enzyme kinetics is

$$
\begin{aligned}
\frac{\partial}{\partial t}p(x;t|x_0;t_0) =& c_1(x_1+1)(x_2+1)p(x_1+1,x_2+1,x_3,x_4;t|x_0;t_0) \\
&- c_1x_1x_2p(x_1,x_2,x_3,x_4;t|x_0;t_0) + c_2(x_3+1)p(x_1,x_2,x_3+1,x_4;t|x_0;t_0) \\
&- c_2x_3p(x_1,x_2,x_3,x_4;t|x_0;t_0) + c_3(x_3+1)p(x_1,x_2,x_3+1,x_4;t|x_0;t_0) \\
&- (c_3x_3)p(x_1,x_2,x_3,x_4;t|x_0;t_0)
\end{aligned}
$$

Figure 1.3b depicts one iteration of Gillespie's stochastic simulation with direct method. Further, Figure 1.4 shows the relationship between expected value of stochastic model and deterministic model. It can be seen that both solutions are not the same because of existence of bimolecular reaction in Michaelis-Menten enzyme kinetics. In Figure 1.5 density histogram of the trajectories of the SSA can be seen at different time steps.

CLE representation of Michaelis-Menten system is as follows:

$$
\begin{aligned}
dZ_1(t) &= \frac{1}{\sqrt{V}}\bigg(-\sqrt{k_1X_1(t)X_2(t)}dW_1(t) + \sqrt{k_2X_3(t)}dW_2(t)\bigg) \\
&\quad - \sqrt{k_1X_1(t)X_2(t)}dt + \sqrt{k_2X_3(t)}dt \\
dZ_2(t) &= \frac{1}{\sqrt{V}}\bigg(-\sqrt{k_1X_1(t)X_2(t)}dW_1(t) + \sqrt{k_2X_3(t)}dW_2(t) + \sqrt{k_3X_3(t)}dW_3(t)\bigg) \\
&\quad + -\sqrt{k_1X_1(t)X_2(t)}dt + \sqrt{k_2X_3(t)}dt + \sqrt{k_3X_3(t)}dt \\
dZ_3(t) &= \frac{1}{\sqrt{V}}\bigg(\sqrt{k_1X_1(t)X_2(t)}dW_1(t) - \sqrt{k_2X_3(t)}dW_2(t) - \sqrt{k_3X_3(t)}dW_3(t)\bigg) \\
&\quad + \sqrt{k_1X_1(t)X_2(t)}dt - \sqrt{k_2X_3(t)}dt - \sqrt{k_3X_3(t)}dt \\
dZ_4(t) &= \frac{1}{\sqrt{V}}\bigg(\sqrt{k_3X_3(t)}dW_3(t)\bigg) + \sqrt{k_3X_3(t)}dt
\end{aligned}
$$

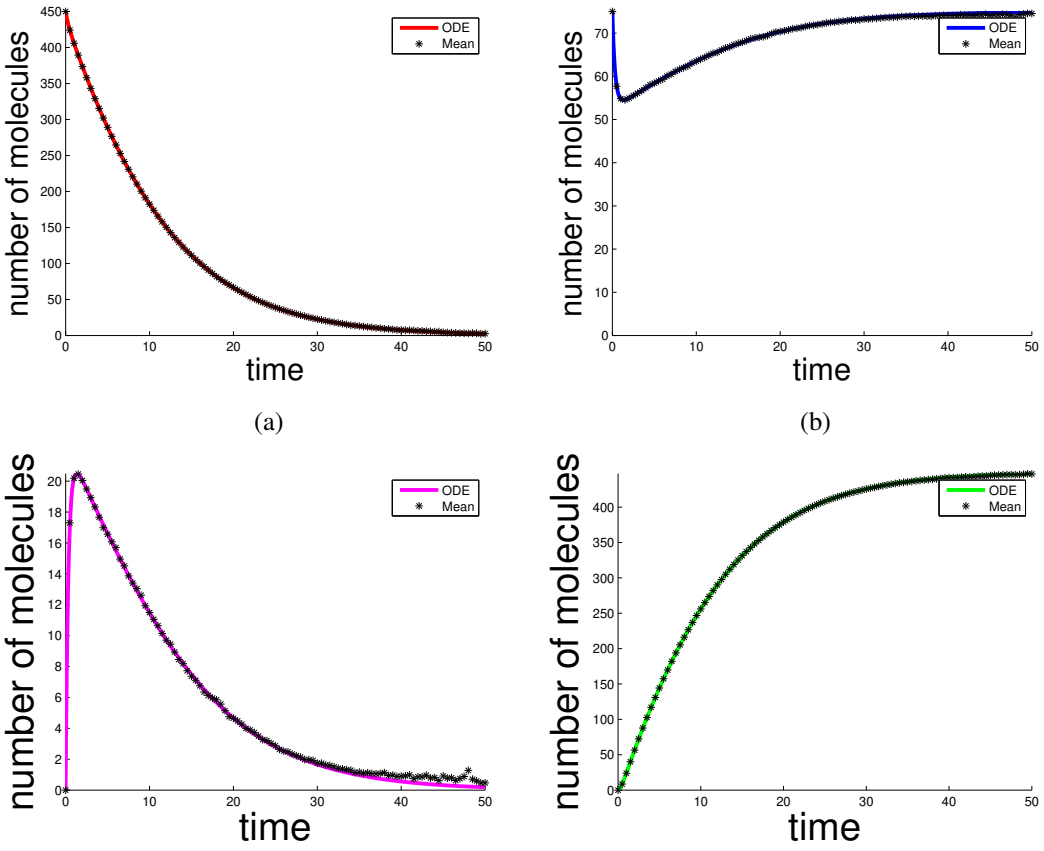
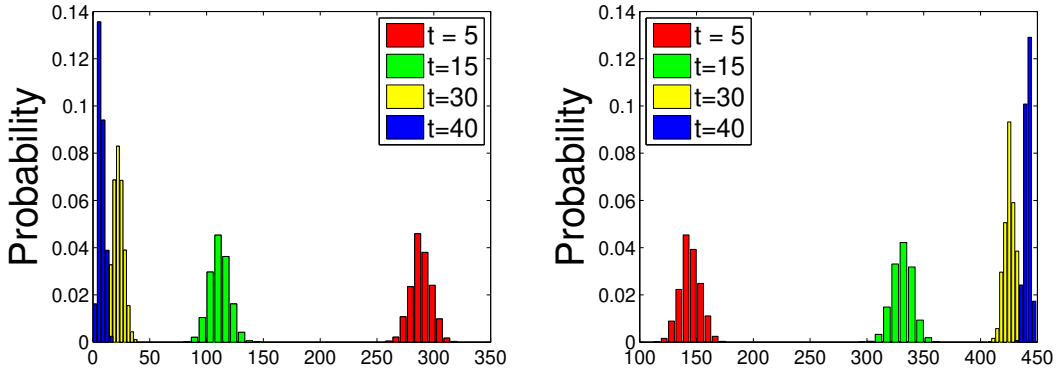


Figure 1.4: (a) Red curve indicates ODE solution of substrate (S). Black dots are mean of state vectors for S after 1000 iterations of SSA (b) Blue curve indicates ODE solution of enzyme (E). Black dots are mean of state vectors for E after 1000 iterations of SSA (c) Green curve indicates ODE solution of enzyme-substrate complex (ES). Black dots are mean of state vectors for ES after 1000 iterations of SSA (d) Pink curve indicates ODE solution of product (P). Black dots are mean of state vectors for P after 1000 iterations of SSA

On the other hand, FPE representation of Michaelis-Menten system becomes:

$$\frac{\partial}{\partial t}p(x;t|x_0;t_0) = \frac{\partial}{\partial x_1}\Big[(-c_1x_1x_2 + c_2x_3)p(x;t|x_0;t_0)\Big] + \frac{\partial}{\partial x_2}\Big[(-c_1x_1x_2 + c_2x_3 + c_3x_3)p(x;t|x_0;t_0)\Big]$$
$$+ \frac{\partial}{\partial x_3}\Big[(c_1x_1x_2 - c_2x_3 - c_3x_3p(x;t|x_0;t_0)\Big] + \frac{\partial}{\partial x_4}\Big[(c_3x_3)P(x;t|x_0;t_0)\Big]$$
$$+ \frac{1}{2}\frac{\partial^2}{\partial x_1^2}\Big[(c_1x_1x_2 + c_2x_3)p(x;t|x_0;t_0)\Big] + \frac{1}{2}\frac{\partial^2}{\partial x_2^2}\Big[(c_1x_1x_2 + c_2x_3 + c_3x_3)p(x;t|x_0;t_0)\Big]$$
$$+ \frac{1}{2}\frac{\partial^2}{\partial x_3^2}\Big[(c_1x_1x_2 + c_2x_3 + c_3x_3)p(x;t|x_0;t_0)\Big] + \frac{1}{2}\frac{\partial^2}{\partial x_4^2}\Big[(c_3x_3)p(x;t|x_0;t_0)\Big]$$
$$+ \frac{\partial^2}{\partial x_1x_2}\Big[(c_1x_1x_2 + c_2x_3)p(x;t|x_0;t_0)\Big] + \frac{\partial^2}{\partial x_1x_3}\Big[(-c_1x_1x_2 - c_2x_3 + c_3x_3)P(x;t|x_0;t_0)\Big]$$
$$+ \frac{\partial}{\partial x_3}\Big[(c_1x_1x_2 - c_2x_3)p(x;t|x_0;t_0)\Big] + \frac{\partial^2}{\partial x_2x_3}\Big[(-c-1x_1x_2 - c_2x_3 - c_3x_3)p(x;t|x_0;t_0)\Big]$$
$$+ \frac{\partial^2}{\partial x_2x_4}\Big[(c_3x_3)p(x;t|x_0;t_0)\Big] + \frac{\partial^2}{\partial x_3x_4}\Big[(-c_3x_3)p(x;t|x_0;t_0)\Big]$$



(a) Estimated probability mass functions of species S for different time steps

(b) Estimated probability mass functions of species P for different time steps

Figure 1.5: Estimated probability mass functions of species S and for time steps $t = 5$, $t = 15, t = 30, t = 40$, based on 10,000 runs

1.4.3 The Janus Kinase Signal Transducer and Activator of Transcription (JAK-STAT) Pathway

In general, signaling pathways deal with transformation or processing of information. JAK-STAT signaling pathway transforms information from outside of cell to cell nucleus. The biological system consists of 41 reactions with 38 species. List of reactions are given in Table A.2 and reaction constants are given in Table A.3. Also, initial molecule numbers are chosen arbitrarily as 100 for each species of JAK-STAT signaling pathway. Figure 1.6 depicts one realization of all species in JAK-STAT signaling pathway obtained through Gillespie's SSA.

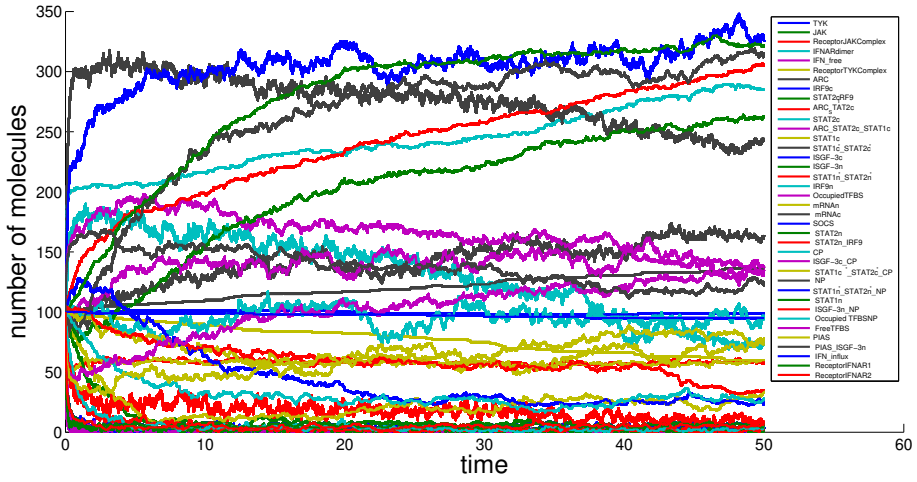
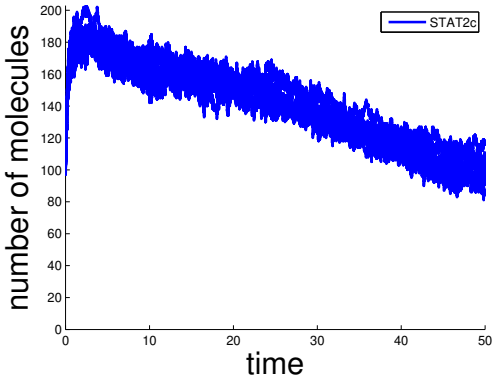
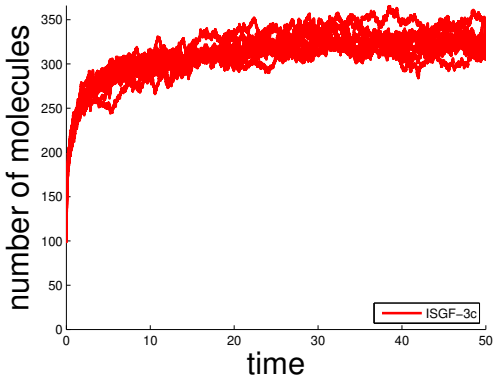


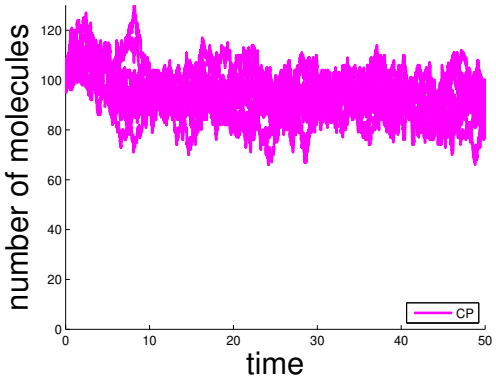
Figure 1.6: One iteration of SSA for JAK-STAT Pathway



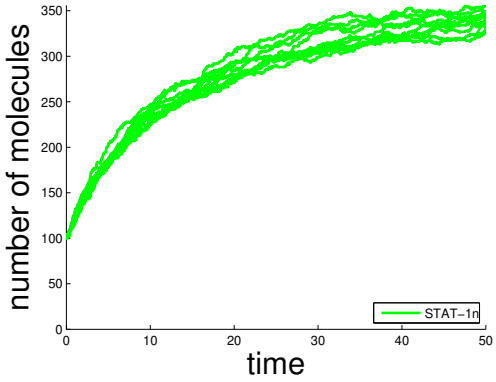
(a) 10 iterations with Gillespie's SSA for the species STAT-2C



(b) 10 iterations with Gillespie's SSA for the species ISGF-3C



(c) 10 iterations with Gillespie's SSA for the species CP



(d) 10 iterations with Gillespie's SSA for the species STAT-1n

Figure 1.7: 10 realizations of Gillespie's SSA for different species

In Figure 1.7, it can be seen that 10 realizations of the Gillespie's SSA. If the number of realizations is increased, it is possible to obtain more accurate estimate for the state of species.

CHAPTER 2

PROBABILISTIC INFERENCE FOR STATE-SPACE MODELS

Estimation theory is a branch of statistics and it is widely used in many scientific areas such as target tracking, signal processing, econometric systems and so on. In a few words, it is the study of inferring the value or the probability distribution of a quantity based on indirect, uncertain measurements [4, 43]. Estimators can be classified into two categories based on the quantity of the estimation: (i) Parameter estimators (ii) State estimators. *A parameter* is a time-invariant quantity which characterizes a population or a probability distribution. On the other hand, *the state* of a dynamic system is usually a time-evolving vector which describes the system under study. The state vector may contain dynamic variables such as velocity, position and orientation [43]. Traditionally, to define methods for estimating the state of a time-varying system according to indirect measurements, the term *filtering* is used [43]. *Smoothing* also defines estimation methods for the state of a time-varying system. Yet, there is a single difference between filtering and smoothing. In filtering, current state is estimated by using measurements up to that time. On the other hand, smoothing is used to estimate former state of the system based on all measurements up to given time [17].

This chapter begins with general description of the state-space model for representation of a dynamic system. Then, Bayesian framework for the state estimation is explained. After that, some state estimation methods based on specific assumptions are introduced.

2.1 State Estimation

A dynamic system is usually represented in state-space form to develop estimation algorithms. *A state-space model* represents probabilistic relationship between hidden states and observed measurements as well as evolution of hidden states of the system under consideration [3, 43].

Let x_t and y_t represent state of dynamic system at time t and measurement obtained at time t, respectively. A general state-space model contains two equations for the hidden state and the measurement at time t as follows [3, 11]:

$$\text{Transition Model: } x_t = \alpha(x_{t-1}, u_t), \;\; t \in \mathbb{N}^*, \;\; x_t \in \mathbb{R}^{n_x}, \tag{2.1}$$

$$\text{Measurement Model: } y_t = \beta(x_t, w_t), \; t \in \mathbb{N}^*, \;\; y_t \in \mathbb{R}^{n_y}, \tag{2.2}$$

where n_x and n_y are dimensions of state and measurement vectors, respectively. Here, $u_t \in \mathbb{R}^{n_u}$ and $w_t \in \mathbb{R}^{n_w}$ represent *process noise* and *measurement noise*, respectively. The former one is a way of modeling uncertainties in the system under consideration while the latter one is due to the assumption that measurements are uncertain [43].

For (2.1) and (2.2), it is assumed that u_t and w_t are independent and identically distributed (i.i.d) random variables. Also, functional form of $\alpha : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x}$, $\beta : \mathbb{R}^{n_x} \times \mathbb{R}^{n_w} \rightarrow \mathbb{R}^{n_y}$ are known but not need to be linear. Further, probability distribution function of the initial state vector x_0 is given as $p(x_0)$ [3]. According to given model, it can be understood that initial state x_0 is used to generate the next state x_1 at the first place. Then, the first measurement y_1 is generated by using this state.

Sequential form of hidden states and measurements generated according to (2.1) and (2.2) from time 0 to the current time t will be shown through the chapter as follows:

$$\text{Hidden states : } x_{0:t} = \{x_0, x_1, \ldots, x_t\}, \tag{2.3}$$

$$\text{Measurements : } y_{1:t} = \{y_1, y_2, \ldots, y_t\}. \tag{2.4}$$

In general, state-space models are based on the following two assumptions [11]: (i) Hidden states, $x_{0:t}$, are Markovian. (ii) Measurements y_t, $t \in \mathbb{N}^*$, given states $x_{0:t}$, and previous measurements $y_{1:t-1}$, depend only on the current hidden state x_t. Indeed, $p(y_t|x_{0:t}, y_{1:t-1}) = p(y_t|x_t)$. The first assumption assures that the current state x_t,

$t \in \mathbb{N}$, only depends on the previous state x_{t-1}, i.e $p(x_t|x_{0:t-1}) = p(x_t|x_{t-1})$. The second assumption means that although measurements are independent from each other, they are dependent on hidden states of the system under study.

In some cases, it is convenient to represent state-space model based on conditional distributions [11, 43] as follows:

$$\text{Initial Distribution: } x_0 \sim p(x_0) \tag{2.5}$$

$$\text{Transition Model: } x_t \sim p(x_t|x_{t-1}) \tag{2.6}$$

$$\text{Measurement Model: } y_t \sim p(y_t|x_t). \tag{2.7}$$

Based on given definitions and assumptions, relationships between states and measurements of a state space model can be seen in Figure 2.1.

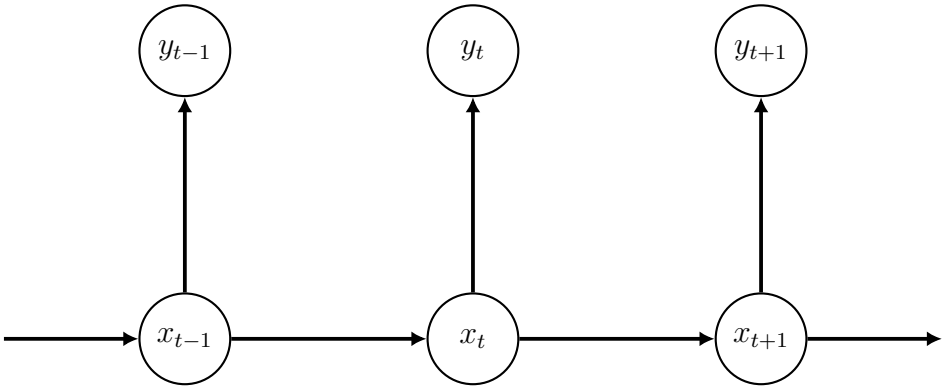


Figure 2.1: Graphical representation of a state-space model [11]

As stated before, state vector of a dynamic system contains all information to describe the system under consideration. In the following sections, main purpose will be obtaining an estimate for the state x_t for given observations $y_{1:t}$ provided that transition and measurement equations are known. The process can be considered as an inverse problem [47]. The solution for the inverse problem can be obtained through Bayesian methods which are based on Bayes' rule defined on probabilities as follows [20]:

$$p(x_{0:t}|y_{1:t}) = \frac{p(x_{0:t}, y_{1:t})}{p(y_{1:t})} = \frac{p(y_{1:t}|x_{0:t})p(x_{0:t})}{p(y_{1:t})}, \tag{2.8}$$

where

- $p(x_{0:t})$ is the prior distribution. It gives marginal information about hidden states before having any measurement.

- $p(y_{1:t}|x_{0:t})$ is the likelihood distribution. It describes inaccurate relationship between hidden states and measurements. In other words, it describes the changes in $p(x_{0:t}|y_{1:t})$ as a new observation is obtained.
- $p(x_{0:t}|y_{1:t})$ is the posterior distribution. It is conditional distribution of hidden states for given measurements.
- $p(y_{1:t})$ is normalizing constant, $p(y_{1:t}) = \int p(y_{1:t}|x_{0:t})p(x_{0:t})dx_{0:t}$. $p(y_{1:t})$ can be considered as constant since it does not depend on $x_{0:t}$ with fixed $y_{1:t}$. Equivalent form of (2.8) omits the normalizing constant and yields unnormalized posterior distribution as:

$$p(x_{0:t}|y_{1:t}) \propto p(y_{1:t}|x_{0:t})p(x_{0:t}) \tag{2.9}$$

Apart from posterior distribution, it is possible to obtain other statistical properties of hidden states by means of Bayes' rule. For instance, marginal distribution of a state can be found as follows: for given joint posterior distribution of hidden states $p(x_{0:t}|y_{1:t})$

$$p(x_t|y_{1:t}) = \int p(x_{0:t}|y_{1:t})dx_{0:t-1}.$$

Also, expectation of a function of x_t can be obtained as:

$$E[f(x_t)] = \int f(x_t)p(x_{0:t}|y_{1:t})dx.$$

Based on Bayes' rule it can be seen that existence of posterior distribution allows us to reach quality of estimate, such as measure of uncertainty and expectation of a state. Therefore, in state-space model all of uncertain quantities, states and measurements, are treated as random variables and described by probability distribution functions. Besides, to reduce computational complexity it is possible to derive recursive way of obtaining probability distributions related to states through Bayes' rule. Bayesian way of formulating filtering is called *Bayesian filtering* [43].

2.1.1 Recursive Bayesian Estimation

If the state space model is considered as a Bayesian model, (2.5) and (2.6) define prior distribution and (2.7) defines the likelihood function [17]. Based on Markovian and independence assumptions explained in the previous section, the following

factorizations are possible:

$$p(x_{0:t}) = p(x_0) \prod_{m=1}^{t} p(x_m|x_{m-1}) \tag{2.10}$$

$$p(y_{1:t}|x_{0:t}) = \prod_{m=1}^{t} p(y_m|x_m). \tag{2.11}$$

Proportionality for the posterior distribution of states can be written through Bayes' theorem as follows:

$$p(x_{0:t}|y_{1:t}) \propto p(x_0) \prod_{m=1}^{t} p(x_m|x_{m-1}) \prod_{m=1}^{t} p(y_m|x_m) \tag{2.12}$$

It is possible to derive recursive formula for the joint posterior distribution of states $p(x_{0:t}|y_{1:t})$. By this way, we can reduce computational cost because we do not need to recompute joint prior and likelihood functions at every time. Instead, transition and measurement equations of the next time step are used. To obtain recursive formula, assume that joint posterior distribution at time $t-1$ is obtained. Conditional independence of measurements, $p(y_t|x_{1:t}, y_{1:t-1}) = p(y_t|x_t)$, and Markov property of states, $p(x_t|x_{1:t-1}, y_{1:t-1}) = p(x_t|x_{t-1})$ allow us to write the following factorizations for the elements of (2.8):

$$\begin{aligned}
p(x_{0:t}, y_{1:t}) &= p(x_{0:t}, y_{1:t-1}, y_t) = p(y_t|x_{0:t}, y_{1:t-1})p(x_{0:t}, y_{1:t-1}) \\
&= p(y_t|x_t)p(x_t, x_{0:t-1}, y_{1:t-1}) \\
&= p(y_t|x_t)p(x_t|x_{0:t-1}, y_{1:t-1})p(x_{0:t-1}, y_{1:t-1}) \\
&= p(y_t|x_t)p(x_t|x_{t-1})p(x_{0:t-1}, y_{1:t-1}) \\
&= p(y_t|x_t)p(x_t|x_{t-1})p(x_{0:t-1}|y_{1:t-1})p(y_{1:t-1})
\end{aligned} \tag{2.13}$$

$$p(y_{1:t}) = p(y_t, y_{1:t-1}) = p(y_t|y_{1:t-1})p(y_{1:t-1}) \tag{2.14}$$

If (2.13) and (2.14) are inserted into (2.8), the following recursive formula for the joint probability function of states up to time t will be obtained as:

$$p(x_{0:t}|y_{1:t}) = p(x_{0:t-1}|y_{1:t-1}) \frac{p(x_t|x_{t-1})p(y_t|x_t)}{p(y_t|y_{1:t-1})}. \tag{2.15}$$

It seems simple to obtain joint posterior distribution recursively through Bayes' rule. However, dimensionality of the posterior distribution increases when a new measurement is given. As a result, computational complexity of a single time step increases. In this case, finding specific marginal distribution $p(x_t|y_{1:t})$ may be more reasonable than finding joint posterior distribution of all states. There are three types of marginal distributions which can be obtain by filtering or smoothing methods [43] (See Figure 2.2):

- *Prediction Distribution*, $p(x_t|y_{1:t-1})$, is marginal distribution of the current state, x_t, given all previous measurements, $y_{1:t-1}$. It is computed at prediction step of a Bayesian filter.
- *Filtering Distribution,* $p(x_t|y_{1:t})$, is marginal distribution of current state, x_t, all measurements up to current time, $y_{1:t}$. It is obtained at update step of a Bayesian filter.
- *Smoothing Distribution*, $p(x_t|y_{1:T})$, is marginal distribution of former state, x_t, given all measurements, $y_{1:T}$ where $T > t$. It is computed through a Bayesian smoother.

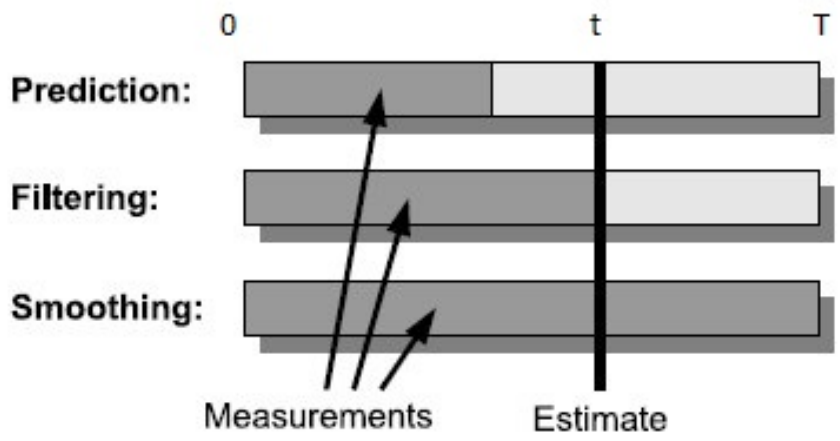


Figure 2.2: Marginal Distributions of States [43]

Theoretically, the simplest way of obtaining filtering distribution is marginalization of (2.12) over $x_{0:t-1}$. However, recursive way, which is algorithmically more convenient than marginalizing, can be derived. Let us assume that estimation for the probability

distribution $p(x_{t-1}|y_{1:t-1})$ of the state at time step $t-1$ is known. Then, the joint distribution $p(x_t, x_{t-1}|y_{1:t-1})$ is obtained as follows:

$$\begin{aligned} p(x_t, x_{t-1}|y_{1:t-1}) &= \frac{p(x_t, x_{t-1}, y_{1:t-1})}{p(y_{1:t-1})} = \frac{p(x_t|x_{t-1}, y_{1:t-1})p(x_{t-1}, y_{1:t-1})}{p(y_{1:t-1})} \\ &= p(x_t|x_{t-1}, y_{1:t-1})p(x_{t-1}|y_{1:t-1}) \\ &= p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1}). \end{aligned}$$

The last equality is written by Markovian assumption on states. Prediction equation is simply marginalization of the above equality

$$p(x_t|y_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}. \tag{2.16}$$

Then, filtering equation can be obtained according to Bayes' rule:

$$\begin{aligned} p(x_t|y_{1:t}) = p(x_t|y_{1:t-1}, y_t) &= \frac{p(x_t, y_{1:t-1}, y_t)}{p(y_{1:t})} = \frac{p(y_t|y_{1:t-1}, x_t)p(y_{1:t-1}, x_t)}{p(y_{1:t-1}, y_t)} \\ &= \frac{p(y_t|y_{1:t-1}, x_t)p(x_t|y_{1:t-1})p(y_{1:t-1})}{p(y_t|y_{1:t-1})p(y_{1:t-1})} \\ &= \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{p(y_t|y_{1:t-1})}. \end{aligned} \tag{2.17}$$

Here, the last equality holds by conditional independence of the measurements.

The process of obtaining (2.16) and (2.17) are called *prediction step* and *update step* of filtering, respectively [16, 17]. In prediction step, system model is used to predict probability distribution function of the state at the next measurement time step, t, for given measurements up to current time step $t-1$, $y_{1:t-1}$. Then, updating step modifies the prediction based on the new observation, y_t. To sum up, filtering starts from prior distribution given in (2.5). Then, turn into a cycle as shown in Figure 2.3 in which a prediction step is followed by an update step.

As it is explained above, Bayesian inference provides recursive model for the state estimation problem. However, this recursive estimation of posterior density provides theoretical solution in general. The reason for this is that given equations have closed form analytical solutions only for a few cases. If state and measurement equations are linear and noises are Gaussian, Kalman filter (KF) gives analytical solution. Also, if there are finite numbers of hidden sates and measurements, grid based methods

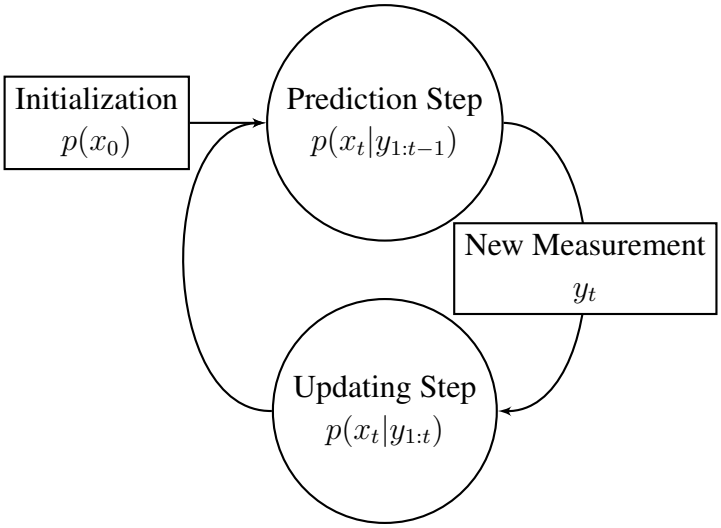


Figure 2.3: Filtering Cycle

provides analytical solution. However, for non-linear and non-Gaussian state space models, analytical expression for the state cannot be obtained because of the dimension of integrals. Hence, extended Kalman filter (EKF) and approximate grid based methods are proposed. These methods are approximations to the KF and grid-based filter. Another idea is that using statistical sampling techniques to represent probability distributions. *Sequential Monte Carlo (SMC) methods* or *particle filters (PF)*, which combine Monte Carlo techniques and Bayesian inference, are used to achieve this.

2.1.2 Accurate State Estimation Methods

Under specific assumptions of KF and grid-based methods provides accurate filtering distributions. Below we investigate these methods.

2.1.2.1 Kalman Filter

KF provides filtering distribution parameterized by mean and covariance. It is based on the fact that if filtering distribution at time $t-1$, $p(x_{t-1}|y_{1:t-1})$, is Gaussian, then $p(x_t|y_{1:t})$ is also Gaussian with the following assumptions [3]:

- Process noise u_t, measurement noise w_t in (2.1) and (2.2) are drawn from Gaus-

sian distribution with zero mean and covariance matrices Q_t and R_t, respectively, i.e, $u_t \sim \mathcal{N}(0, Q_t)$ and $w_t \sim \mathcal{N}(0, R_k)$.

- $\alpha(x_{t-1}, u_t)$, defined in (2.1) and $\beta(x_t, w_t)$, defined in (2.2) are linear functions.
- Initial state x_0 has a Gaussian distribution.

Then, transition model and measurement model can be rewritten as follows:

$$\text{State Model: } x_t = A_t x_{t-1} + u_t,$$
$$\text{Measurement Model: } y_t = B_t x_t + w_t,$$

where A_t and B_t are transition matrix of the dynamic system under study and measurement model matrix, respectively [3, 43].

Recursive formulations given in (2.16) and (2.17) form foundation of KF, derived in the original paper through mean square estimator [19, 43, 45]. It is called optimal algorithm for linear state space models since it minimizes the mean square error. Yet, it is possible to derive based on the fact that product of two Gaussian distribution is again a Gaussian distribution [19, 43]. As a result, KF has the following recursive relations:

Current Filtering Distribution: $p(x_{t-1}|y_{1:t-1}) = \mathcal{N}(x_{t-1}; m_{t-1|t-1}, P_{t-1|t-1})$,

Prediction Step: $p(x_t|y_{1:t-1}) = \mathcal{N}(x_t; m_{t|t-1}, P_{t|t-1})$,

Updating Step: $p(x_t|y_{1:t}) = \mathcal{N}(x_t; m_{t|t}, P_{t|t})$,

where $\mathcal{N}(x; m, c)$ is the normal distribution with mean m and covariance c. Also,

$$\begin{aligned} m_{t|t-1} &= A_t m_{t-1|t-1}, \\ P_{t|t-1} &= Q_{t-1} + A_t P_{t-1|t-1} A_t^T \\ m_{t|t} &= m_{t|t-1} + K_t(y_t - B_t m_{t|t-1}) \\ P_{t|t} &= P_{t|t-1} K_t B_t P_{t|t-1}, \end{aligned}$$

where

$$K_t = \frac{P_{t|t-1} B_t^T}{B_t P_{t|t-1} B_t^T + R_t}. \tag{2.18}$$

Here, K_t in (2.18) is known as the *Kalman gain*. It measures the correction of estimation of the state according to the given measurement.

2.1.2.2 Grid Based Filter

If the state space consists of finite number of discrete spaces, then grid-based filter, which is also known as *point-mass filter*, provides an optimal solution for filtering distribution $p(x_t|y_{1:t})$ [3, 31]. In KF, state space was described by a linear equation. Here, assume that state space at time $t-1$ consists of discrete states $x_{t-1}^i, i = 1, 2, \ldots, N$. Then, conditional distribution of each state x_{t-1}^i for given measurements turns to be [3]:

$$p(x_{t-1}|y_{1:t-1}) = \sum_{i=1}^{N} p(x_{t-1} = x_{t-1}^{(i)}|y_{1:t-1})\delta(x_{t-1} - x_{t-1}^{(i)})$$

where $\delta(\cdot)$ is the Dirac-delta function.

Based on this fact, prediction and update states can be rewritten as follows:

$$\text{Prediction Step : } p(x_t|y_{1:t-1}) = \sum_{i=1}^{N} \left(\sum_{j=1}^{N} p(x_{t-1} = x_{t-1}^{(i)}|y_{1:t-1}) p(x_t^{(i)}|x_{t-1}^{(j)}) \right) \delta(x_t - x_t^{(i)})$$

$$\text{Updating Step : } p(x_t|y_{1:t}) = \sum_{i=1}^{N} \frac{\sum_{j=1}^{N} p(x_{t-1} = x_{t-1}^{(i)}|y_{1:t-1}) p(x_t^{(i)}|x_{t-1}^{(i)}) p(y_t|x_t^{(i)})}{\sum_{i=1}^{N}\sum_{j=1}^{N} p(x_{t-1} = x_{t-1}^{(i)}|y_{1:t-1}) p(x_t^{(i)}|x_{t-1}^{(i)}) p(y_t|x_t^{(i)})}.$$

Grid based method provides optimal solution in case $p(x_{t+1}^{(i)}|x_t^{(j)})$ and $p(y_{t+1}|x_{t+1}^{(i)})$ are known [3].

2.1.3 Approximation Methods

In most of the state estimation problems, assumptions on the linearity of state space model or having finite number of discrete states in state space do not hold. Hence, approximate methods are proposed to extend application areas of optimal algorithms.

2.1.3.1 Extended Kalman Filter

EKF is applicable to non-linear filtering problems. It begins with linearization of the non-linear state-space model. Then, KF is applied to system for which assumptions

are satisfied.

Assume that state and measurement equations defined in (2.1) and (2.2) has the following forms:

$$\text{State Model: } x_t = \alpha(x_{t-1}) + u_t, \tag{2.19}$$

$$\text{Measurement model: } y_t = \beta(x_t) + w_t, \tag{2.20}$$

where $\alpha(x_t)$ and $\beta(x_t)$ are nonlinear functions of x_t. The EKF is based on the idea that filtering distribution $p(x_t|y_{1:t})$ is approximated by Gaussian distribution. Linearizations of functions $\alpha(x_t)$ and $\beta(x_t)$ are obtained through their Taylor series expansions. For details of derivation we refer to [43]. Prediction and update steps of EKF are as follows:

$$\text{Current Estimate: } p(x_{t-1}|y_{1:t-1}) \approx \mathcal{N}(x_{t-1}; m_{t-1|t-1}, P_{t-1|t-1}), \tag{2.21}$$

$$\text{Prediction Step: } p(x_t|y_{1:t-1}) \approx \mathcal{N}(x_t; m_{t|t-1}, P_{t|t-1}), \tag{2.22}$$

$$\text{Updating Step: } p(x_t|y_{1:t}) \approx \mathcal{N}(x_t; m_{t|t}, P_{t|t}). \tag{2.23}$$

Also,

$$\begin{aligned} m_{t|t-1} &= \alpha(m_{t-1|t-1}), \\ P_{t|t-1} &= Q_{t-1} + \hat{F}_t P_{t-1|t-1} \hat{F}_t^T \\ m_{t|t} &= m_{t|t-1} + K_t(y_t - \beta(m_{t|t-1})) \\ P_{t|t} &= P_{t|t-1} - K_t \hat{H}_t P_{t|t-1}, \end{aligned} \tag{2.24}$$

where $\hat{F}_t$ and $\hat{H}_t$ are linearization of functions $\alpha(x_t)$ and $\beta(x_{n_t})$, respectively. In other words, $\hat{F}$ and $\hat{H}$ are the first terms of Taylor expansion of $\alpha(x_t)$ and $\beta(x_t)$, respectively. Also, K_t in (2.24)

$$K_t = \frac{P_{t|t-1}\hat{H}^T}{\hat{H} P_{t|t-1} \hat{H}^T + R_t}$$

represents Kalman gain as standard KF.

The EKF described above is called first order EKF with additive noise since noises are added to nonlinear functions in the state and the measurements models and first order Taylor series expansion is used. In the case of second order Taylor series expansion, the filter is called second order EKF with additive noise. Besides, the filter is named

EKF with non-additive noise if the state and the measurement models are given as follows [43]:

$$\text{State Model: } x_t = \alpha(x_{t-1}, u_t), \tag{2.25}$$

$$\text{Measurement Model: } y_t = \beta(x_t, w_t). \tag{2.26}$$

Although higher order Taylor series expansions might be used to increase accuracy, these methods are not practical due to dimensional complexity.

An advantage of EKF is its simplicity by linearization. Linearization is a widely used approximation technique to non-linear systems. However, disadvantage of it is that filtering distribution is always approximated by a Gaussian distribution. Hence, EKF usually gives better approximations if the true posterior distribution is symmetric and unimodal [31].

2.1.3.2 Approximate Grid-Based Filter

In Section 2.1.2.2, it is stated that grid based filter gives optimal filtering distribution for finite and discrete state space models. The same idea can be used for nonlinear and non-Gaussian state space models which can also be decomposed as follows:

$$p(x_t|y_{1:t}) \approx \sum_{i=1}^{N} w_{t|t}^{(i)} \delta(x_{k-1}|x_{k-1}^{(i)}), \tag{2.27}$$

where $\delta(\cdot)$ is again the Dirac-delta function.

Consequently, the prediction and the update steps take the form [3]:

$$\text{Prediction Step: } p(x_{t+1}|y_{1:t}) \approx \sum_{i=1}^{N} w_{t+1|t}^{(i)} \delta(x_{t+1} - x_{t+1}^{(i)}),$$

$$\text{Updating Step: } p(x_{t+1}|y_{1:t+1}) \approx \sum_{i=1}^{N} w_{t+1|t+1}^{(i)} \delta(x_{t+1} - x_{t+1}^{(i)}),$$

where

$$w_{t+1|t}^{(i)} = \sum_{j=1}^{N} w_{t|t}^{(j)} \int_{x \in x_{t+1}^{(i)}} p(x_{t+1}^{(i)} | \bar{x}_t^{(j)}) dx$$

$$w_{t+1|t+1}^{(i)} = \frac{w_{t+1|t}^{(i)} \int_{x \in x_{t+1}^{(i)}} p(y_{t+1}|x) dx}{\sum_{j=1}^{N} w_{t+1|t}^{j} \int_{x \in x_{t+1}^{(i)}} p(y_{t+1}|x) dx}.$$

Here, $\bar{x}_t^{(j)}$ denotes the center of cell at time t.

Apart from EKF and approximate grid based filter, MC techniques can be considered to obtain numerical approximation to the posterior distribution and the filtering distributions.

2.2 Filtering for Non-linear State-Space Models: Sequential Monte Carlo Methods

Direct application of Bayes' rule provides an exact solution for the posterior distribution of the state of process of interest. However, it is not appropriate for most of the cases because of existence of high dimensional integrals in normalizing constant and marginalization of the posterior distribution. Hence, methods based on several assumptions are proposed. As mentioned in previous section, if dynamical system is modeled by linear-Gaussian state space model, then the Kalman filter is an optimal estimation method. It solves these integrals analytically and solutions are based on parameters mean and variance of process and measurement noises. Moreover, if the state space model consists of finite number of states, then grid based filter is the optimal method. However, in real life underlying assumptions usually do not hold. Therefore, some approximate methods such as extended Kalman filter and approximate grid based filter are proposed. These methods are still inefficient because they are not easy to implement. Also, these methods require strict assumptions too.

Monte Carlo integration is a statistical technique for numerical integration using random samples from a probability distribution [13]. It is an alternative approach to handle high dimensional integrals appearing in (2.16) and (2.17). There are different

ways to apply Monte Carlo integration such as importance sampling, sequential importance sampling and sequential Monte Carlo which is also sometimes referred to as particle filtering.

SMC method is a combination of Monte Carlo sampling methods with Bayesian inference [13]. The derivation of particle filter (PF) is done in the following sections. First, Monte Carlo sampling methods are described. Then, the idea is applied to recursive Bayesian estimation to obtain the particle filter algorithm.

2.2.1 Monte Carlo Sampling

The reason for using Monte Carlo method in state estimation is to approximate posterior distribution $p(x_{0:t}|y_{1:t})$. Let us assume that there are N random samples $\{x_{0:t}^{(i)}, i = 1, 2, \ldots, N\}$ from this posterior distribution. These random samples are also known as *particles*. Then, approximate probability density function, $\hat{p}(x_{0:t}|y_{1:t})$, can be obtained through Monte Carlo method as follows [16, 17]:

$$\hat{p}(x_{0:t}|y_{1:t}) = \frac{1}{N}\sum_{i=1}^{N}\delta(x_{0:t} - x_{0:t}^{(i)}). \tag{2.28}$$

Based on this fact, marginal of the posterior distribution can be written as:

$$\hat{p}(x_t|y_{1:t}) = \frac{1}{N}\sum_{i=1}^{N}\delta(x_t - x_t^{(i)}). \tag{2.29}$$

In some cases, estimation problem requires the expectation of an arbitrary function $h(x_{0:t})$ over the posterior distribution instead of probability value for marginal of the posterior distribution:

$$I(h(x_{0:t})) = \int h(x_{0:t})p(x_{0:t}|y_{1:t})dx_{0:t}. \tag{2.30}$$

Based on (2.28) and (2.30), the following estimation for the expectation of the function $h(x_{0:t})$ can be obtained:

$$\hat{I}(h(x_{0:t})) = \int h(x_{0:t})\hat{p}(x_{0:t}|y_{1:t})dx_{0:t} = \frac{1}{N}\sum_{i=1}^{N}h(x_{0:t}^{(i)}). \tag{2.31}$$

As stated above, in Monte Carlo sampling, random samples are drawn directly from the posterior distribution. Therefore, estimation given in (2.31) is an unbiased estimation [16, 17]. Also, variance of the estimation decreases when the number of random samples N increases. However, these advantages yields the following two disadvantages: (i) direct samples from a probability distribution function is possible for a few cases, i.e. standard probability density functions, such as normal or uniform distributions; (ii) computational complexity increases as t increases since sampling directly from probability density $p(x_{0:t}|y_{1:t})$ requires a longer time period. To overcome the first disadvantage, importance sampling (IS) method is proposed.

2.2.2 Importance Sampling

In this method, *importance density* or *proposal density* $q(x_{0:t}|y_{1:t})$ is introduced [17]. It has similar features with the posterior distribution and easy to sample from. In case direct sampling from the posterior distribution $p(x_{0:t}|y_{1:t})$ is not possible, indirect samples through importance density $q(x_{0:t}|y_{1:t})$ can be used as an alternative [3, 17].

Expectation of function $h(x_{0:t})$ over the posterior distribution given in (2.31) can also be rewritten as:

$$I(h(x_{0:t})) = \int h(x_{0:t}) \frac{p(x_{0:t}|y_{1:t})}{q(x_{0:t}|y_{1:t})} q(x_{0:t}|y_{1:t}) dx_{0:t}.$$

As a result, we consider the expectation of $h(x_{0:t}) \dfrac{p(x_{0:t}|y_{1:t})}{q(x_{0:t}|y_{1:t})}$ over importance density. Therefore, if N particles $\{x_{0:t}^{(i)}, i = 1, 2, \ldots, N\}$ are available from importance density, Monte Carlo approximation allows us to write:

$$\hat{I}(h(x_{0:t})) = \frac{1}{N} \sum_{i=1}^{N} \frac{p(x_{0:t}^{(i)}|y_{1:t})}{q(x_{0:t}^{(i)}|y_{1:t})} h(x_{0:t}^{(i)}) = \frac{1}{N} \sum_{i=1}^{N} w(x_{0:t}^{(i)}) h(x_{0:t}^{(i)}),$$

where

$$w(x_{0:t}^{(i)}) = \frac{p(x_{0:t}^{(i)}|y_{1:t})}{q(x_{0:t}^{(i)}|y_{1:t})} \tag{2.32}$$

is called *importance weight*. However, it can be seen that values of $p(x_{0:t}^{(i)}|y_{1:t})$ for $i = 1, 2, \ldots, N$ need to be known to apply IS. Therefore, the posterior distribution

can be written by means of Bayes' theorem as:

$$p(x_{0:t}^{(i)}|y_{1:t}) = \frac{p(y_{1:t}|x_{0:t}^{(i)})p(x_{0:t}^{(i)})}{\int p(y_{1:t}|x_{0:t})p(x_{0:t})dx},$$

where the prior $p(x_{0:t}^{(i)})$ and the likelihood $p(y_{1:t}|x_{0:t}^{(i)})$ functions can be obtained (2.10) and (2.11), respectively. Yet, integral in denominator usually cannot be computed easily because of high dimensionality. To handle this problem, IS can be applied to the normalizing constant to get an approximation for it. So, (2.31) can be rewritten as [17]:

$$\begin{aligned} I(h(x_{0:t})) &= \frac{\int h(x_{0:t})p(y_{1:t}|x_{0:t})p(x_{0:t})dx_{0:t}}{\int p(y_{1:t}|x_{0:t})p(x_{0:t})dx_{0:t}} \\ &= \frac{\int \frac{h(x_{0:t})p(y_{1:t}|x_{0:t})p(x_{0:t})}{q(x_{0:t}|y_{1:t})} q(x_{0:t}|y_{1:t})dx_{0:t}}{\int \frac{p(y_{1:t}|x_{0:t})p(x_{0:t})}{q(x_{0:t}|y_{1:t})} q(x_{0:t}|y_{1:t})dx_{0:t}}. \end{aligned}$$

Hence,

$$\hat{I}(h(x_{0:t})) = \frac{\frac{1}{N}\sum_{i=1}^{N} h(x_{0:t})w(x_{0:t}^{(i)})}{\frac{1}{N}\sum_{j=1}^{N} w(x_{0:t}^{(j)})} = \sum_{i=1}^{N} \frac{w(x_{0:t}^{(i)})}{\sum_{j=1}^{N} w(x_{0:t}^{(j)})} h(x_{0:t}^{(i)}) = \sum_{i=1}^{N} \tilde{w}(x_{0:t}^{(i)})h(x_{0:t}^{(i)}),$$

where

$$\tilde{w}(x_{0:t}^{(i)}) = \frac{w(x_{0:t}^{(i)})}{\sum_{j=1}^{N} w(x_{0:t}^{(j)})}, \tag{2.33}$$

is called *normalized importance weight*. Also, Monte Carlo approximation for the posterior distribution can be written as

$$\hat{p}(x_{0:t}|y_{1:t}) = \sum_{i=1}^{N} \tilde{w}(x_{0:t}^{(i)})\delta(x_{0:t} - x_{0:t}^{(i)}). \tag{2.34}$$

A pseudo-algorithm for IS may be as follows [43]:

0. Prior distribution $p(x_{0:t})$ defined in (2.10), likelihood distribution $p(y_{1:t}|x_{0:t})$ defined in (2.11) and importance density $q(x_{0:t}|y_{1:t})$ are given.

1. Sample N particles $x_{0:t}^{(i)}$, $i = 1, 2, \ldots, N$ from importance density $q(x_{0:t}|y_{1:t})$.

2. Compute importance weights of each particles: $w(x_{0:t}^{(i)}) = \dfrac{p(y_{1:t}|x_{0:t}^{(i)})p(x_{0:t}^{(i)})}{q(x_{0:t}^{(i)}|y_{1:t})}$.

3. Normalize the importance weights and assign to $\tilde{w}(x_{0:t}^{(i)})$.

4. Find approximation for the expectation of function $h(x_{0:t})$ as

$$I(h(x_{0:t})) \approx \sum_{i=1}^{N} \tilde{w}(x_{0:t}^{(i)})h(x_{0:t}^{(i)})$$

Having efficient estimate by IS depends on the choice of importance density and the number of random samples [17]. Despite this, calculating importance weight at every time step becomes time consuming when the number of time steps increases. Hence, a recursive method *Sequential importance sampling (SIS)* is also proposed.

2.2.3 Sequential Importance Sampling

This method is proposed to overcome second disadvantage of the general Monte Carlo sampling. Basically, SIS fixes computational cost at every time step to approximate posterior distribution. If the general state space model is given by (2.5) - (2.7), then SIS can be used. In SIS, importance density is chosen as [17]

$$q(x_{0:t}|y_{1:t}) = q(x_{0:t-1}|y_{1:t-1})q(x_t|x_{0:t-1}, y_t). \tag{2.35}$$

Also, the posterior distribution will have the following recursion through Markov properties of the state space model:

$$p(x_{0:t}|y_{1:t}) = p(x_{0:t-1}|y_{1:t-1})p(x_t|x_{t-1})p(y_t|x_t).$$

As a result, importance weight becomes

$$w(x_{0:t}^{(i)}) = \frac{p(x_{0:t}^{(i)}|y_{1:t})}{q(x_{0:t}^{(i)}|y_{1:t})} = \frac{p(x_{0:t-1}^{(i)}|y_{1:t-1})p(x_t^{(i)}|x_{t-1}^{(i)})p(y_t|x_t^{(i)})}{q(x_{0:t-1}^{(i)}|y_{1:t-1})q(x_t^{(i)}|x_{t-1}^{(i)}, y_t)} \tag{2.36}$$

$$\propto w(x_{0:t-1}^{(i)})\gamma(x_t^{(i)}), \tag{2.37}$$

where

$$\gamma(x_t^{(i)}) = \frac{p(x_t^{(i)}|x_{t-1}^{(i)})p(y_t|x_t^{(i)})}{q(x_t^{(i)}|x_{t-1}^{(i)}, y_t)} \tag{2.38}$$

called as *incremental importance weight* [17].

It can be inferred that to obtain random sample $x_{0:t}^{(i)} \sim q(x_{0:t}|y_{1:t})$, the particle $x_k^{(i)} \sim q(x_t|x_{0:t-1}^{(i)}, y_t)$ can be added to set $x_{0:t-1}^{(i)}$. This is the reason for the following algorithm is called sequential. A pseudo-algorithm for SIS may be given as follows [43]:

0. Initial distribution, transition and measurement equations as in (2.5) - (2.7) and importance density are given.

1. Sample N particles $x_0^{(i)}$, $i = 1, 2, \ldots, N$ from prior distribution $p(x_0)$. Also, set initial importance weights $w(x_0^{(i)})$ to $\frac{1}{N}$.

2. For each time step $k = 1, 2, \ldots, t$:

   • Draw samples $x_k^{(i)} \sim q(x_t|x_{0:t-1}, y_t)$.

   • Calculate weights $w(x_{0:t}^{(i)})$ according to (2.36).

   • Normalize the importance weights and assign to $\tilde{w}(x_{0:t}^{(i)})$.

   • Find approximation for the expectation of function $h(x_{0:t})$ as

$$I(h(x_{0:t})) \approx \sum_{i=1}^{N} \tilde{w}(x_{0:t}^{(i)}) h(x_{0:t}^{(i)})$$

We realize that, if importance distribution is chosen Markovian, namely, $q(x_t|x_{0:t-1}, y_t) = q(x_t|x_{t-1}, y_t)$, then storage cost will be reduced [43]. Storing the current state $x_k^{(i)}$ is sufficient instead of keeping the whole history of the state $x_{0:t}^{(i)}$.

SIS is efficient in terms of computational cost. However, if the number of time steps increases, the method experiences the of dimensionality.

2.2.4 Sequential Importance Resampling: Particle Filtering

In SIS, importance weights can be updated when a new measurement is available. However, importance weights become zero or nearly zero after a few time steps. The

reason for this is that sample size N is kept constant while dimensionality of the state space is increasing at each additional time step. So, variance of importance weights increases over time. The defined problem is called *degeneracy phenomenon*. Disadvantage of degeneracy is that computational power is used to update particles which has almost zero contribution for estimation of the posterior distribution. A measure for degeneracy of an algorithm is *effective sample size* which is defined as [3]

$$N_{\text{eff}} = \frac{N}{1 + V[\hat{w}(x_{0:t}^{(i)})]}, \tag{2.39}$$

where N is number of random sample drawn from importance density, $V[\cdot]$ represents the variance and $\hat{w}(x_{0:t}^{(i)}) = \dfrac{p(x_t^{(i)}|y_{1:t})}{q(x_t|x_{t-1}, y_{1:t})}$ is called the *true weight*. Although the exact value of effective sample size cannot be evaluated, an approximation can be obtained via [3]

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N} \tilde{w}(x_{0:t}^{(i)})}. \tag{2.40}$$

To avoid the degeneracy, N should be chosen so that $N > N_{\text{eff}}$. Obviously, N can be chosen very large but in general this is not easy. Instead, variance of the *true weight* $\hat{w}(x_{0:t}^{(i)})$ is reduced by following two ways: (i) good choice of importance density, (ii) using resampling in sequential importance sampling (sequential importance resampling).

Good Choice Importance Density

"Good choice" means that importance density is chosen as close as possible to the posterior distribution. By this way, random samples describing importance density will be close to the posterior distribution. So, the possibility of having samples with small weights will be reduced [3].

A possible choice for the importance density is the prior distribution of state, which leads to [43]

$$w(x_{0:t}^{(i)}) = \frac{p(x_{0:t}^{(i)}|y_{1:t})}{p(x_{0:t}^{(i)})} = \frac{p(x_{0:t-1}^{(i)}|y_{1:t-1})p(x_t|x_{t-1}^{(i)})p(y_t|x_t^{(i)})}{p(x_{0:t-1}^{(i)})p(x_t|x_{t-1}^{(i)})}$$

so that

$$q(x_t|x_{0:t-1}^{(i)}, y_t) = p(x_t|x_{t-1}^{(i)}).$$

Hence, we have

$$w(x_{0:t}^{(i)}) \propto w(x_{0:t-1}^{(i)})p(y_t|x_t^{(i)}).$$

The prior distribution is the most popular choice on importance density. The reason for this is that the prior distribution is easy to sample and incremental weight depends only on the likelihood of the observations. However, it provides a poor estimation especially in case that prior is non-informative or very informative.

If the likelihood distribution is more informative than the prior distribution and it is integrable over x_t, then likelihood distribution can be chosen as importance density as follows [31]:

$$q(x_t|x_{t-1}^{(i)}, y_t) \propto p(y_t|x_t),$$

which yields

$$w(x_{0:t}^{(i)}) \propto w(x_{0:t-1}^{(i)})p(x_t|x_{t-1}^{(i)}).$$

Another choice for the importance density is the posterior distribution itself which is called *optimal importance density* [17]. Optimal importance density leads second term of the right hand side of (2.35) to be written as:

$$q_{\text{optimal}}(x_t|x_{t-1}^{(i)}, y_t) = p(x_t|x_{t-1}^{(i)}, y_t) = \frac{p(y_t|x_t)p(x_t|x_{t-1}^{(i)})}{p(y_t|x_{t-1}^{(i)})}. \tag{2.41}$$

Therefore, the recursive importance weight becomes

$$w(x_{0:t}^{(i)}) \propto w(x_{0:t-1}^{(i)})p(y_t|x_{t-1}^{(i)}) = w(x_{0:t-1}^{(i)}) \int p(y_t|x_t)p(x_t|x_{t-1}^{(i)})dx_t. \tag{2.42}$$

The equality holds due to the Chapman-Kolmogorov equation. Despite optimal importance density minimizes $V[\hat{w}(x_{0:n_t}^{(i)})]$ and maximizes N_{eff} given in (2.39), it has two major drawbacks. First, direct samples from posterior distribution is usually not possible and second, the integral in (2.42) cannot be evaluated easily.

Resampling

As a second method, resampling can be used to reduce degeneracy of the particles. The idea of resampling is eliminating particles with low weight while duplicating particles with high weights [17]. Recall that in IS, approximate posterior distribution $\hat{p}(x_{0:t}|y_{1:t})$ defined in (2.34) is based on random samples from importance density and can be considered as a discrete distribution. If we are interested in approximate random samples from the posterior distribution, we can basically sample from $\hat{p}(x_{0:t}|y_{1:t})$ according to the weights of particles. This process is called *resampling.* In short, resampling is sampling from an approximation which consists of samples. At the end, number of offspring $\{N^{(i)} : i = 1, 2, \ldots, N\}$ are associated to each particle $\{x_{0:t}^{(i)} : i = 1, 2, \ldots, N\}$. Then, particles with $N^{(i)} = 0$ are eliminated and remaining particles are approximately distributed according to $p(x_{0:t}|y_{1:t})$. Resampling procedure can be summarized in the following algorithm:

1. Consider weights $w_k^{(i)}$ as the probability of obtaining ith particle from the set $\{x_k^{(i)} : i = 1, 2, \ldots, N\}$.

2. Draw N samples from Multinomial distribution with parameters $(N, w_k^{(i)})$ and set as new sample set.

3. Assign $\dfrac{1}{N}$ to all weights.

The described algorithm is the simplest, unbiased resampling method and it is called Multinomial resampling. Other improved unbiased resampling methods with smaller variance and their comparisons can be found in [15].

Resampling need not be performed at every time step. If k is a predefined constant, resampling can be performed at every kth step. The advantage of this method is that it is unbiased. Moreover, resampling can be performed only when it is needed. Necessity of resampling can be measured by approximate effective sample size given by (2.40). If $\hat{N}_{\text{eff}} << N$, then resampling is performed. The method is called *adaptive resampling.*

Sequential importance resampling or particle filter is a combination of sequential

importance sampling and resampling. The general algorithm can be stated as follows [16, 43]:

0. Initial distribution, transition and measurement equations as in (2.5) - (2.7) and importance density are given.
1. Sample N particles $x_0^{(i)}$, $i = 1, 2, \ldots, N$ from prior distribution $p(x_0)$. Also, set initial importance weights $w(x_0^{(i)})$ to $\frac{1}{N}$.
2. Draw N random samples from importance distribution $q(x_{0:t}|y_{1:t})$.
3. Calculate importance weights by (2.32) and normalized importance weights by (2.33).
4. Generate a new random sample set by using one of the resampling methods explained above, for instance, adaptive resampling.
5. Approximate filtering distribution:

$$p(x_t|y_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta(x_t - x_t^{(i)}) \tag{2.43}$$

If importance density is chosen as $p(x_t|x_{t-1})$, then particle filter is called *bootstrap particle filter*.

Up to now, filtering algorithms are considered. These algorithms provide filtering distribution, approximate posterior distribution for the state of a dynamic system and estimate for state itself according to measurements given before and current time step. Another approach for the state estimation is based on smoothing algorithms. These algorithms provide smoothing distribution based on measurements observed for some later time. Indeed, $p(x_t|y_{1:T})$ should be obtained when $T > t$ [43].

2.3 Smoothing for Non-linear State-Space Models

Historically, smoothing algorithms have not get enough attention because of the following two reasons [35]: (i) Computational cost is high when it is compared to particle filters, (ii) They require strong assumptions that are not usually satisfied in real

systems. However, smoothing algorithms provide more accurate estimations because of the usage of future measurements to estimate the current state [11].

The simplest way of calculating the smoothing distribution $p(x_t|y_{1:T})$ is very similar to the case in the filtering distribution. First, calculate the joint posterior distribution $p(x_{0:T}|y_{1:T})$ according to (2.12). Then, marginalize over $x_{1:t-1}$ and $x_{t+1:T}$. Besides, recursive formulation which is called "*forward filtering - backward smoothing (FFBS)*" also exists [43].

2.3.1 Forward Filtering - Backward Smoothing

The method is given that name since firstly all the prediction $p(x_t|y_{1:t-1})$ and the filtering distributions $p(x_t|y_{1:t})$ over the time steps $t = 1, 2, \ldots, T$ are computed. Then, smoothing distributions are obtained recursively backward in time starting from time step T [6, 17, 35]. To derive this, the smoothing distribution can be factorized as follows:

$$p(x_t|y_{1:T}) = \int p(x_t, x_{t+1}|y_{1:T})dx_{t+1} \tag{2.44}$$

$$= \int p(x_t|x_{t+1}, y_{1:T})p(x_{t+1}|y_{1:T})dx_{t+1}, \tag{2.45}$$

where

$$\begin{aligned} p(x_t|x_{t+1}, y_{1:T}) &= p(x_t|x_{t+1}, y_{1:t}) \\ &= \frac{p(x_t, x_{t+1}, y_{1:t})}{p(x_{t+1}, y_{1:t})} = \frac{p(x_t, y_{1:t})p(x_{t+1}|x_t, y_{1:t})}{p(x_{t+1}|y_{1:t})p(y_{1:t})} \\ &= \frac{p(x_t|y_{1:t})p(x_{t+1}|x_t, y_{1:t})}{p(x_{t+1}|y_{1:t})} \\ &= \frac{p(x_t|y_{1:t})p(x_{t+1}|x_t)}{p(x_{t+1}|y_{1:t})}. \end{aligned} \tag{2.46}$$

The first and the line of equations are the result of Markov property of measurements and states. When (2.46) is inserted into (2.44), then the following FFBS recursion will be obtained:

$$p(x_t|y_{1:T}) = p(x_t|y_{1:t}) \int \frac{p(x_{t+1}|x_t)p(x_{t+1}|y_{1:T})}{p(x_{t+1}|y_{1:T})}dx_{t+1}, \tag{2.47}$$

where $p(x_{t+1}|y_{1:T}) = \int p(x_{t+1}|x_t)p(x_{t+1}|y_{1:T})dx_t$ by Chapman-Kolmogorov equation.

Note that in (2.47) the term $p(x_{t+1}|y_{1:T})$ in the numerator is the smoothing distribution obtained just in the previous time step. Another method to obtain the smoothing distribution is based on two filtering distributions. The method is called *Two-Filter Smoothing (TFS)*, which we describe next.

2.3.2 Two-Filter Smoothing

In this approach, smoothing distribution is obtained by combination of two filters [6, 35, 43]. The first one is the standard Bayesian filter. The second one is the backward information filter which will be defined below.

To derive TFS, the following factorization of smoothing distribution is used:

$$\begin{aligned} p(x_t|y_{1:T}) &= p(x_t|y_{1:t-1}, y_{t:T}) = \frac{p(x_t, y_{1:t-1}, y_{t:T})}{p(y_{1:t-1}, y_{t:T})} \\ &= \frac{p(y_{t:T}|x_t, y_{1:t-1})p(x_t|y_{1:t-1})}{p(y_{t:T}|y_{1:t-1})} \\ &= \frac{p(y_{t:T}|x_t)p(x_t|y_{1:t-1})}{p(y_{t:T}|y_{1:t-1})}. \end{aligned} \tag{2.48}$$

The last line of (2.48) is due to conditional independence of measurements. As a result (2.48) implies that,

$$p(x_t|y_{1:T}) \propto p(y_{t:T}|x_t)p(x_t|y_{1:t-1}). \tag{2.49}$$

It can be seen that the second term of proportionality is the prediction step of Bayesian filter. The first term is the backward information filter and it is computed as:

$$\begin{aligned} p(y_{t:T}|x_t) &= p(y_t, y_{t+1:T}|x_t) = p(y_t|x_t)p(y_{t+1:T}|x_t) \\ &= p(y_t|x_t) \int p(y_{t+1:T}, x_{t+1}|x_t)dx_{t+1} \\ &= p(y_t|x_t) \int \frac{p(y_{t+1:T}, x_{t+1}, x_t)}{p(x_t)}dx_{t+1} \\ &= p(y_t|x_t) \int p(y_{t+1:T}|x_{t+1}, x_t)p(x_{t+1}|x_t)dx_{t+1}. \end{aligned}$$

Hence,

$$p(y_{t:T}|x_t) = \int p(y_{t+1:T}|x_{t+1}, x_t)p(x_{t+1}|x_t)p(y_t|x_t)dx_{t+1}.$$

Closed form solution for the integral defined above is not possible except for linear cases. Another drawback of the two filter smoothing formula is that SMC methods are not applicable to approximate the information filter. The reason is that $p(y_{t:T}|x_t)$ is not probability distribution of x_t. Therefore, it cannot be normalized with respect to x_t. Indeed, there is a possibility of having $\int p(y_{t:T}|x_t)dx_t = \infty$. To overcome this problem, *artificial probability distribution*, denoted by $\gamma_(x_t)$, with the following properties is proposed in the following [6, 35, 43].

1. $\gamma_t(x_t) > 0$ whenever $p(y_{t:T}|x_t) > 0$.
2. Define $\tilde{p}(x_T|y_T) = \dfrac{\gamma_t(x_t)p(y_T|x_T)}{p(y_T)}$ where $p(y_T) = \int \gamma_t(x_t)p(y_T|x_T)dx_T$
3. Based on the previous property and property of joint distributions define also

$$\tilde{p}(x_{t:T}|y_{t:T}) = \frac{\gamma_t(x_t)\prod_{n=t+1}^{T} p(x_{n+1}|x_n)\prod_{n=t}^{T} p(y_n|x_n)}{\tilde{p}(y_{t:T})},$$

where

$$\tilde{p}(y_{t:T}) = \int \dots \int \gamma_t(x_t)\prod_{n=t+1}^{T} p(x_{n+1}|x_n)\prod_{n=t}^{T} p(y_n|x_n)dx_{t:T+1}.$$

Then, backward information filter can be rewritten as follows:

$$\begin{aligned} p(y_{t:T}|x_t) &= \int \dots \int p(y_{t:T}, x_{t+1:T}|x_t)dx_{t+1:T} = \int \dots \int \frac{p(y_{t:T}, x_{t:T})}{p(x_t)}dx_{t+1:T} \\ &= \int \dots \int p(y_{t:T}|x_{t:T})p(x_{t+1:T}|x_t)dx_{t+1:T} \end{aligned}$$

Joint probability distribution can be written as the product of the marginal distributions:

$$p(y_{t:T}|x_t) = \int \dots \int \prod_{n=t}^{T} p(y_n|x_n)\prod_{n=t+1}^{T} p(x_n|x_{n-1})dx_{t+1:T}$$

Multiplying and dividing by artificial distribution $\gamma_t(x_t)$

$$p(y_{t:T}|x_t) = \int \dots \int \frac{\gamma_t(x_t)}{\gamma_t(x_t)} \prod_{n=t}^{T} p(y_n|x_n) \prod_{n=t+1}^{T} p(x_n|x_{n-1}) dx_{t+1:T}$$

According to property 3 above, we have

$$p(y_{t:T}|x_t) = \int \dots \int \frac{\tilde{p}(x_{t:T}|y_{t:T})\tilde{p}(y_{t:T})}{\gamma_t(x_t)} dx_{t+1:T} = \frac{\tilde{p}(x_t|y_{t:T})\tilde{p}(y_{t:T})}{\gamma_t(x_t)}.$$

or equivalently,

$$p(y_{t:T}|x_t) \propto \frac{\tilde{p}(x_t|y_{t:T})}{\gamma_t(x_t)}.$$

Two-step recursive formula for $\tilde{p}(x_t|y_{t:T})$ can be obtained. Detail of the derivation is found in [6]. The recursion begins with probability density which is defined in the second property of artificial distribution.Below is the two step recursive formula for $\tilde{p}(x_t|y_{t:T})$

Prediction step: $\tilde{p}(x_t|y_{t+1:T}) = \int \tilde{p}(x_{t+1}|y_{t+1:T}) \frac{p(x_{t+1}|x_t)\gamma_t(x_t)}{\gamma_{t+1}(x_{t+1})} dx_{t+1}$

Update step :$\tilde{p}(x_t|y_{t:T}) = \frac{p(y_t|x_t)\tilde{p}(x_t|y_{t+1:T})}{\int p(y_t|x_t)\tilde{p}(x_t|y_{t+1:T})dx_t}$7

This form of information filter enables us to derive two-filter smoothing algorithms for non-linear state space models as well as particle smoothing algorithms.

In Section 2.2, it is stated that SMC methods provide approximation to the filtering distribution. The same idea can be applied to the smoothing distribution as well.

2.3.3 Particle Smoothing

If Monte Carlo methods are used to approximate smoothing methods, then such a smoothing algorithm is called *particle smoother* [43].

2.3.3.1 Sequential Importance Resampling Smoother

SIR can also be considered to approximate smoothing distribution $p(x_t|y_{1:T})$ for $t = 1, 2, \dots, T$. The solution is obtained by keeping whole state history $x_{0:T}$. The

algorithm below will be similar to the SIR except for the approximation by Monte Carlo method [17].

0. Initial distribution, transition and measurement equations as in (2.5) - (2.7) and importance density are given.

1. Sample N particles $x_0^{(i)}$, $i = 1, 2, \ldots, N$ from prior distribution $p(x_0)$. Also, set initial importance weights $w(x_0^{(i)})$ to $\frac{1}{N}$.

2. For each time step $k = 1, 2, \ldots, t$:

   - Draw samples $x_k^{(i)} \sim q(x_t|x_{0:t-1}, y_t)$.
   - Calculate weights $w(x_{0:t}^{(i)})$ according to (2.36).
   - Normalize the importance weights and assign them to $\tilde{w}(x_{0:t}^{(i)})$.
   - Set $x_{0:t}^{(i)} = \{x_{0:t-1}, x_t\}$.
   - Resample particles if it is necessary

3. Find approximation for the smoothing distribution as:

$$p(x_t|y_{1:T}) \approx \sum_{i=1}^{N} w_T^{(i)} \delta(x_t - x_t^{(i)})$$

The disadvantage of SIR smoother is that if T is much bigger than t, then it generates poor approximations to smoothing distributions. Better approximations can be obtained if filtering distributions are used instead of past states.

2.3.3.2 Backward Simulation Particle Smoothing(BSPS)

Assume that particle filtering is processed and approximations to the filtering distributions are obtained for each time step from 1 up to T. Constitute weighted set of particles as $\{(w_t^{(i)}, x_t^{(i)}) : i = 1, 2, \ldots, N, t = 1, 2, \ldots, T\}$. BSPS can be derived based on the following fact: according to (2.46)

$$p(x_t|x_{t+1}, y_{1:T}) = \frac{p(x_{t+1}|x_t)p(x_t|y_{1:t})}{p(x_{t+1}|y_{1:t})}$$

Also, approximation to filtering distribution is given by (2.43). Hence, if this approximation is substituted into the above equation we get:

$$p(x_t|x_{t+1}, y_{1:T}) \propto \sum_{i=1}^{N} w_t^{(i)} p(x_{t+1}|x_t)\delta(x_t - x_t^{(i)}). \tag{2.50}$$

As a result, pseudo-algorithm may be stated as follows:

0. Obtain set of particles and associated weights $\{(x_t^{(i)}, w_t^{(i)}) : i = 1, 2, \ldots, N, t = 1, 2, \ldots, T\}$ through particle filter algorithm.
1. Choose $x_T = x_T^{(i)}$ with probability $w_T^{(i)}$
2. For each time step $t = T-1, T-2, \ldots, 1$:

   Calculate $\hat{w}_t^{(i)} = w_t^{(i)} p(x_{t+1}|x_t^{(i)})$

   Choose $x_t = x_t^{(i)}$ according to the weights $\hat{w}_t^{(i)}$.

   Obtain the following approximation:

$$p(x_{0:T}|y_{1:t}) = \sum_{i=1}^{N} \frac{1}{N} \delta(x_{0:T}|y_{1:t})$$

2.4 Application

In this section bootstrap particle filtering is applied to the following model:

$$\emptyset \xrightarrow{c_1} 10\ S,$$
$$S \xrightarrow{c_2} \emptyset,$$

where reaction constants of the first and the second reactions are $c_1 = 1$ and $c_2 = 2$, respectively.

Recall that if the importance density is chosen as $p(x_t|x_{t-1})$, then particle filter is called bootstrap particle filter. So, importance weights become

$$w_t^{(i)} = \frac{p(x_{0:t}|y_{1:t})}{q(x_{0:t}|y_{1:t})} = \frac{p(x_{0:t-1}|y_{1:t-1})p(y_t|x_t)p(x_t|x_{t-1})}{p(x_t|x_{t-1})} \propto p(y_t|x_t). \tag{2.51}$$

In the model given above, output of Gillespie's direct method is used as sample from the importance density. The bootstrap algorithm, based on [16] proceeds as follows:

0. *Initialization:* Determine the number of particles N. Set time steps $t = t_0, t_1, \ldots, t_M$. Specify the measurements.

1. For $t = 0$, draw N random samples from initial distribution, which is chosen as Normal distribution. At the end, we have $\{x_0^{(i)}, i = 1, 2, \ldots, N\}$

2. *Importance Sampling Step:* At time step $t = t_j$, $j = 1, 2, \ldots, M$, run the Gillespie's algorithm for each particle to obtain N random samples from importance density. At the end, we have $\{x_{0:t}^{(i)} : i = 1, 2, \ldots, N\}$

3. Calculate the importance weights according to (2.51) and normalize them. At the end, we have $\{w_t^{(i)}, i = 1, 2, \ldots, N\}$.

4. *Resampling:*Apply multinomial resampling according to [15]. At the end, N particles will be obtained according to their normalized importance weights. In other words, particles with low importance weights will be eliminated while particles with high importance weights will be sampled more than once.

5. Set $j = j + 1$ and return to the step 2.

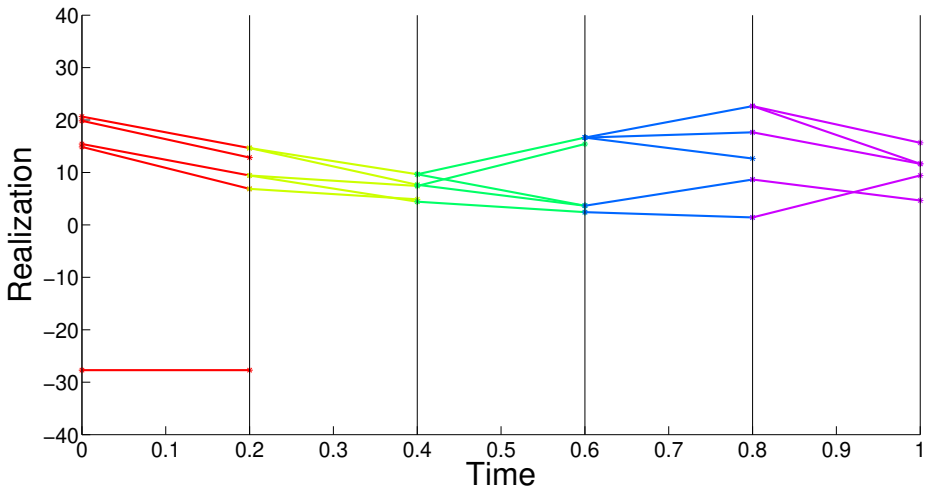


Figure 2.4: Elimination and replication of particles in bootstrap algorithm

In Figure 2.4, particle filter algorithm begins with 5 particles, and time steps are taken to be $\{0, 0.2, \ldots, 1.00\}$. At the end of time interval $[0, 0.2]$, the first and the third particles are duplicated because of high importance weight while the second and the last particles are eliminated because of low importance weights. At each time intervals, weights are calculated and resampling is applied to the particles.

Filtering distribution estimates obtained with 500,1000 and 2000 particles are depicted in Figures 2.5-2.7:

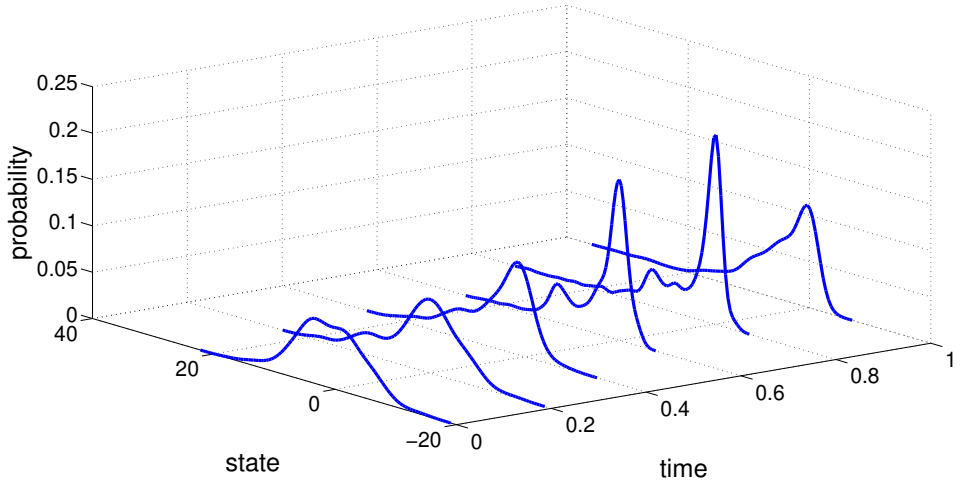


Figure 2.5: Estimated probability distributions for $p(x_t|y_{1:t})$ with 500 particles

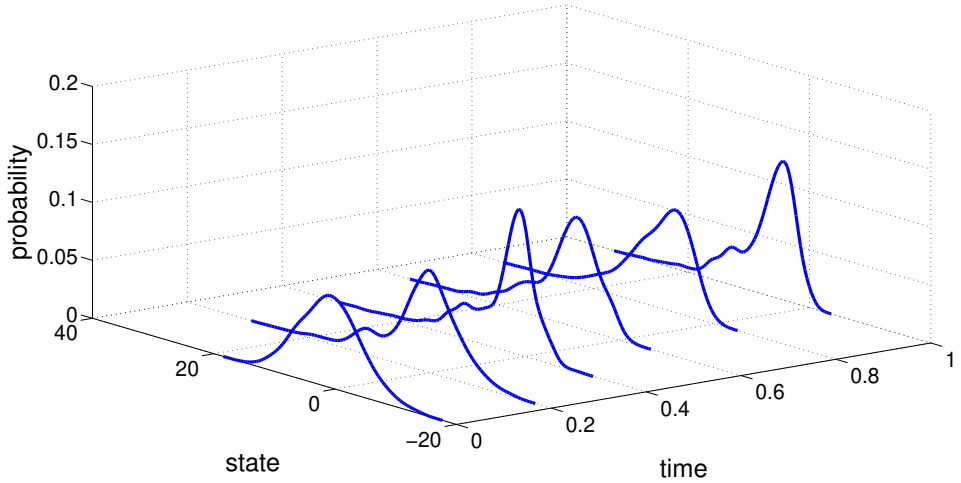


Figure 2.6: Estimated probability distributions for $p(x_t|y_{1:t})$ with 1000 particles

When number of the particles is increased, variance of the filtering distribution become smaller which means that more accurate estimation is obtained. Changing the number of particles allows us trade off the accuracy of estimation for rapid results. Indeed, if the number of particles is reduced, there will be less work to do but estimation will be further from the exact result.

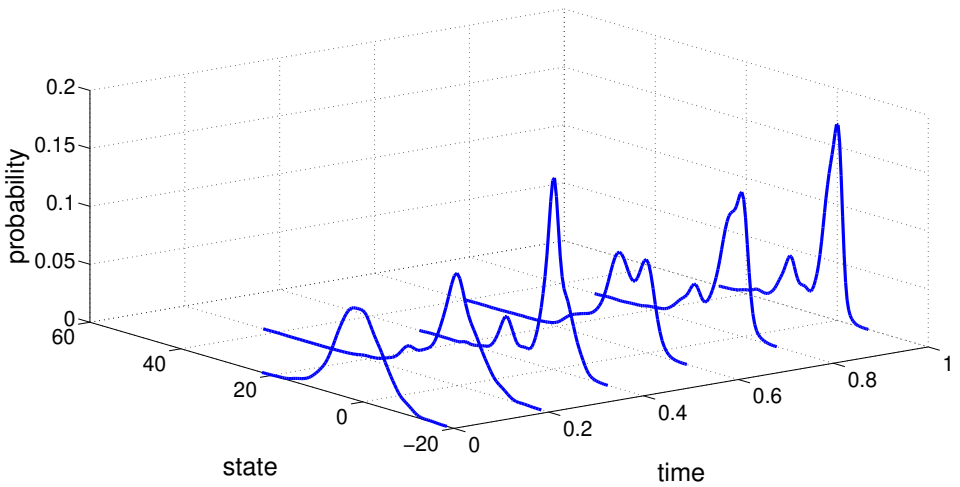


Figure 2.7: Estimated probability distributions for $p(x_t|y_{1:t})$ with 2000 particles

CHAPTER 3

CONCLUSION

In the first chapter of this thesis, different mathematical models and simulation algorithms to describe the time evolution of a biochemical reaction network and their relationships to each other are investigated. Mathematical models are categorized based on whether uncertainty is included into the model through random variables and probability distributions. After the introduction of a general chemical reaction channel, deterministic model, based on numerical integration of set of first-order ODE system, is considered. There, the system's state is taken to be concentrations. Also, randomness in molecule abundances and external noises are ignored. Finally, unique solution for given initial state is obtained.

Then, stochastic approaches are examined beginning from CTMC. If molecular populations of species in a biochemical reaction system is low so that it can be represented as integer values, i.e., discrete state space, then the state of the system satisfies RTCM. Further, time evolution of the corresponding probability mass distribution is given by CME. If the species in the reaction network have high copy number, then the time evolution of the state can be approximated by a diffusion process. The state of the system in terms of concentrations can be represented as stochastic integral equation which satisfies stochastic differential equation, namely CLE. Further, in case thermodynamic limit condition is satisfied, CLE approaches to a system of ODEs. The time evolution of the corresponding probability density distribution is given as CFPE. Moreover, if all the reactions in the network is unimolecular, expected value of the realizations of the state, which satisfies CME or CFPE, satisfies ODE system, as well. In summary, relationships between different mathematical models can be seen in Fig-

ure 3.1.

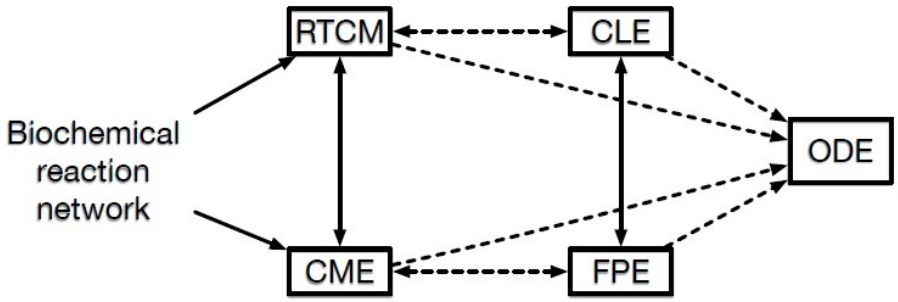


Figure 3.1: Relationships between different mathematical models. [1]

Another method to analyze the dynamics of biochemical reaction network is finding realizations of the state of the system under consideration. SSAs can be categorized according to the methodology of finding the solution. Exact SSAs provide the exact solution but these algorithms require high computational time. To reduce this computational time, approximate SSAs which scarify the exactness of the solution, are proposed. Also, as an alternative hybrid SSAs, combining both exact and approximate algorithms, are derived.

In the application section, deterministic and stochastic modeling approaches are applied to biological systems, LV model, MM enzyme kinetics and JACK-STAT signaling pathway. Also, numerical solution of ODE system and realizations obtained through Gillespie's SSA with direct method are compared.

In the second chapter, state estimation techniques for the general state-space models are reviewed. The aim is making inference on the state of a dynamic system through measurements. It is usually accomplished by obtaining filtering distribution $p(x_t|y_{1:t})$. The simplest method is a direct application of Bayes' rule and recursive version of it to reduce computational cost. However, these methods do not provide analytical solutions because of the difficulty of high dimensional integration in normalizing constant and marginal of joint posterior distribution. To overcome these problem, different filtering algorithms are proposed. For example Kalman filter for linear Gaussian state space models and grid-based filter for the finite state-space model provide more accurate estimation for the probability distribution. Also, to extend feasibility, the methods such as extended Kalman filter and approximate grid

based filter are proposed. As an alternative to approximation of filtering distribution, Monte Carlo methods might be used. In addition to Monte Carlo sampling, IS to overcome sampling from the posterior distribution and SIS to reduce computational cost are addressed. Also, different choice of resampling methods and importance densities exist to overcome degeneracy problem due to increase in dimension of the state space.

Another way of obtaining estimation for the state of a dynamic system is attaining its smoothing distribution $p(x_t|y_{1:T})$, for $T > t$. Smoothing algorithms require filtering distributions at each time step; this increases the computational cost of the smoothing algorithms. However, smoothing algorithms provide more accurate information about the state of the dynamic system by means of the measurements available for later time. Two recursive smoothing algorithms reviewed in this thesis are FFBS and TFS. Monte Carlo methods can also be used to approximate smoothing distribution. SIR smoother includes direct usage of Monte Carlo sampling. Yet, increase in time step T yields decrease in efficiency of the algorithm. Moreover, BSPS provides approximation to smoothing distribution through approximated filtering distributions.

In the application section of this chapter, bootstrap particle filter is applied to the birth-death process. Posterior distribution is obtained by using different number of particles. It can be seen that variances is getting smaller for each time steps, i.e., accurate approximations are obtained, when the number of particles is increased.

Statistical inference based algorithms do not provide exact results and it is always possible to obtain better estimations. For example, theoretically we know that using optimal importance density instead of prior distribution leads to more accurate filtering distribution in bootstrap particle filter. This study may be a starting point to develop enhanced filtering algorithms as a future work. Besides, it is also possible to apply particle filtering algorithms to complex models differ from biological processes.

REFERENCES

[1] D. Altıntan, J. Diemer, and H. Koeppl, Analysis and modeling of single cell data, in J. Dash and C. Jayaprakashi, editors, *Systems Immunology*, chapter 10, CRC Press, 2017.

[2] D. F. Anderson and T. G. Kurtz, Continuous markov chain models for chemical reaction networks, in H. Koeppl, G. Setti, d. M. Bernardo, and D. Densmore, editors, *Design and Analysis of Biomolecular Circuits*, chapter 1, Springer-Verlag, 2011.

[3] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking, IEEE Transactions on Signal Processing, 50, pp. 174–188, 2002.

[4] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*, John Wiley & Sons, Inc., 2001.

[5] E. Bibbona and R. Sirovich, Strong approximation of density dependent markov chains on bounded domains, arXiv:1704.07481v2, pp. 1–27, 2017.

[6] M. Briers, A. Doucet, and S. Maskell, Smoothing algorithms for state–space models, Annals of the Institute of Statistical Mathematics, 62, pp. 61–89, 2009.

[7] K. Burrage, P. M. Burrage, and B. Tian, Numerical methods for strong solution of stochastic differential equations : an overview, Proc. R. Soc. London A., 460(2), pp. 373–402, 2004.

[8] K. Burrage, T. Tian, and B. P., A multi-scaled approach for simulating chemical reaction systems., Prog. Biophys. Mol. Biol., 85, pp. 217–234, 2004.

[9] Y. Cao, D. T. Gillespie, and L. R. Petzold, Efficient step size selection for the tau-leaping simulation method, The Journal of Chemical Physics, 124, p. 044109, 2006.

[10] Y. Cao, H. Li, and L. Petzold, Efficient formulation of the stochastic simulation algorithm for chemically reacting system, Journal of Chemical Physics, 121(9), pp. 4059–4067, 2004.

[11] O. Cappe, S. J. Godsill, and E. Moulines, An overwiev of existing methods and recent advances in sequential monte carlo, Proceedings of the IEEE, 95(5), pp. 899–924, 2007.

[12] A. Chatterjee, D. G. Vlachos, and M. A. Katsoulakis, Binomial distribution based tau-leap accelerated stochastic simulation, The Journal of Chemical Physics, 122, p. 024112, 2005.

[13] Z. Chen, Bayesian filtering: From kalman filters to particle filters, and beyond, Journal of Theoretical and Applied Statistics, 182, 2003.

[14] S. Ditlevsen and A. Samson, Introduction to stochastic models in biology, in M. B. et al., editor, *Stochastic Biomathematical Models, Lecture Notes in Mathematics*, chapter 1, Springer-Verlag Berlin Heidelberg, 2013.

[15] R. Douc and O. Cappe, Comparison of resampling schemes for particle filtering, in *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005.*, pp. 64–69, 2005, ISSN 1845-5921.

[16] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo methods in practice*, New York : Springer, 2001.

[17] A. Doucet and A. Johansen, A tutorial on particle filtering and smoothing: Fifteen years later, Handbook of Nonlinear Filtering, 12, 2009.

[18] R. Emori and D. Schuring, *Scale Models in Engineering*, Pergamon Press, 1977.

[19] R. Faragher, Understanding the basis of the kalman filter via a simple and intuitive derivation, IEEE Signal Processing Magazine, pp. 128–132, 2012.

[20] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*, Chapman & Hall/CRC, 2004.

[21] M. A. Gibson and J. Bruck, Efficient exact stochastic simulation of chemical systems with many species and many channels, The Journal of Physical Chemistry A, 104(9), pp. 1876–1889, 2000.

[22] D. T. Gillespie, A general method for numerically simulating the stochastic time evolution of coupled chemical reactions, Journal of Computational Physics, 22, pp. 403 – 434, 1976.

[23] D. T. Gillespie, Exact stochastic simulation of coupled chemical reactions, Journal of Chemical Physics, 81(25), pp. 2340 – 2361, 1977.

[24] D. T. Gillespie, A rigorous derivation of the chemical master equation, Physica A, 188, pp. 404–425, 1992.

[25] D. T. Gillespie, Approximate accelerated stochastic simulation of chemically reacting systems, Journal of Chemical Physics, 115, pp. 1716–1733, 2001.

[26] D. T. Gillespie, The chemical langevin and fokker-planck equations for the reversible isomerization reaction, Journal of Physical Chemistry, 106, pp. 5063–5071, 2001.

[27] D. T. Gillespie, The chemical langevin equation, Journal of Chemical Physics, 113(1), pp. 297 – 306, 2001.

[28] D. T. Gillespie, Stochastic simulation of chemical kinetics, The Annual Review of Physical Chemistry, 58, pp. 35–55, 2007.

[29] D. T. Gillespie and L. R. Petzold, Improved leap-size selection for accelerated stochastic simulation, Journal of Chemical Physics, 119(16), pp. 8229–8234, 2003.

[30] D. F. Griffiths and D. J. Higham, *Numerical Methods for Ordinary Differential Equations Initial Value Problems*, Springer, 2010.

[31] F. Gustafsson, Particle filter theory and practice with positioning applications, Aerospace and Electronic Systems Magazine, IEEE, 25, pp. 53 – 82, 2010.

[32] M. Jackson, Some notes on models and modeling, in A. B. et al., editor, *Conceptual Modeling: Foundations and Applications Essays in Honor of John Mylopoulos*, pp. 68–81, Springer-Verlag Berlin Heidelberg, 2009.

[33] D. Khoshnevisan, Lecture notes on donsker's theorem, 2005.

[34] T. R. Kiehl, R. M. Mattheyses, and M. K. Simmons, Hybrid simulation of cellular behavior, Bioinformatics, 20, pp. 316–322, 2004.

[35] M. Klaas, M. Briers, N. de Freitas, A. Doucet, S. Maskell, and D. Lang, Fast particle smoothing: If i had a million particles, ACM International Conference Proceeding Series, 148, pp. 481–488, 2006.

[36] E. Klipp, W. Liebermeister, C. Wierling, A. Kowald, H. Lehrach, and R. Herwig, *Systems Biology: A Textbook*, WILEY-VCH Verlag GmbH & Co. KGaA, 2009.

[37] L. Marchetti, C. Priami, and V. H. Thanh, *Simulation Algorithms for Computational Systems Biology*, Springer, 2017.

[38] J. Pahle, *Stochastic simulation and analysis of biochemical networks*, Ph.D. thesis, Humboldt University, 2008.

[39] H. Petrucci, Ralph, F. Herring, Geoffrey, D. Madura, Jeffry, and C. Bissonnette, *Genaral Chemistry: Principles and Modern Applications*, Pearson, 2011.

[40] J. Puchalka and A. M. Kierzek, Briding the gap between stochastic and deterministic regimes in the kinetic simulation of the biochemical reaction networks, Biophysical Journal, 86, pp. 1357–1372, 2004.

[41] P. Renard, A. Alcolea, and D. Ginsbourger, Stochastic versus deterministic approaches, in J. Wainwright and M. Mulligan, editors, *Environmental Modelling: Finding Simplicity in Complexity*, chapter 8, John Wiley & Sons, Ltd., 2013.

[42] K. Sanft, D. T. Gillespie, and L. Petzold, Legitimacy of the stochastic michaelis menten approximation, IET Systems Biology, 5, pp. 58–69, 2011.

[43] S. Sarkka, *Bayesian Filtering and Smoothing*, Cambridge University Press, 2013.

[44] P. Smadbeck and Y. Kaznessis, Stochastic model reduction using a modified hill-type kinetic rate law, The Journal of Chemical Physicsl, 137, 2012.

[45] G. Welch and G. Bishop, An introduction to the kalman filter, UNC-Chapel Hill, 2006.

[46] D. J. Wilkinson, *Stochastic Modelling for System Biology*, Chapman & Hall/CRC, 2006.

[47] C. Zechner, *Stochastic Biochemical Networks in Random Environments: Probabilistic Modeling and Inference*, Ph.D. thesis, ETH Zurich, 2014.

[48] J. Zhang, *Numerical Methods for the Chemical Master Equation*, Ph.D. thesis, the Faculty of the Virginia Polytechnic Institute and State University, 2009.

APPENDIX A

JACK-STAT SIGNALING PATHWAY

Reactions

$R_1 : ReceptorIFNAR1 + TYK \leftrightharpoons ReceptorTYKComplex$
$R_2 : ReceptorIFNAR2 + JAK \leftrightharpoons ReceptorJAKComplex$
$R_3 : ReceptorJakComplex + ReceptorTykComplex + IFN\_free \leftrightharpoons IFNARdimer$
$R_4 : IFNARdimer \longrightarrow ActiveReceptorComplex$
$R_5 : STAT2c\_IRF9 + ActiveReceptorComplex \longrightarrow ActiveReceptorComplex\_STAT2c + IRF9c$
$R_6 : STAT2c + ActiveReceptorComplex \leftrightharpoons ActiveReceptorComplex\_STAT2c$
$R_7 : STAT1c + ActiveReceptorComplex\_STAT2c \leftrightharpoons ActiveReceptorComplex\_STAT2c\_STAT1c$
$R_8 : ActiveReceptorComplex\_STAT2c\_STAT1c \longrightarrow ActiveReceptorComplex + STAT1c^*\_STAT2c^*$
$R_9 : IRF9c + STAT1c^*\_STAT2c^* \leftrightharpoons ISGF - 3c$
$R_{10} : ISGF - 3c \leftrightharpoons ISGF - 3n$
$R_{11} : STAT1c^*\_STAT2c^* \leftrightharpoons STAT1n^*\_STAT2n^*$
$R_{12} : STAT1n^*\_STAT2n^* + IRF9n \leftrightharpoons ISGF - 3n$
$R_{13} : ISGF - 3n + FreeTranscriptionFactorBindingSite(TFBS) \leftrightharpoons OccupiedTFBS$
$R_{14} : \emptyset \longrightarrow mRNAn$
$R_{15} : mRNAn \leftrightharpoons mRNAc$
$R_{16} : \emptyset \leftrightharpoons IRF9c$
$R_{17} : \emptyset \leftrightharpoons SOCS$
$R_{18} : ActiveReceptorComplex \leftrightharpoons ReceptorIFNAR1 + ReceptorIFNAR2 + JAK + TYK$
$R_{19} : ActiveReceptorComplex \longrightarrow IFNARdimer$
$R_{20} : IRF9n \longrightarrow \emptyset$
$R_{21} : STAT2c\_IRF9 \longrightarrow STAT2c$
$R_{22} : STAT2n\_IRF9 \longrightarrow STAT2n$
$R_{23} : ISGF - 3c + Cytoplasmicphosphatase(CP) \leftrightharpoons ISGF - 3c\_CP$
$R_{24} : ISGF - 3c\_CP \longrightarrow STAT1c + STAT2c + CP + IRF9c$
$R_{25} : STAT1c^*\_STAT2c^* + CP \leftrightharpoons STAT1c^*\_STAT2c^*\_CP$
$R_{26} : STAT1c^*\_STAT2c^*\_CP \longrightarrow STAT1c + STAT2c + CP$
$R_{27} : STAT1n^*\_STAT2n^* + Nuclearphosphatase(NP) \leftrightharpoons STAT1n^*\_STAT2n^*\_NP$
$R_{28} : STAT1n^*STAT2n^*\_NP \longrightarrow STAT1n + STAT2n + NP$
$R_{29} : ISGF - 3n + NP \leftrightharpoons ISGF - 3n\_NP$
$R_{30} : ISGF - 3n\_NP \longrightarrow STAT1n + STAT2n + NP + IRF9n$
$R_{31} : OccupiedTFBS + NP \leftrightharpoons OccupiedTFBSNP$

Table A.1: List of reactions for JAK-STAT signaling pathway

Reactions

$R_{32} : OccupiedTFBS\_NP \longrightarrow STAT1n + STAT2n + IRF9n + FreeTFBS + IRF9n$
$R_{33} : PIAS + ISGF - 3n \leftrightharpoons PIAS\_ISGF - 3n$
$R_{34} : mRNAc \longrightarrow \emptyset$
$R_{35} : STAT1c \leftrightharpoons STAT1n$
$R_{36} : STAT2c \leftrightharpoons STAT2n$
$R_{37} : STAT2c + IRF9c \leftrightharpoons STAT2c\_IRF9$
$R_{38} : STAT2n + IRF9n \leftrightharpoons STAT2n\_IRF9$
$R_{39} : STAT2c\_IRF9 \leftrightharpoons STAT2n\_IRF9$
$R_{40} : IRF9c \leftrightharpoons IRF9n$
$R_{41} : IFN\_influx \longrightarrow IFN\_free$

Table A.2: List of reactions for JAK-STAT signaling pathway, (ctd')

Reaction Rate Constants

$R_1 : k_1 = 0.1 \quad k_{1^-} = 0.05$
$R_2 : k_2 = 0.1 \quad k_{2^-} = 0.05$
$R_3 : k_3 = 0.01 \quad k_{3^-} = 0.01$
$R_4 : k_4 = 0.005$
$R_5 : k_5 = 0.002$
$R_6 : k_6 = 0.002 \quad k_{6^-} = 4$
$R_7 : k_7 = 0.002 \quad k_{7^-} = 4$
$R_8 : k_8 = 8$
$R_9 : k_9 = 0.1 \quad k_{9^-} = 0.1$
$R_{10} : k_{10} = 0.015 \quad k_{10^-} = 0.015$
$R_{11} : k_{11} = 0.01 \quad k_{11^-} = 0.01$
$R_{12} : k_{12} = 0.1 \quad k_{12^-} = 0.1$
$R_{13} : k_{13} = 0.00025 \quad k_{13^-} = 0.01$
$R_{14} : k_{14} = 0.05$
$R_{15} : k_{15} = 0.045 \quad k_{15^-} = 0.045$
$R_{16} : k_{16} = 0.0005 \quad k_{16^-} = 0.0003$
$R_{17} : k_{17} = 0.000012 \quad k_{17^-} = 0.01$
$R_{18} : k_{18} = 0.0001 \quad k_{18^-} = 0.0001$
$R_{19} : k_{19} = 0.0001$
$R_{20} : k_{20} = 0.001$

Reaction Rate Constants

$R_{21} : k_{21} = 0.2$
$R_{22} : k_{22} = 0.003$
$R_{23} : k_{23} = 0.001 \quad k_{23^-} = 0.2$
$R_{24} : k_{24} = 0.003$
$R_{25} : k_{25} = 0.01 \quad k_{25^-} = 0.1$
$R_{26} : k_{26} = 0.01$
$R_{27} : k_{27} = 0.01 \quad k_{27^-} = 0.1$
$R_{28} : k_{28} = 0.1$
$R_{29} : k_{29} = 0.01 \quad k_{29^-} = 0.1$
$R_{30} : k_{30} = 0.002$
$R_{31} : k_{31} = 0.0001 \quad k_{31^-} = 0.1$
$R_{32} : k_{32} = 0.1$
$R_{33} : k_{33} = 0.1 \quad k_{33^-} = 0.1$
$R_{34} : k_{34} = 0.0005$
$R_{35} : k_{35} = 0.00125 \quad k_{35^-} = 0.01$
$R_{36} : k_{36} = 0.0000817 \quad k_{36^-} = 0.0014$
$R_{37} : k_{37} = 0.01 \quad k_{37^-} = 0.01$
$R_{38} : k_{38} = 0.01 \quad k_{38^-} = 0.01$
$R_{39} : k_{39} = 0.00125 \quad k_{39^-} = 0.0014$
$R_{40} : k_{40} = 0.02 \quad k_{40^-} = 0.005$
$R_{41} : k_{41} = 0.00069$

Table A.3: Reaction rate constants of JACK-STAT signaling pathway

Reaction Rate Equations

$\frac{d}{dt}[TYK] = k_{1\text{-}}[ReceptorTYKComplex] + k_{18}[ActiveReceptorComplex] - k_1[ReceptorIFNAR1][TYK]$

$\frac{d}{dt}[JAK] = k_{2\text{-}}[ReceptorJAKComplex] + k_{18}[ActiveReceptorComplex] - k_2[ReceptorIFNAR2][JAK]$

$\frac{d}{dt}[ReceptorJAKComplex] = k_2[receptorIFNAR2][JAK] - k_{2\text{-}}[ReceptorJAKComplex] + k_{3\text{-}}[IFNARdimer]$

$-k_3[ReceptorJAKComplex][ReceptorTYKComplex][IFN\_free]$

$\frac{d}{dt}[IFNARdimer] = k_3[ReceptorJAKComplex][ReceptorTYKComplex][IFN\_free] - k_{3\text{-}}[IFNARdimer]$

$-k_4[IFNARdimer]$

$\frac{d}{dt}[IFN\_free] = k_{3\text{-}}[IFNARdimer] - k_{41}[IFN\_influx] - k_3[ReceptorJAKComplex][ReceptorTYKComplex][IFN\_free]$

$\frac{d}{dt}[ReceptorTYKComplex] = k_1[ReceptorIFNAR1][TYK] - k_{1\text{-}}[ReceptorTYKComplex] + k_{3\text{-}}[IFNARdimer]$

$-k_3[ReceptorJAKComplex][ReceptorTYKComplex][IFN\_free]$

$\frac{d}{dt}[ActiveReceptorComplex] = k_4[IFNARdimer] + k_{6\text{-}}[ActiveReceptorComplex_STAT2c]$

$+k_8[ActiveReceptorComplex\_STAT2c\_STAT1c] - k_5[STAT2c\_IRF9][ActiveReceptorComplex]$

$-k_6[STAT2c][ActiveReceptorComplex] - k_{18}[ActiveReceptorComplex] - k_{19}[ActiveReceptorComplex]$

$\frac{d}{dt}[IRF9c] = k_5[STAT2c\_IRF9][ActiveReceptorComplex] + k_{9\text{-}}[ISGF-3c] + k_{16} + k_{37\text{-}}[STAT2c\_IRF9]$

$+k_{40\text{-}}[IRF9n] - k_9[IRF9c][STAT1c^*\_STAT2c^*] - k_{16\text{-}}[IRF9c] - k_{37}[STAT2c][IRF9c] - k_{40}[IRF9c]$

$\frac{d}{dt}[STAT2c\_IRF9] = k_{39\text{-}}[STAT2n\_IRF9] + k_{37}[STAT2c][IRF9] - k_5[STAT2c\_IRF9][ActiveReceptorComplex]$

$-k_{37\text{-}}[STAT2c\_IRF9] - k_{39}[STAT2c\_IRF9]$

$\frac{d}{dt}[ActiveReceptorComplex\_STAT2c] = k_5[STAT2c\_IRF9][ActiveReceptorComplex]$

$+k_6[STAT2c][ActiveReceptorComplex] + k_{7\text{-}}[ActiveReceptorComplex\_STAT2c\_STAT1c]$

$-k_{6\text{-}}[ActiveReceptorComplex\_STAT2c] - k_7[STAT1c][ActiveReceptorComplex\_STAT2c]$

$\frac{d}{dt}[STAT2c] = k_{6\text{-}}[ActiveReceptorComplex\_STAT2c] + k_{21}[STAT2c_IRF9] + k_{24}[ISGF-3cCP]$

$+k_{26}[STAT1c^*STAT2c^*CP]k_{36\text{-}}[STAT2n] + k_{37\text{-}}[STAT2c\_IRF9] - k_6[STAT2c][ActiveReceptorComplex]$

$-k_{36}[STAT2c] - k_{37}[STAT2c][IRF9c]$

$\frac{d}{dt}[ActiveReceptorComplex\_STAT2c\_STAT1c] = k_7[STAT1c][ActiveReceptorComplec\_STAT2c]$

$k_{7\text{-}}[ActiveReceptorComplex\_STAT2c\_STAT1c] - k_8[ActiveReceptorComplex\_STAT2c\_STAT1c]$

$\frac{d}{dt}[STAT1c] = k_{7\text{-}}[ActiveReceptorComplex\_STAT2c\_STAT1c] + k_{24}[ISGF-3c\_CP]$

$+k_{26}[STAT1c^*STAT2c^*\_CP] + k_{35\text{-}}[STAT1n] - k_7[STAT1c][ActiveReceptorComplex\_STAT2c]$

$-k_{35}[STAT1c]$

$\frac{d}{dt}[STAT1c^*STAT2c^*] = k_8[ActiveReceptorComplex\_STAT2c\_STAT1c] + k_{11\text{-}}[STAT1n^*STAT2n^*]$

$+k_{25\text{-}}[STAT1c^*STAT2c^*\_CP] - k_{11}[STAT1c^*STAT2c^*] - k_{25}[STAT1c^*STAT2c^*][CP]$

$\frac{d}{dt}[ISGF-3c] = k_9[IRF9c][STAT1c^*\_STAT2c^*] + k_{10\text{-}}[ISGF-3n] + k_{23\text{-}}[ISGF-3c\_CP]$

$-k_{9\text{-}}[ISGF-3c] - k_{10}[ISGF-3c] - k_{23}[ISGF-3c][CP]$

Table A.4: Reaction Rate Equations(ctd')

$\frac{d}{dt}[ISGF-3n] = k_{10}[ISGF-3c] + k_{12}[STAT1n^*\_STAT2n^*][IRF9n] + k_{13^-}[OccupiedTFBS]$

$+k_{33^-}[PIAS\_ISGF-3n] - k_{10^-}[ISGF-3n] - k_{12^-}[ISGF-3n] - k_{13}[ISGF-3n][TFBS]$

$-k_{29}[ISGF-3n][NP] - k_{33}[PIAS][ISGF-3n]$

$\frac{d}{dt}[STAT1n^*\_STAT2n^*] = k_{11}[STAT1c^*\_STAT2c^*] + k_{12^-}[ISGF-3n] + k_{27^-}[STAT1n^*\_STAT2n^*\_NP]$

$-k_{11^-}[STAT1n^*STAT2n^*] - k_{12}[STAT1n^*\_STAT2n^*][IRF9n] - k_{27}[STAT1n^*\_STAT2n^*][NP]$

$-k_{28}[STAT1n^*\_STAT2n^*\_NP]$

$\frac{d}{dt}[IRF9n] = k_{12^-}[ISGF-3n] + k_{38^-}[STAT2n_IRF9] + k_{40}[IRF9c] - k_{12}[STAT1n^*\_STAT2n^*][IRF9n]$

$-k_{38}[STAT2n][IRF9n] - k_{40^-}[IRF9c]$

$\frac{d}{dt}[OccupiedTFBS] = k_{13}[ISGF-3n][FreeTFBS] + k_{31^-}[OccupiedTFBS\_NP] - k_{13^-}[OccupiedTFBS]$

$-k_{31}[OccupiedTFBS][NP]$

$\frac{d}{dt}[mRNAn] = k_{15^-}[mRNAc] - k_{15}[mRNAn]$

$\frac{d}{dt}[mRNAc] = k_{14} + k_{15}[mRNAn] - k_{15^-}[mRNAc] - k_{34}[mRNAc]$

$\frac{d}{dt}[SOCS] = k_{17} - k_{17^-}[SOCS]$

$\frac{d}{dt}[STAT2n] = k_{22}[STAT2n_IRF9] + k_{28}[STAT1n^*STAT2n^*\_NP] + k_{30}[ISGF-3n\_NP] + k_{32}[OccupiedTFBS\_NP]$

$+k_{36}[STAT2c] - k_{36^-}[STAT2n]$

$\frac{d}{dt}[STAT2n\_IRF9] = k_{38}[STAT2n][IRF9] + k_{39}[STAT2c\_IRF9] - k_{22}[STAT2n\_IRF9] - k_{38^-}[STAT2n\_IRF9]$

$-k_{39^-}[STAT2n\_IRF9]$

$\frac{d}{dt}[CP] = k_{23^-}[ISGF-3c\_CP] + k_{24}[ISGF-3c\_CP] + k_{25^-}[STAT1c^*STAT2c^*\_CP] + k_{26}[STAT1c^*STAT2c^*\_CP]$

$-k_{23}[ISGF-3c][CP] - k_{25}[STAT1c^*\_STAT2c^*][CP]$

$\frac{d}{dt}[ISGF-3c\_CP] = k_{23}[ISGF-3c][CP] - k_{24}[ISGF-3c\_CP]$

$\frac{d}{dt}[STAT1c^*STAT2c^*\_CP] = k_{25}[STAT1c^*STAT2c^*][CP] - k_{25^-}[STAT1c^*STAT2c^*\_CP] - k_{26}[STAT1c^*STAT2c^*\_CP]$

$\frac{d}{dt}[NP] = k_{27^-}[STAT1n*STAT2n^*\_NP] + k_{28}[STAT1n*STAT2n^*\_NP] + k_{30}[ISGF-3n\_NP] + k_{31^-}[OccupiedTFBS\_NP]$

$-k_{27}[STAT1n^*STAT2n^*][NP] - k_{29}[ISGF-3n][NP] - k_{31}[OccupiesdTFBS][NP]$

$\frac{d}{dt}[STAT1n^*STAT2n^*\_NP] = k_{27}[STAT1n^*STAT2n^*][NP] - k_{27^-}[STAT1n*STAT2n^*\_NP] - k_{28}[STAT1n*STAT2n^*\_NP]$

$\frac{d}{dt}[STAT1n] = k_{28}[STAT1n*STAT2n^*\_NP] + k_{30}[ISGF-3n\_NP] + k_{32}[OccupiedTFBS\_NP] + k_{35}[STAT1c] - k_{35^-}[STAT1n]$

$\frac{d}{dt}[ISGF-3n\_NP] = k_{29}[ISGF-3n][NP] - k_{30}[ISGF-3n\_NP]$

$\frac{d}{dt}[OccupiedTFBS\_NP] = k_{13}[ISGF-3n][FreeTFBS] + k_{31}[OccupiedTFBS][NP] - k_{13^-}[OccupiedTFBS\_NP]$

$-k_{31^-}[OccupiedTFBS\_NP] - k_{32^-}[OccupiedTFBS\_NP]$

$\frac{d}{dt}[FreeTFBS] = k_{13^-}[OccupiedTFBS\_NP] + k_{32^-}[OccupiedTFBS\_NP] - k_{13}[ISGF-3n][FreeTFBS]$

$\frac{d}{dt}[PIAS] = k_{33^-}[PIASISGF-3n] - k_{33}[PIAS][ISGF]$

$\frac{d}{dt}[PIASISGF-3n] = k_{33}[PIAS][ISGF] - k_{33^-}[PIASISGF-3n]$

$\frac{d}{dt}[IFNinflux] = -k_{41}[IFN\_influx]$

$\frac{d}{dt}[ReceptorIFNAR1] = k_{1^-}[ReceptorTYKComplex] + k_{18}[ActiveReceptorComplex] - k_1[ReceptorIFNAR1][TYK]$

$\frac{d}{dt}[ReceptorIFNAR2] = k_{2^-}[ReceptorJAKComplex] + k_{18}[ActiveReceptorComplex] - k_2[ReceptorIFNAR2][JAK]$

Table A.5: Reaction Rate Equations(ctd')

APPENDIX B

MATLAB CODES

B.1 MATLAB Source Code for Gillespie's SSA with Direct Method

```
function[ state,time,x_step,t_points,s,X_Step] = gillespie_merve(x0, c, pre,...
 post, t_initial, t_final, iteration, h)
 for i = 1:iteration
n_r = length(c); %number of chemical reactions
n_s = length(x0);  %number of chemical species
k   = 1; j   = 2;
t_points = t_initial:h:t_final; %time points for given step size
s        = length(t_points);
x_step   = zeros(n_s,s); %states for time time points
 % 0.Initialize the time and the state
x_step(:,1) = x0;
x(:,k)      = x0;
t(k)        = t_initial;
 % 1. Compute the propensity functions and total propensity for the given
 % state and time
 [ a,a_0 ]   = propensity_merve ( x0, pre, c );
 while (t(k) <= t_final) && (a_0 > 0)
 % 2. Generate tau and mu according to the Direct Method
     r      = rand(1,2);
     tau(k) = (1/a_0)*log(1/r(1));
     mu(k)  = find(cumsum(a)> r(2)*a_0, 1);
```

```
% Update the time and the state according to tau and mu
    t(k+1)     = t(k) + tau(k);
    x(:,k+1)   = x(:,k) + post(:,mu(k)) - pre(:,mu(k));
    k          = k+1;
    [ a,a_0 ] = propensity_merve ( x(:,k), pre, c );
    end
for  j = 2:length(t_points)
    [M,I] = min(abs(t_points(j)-t));
    x_step(:,j) = x(:,I);
    j = j+1;
end
state{i}  = x;
time{i}   = t;
X_Step{i} = x_step;
end
```

B.2 MATLAB Source Code for Bootstrap Particle Filter

```
function [ time_steps,x_int,Y,state_history_all,time_history_all,...
post, state_at_time_pts,W,normal_W,N_of_rxns] = particlefilter( t_initial,... \\
post,  t_final, h, mu, sigma,N,c,pre,post)
% Particle filter
% Given : time steps
time_steps = t_initial:h:t_final;
% Given : Measurements
Y = measurments( c, pre, post, t_initial, t_final, h,sigma);
%% Initialization
% Prelocation
state_history_all = cell(N,length(time_steps)-1);
time_history_all = cell(N,length(time_steps)-1);
state_at_time_pts = zeros(N,length(time_steps));
```

```
N_of_rxns = zeros(N,length(time_steps));
color = hsv(N);
% Draw N random samples from initial distribution which is chosen as
% Gaussian distribution with mean mu and variance sigma
x_int = normrnd(mu,sigma,1,N);
state_at_time_pts(:,1) = x_int;
N_of_rxns(:,1) = ones(N,1);
for  j = 1:length(time_steps)-1
%% Importance Sampling
% Sample from importance density which is chosen as p(x_t|x_{t-1}) and
% obtained by Gillespie algorithm
    counter = 0;
for  i = 1:N
    if  N_of_rxns(i,j) == 0
        i = i+1;
    else
    for  ii = 1:N_of_rxns(i,j)
[  state,time,x_step,t_points] = ...
  gillespie_merve(state_at_time_pts(i,j), c, pre, post, time_steps(j),...
post,   time_steps(j+1),1, h);
counter = counter+1;
state_history_all(counter,j) = state;
time_history_all  (counter,j) = time;
state_at_time_pts  (counter,j+1) = x_step(end);
figure(1)
hold  on
            set(gca,'Fontsize',20)
            plot(t_points,x_step,'color',color(j,:),'LineWidth',2)
            plot(t_points,x_step,'*','color',color(j,:))
            xlabel('Time','Fontsize',30)
            ylabel('Realization','Fontsize',30)
    end
    end
```

```
43  end
44  % Calculate the importance weights w_t^(i) ~ p(y_t|x_t^(i))
45  [ W,normal_W ] = importance_weight(Y,state_at_time_pts (:,j+1),N,time_steps,sigma);

46  %% Resampling Step(Multinomial Resampling)
47  %  Determine the particles which have large importance weights
48  Q = cumsum(normal_W(:,j));
49  index = zeros(1, N);
50  m = 0;
51  while  m < N
52      m = m + 1;
53      sampl = rand;   % ~(0,1]
54      a = 1;
55      while  Q(a) < sampl
56          a = a + 1;
57      end;
58      index(m)= a;
59  end
60  index;
61  n_of_rxn = zeros(1,N);
62  for  p = 1:N
63      n_of_rxn(index(p)) = n_of_rxn(index(p))+1;
64  end
65  N_of_rxns(:,j+1) = n_of_rxn;
66  end
67  yLimits = get(gca,'YLim');
68  for  jj=1:length(time_steps-1)
69      line([time_steps(jj) time_steps(jj)],[yLimits(1) yLimits(2)],'color','k'
    );
70  end
71  end
```