Comparison of Approximation Schemes in Stochastic Simulation Methods for Stiff Chemical Systems

by

Chad Richard Wells

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Applied Mathematics

Waterloo, Ontario, Canada, 2009

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Interest in stochastic simulations of chemical systems is growing. One of the aspects of simulation of chemical systems that has been the prime focus over the past few years is accelerated simulation methods applicable when there is a separation of time scale. With so many new methods being developed we have decided to look at four methods that we consider to be the main foundation for this research area.

The four methods that will be the focus of this thesis are: the slow scale stochastic simulation algorithm, the quasi steady state assumption applied to the stochastic simulation algorithm, the nested stochastic simulation algorithm and the implicit tau leaping method. These four methods are designed to deal with stiff chemical systems so that the computational time is decreased from that of the "gold standard" Gillespie algorithm, the stochastic simulation algorithm.

These approximation methods will be tested against a variety of stiff examples such as: a fast reversible dimerization, a network of isomerizations, a fast species acting as a catalyst, an oscillatory system and a bistable system. Also, these methods will be tested against examples that are marginally stiff, where the time scale separation is not that distinct.

From the results of testing stiff examples, the slow scale SSA was typically the best approximation method to use. The slow scale SSA was highly accurate and extremely fast in comparison with the other methods. We also found for certain cases, where the time scale separation was not as distinct, that the nested SSA was the best approximation method to use.

Acknowledgements

I would like to thank my supervisors, Brian Ingalls and Zoran Miskovic, for their guidance throughout the process of this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Biochemical systems are all around us, some have extremely complicated structures, e.g. those involved in bacterial growth. In [1] a numerical study was done on the intercellular growth of bacteriophage T7. Bacteria and viruses affect and are involved in our everyday lives. For example, the Spanish influenza pandemic that occurred from 1918-1920, had a global mortality rate of around 50 million [2]. One way to understand these biochemical systems is through a mathematical model, as mentioned previously with the example from [1]. A good example of how a virus can affect a living organism can be found in [3], this is a model of a virus infecting a cell. One of the interesting aspects of this model is that the model could enter two different states: all the species become populated or all the species become extinct [3]. From what has been mention so far, we see that mathematical models are an important role in understanding biochemical systems.

There are many aspects of a chemical system's environment that have an impact on the behaviour of the system. A system involving a large number of variables and parameters can be incredibly complex, making it difficult to fully understand its dynamics [4]. With experimental data [5] and approximation models [4] now becoming more available for complicated systems, many simulation tools that determine the dynamics of a chemical system are being or have been developed [5]. Computer simulations have become a necessity when it comes to the study of dynamics of a chemical system because the majority of the time these chemical systems are very complex [6]. This need for simulations has become evident in experimental and theoretical studies of different chemical and genetic networks [7].

In the study of chemical kinetics, there are many different mathematical approaches to determine the behaviour of a chemical system [6]. One of the common ways to determine the behaviour of the system is to model it by a set of coupled ordinary differential equations, which has a deterministic behaviour. Another common way to determine the behaviour of the system is by using a stochastic approach (e.g. random walk). This stochastic approach has a clearer physical basis than the deterministic approach [8]. A good example of this would be genetic switching [9], where

the system has two steady states. A deterministic trajectory will converge to one steady state and stay in that state, where as the stochastic approach accounts for the noise in the system which plays an important part in genetic switching [9]. The reason noise is so important to genetic switching is that it allows the concentration of a species to go from one steady state to another steady state by "pushing" it out of its current steady state. Stochastic modeling and simulation are important to chemical kinetics because reactions randomly occur in a chemical system [6] and the stochastic nature of these reactions produce fluctuations, causing an effect on the system [10], i.e. genetic switching. The importance of stochastic modeling is also evident in the chemical reactions occurring within a cell where the volume and number of reacting species are relatively small, so that any fluctuations in the number of chemical species can become relatively large. Therefore, further development and investigations of stochastic models are needed to determine the chemical dynamics of interest in biological systems [10].

The basis for the majority of the stochastic approaches in chemical dynamics is the *master equation*, which describes the evolution of the probability distribution over a period of time [11]. Another approach to approximate the solution to the master equation is the explicit tau leaping method, which solves the associated *chemical Langevin equation*, a stochastic differential equation, using a solution method similar to the explicit Euler method [12]. The explicit tau leaping method is an accelerated approximation method, compared to the SSA, to determine the behaviour of the system [12].The Langevin equation can be related back to the master equation through the *Fokker-Planck equation*, a stochastic partial differential equation that describes the time evolution of the probability density function. There is a relation between the Fokker-Planck equation and the Langevin equation that will be discussed later on in this thesis. Even though the method of solving the Langevin equation is a different approach, it yields similar results to the SSA, [12].

The SSA and the explicit tau leaping method are able to handle chemical systems that are not *stiff*. A chemical system is stiff if there is a separation in time scales between its reactions. The problem that the explicit tau method has with a stiff system is that it becomes unstable for large Δt, in the same manner that the Explicit Euler method would become unstable in addressing stiff systems. The SSA does not become unstable with a stiff chemical system because the SSA is an *exact* method, in the sense that it accounts for all the fluctuations and correlations implied by the reactants [8]. However, the SSA is slow when dealing with stiff chemical systems [13] because it must simulate all of the reactions.

A main focus in the area of analysing the behaviour of chemical systems, stiff or not, is to find an approximation method that is as accurate as the SSA and is able to produce the results much faster. There are many papers describing these new methods dealing with non-stiff and stiff chemical systems such as: the binomial tau leaping method [7, 14], the multinomial tau leaping method [15], a hybrid method [10] and tau leaping with random correction [16] to name a few. In this

thesis, we will be focusing on four approximation methods: the quasi steady state assumption applied to the SSA, (QSSA algorithm) [13]; the slow scale SSA [17]; the nested SSA [18, 19]; and the implicit tau leaping method [20].

The explicit tau leaping method is closely related to the implicit tau leaping method [20]. The implicit tau leaping method, similar to the implicit Euler method, is able to handle a stiff chemical system much better than the explicit tau leaping method [20]. We will consider three different ways to implement the implicit tau leaping method. The first way is to select a *time jump* that is on the *slow time scale*, the time scale where *slow reactions* take place. The second way of implementing the implicit tau is to use a fixed time jump through the entire simulation. The third way we will implement the implicit tau leaping method is by taking a sequence of small time steps and then taking a larger time step or vice versa. In [20], it is shown that the sequence of small jumps will restore the overly damped fluctuations in the fast variables caused by the larger jump on the slow time scale.

The next method that we will be looking at is described in [13]. This is the QSSA algorithm, a method which uses the quasi steady state assumption to reduce the computational effort of the SSA. The quasi steady state assumption allows us to put the species on the fast time scale into steady state, thereby removing the stiffness from the problem by eliminating the fast reactions. We will implement the QSSA algorithm by approximating the fast species by its steady state, obtained from the deterministic equations. A probabilistic approach to the QSSA algorithm involves sampling from a probability distribution based on the fast system being in a steady state. This probabilistic approach to the QSSA algorithm is called the slow scale SSA [17]. We will be looking at a few examples that use the binomial distribution to approximate the fast species and another example that uses a normal distribution. The approach of the slow scale SSA is a more "theoretical" version of the nested SSA [19], which we will discuss next.

Introduced in [19], the nested SSA takes an approach slightly different from that of the slow scale SSA. The nested SSA involves approximating the fast species by running a number of SSA's, for the fast reactions only and for a certain amount of time, during which the fast system is presumed to reach steady state. The nested SSA numerically produces a probability distribution and samples from it. The nested SSA involves three design parameters: n, the number of times to run the inner SSA, T_0 the time to truncate the transient and T_f the time to run after the time of truncation. We will be looking at different values of n and lengths for the inner SSA to see which optimizes the CPU time of the nested SSA, as well as the parameters that produce the best results. In [21], the authors mention that over-parametrization can lead to an infinite number of parameter combinations that lead to equivalent results.

In this thesis, we will look at the above four methods (QSSA algorithm, slow scale SSA, nested SSA and the implicit tau leaping method) in greater detail to see

how they are similar, as well as how they differ from each other and how well they describe the behaviour of a chemical system. In comparing the performance of the methods for different examples, we will use the SSA as a "gold standard" since it is an exact method [8]. We will be looking at the CPU times for all of the approximation methods and see which one is the fastest and compare with the SSA. We also want to see which approximation method, out of the four we have chosen, is the most accurate when compared to the SSA; we will compute an ensemble of sample paths of species in each approximation method. This enables us to conduct a comparison on the statistical level. When comparing the approximation methods to the SSA we will be looking at the means and the standard deviation of all the species. Also we will look at the skewness of the distribution of each of the species produced by each method which gives us insight to what the probability distribution looks like for each of the methods. With this in mind we will also look at a histogram of the species at the final time point of each example. This will allow us to see if the probability distribution of the approximation method is similar to the SSA's or not.

We will also be focusing on how well each approximation method approximates the *virtual fast process*, the fast species in the fast reactions [17]. We will investigate how well each method approximates the distribution of this fast process by creating a histogram of the fast species just before a slow reaction occurs. If an approximation of the fast system is bad, it might be a poor method to use in that case. With this information we will come to a conclusion as to which approximation method is best overall or is best for certain examples.

We will be also looking at two elaborate examples, an oscillatory system and a bistable system. These approximation methods have not previously been tested against these more elaborate examples.

If, for some reason, the results of the approximation methods do not share similar characteristics to that of the SSA's results, then the best approach for these more elaborate examples is simply the SSA.

The results of this thesis will guide the choice of the appropriate approximation algorithm for a given stiff system. This thesis will also point out any possible flaws or problems that each method has. Knowing the flaws and problems of a method prior to implementation is important, as it may have influence on whether one wants to use that approximation method or not.

Chapter 2

Background

2.1 Chemistry Background

We will consider a system of N chemical *species* $(S_1, S_2, \cdots, S_N)$ involved in M *reactions* $(R_1, R_2, \cdots, R_M)$. The *state* of the system at time t will be denoted as $\mathbf{X}(t) = (X_1(t), X_2(t), \cdots, X_N(t))$, where $X_i(t)$ indicates the number of molecules of species S_i at time t. The *reaction rates* are denoted as $\mathbf{c}$, where c_i corresponds to R_i. To each reaction R_i is an associated state vector change or *stoichiometry*, $\mathbf{v}_i$. When R_i occurs, the state is updated by $\mathbf{x} \mapsto \mathbf{x} + \mathbf{v}_j$, where $\mathbf{v}_i = (v_{i1}, v_{i2}, \cdots, v_{iN})$. Each reaction R_i is characterized by a *propensity function* $a_i(\mathbf{x})$, that depend on the state. The propensity function for R_i is a function of c_i and the reactants in R_i. This relates to the *law of mass action* which states that the rate of a reaction is proportional to the product of the reactants *concentrations*[22]. In Table 2.1.1 we see examples of propensity functions and stoichiometry values for certain reactions. However, when we deal with the stochastic modeling one must take into account individual molecules. We see in Table 2.1.2, how the propensities from stochastic modeling differ from Table 2.1.1. Both stochastic and deterministic approaches rely on the condition of a *well stirred* system. A system is well stirred if there is spatial homogeneity in the system, which can be caused by non-reactive molecular

Reaction	Deterministic Reaction Rate	Stoichiometry Vector
$S_1 \xrightarrow{c_1} S_2$	$c_1 x_1$	$\mathbf{v} = (-1, 1)$
$S_1 + S_2 \xrightarrow{c_1} S_3$	$c_1 x_1 x_2$	$\mathbf{v} = (-1, -1, 1)$
$S_1 + S_2 \xrightarrow{c_1} S_2$	$c_1 x_1 x_2$	$\mathbf{v} = (-1, 0)$
$S_1 + S_1 \xrightarrow{c_1} S_2$	$\frac{c_1 x_1^2}{2}$	$\mathbf{v} = (-2, 1)$
$S_1 + S_1 + S_1 \xrightarrow{c_1} 2S_2$	$\frac{c_1 x_1^3}{6}$	$\mathbf{v} = (-3, 2)$

Table 2.1.1: Examples of reactions, deterministic reaction rates and stoichiometry vectors

Reaction	Propensity Function
$S_1 \xrightarrow{c_1} S_2$	$c_1 x_1$
$S_1 + S_2 \xrightarrow{c_1} S_3$	$c_1 x_1 x_2$
$S_1 + S_2 \xrightarrow{c_1} S_2$	$c_1 x_1 x_2$
$S_1 + S_1 \xrightarrow{c_1} S_2$	$\frac{c_1 x_1 (x_1 - 1)}{2}$
$S_1 + S_1 + S_1 \xrightarrow{c_1} 2S_2$	$\frac{c_1 x_1 (x_1 - 1)(x_1 - 2)}{6}$

Table 2.1.2: Examples of reactions and propensity functions for stochastic modeling

collisions causing a self-stirring effect[23]. In some chemical systems some reactions occur much more often than others, due to high reactant population size or the values of the reaction rates. When some reactions occur more often than others the system is referred to as a *stiff* system: there is a separation of time scales between two sets of reactions. The two sets of reactions are the *fast reactions* or *slow reactions*.

Fast reactions have high propensity values compared to the slow reactions propensity values. Any species whose stoichiometry value is non zero in a fast reaction is called a *fast species*, where any species whose stoichiometry value is zero in all the fast reactions is called a *slow species*. In a stiff system there is a set of n_f fast species $(S_1^f, \cdots, S_{n_f}^f)$ and a set of n_s slow species $(S_1^s, \cdots, S_{n_s}^s)$, where $N = n_f + n_s$. In a stiff system there is also a set of m_f fast reactions $(R_1^f, \cdots, R_{m_f}^f)$ and a set of m_s slow reactions $(R_1^s, \cdots, R_{m_s}^s)$, where $M = m_f + m_s$. The state of the fast species at time t is denoted as $\mathbf{X}^f(t) = (X_1^f(t), \cdots, X_{n_f}^f(t))$, where $X_i^f(t)$ indicates the number of molecules of the fast species S_i^f at time t. Similarly, $X_i^s(t)$ indicates the number of molecules of the slow species S_i^s at time t, where $\mathbf{X}^s(t)$ is the state of the slow species at time t. Each slow reaction R_i^s will be characterized by a slow propensity function $a_i^s(\mathbf{x})$, where $\mathbf{X}(t) = \mathbf{x}$. Similarly, each fast reaction R_i^s will be characterized by a fast propensity function $a_i^f(\mathbf{x})$, where $\mathbf{X}(t) = \mathbf{x}$. To each fast reaction is an associated fast stoichiometry vector $\mathbf{v}^f$ and each slow reaction there is an associates slow stoichiometry vector $\mathbf{v}^s$. As mentioned in the definition of a slow species, in all of the fast stoichiometry vectors, the value for all slow species is zero.

Example 2.1.1

$$S_1 \rightarrow S_2 \tag{2.1.1}$$

$$S_2 + S_1 \rightarrow S_3 \tag{2.1.2}$$

If (2.1.1) is a fast reaction and (2.1.2) is a slow reaction, then S_1 and S_2 are fast species and S_3 is a slow species.

Example 2.1.2

$$S_1 + S_4 \rightarrow S_2 + S_4 \tag{2.1.3}$$
$$S_2 + S_3 \rightarrow S_4 \tag{2.1.4}$$

If (2.1.3) is a fast reaction and (2.1.4) is a slow reaction, then S_1 and S_2 are fast species and S_3 and S_4 are slow species. S_4 is a slow species because its stoichiometry value in the fast reaction (2.1.3) is zero.

One approach to handling a stiff system is by using the *quasi steady state assumption*. Since we are dealing with stiffness due to a time scale separation, the quasi steady state assumption allows us to place the fast species into their steady state and focus only on the slow time scale. Therefore, the quasi steady state assumption eliminates the fast dynamics of the system, which means when dealing with deterministic equations we eliminate the differential equations of the fast species by finding their steady state [13]. Following is an example of how the quasi steady state assumption is applied to a stiff chemical system.

Example 2.1.3

$$\emptyset \xrightarrow{c_1} S_1 \tag{2.1.5}$$
$$S_1 + S_2 \xrightarrow{c_2} 2S_2 \tag{2.1.6}$$
$$S_2 + S_3 \xrightarrow{c_3} S_3 \tag{2.1.7}$$
$$S_2 \xrightarrow{c_4} S_3 \tag{2.1.8}$$

The deterministic equations for the system are

$$\frac{dS_1}{dt} = c_1 - c_2 S_2 S_1$$
$$\frac{dS_2}{dt} = c_2 S_2 S_1 - c_3 S_2 S_3 - c_4 S_2$$
$$\frac{dS_3}{dt} = c_4 S_2$$

If (2.1.5),(2.1.6) and (2.1.7) are fast reactions and (2.1.8) is a slow reaction then S_1 and S_2 are fast species and S_3 is a slow species, since its stoichiometry value in (2.1.7) is zero. Applying the quasi steady state assumption we set, $c_1 - c_2 S_2 S_1 = 0$ and $c_2 S_2 S_1 - c_3 S_2 S_3 = 0$ so that $S_1 = (c_3 S_3 + c_4)/c_2$ and $S_2 = c_1/(c_3 S_3 + c_4)$. After applying the applying the quasi steady state assumption the deterministic equations are now reduced to

$$\frac{dS_3}{dt} = c_4 S_2 = \frac{c_4 c_1}{c_3 S_3 + c_4}. \tag{2.1.9}$$

We will see later on in the thesis how this reduces the computational time in analysis of a stiff system. We next turn to the probability theory which underlines the methods used in this thesis.

2.2 Probability Theory

Definition 2.2.1 *A **sample space**, $\mathcal{S}$, is a set of all possible outcomes for an experiment.*

Definition 2.2.2 *An **event**, E, is a certain set of outcomes that occur in an experiment.*

Example 2.2.1 *Consider an experiment where we are observing two rolls of a die, where order does not matter, i.e. $(1,4) = (4,1)$. This means that the possible outcomes of this experiment is the sample space,*

$$\begin{aligned}\mathcal{S} \ &= \{(1,1),(1,2),(1,3),(1,4),(1,5),(1,6),(2,2),(2,3),(2,4),(2,5),(2,6),\\ &\quad (3,3),(3,4),(3,5),(3,6),(4,4),(4,5),(4,6),(5,5),(5,6),(6,6)\}\end{aligned}$$

Some possible events for the experiment are

$$\begin{aligned}E_1 &= \text{One of the rolls was even},\\ E_2 &= \text{The sum of the two rolls is 6}.\end{aligned}$$

Definition 2.2.3 *A **random variable** is a function defined on a sample space that maps $\mathcal{S} \to \mathbb{R}$.*

Definition 2.2.4 *A random variable is called a **discrete random variable** if the set of possible values for the random variable is countable.*

Definition 2.2.5 *A random variable is called a **continuous random variable** if it is not a discrete random variable.*

Example 2.2.2 *Consider the experiment of tossing a coin n times and define*

$$X_j = \begin{cases} 1 & j^{th} \text{ toss is a head} \\ 0 & j^{th} \text{ toss is a tail} \end{cases}$$

Then X_j is a discrete random variable.

Example 2.2.3 *An **exponential random variable** is a continuous random variable. The exponential random variable X, with the rate λ, has the probability distribution function*

$$f(x) = \begin{cases} \lambda \exp(-\lambda x) & x \geq 0 \\ 0 & x < 0 \end{cases}.$$

Definition 2.2.6 *Two events are called **mutually exclusive** if, no matter the outcome of the experiment, they cannot be simultaneously satisfied.*

Definition 2.2.7 *The* ***probability*** *of an event E, $P(E)$, is defined as the frequency of an event occurring in a sample space and must satisfy the following axioms:*
1.$P(E) \geqslant 0$ *for every event E and that $P(\mathcal{S}) = 1$ for the sample space $\mathcal{S}$.*
2. *If E and F are mutually exclusive events then $P(E \cup F) = P(E) + P(F)$.*

Definition 2.2.8 *Events E and F are* ***statistically independent*** *if $P(E \cap F) = P(E)P(F)$. Events E_i are* ***mutually independent*** *if*

$$P(\bigcap_{i=1}^{n} E_i) = \prod_{i=1}^{n} P(E_i).$$

Definition 2.2.9 *The probability that event E occurs given that event F has occurred is called the* ***conditional probability****, denoted as $P(E|F)$, and defined by*

$$P(E|F) = \frac{P(E \cap F)}{P(F)}.$$

Definition 2.2.10 *The* ***probability distribution function****, $f(x)$, for a discrete random variable X is*

$$f(x) = P(X = x).$$

The ***cumulative probability distribution function****, $F(x)$, for a discrete random variable is*

$$F(a) = \sum_{x=-\infty}^{a} f(x).$$

The probability that the value of a continuous random variable X lies somewhere in the interval (a, b) is

$$P(a < X < b) = \int_a^b f(x) \quad dx,$$

where $f(x)$ is the probability distribution function. The cumulative probability distribution function, $F(x)$, for a continuous random variable is

$$F(a) = \int_{-\infty}^{a} f(x) \quad dx.$$

A probability distribution function has the following properties

$$f(x) \geq 0 \quad \forall \quad x$$

$$\sum_{x=-\infty}^{\infty} f(x) = 1 \quad \text{for a discrete random variable}$$

$$\int_{-\infty}^{\infty} f(x)dx = 1 \text{ for a continuous random variable}$$

Definition 2.2.11 *The **expected value** of the random variable X is denoted by $\langle X \rangle$. The expected value of a discrete random variable X is defined as:*

$$\langle X \rangle = \sum_{x=-\infty}^{\infty} xf(x).$$

The expected value of a function of a discrete random variable is

$$\langle g(X) \rangle = \sum_{x=-\infty}^{\infty} g(x)f(x).$$

The expected value of a continuous random variable X is defined as

$$\langle X \rangle = \int_{-\infty}^{\infty} xf(x) \quad dx.$$

The expected value of a function of a continuous random variable is

$$\langle g(X) \rangle = \int_{-\infty}^{\infty} g(x)f(x) \quad dx.$$

Definition 2.2.12 *The $\boldsymbol{n^{th}}$ **moment** of the random variable X is $\langle X^n \rangle$. The $\boldsymbol{n^{th}}$ **central moment** of the random variable X is $\langle (X - \langle X \rangle)^n \rangle$.*

Definition 2.2.13 *The **variance** of a random variable X is denoted as $\langle\langle X \rangle\rangle$ and is the second central moment.*

Definition 2.2.14 *The **skewness** of a random variable X is defined as*

$$\frac{\langle (X - \langle X \rangle)^3 \rangle}{\langle\langle X \rangle\rangle^{3/2}}.$$

If the skewness value is negative then the left tail of the distribution is longer and majority of the probability lies to the right. If the skewness value is positive then the right tail of the distribution is longer and the majority of the probability lies to the left. If the skewness value is zero then we know that the distribution is symmetric.

Definition 2.2.15 *The **covariance** of two random variables X and Y is denoted as $\langle\langle XY \rangle\rangle$ and is defined as*

$$\langle\langle XY \rangle\rangle = \langle XY \rangle - \langle X \rangle \langle Y \rangle.$$

Definition 2.2.16 *The random variables X and Y are **uncorrelated** if and only if $\langle\langle XY\rangle\rangle = 0$.*

Definition 2.2.17 *The **joint probability distribution function** of the random variable $X_1, X_2, \cdots, X_n$ is denoted as $f(x_1, x_2, \cdots, x_n)$. For discrete random variables*

$$f(x_1, x_2, \cdots, x_n) = P(X_1 = x_1, X_2 = x_2, \cdots, X_n = x_n).$$

*The **marginal distribution** for a discrete random variable, X_i, is found by summing the joint probability distribution function of $X_1, X_2, \cdots, X_n$ over all states except X_i. The marginal distribution of X_i is denoted as $f_i(X_i)$.*

$$f_i(x_i) = \sum_{x_1=-\infty}^{\infty} \cdots \sum_{x_{i-1}=-\infty}^{\infty} \sum_{x_{i+1}=-\infty}^{\infty} \cdots \sum_{x_n=-\infty}^{\infty} f(x_1, x_2, \cdots, x_n)$$

The probability that the value of each random variable X_i lies somewhere in the interval (a_i, b_i) is

$$P(a_1 < X_1 < b_1, \cdots, a_n < X_n < b_n) = \int_{a_n}^{b_n} \cdots \int_{a_1}^{b_1} f(x_1, x_2, \cdots, x_n) \;\; dx_1 \cdots dx_n,$$

where $f(x_1, x_2, \cdots, x_n)$ is the joint probability distribution function of $X_1, X_2, \cdots, X_n$. For a continuous random variable the marginal distribution for X_i is

$$f_i(x_i) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f(x_1, x_2, \cdots, x_n) \;\; dx_1 \cdots dx_{i-1} dx_{i+1} \cdots dx_n.$$

To find the marginal distribution of more variables other than X_i, say X_i, X_j and X_k, sum or integrate over every X_l, $l \neq i, j, k$. The marginal distribution here will be denoted as $f_{i,j,k}(x_i, x_j, x_k)$. The expectation of a function of discrete random variables, $g(X_1, \cdots, X_n)$, is

$$\langle g(X_1, \cdots, X_n)\rangle = \sum_{x_1=-\infty}^{\infty} \cdots \sum_{x_n=-\infty}^{\infty} g(x_1, \cdots, x_n) f(x_1, \cdots, x_n).$$

The expectation of a function of continuous random variables, $g(X_1, \cdots, X_n)$, is

$$\langle g(X_1, \cdots, X_n)\rangle = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} g(x_1, \cdots, x_n) f(x_1, \cdots, x_n) \;\; dx_1 \cdots dx_n.$$

Definition 2.2.18 *The **joint conditional probability density function** for $X_i, X_j, X_k, \cdots$ conditional on $X_m, X_p, X_q, \cdots$ is denoted as*

$$f(x_i, x_j, x_k, \cdots | x_m, x_p, x_q, \cdots).$$

The joint conditional probability density function is computed by dividing the joint

probability density function by the marginal distribution of the conditional variables,

$$f(x_i, x_j, x_k, \cdots | x_m, x_p, x_q, \cdots) = \frac{f(x_1, \cdots, x_n)}{f_{m,p,q,\cdots}(x_m, x_p, x_q, \cdots)}.$$

This is also known as Bayes' relation.

The *conditional expectation* of a random variable X_i is denoted as

$$\langle X_i | X_1, \cdots, X_{i-1}, X_{i+1}, \cdots, X_n \rangle.$$

The conditional expectation of a function of discrete random variables, e.g. $g(X_i, X_j, X_k)$, is

$$\begin{aligned} &\langle g(X_i, X_j, X_k) | X_l \quad l \neq i, j, k \rangle \\ &= \sum_{x_i=-\infty}^{\infty} \sum_{x_j=-\infty}^{\infty} \sum_{x_k=-\infty}^{\infty} g(x_i, x_j, x_k) f(x_i, x_j, x_k | x_1, \cdots, x_n). \end{aligned}$$

The conditional expectation of a function of continuous random variables, e.g. $g(X_i, X_j, X_k)$, is

$$\begin{aligned} &\langle g(X_i, X_j, X_k) | X_l \quad l \neq i, j, k \rangle \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x_i, x_j, x_k) f(x_i, x_j, x_k | x_1, \cdots, x_n) dx_i dx_j dx_k. \end{aligned}$$

2.3 A Background in Stochastics

Definition 2.3.1 *A **stochastic process** is a family of random variables. A discrete time stochastic process uses a discrete index, where the family is denoted by* $\{X(n)\}$, $n \in \mathbb{Z}$. *A continuous time stochastic process has a continuous index, where the family is denoted by* $\{X(t)\}$, $t \in \mathbb{R}$.

Definition 2.3.2 *A stochastic process $X(t)$ has **discrete states** if the set of possible values of $X(t)$ is countable.*

Definition 2.3.3 *A stochastic process $X(t)$ has **continuous states** if the set of possible values of $X(t)$ is not countable.*

Example 2.3.1 *An example of a discrete random process is tossing a coin, as in Example 2.2.2, where the stochastic process is $X = \{X_i\}$ for $i = 1, \cdots, n$. If there were 10 tosses, then one possible outcome of the stochastic process is $X = \{H, T, T, H, H, T, H, T, H, H, H\}$. An example of a continuous stochastic process is a stock in the stock market. The stochastic process for the stock is $X =$* $\{$the price of the stock at time $t\}$.

Definition 2.3.4 *A stochastic process $X(t)$ is said to have **independent increments** if $X(t_i) - X(t_{i-1})$ are independent random variables for $i = 2, 3, 4, \cdots, n$. A stochastic process $X(t)$ has **stationary independent increments** if $X(t_i) - X(t_{i-1})$ can be written in terms of the time difference only,*

$$X(t_i) - X(t_{i-1}) = \Delta X(t_i - t_{i-1}),$$

where $X(t)$ has independent increments.

Here are a few examples of some well known stochastic processes.

Definition 2.3.5 *A **Poisson process** is a discrete state stochastic process. A stochastic process $X(t)$ is called a Poisson process, with rate λ, if and only if $X(0) = 0$, $X(t)$ has stationary independent increments and $X(t)$ has the probability distribution function*

$$P(X(t) = n) = \frac{\exp(-\lambda t)(\lambda t)^n}{n!}.$$

Definition 2.3.6 *A stochastic process $X(t)$, with continuous state, is called **Brownian Motion** if and only if $X(0) = 0$, $X(t)$ has stationary independent increments and $X(t)$ has the probability distribution function of a normal random variable with mean 0 and variance t.*

Definition 2.3.7 ***Gaussian white noise**, $\Gamma(t)$ is a stochastic process such that $\langle \Gamma(t) \rangle = 0$ and $\langle \Gamma(t_1)\Gamma(t_2) \rangle = \delta(t_1 - t_2)$. It can be considered as an ill defined limit*

$$\Gamma(t) = \lim_{dt \to 0} \mathcal{N}(0, 1/dt),$$

where $\mathcal{N}(0, 1/dt)$ is a normal distribution with mean zero and variance $1/dt$ [24].

Definition 2.3.8 *If a stochastic process is not influenced by any information from the past and is only influenced by its most recent state then the stochastic process is called a **Markov process**. For a discrete time Markov process, we have*

$$P(X_{n+1} = x_{n+1} | X_n = x_n, \cdots, X_1 = x_1) = P(X_{n+1} = x_{n+1} | X_n = x_n)$$

and for a continuous time Markov process,

$$P(X(t_{n+1}) = x_{n+1} | X(t_n) = x_n; \cdots; X(t_0) = x_0) = P(X(t_{n+1}) = x_{n+1} | X(t_n) = x_n),$$

where $t_{i+1} > t_i$ for $i = 0, \cdots, n$.

From here on, we will only be referring to continuous time stochastic processes. The joint probability density function for a Markov process can be written as

$$f(x_1, t_1; \cdots; x_n, t_n) = \left[\prod_{i=2}^{n} f(x_i, t_i | x_{i-1}, t_{i-1}) \right] f(x_1, t_1).$$

The conditional probability $f(x_i, t_i|x_{i-1}, t_{i-1})$ is referred to as *the transition probability*. Using the Bayesian relation

$$f(x_1, t_1; x_2, t_2) = f(x_2, t_2|x_1, t_1)f(x_1, t_1)$$

in calculating the marginal distribution for X_2,

$$\int_{-\infty}^{\infty} f(x_1, t_1; x_2, t_2)dx_1 = f(x_2, t_2)$$

which gives the first Chapman-Kolmogorov equation

$$\int_{-\infty}^{\infty} f(x_2, t_2|x_1, t_1)f(x_1, t_1)dx_1 = f(x_2, t_2).$$

The second Chapman-Kolmogorov equation involves the use of the joint probability distribution functions

$$f(x_1, t_1; x_2, t_2; x_3, t_3) = f(x_3, t_3|x_2, t_2)f(x_2, t_2|x_1, t_1)f(x_1, t_1). \tag{2.3.1}$$

Integrating (2.3.1) with respect to x_2, gives

$$f_{1,3}(x_1, t_1; x_3, t_3) = \int_{-\infty}^{\infty} f(x_3, t_3|x_2, t_2)f(x_2, t_2|x_1, t_1)f(x_1, t_1)dx_2.$$

The above equation can be written as

$$f_{1,3}(x_3, t_3|x_1, t_1)f(x_1, t_1) = \int_{-\infty}^{\infty} f(x_3, t_3|x_2, t_2)f(x_2, t_2|x_1, t_1)f(x_1, t_1)dx_2.$$

Dividing out $f(x_1, t_1)$ results in the second Chapman-Kolmogorov equation

$$f_{1,3}(x_3, t_3|x_1, t_1) = \int_{-\infty}^{\infty} f(x_3, t_3|x_2, t_2)f(x_2, t_2|x_1, t_1)dx_2. \tag{2.3.2}$$

The master equation can be derived from the Chapman-Kolmogorov equation [25, 26].

2.3.1 The Master Equation

The master equation for the probability density with discrete states is

$$\frac{\partial f(x,t)}{\partial t} = \sum_y w(x|y)f(y,t) - w(y|x)f(x,t), \tag{2.3.3}$$

where y is summed over all the possible states of x and $w(y|x)$ is the probability per unit time of transitioning from state x to state y. The master equation for the

probability density with continuous states is

$$\frac{\partial f(x,t)}{\partial t} = \int_{-\infty}^{\infty} w(x|y)f(y,t) - w(y|x)f(x,t)dy. \tag{2.3.4}$$

Analogous master equations can also be derived for transition probabilities. The master equation describes the evolution of the transition probability over *mesoscopically* long intervals of time, the scale between the microscopic and macroscopic,[11] allowing us to determine the dynamic behaviour of a stochastic process [27].

Definition 2.3.9 *A stochastic process is a* ***birth and death process*** *if there is a sequence of birth rates $b(x) \geq 0$ and a sequence of death rates $d(x) \geq 0$ for $x \geq 0$ such that the time spent in the state $X = x$ is an exponential random variable with rate $b(x) + d(x)$, the states $x \pm 1$ are the only states that can be entered from state x and the elapsed time and the probability of jumping to either state $x + 1$, a birth, or state $x - 1$, a death, are mutually independent [28].*

The master equation for a birth and death process is

$$\frac{\partial f(x,t)}{\partial t} = b(x-1)f(x-1,t) - b(x)f(x,t) + d(x+1)f(x+1,t) - d(x)f(x,t).$$

Van Kampen obtained an expansion of the master equation in terms of the large volume, where the truncation of the Taylor expansion in $1/\sqrt{\text{volume}}$, took the form of a partial differential equation known as the Fokker-Planck equation [29].

2.3.2 Fokker-Planck Equation

The Fokker-Planck equation can be used to derive approximate time evolution equations for the probability density or the transition probability of a stochastic process with continuous states. The equation

$$\frac{\partial f(x,t|x_0,t_0)}{\partial t} = -\frac{\partial(\alpha_1(x,t)f(x,t|x_0,t_0))}{\partial x} + \frac{1}{2}\frac{\partial^2(\alpha_2(x,t)f(x,t|x_0,t_0))}{\partial x^2} \tag{2.3.5}$$

is referred to as the forward Fokker-Planck equation [30]. The backward Fokker Planck equation

$$-\frac{\partial f(x,t|x_0,t_0)}{\partial t_0} = \alpha_1(x_0,t_0)\frac{\partial f(x,t|x_0,t_0)}{\partial x_0} + \frac{1}{2}\alpha_2(x_0,t_0)\frac{\partial^2 f(x,t|x_0,t_0)}{\partial x_0^2}$$

can be used to calculate the time it takes for the process to first exit an interval [30]. In (2.3.5), $\alpha_i(x)$ is the i^{th} jump moment for the stochastic process $X(t)$, where

$$\alpha_i(x) = \int_{-\infty}^{\infty} (y-x)^i w(y|x)dy. \tag{2.3.6}$$

Using the i^{th} jump moment equations in (2.3.6), one can find the *macroscopic equation* for any moment, i.e an ordinary differential equation that describes the time evolution of a moment of a stochastic process. The general formula for the hierarchy of macroscopic equations, given by

$$\frac{d\langle X^n(t)\rangle}{dt} = \sum_{i=1}^{n} \begin{pmatrix} n \\ i \end{pmatrix} \langle X^{n-i}(t)\alpha_i(X(t))\rangle, \tag{2.3.7}$$

can be derived from either the master equation or the Fokker-Planck equation. The macroscopic equation provides the time evolution equations for the n^{th} moment.

Next, we will focus on the Langevin equation, which is closely related to the Fokker Planck equation although it takes a completely different form.

2.3.3 The Langevin Equation

The Langevin equation is a stochastic differential equation. Before defining the Langevin equation, we will discuss the difference between a stochastic differential equation and an ordinary differential equation. Looking at the ordinary differential equation

$$\frac{dX(t)}{dt} = A(X(t), t), \tag{2.3.8}$$

where $A(X(t), t)$ is an arbitrary function, we can rewrite it in update form as

$$X(t + dt) = X(t) + A(X(t), t)dt. \tag{2.3.9}$$

In (2.3.9), dt is a non-negative infinitesimal value, implying that any higher order terms of dt are so small that they are negligible [24]. The self consistency condition constrains a Markov process to evolve according to generalizations of (2.3.8) and (2.3.9), known as the Langevin equation. The Langevin equation in its standard form is written as

$$X(t + dt) = X(t) + \beta(X(t), t)dt + \gamma(X(t), t)\mathcal{N}(0, 1)\sqrt{dt}, \tag{2.3.10}$$

where $\mathcal{N}(0, 1)$ is a temporally uncorrelated standard normal random variable with zero mean and unit variance. Dividing (2.3.10) by dt and taking the limit as dt approaches zero we have

$$\frac{dX(t)}{dt} = \beta(X(t), t) + \gamma(X(t), t)\Gamma(t), \tag{2.3.11}$$

where $\Gamma(t)$ is Gaussian white noise. The Langevin equation has a close relation to the Fokker-Planck equation. For the random variable $X(t)$ described in the

Langevin equations (2.3.11), the probability distribution function $f(x,t)$ satisfies

$$\frac{\partial f(x,t)}{dt} = -\frac{\partial(\beta(x,t)f(x,t))}{\partial x} + \frac{1}{2}\frac{\partial^2(\gamma^2(x,t)f(x,t))}{\partial x^2},$$

which is a Fokker-Planck equation. For more details of the derivation see [24]. We will next focus on the special case of the master equation and the Langevin equation that describes a chemical reaction systems.

2.3.4 Chemical Master Equation

Definition 2.3.10 *The time evolution of a well stirred system as a discrete stochastic process is described by* ***stochastic chemical kinetics****.* ***Deterministic chemical kinetics*** *describes a well stirred system by a set of coupled, first order, ordinary differential equations. [23]*

For both the chemical master equation and chemical Langevin equation, we assume that the system is in thermal equilibrium, is well stirred and the system is confined to a constant volume [23]. In [31], Gillespie derives the chemical master equation. We will briefly highlight some of the main points from the paper [31].

Theorem 2.3.1 *If $\boldsymbol{X}(t) = \boldsymbol{x}$, the probability that exactly one reaction, R_i, will occur in the time interval $[t, t+dt)$ is $a_i(\boldsymbol{x})dt + o(dt)$.*

Theorem 2.3.2 *If $\boldsymbol{X}(t) = \boldsymbol{x}$, the probability that no reactions will occur in the time interval $[t, t+dt)$ is $1 - \sum_{i=1}^{M} a_i(\boldsymbol{x})dt + o(dt)$.*

Theorem 2.3.3 *If $\boldsymbol{X}(t) = \boldsymbol{x}$, the probability that more than one reaction occurring in the time interval $[t, t+dt)$ is $o(dt)$.*

Gillespie expressed $P(\mathbf{x}, t+dt|\mathbf{x}_0, t_0)$, the probability that $\mathbf{X}(t) = \mathbf{x}$ conditional on $\mathbf{X}(t_0) = \mathbf{x}_0$, $P(\mathbf{x}, t|\mathbf{x}_0, t_0)$, as the sum of the mutually exclusive events discussed in the three theorems above. This means that

$$\begin{aligned} P(\mathbf{x}, t+dt|\mathbf{x}_0, t_0) = \;& P(\mathbf{x}, t|\mathbf{x}_0, t_0)\big(1 - \sum_{i=1}^{M} a_i(\mathbf{x})dt + o(dt)\big) \\ & + \sum_{i=1}^{M} P(\mathbf{x} - \mathbf{v}_i, t|\mathbf{x}_0, t_0)\big(a_i(\mathbf{x})dt + o(dt)\big) + o(dt) \end{aligned} \tag{2.3.12}$$

Subtracting $P(\mathbf{x}, t|\mathbf{x}_0, t_0)$ from both sides, dividing (2.3.12) by dt and then taking the limit as dt goes to zero, we end up with the chemical master equation

$$\frac{\partial P(\mathbf{x}, t)}{\partial t} = \sum_{i=1}^{M} a_i(\mathbf{x} - \mathbf{v}_i)P(\mathbf{x} - \mathbf{v}_i, t) - a_i(\mathbf{x})P(\mathbf{x}, t), \tag{2.3.13}$$

a special case of the master equation. Next we will look at the form of the chemical Langevin equation.

2.3.5 Chemical Langevin Equation

A derivation of the chemical Langevin equation starts with the update equation

$$X_i(t+dt) = X_i(t) + \sum_{j=1}^{M} v_{ji}\mathcal{K}_j(\mathbf{X}(t), dt), \tag{2.3.14}$$

where $\mathcal{K}_j(\mathbf{X}(t), dt)$ is a random variable that is the number of reactions that occur in the interval $[t, t+dt]$ [27]. We impose the condition that dt is so small that the change in the state during $[t, t+dt]$ is such that $a_j(\mathbf{X}(t')) \approx a_j(\mathbf{X}(t))$ for all $t' \in [t, t+dt]$. This allows us to approximate the $\mathcal{K}_j(\mathbf{X}(t), dt)$ as independent Poisson random variables, $\mathcal{P}(a_j(\mathbf{X}(t)), dt)$ [27]. The second condition that we will impose is that dt is large enough that the expected number of reactions R_j occurring in $[t, t+dt]$ is much larger than 1. This means that

$$a_j(\mathbf{X}(t))dt \gg 1, \, \forall j = 1, \cdots, M.$$

This condition allows the approximation of a Poisson random variable with a normal random variable with the mean $a_j(\mathbf{X}(t))dt$ and the variance $a_j(\mathbf{X}(t))dt$ [27]. Equation (2.3.14) can then be approximated as

$$X_i(t+dt) = X_i(t) + \sum_{j=1}^{M} v_{ji}\mathcal{N}_j(a_j(\mathbf{X}(t))dt, a_j(\mathbf{X}(t))dt), \tag{2.3.15}$$

where $\mathcal{N}(\mu, \sigma^2)$ is a normal random variable with mean μ and variance σ^2. Since $\mathcal{N}(\mu, \sigma^2) = \mu + \sigma\mathcal{N}(0, 1)$, (2.3.15) can be written as

$$X_i(t+dt) = X_i(t) + \sum_{j=1}^{M} v_{ji}a_j(\mathbf{X}(t))dt + v_{ji}\sqrt{a_j(\mathbf{X}(t))dt}\mathcal{N}_j(0, 1) \tag{2.3.16}$$

where $\mathcal{N}_j(0, 1)$ are independent unit normals. Equation (2.3.16) is the standard form of the Langevin equation; putting (2.3.16) in differential form we have the chemical Langevin equation

$$\frac{dX_i(t)}{dt} = \sum_{j=1}^{M} v_{ji}a_j(\mathbf{X}(t)) + v_{ji}\sqrt{a_j(\mathbf{X}(t))}\Gamma_j(t) \tag{2.3.17}$$

where $\Gamma_j(t)$ are temporally uncorrelated, independent Gaussian white noise terms [27].

From the chemical master equation and its approximation in the form of the chemical Langevin equation, one can derive simulation algorithms that accurately describe the behaviour of chemical reaction systems. The SSA provides an exact simulation of the behaviour of a chemical system. An alternative simulation method, explicit tau leaping, was derived from the chemical Langevin equation, (2.3.16) [12]. There have been other approximation methods derived that are able to handle stiff chemical systems. These methods were developed for stiff systems because the SSA runs very slow for a stiff system. As for the explicit tau leaping method, it becomes unstable when dealing with a stiff chemical system. The methods developed to handle stiff systems include the QSSA algorithm, the slow scale SSA, the nested SSA and the implicit tau leaping method.

Chapter 3

Simulation Methods

The following methods simulate well stirred systems at fixes volume and constant temperature. These assumptions allow for the Markov property to be applied to the system [32]. The majority of the methods approximating a chemical system are derived from the chemical master equation [32].

3.1 Stochastic Simulation Algorithm (SSA)

The SSA simulates a realization of the chemical master equation. To approximate the solution to the chemical master equation by the SSA one computes the values of the propensity functions of the system in order to determine which reaction will occur next as well as the time at which that reaction occurs. The SSA makes no distinction between fast and slow reactions. Thus, the computational time is slow for stiff systems, since the SSA computes all of the many fast reactions occurring in between each slow reaction event. In the SSA, the time to the next reaction, τ, is computed as

$$\tau = -\frac{\ln(r_1)}{a_0(\mathbf{x})}, \tag{3.1.1}$$

where $a_0(\mathbf{x})$ is the sum of the propensities:

$$a_0(\mathbf{x}) = \sum_{i=1}^{M} a_i(\mathbf{x}), \tag{3.1.2}$$

and r_1 is a unit uniform random number. The index, j, of the reaction, R_j, that occurs at this time is found by

$$j = \text{smallest integer} \quad \text{such that} \quad \sum_{i=1}^{j} a_i(\mathbf{x}) \geq r_2 a_0(\mathbf{x}), \tag{3.1.3}$$

where r_2 is a unit uniform random number independent of r_1. The following steps are used to implement the SSA over a time interval $[t_0, t_f]$.

Step 1: Set initial conditions of the system, $\mathbf{X}(t_0) = \mathbf{x}_0$.
Step 2: Compute the values of the propensity functions at the current state.
Step 3: Sample two unit uniform random numbers, r_1 and r_2. Compute the time to the next reaction using (3.1.1). Compute the index j of the reaction to occur using (3.1.3).
Step 4: Update the species, $\mathbf{X}(t + \tau) = \mathbf{X}(t) + \mathbf{v}_j$.
Step 5: Update time, $t = t + \tau$.
Step 6: If $t < t_f$ repeat *Steps 2-6.*

3.2 Quasi-Steady State Assumption Applied to the Stochastic Simulation Algorithm (QSSA algorithm)

The quasi steady state assumption can be applied when there is a separation of time scales in the dynamics. It states that a subset of species reach steady state on the fast time scale, allowing the number of species and reactions in a stiff system to be reduced. A species is eliminated when it depends only on fast reactions; its behaviour is given by its steady state. When the quasi steady state assumption is applied to the SSA, the quasi steady state assumption eliminates the fast dynamics of the system by setting the fast species to their steady state under the fast reactions, where slow species are treated as parameters. Then with the steady state values of the fast species the QSSA algorithm defines the time to the next slow reaction as

$$\tau = -\frac{\ln(r_1)}{a_0^s(\mathbf{x})}, \tag{3.2.1}$$

where $a_0^s(\mathbf{x})$ is the sum of the slow propensities:

$$a_0^s(\mathbf{x}) = \sum_{i=1}^{m_s} a_i^s(\mathbf{x}).$$

To find the index, j, of the next slow reaction the QSSA algorithm uses [13]

$$j = \text{smallest integer} \quad \text{such that} \quad \sum_{i=1}^{j} a_i^s(\mathbf{x}) \geq r_2 a_0^s(\mathbf{x}). \tag{3.2.2}$$

In (3.2.1) and (3.2.2), r_1 and r_2 are unit interval uniform random numbers. Rao and Arkin were able to construct the following algorithm from the previous information for the QSSA algorithm for a time interval $[t_0, t_f]$[13].

Step 1: Partition the fast and slow reactions, followed by partitioning the fast and slow species. Find the equations for the fast species steady state from the deterministic equations of the fast system, treating the slow species as parameters. Then set the initial conditions $\mathbf{X}(t_0) = (x_0^f, x_0^s)$.
Step 2: Set the fast species to its steady state using the equations found in *Step 1:*, taking any conservation into account. Round to the nearest integer greater than zero for each fast species.
Step 3: Compute the values of the slow propensity functions.
Step 4: Sample two unit uniform random numbers, r_1 and r_2. Then compute the time to the next slow reaction from (3.2.1) and the index of the next slow reaction from (3.2.2).
Step 5: Update the species $\mathbf{X}(t+\tau) = \mathbf{X}(t) + \mathbf{v}_j$ and time $t = t + \tau$.
Step 6: If $t < t_f$ repeat *Steps 2-6*.

In [13] there was a probabilistic approach taken to develop the QSSA algorithm. The probabilistic approach to the QSSA algorithm is similar to the slow scale SSA in [17].

3.3 Slow Scale SSA

Definition 3.3.1 *The **virtual fast processes**, denoted as $\hat{\boldsymbol{X}}^f(t)$, consists of the fast species, $\boldsymbol{X}^f(t)$, reacting under the fast reactions only.*

The way that $\hat{\mathbf{X}}^f(t)$ is defined makes it Markovian since the state is effected by the fast reactions and any slow species that participates in a fast reaction, e.g. as a catalyst, is treated as a parameter that is considered to be constant. The virtual fast process obeys the chemical master equation,

$$\begin{aligned} \frac{\partial P(\mathbf{x}^f, t|\mathbf{x}_0, t_0)}{\partial t} &= \sum_{i=1}^{m_f} a_i^f(\mathbf{x}_i^f - \mathbf{v}_i^f, \mathbf{x}_0^s) P(\mathbf{x}_i^f - \mathbf{v}_i^f, t|\mathbf{x}_0, t_0) \\ &- \sum_{i=1}^{m_f} a_i^f(\mathbf{x}_i^f, \mathbf{x}_0^s) P(\mathbf{x}_i^f, t|\mathbf{x}_0, t_0) \end{aligned} \tag{3.3.1}$$

One of the requirements for application of the slow scale SSA is that $\hat{\mathbf{X}}^f(t)$ be stable, implying

$$\lim_{t\to\infty} P(\mathbf{x}^f, t|\mathbf{x}_0, t_0) = P(\mathbf{x}^f, \infty|\mathbf{x}_0)$$

exists, i.e. the distribution of the virtual fast process, $\hat{\mathbf{X}}^f(t)$, will reach a limit as $t \to \infty$. However, this does not mean that the distribution of the fast species, $\mathbf{X}^f(t)$, should reach a limit as $t \to \infty$ because the slow reactions may change the distribution of the fast species, $\mathbf{X}^f(t)$, throughout the simulation. With $\hat{\mathbf{X}}^f(\infty)$ in

steady state (3.3.1) reads as

$$0 = \sum_{i=1}^{m_f} a_i^f(\mathbf{x}^f - \mathbf{v}_i^f, \mathbf{x}_0^s)P(\mathbf{x}^f - \mathbf{v}_i^f, \infty|\mathbf{x}_0) - a_i^f(\mathbf{x}^f, \mathbf{x}_0^s)P(\mathbf{x}^f, \infty|\mathbf{x}_0). \tag{3.3.2}$$

This set of algebraic equations for $\mathbf{x}^f$ is simpler to solve then the set of differential equations in (3.3.1). The slow scale SSA will only provide a good approximation of the system behaviour if the convergence of the virtual fast process, $\hat{\mathbf{X}}^f(t)$, to the virtual fast process in its steady state, $\hat{\mathbf{X}}^f(\infty)$, occurs quickly compared to the slow time scale. In [17], the authors define the *slow scale propensity* function for R_i^s as

$$\bar{a}_i^s(\mathbf{x}^f, \mathbf{x}^s) = \sum_{\mathbf{x}^{f'}} P(\mathbf{x}^{f'}, \infty|\mathbf{x}^f, \mathbf{x}^s)a_i^s(\mathbf{x}^{f'}, x^s). \tag{3.3.3}$$

Treating the fast species as though they were distributed according to $\hat{\mathbf{X}}^f$, the slow scale propensity function is the average of the propensity functions over the fast variables. The slow scale propensity functions can be written in terms of moments of the virtual fast process in its steady state, $\hat{\mathbf{X}}^f(\infty)$, as follows.
If the propensity function is independent of any fast species then

$$\bar{a}_i^s(\mathbf{x}_s) = a_i^s(\mathbf{x}_s). \tag{3.3.4}$$

Otherwise the following rules can be used

$$a_i^s(\mathbf{x}^f, \mathbf{x}^s) = c_i x_j^f \quad \Rightarrow \quad \bar{a}_i^s(\mathbf{x}^f, \mathbf{x}^s) = c_i\langle\hat{X}_j^f(\infty)\rangle \tag{3.3.5}$$

$$a_i^s(\mathbf{x}^f, \mathbf{x}^s) = c_i x_{j'}^s x_j^f \quad \Rightarrow \quad \bar{a}_i^s(\mathbf{x}^f, \mathbf{x}^s) = c_i x_{j'}^s\langle\hat{X}_j^f(\infty)\rangle \tag{3.3.6}$$

$$a_i^s(\mathbf{x}^f, \mathbf{x}^s) = c_i x_j^f(x_j^f - 1) \quad \Rightarrow \quad \bar{a}_i^s(\mathbf{x}^f, \mathbf{x}^s) = c_i\langle\hat{X}_j^f(\infty)(\hat{X}_j^f(\infty) - 1)\rangle \tag{3.3.7}$$

$$a_i^s(\mathbf{x}^f, \mathbf{x}^s) = c_i x_j^f x_{j'}^f \quad \Rightarrow \quad \bar{a}_i^s(\mathbf{x}^f, \mathbf{x}^s) = c_i\langle\hat{X}_j^f(\infty)\hat{X}_{j'}^f(\infty)\rangle \tag{3.3.8}$$

Using the information from above, Cao, Gillespie and Petzold approximate the SSA by having the *slow scale SSA* run on the slow time scale only [17]. This is achieved by summing the slow scale propensity functions.

$$\bar{a}_0^s(\mathbf{x}) = \sum_{i=1}^{m_s} \bar{a}_i^s(\mathbf{x}). \tag{3.3.9}$$

Given two unit interval uniform random numbers, r_1 and r_2, as well as the state $\mathbf{X}(t)$, the time to the next slow reaction is

$$\tau = -\frac{\ln(r_1)}{\bar{a}_0^s(\mathbf{x})}, \tag{3.3.10}$$

and the next slow reaction, R_j^s, is determined from

$$j = \text{smallest integer} \quad \text{such that} \quad \sum_{i=1}^{j} \bar{a}_i^s(\mathbf{x}) \geq r_2 \bar{a}_0^s(\mathbf{x}). \tag{3.3.11}$$

The determination of the time to the next slow reaction and finding the index of the next slow reaction is similar to that of the QSSA algorithm, where the only difference in the propensity functions is that they rely on the moments of the fast species or the moment of the product of the fast species. The slow scale SSA similar is implemented with the following steps for the time interval of $[t_0, t_f]$[17].

Step 1: Set all the parameter values, then partition the fast and slow species as well as the fast and slow reactions. Identify the virtual fast process. Compute the mean and variance from the stationary moments. Find a distribution that will approximate the virtual fast process using the mean and variance from the stationary moments. Provide the initial state of the species, $\mathbf{X}(t_0) = (\mathbf{x}_0^f, \mathbf{x}_0^s)$ and set the starting time t_0.
Step 2: Compute the values of the slow scale propensity functions at $\mathbf{X}(t)$.
Step 3: Compute the time to the next slow reaction using equation (3.3.10) and the index of the next slow reaction using (3.3.11).
Step 4: Update the time and species with the values found in *Step 3*, i.e., $t = t + \tau$ and $\mathbf{X}(t + \tau) = \mathbf{X}(t) + \mathbf{v}_j^s$
Step 5: Sample from the distribution found for the virtual fast process to approximate the fast species. Then round the fast species to the nearest integer that is greater than zero.
Step 6: If $t < t_f$ repeat *Step 2 - Step 6*.

We consider three different ways to find the mean and variance of a normal distribution for the virtual fast process. The first way is set the mean to the equilibrium in the fast system, taking the variance for the fast species as zero. In this case the slow scale SSA reduces to the previously described QSSA algorithm. The second approach to finding the mean and variance for the normal distribution of the virtual fast process is to use birth and death formulas

$$\hat{\alpha}(x) = b(x-1) - d(x) \tag{3.3.12}$$

From [17], $\hat{x} \geq 0$ is a steady state of the virtual fast process if and only if

$$\text{for some } \delta \in [0,1), \hat{\alpha}(\hat{x}+\delta) = 0 \text{ and } \hat{\alpha}'(\hat{x}+\delta) < 0. \tag{3.3.13}$$

Then $\langle \hat{X}(\infty) \rangle = \hat{x}$, if the virtual fast process has only one steady state $\hat{x}$ which can be approximately obtained from $\hat{\alpha}(\hat{x}) = 0$ and $\hat{\alpha}'(\hat{x}) < 0$ [17]. The variance of $\hat{x}$ is

$$\sigma^2 = \frac{d(\hat{x})}{-\hat{\alpha}'(\hat{x})}. \tag{3.3.14}$$

The variance in (3.3.14) will be used in a Gaussian distribution [17]. The final method to finding the mean and variance of the normal distribution for the virtual fast process is to use the stationary moment equations as follows. To find the first moment solve

$$0 = \sum_{i=1}^{m_f} v_{ij}^f \langle a_i^f(\mathbf{X}(\infty))\rangle \quad (j = 1, \cdots, n_f). \tag{3.3.15}$$

To find the second moment solve

$$\begin{aligned} 0 = &\sum_{i=1}^{m_f} v_{ij}^f \langle X_k^f(\infty) a_i^f(\mathbf{X}(\infty))\rangle + v_{ik}^f \langle X_j^f(\infty) a_i^f(\mathbf{X}(\infty))\rangle \\ &+ \sum_{i=1}^{m_f} v_{ij}^f v_{ik}^f \langle a_i^f(\mathbf{X}(\infty))\rangle \quad j = 1, \cdots, n_f \quad k = j, \cdots, n_f \end{aligned} \tag{3.3.16}$$

These moment equations originate from (3.3.2). To find (3.3.15), one would multiply the interior of the summation from (3.3.2) by the a fast species, $X_i^f(t)$. To find (3.3.16), one would multiply the interior of the summation from (3.3.2) by two fast species, $X_i^f(t)$ and $X_j^f(t)$. For details see [17]. In the case where the propensity functions are linear we do not have to truncate any higher moments. If the propensity functions are nonlinear we would have to truncate the higher moments appearing in (3.3.15) and (3.3.16) to be able to find the first and second moments of the virtual fast process.

The QSSA algorithm and slow scale SSA both suppose that the fast system reaches equilibrium quickly compared to the slow time scale. The difference between them is that the QSSA algorithm uses deterministic equations and the slow scale SSA uses stationary moment equations. The slow scale SSA is a probabilistic version of the QSSA algorithm; both algorithms share the assumption that the fast species are Markovian, that the fast system must reach an equilibrium and that the probability distribution of the fast species conditional on the slow species is time invariant.

3.4 Nested SSA

The nested SSA [19] can be briefly described as two SSA simulations in one. One SSA handles the slow reactions while the other, which runs between each slow reaction, handles only the fast reactions. Since most of the SSA's time is spent on computing the fast reactions, the nested SSA's focus is to approximate these fast reactions as accurately as possible. The *inner SSA* handles only the fast reactions and runs n independent SSA's for the time interval $[t, t + T_f + T_0]$. The inner SSA runs between each slow step to approximate the virtual fast process. The inner

SSA also computes the *modified slow rates* of the system, defined as

$$\bar{a}_j^s = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{T_f}\int_{T_0}^{T_0+T_f} a_j^s(\mathbf{x}^i(\gamma))d\gamma,$$

where $\mathbf{x}^i(\gamma)$ is $\mathbf{x}$ is the state at time γ of the i^{th} run. The parameter T_0 is chosen to minimize the error in the approximation of the modified slow rates caused by the transient. Since the inner SSA can be parallelized it can be beneficial to take n large [19], but in [19] the authors take $n = 1$. Of course, taking more than one inner SSA will effect the CPU time of the nested SSA, as we will see in later sections. The authors also suggest it is possible to take $T_f \ll 1$ and still have a good approximation of the fast system, assuming that the slow time scale is of order 1 [18]. The nested SSA algorithm which as follows for the time interval $[t_0, t_f]$ [19].

Step 1: Partition the species into fast species and slow species, the reactions into fast reactions and slow reactions and the propensity functions into fast propensity functions and slow propensity functions. Set the initial conditions $\mathbf{X}(t_0) = \mathbf{x}_0$.
Step 2: Run n SSA's on the fast reactions only for a time from t to $t + T_f + T_0$ with the initial condition being the state $\mathbf{X}(t) = \mathbf{x}$.
Step 3: Find the modified slow rates

$$\bar{a}_j^s = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{T_f}\int_{T_0}^{T_0+T_f} a_j^s(\mathbf{x}^i(\gamma))d\gamma.$$

Step 4: Sample two unit uniform random numbers, r_1 and r_2. Compute the time to the next slow reaction

$$\tau = -\ln(r_1)/\bar{a}^s$$

where $\bar{a}^s = \sum_{i=1}^{m_s}\bar{a}_i^s$ and compute the index of the next slow reaction from

$$j = \text{smallest integer} \quad \text{such that} \quad \sum_{i=1}^{j}\bar{a}_i^s(\mathbf{x}) \geq r_2\bar{a}^s(\mathbf{x}).$$

Step 5: Update the species and the time, $\mathbf{X}(t+\tau) = \mathbf{X}(t) + \mathbf{v}_j$ and $t = t + \tau$. If $X_i(t+\tau) < 0$ then $X_i(t+\tau) = 0$.
Step 6: If $t < t_f$ repeat *Step 2 - Step 6*.

The nested SSA is similar to the slow scale SSA since they both sample from a distribution to approximate the virtual fast process. The difference between the nested SSA and the slow scale SSA is that the nested SSA numerically generates the probability distribution for the fast species, whereas the slow scale SSA determines the probability distribution for the fast species analytically. The next method we will be looking at is the implicit tau leaping method.

3.5 Implicit-Tau Leaping Method

The implicit-tau leaping method is derived from the chemical Langevin equation, similar to the explicit-tau leaping method. The explicit-tau leaping method is very similar to the explicit Euler method [20]. The explicit-tau update step is

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \tau \sum_{i=1}^{M} \mathbf{v}_j a_j(\mathbf{X}(t+\tau)) + \tau^{1/2} \mathbf{v}_j a_j(\mathbf{X}(t))^{1/2} \mathcal{N}_j(0,1),$$

where $\mathcal{N}_j(0,1)$ are independent standard Normals [12]. The explicit-tau method does not handle stiff systems very well, since if the time jump is too big the method becomes unstable. Using small time jumps will not make the explicit tau method run faster than the SSA, which is what the explicit tau was developed for. The implicit-tau leaping method is able to handle a stiff system with little worry of the method becoming unstable and able to run faster than the SSA.

With the implicit-tau leaping method, there is no need to distinguish fast reactions or species as the method directly solves a set of equations, linear or non-linear, for the desired time. The implicit-tau method was developed by Rathinam, Petzold, Cao and Gillespie [20]. Similar to the explicit-tau leaping method the authors attempt to implicitize the generation of Poisson random variables, $\mathcal{P}_j(a_j(\mathbf{x}(t+\tau)), \tau)$, where $\mathbf{X}(t+\tau) = \mathbf{x}(t+\tau)$ is unknown. To resolve this problem they took the random variable $\mathcal{P}_j$ and wrote it as the sum of two parts, one being the mean value of $\mathcal{P}_j$, $a_j\tau$, and the second, $\mathcal{P}_j - a_j\tau$, the zero mean random variable. At the known state $\mathbf{X}(t)$, they evaluated the zero mean random variable and the mean value at the unknown state $\mathbf{X}(t+\tau)$. Then they were able to describe the implicit method, with statistically independent Poisson random variables, as follows

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \sum_{i=1}^{M} \mathbf{v}_j a_j(\mathbf{X}(t+\tau))\tau + \mathbf{v}_j(\mathcal{P}_j(a_j(\mathbf{X}(t)), \tau) - a_j(\mathbf{X}(t))\tau). \quad (3.5.1)$$

Since the Poisson random variables do not depend on the next value of the species, $\mathbf{X}(t+\tau)$, the generation of the Poisson random numbers is greatly simplified. An approximation of the Poisson with a Normal in (3.5.1) gives

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \tau \sum_{i=1}^{M} \mathbf{v}_j a_j(\mathbf{X}(t+\tau)) + \tau^{1/2} \mathbf{v}_j a_j(\mathbf{X}(t))^{1/2} \mathcal{N}_j(0,1), \quad (3.5.2)$$

where $\mathcal{N}_j(0,1)$ are independent standard Normals.

Rathinam, Petzold, Cao and Gillespie developed a method for restoring the true behaviour in the fast species. Any fluctuations that may occur in the fast species due to noise are damped out by the large tau leaps [20]. The idea to reduce the damping is to take tau leaps on the slow time scale of the system followed by a

sequence of smaller tau leaps on the fast time scale. The smaller time steps can be taken using the implicit or explicit tau methods, as there would be little worry of instability. With just the large implicit tau leaps the statistical distributions may become narrow, which is corrected by the sequence of small tau leaps. To implement the implicit tau leaping method for the time interval $[t_0, t_f]$, follow these steps

Step 1: Set the initial values of the species, $\mathbf{X}(t_0) = \mathbf{x}_0$
Step 2: Define the time jump, τ. It is possible to use a τ similar to (3.2.1), or use a fixed τ, or use the method of tau leaps on the slow time scale then a sequence of tau leaps on the fast time scale to reduce the dampening in the fast variables.
Step 3: Using τ, solve the set of implicit equations, (3.5.2), to find the species at the new time, e.g. using Newton's method.
Step 4: Update the time $t = t + \tau$.
Step 5: If $t < t_f$ repeat *Step 2* - *Step5* until desired time is reached.

The implicit tau method takes a different approach than the other methods, since it involves an implicit description of the species at the next time step. The implicit tau method has an approach for time stepping to reduce the dampening of the fluctuations as mentioned previously. This approach is not taken by any of the other approximation methods in this thesis. It would be worth investigating to see if this approach of a sequence of time steps on the fast time scale improve a method's approximation of a chemical reaction system. A significant difference is that the implicit tau method does not require separation of the species or the reactions.

We will now apply the QSSA algorithm, slow scale SSA, the nested SSA and the implicit tau leaping method to a variety of examples to see how each compare to the SSA. The goal of this analysis is to see whether one method prevails for any kind of chemical system or if a certain method excels only for a specific type of chemical system. We will consider a variety of measures to determine what method is best suited for each example.

Chapter 4

Example 1: Fast Reversible Dimerization

This example is from [17]. The system is a fast reversible dimerization, with three species involved in four reactions:

$$\begin{aligned} 2S_1 &\underset{c_2}{\overset{c_1}{\rightleftharpoons}} S_2 \\ S_1 &\xrightarrow{c_3} \emptyset \\ S_2 &\xrightarrow{c_4} S_3 \end{aligned}$$

Following mass action, the propensity functions for this system are

$$\mathbf{a}(\mathbf{x}) = (c_1 x_1(x_1 - 1)/2, c_2 x_2, c_3 x_1, c_4 x_2).$$

The stoichiometry vectors are

$$\mathbf{v}_1 = (-2, 1, 0), \quad \mathbf{v}_2 = (2, -1, 0), \quad \mathbf{v}_3 = (-1, 0, 0), \quad \mathbf{v}_4 = (0, -1, 1).$$

Taking

$$2S_1 \underset{c_2}{\overset{c_1}{\rightleftharpoons}} S_2$$

as the fast reactions, the fast species in this example are S_1 and S_2. The fast stoichiometry vectors are $\mathbf{v}_1^f = (-2, 1)$ and $\mathbf{v}_2^f = (2, -1)$. We consider this example to be a fairly representative example with straightforward dynamics. The two fast species both decay over time, while the slow species grows until S_2 has diminished.

4.1 Theoretical Analysis of the System

The SSA, QSSA algorithm, slow scale SSA and the nested SSA approximate a solution to the chemical master equation which for this example is

$$\begin{aligned}\frac{\partial P(\mathbf{x},t|\mathbf{x}_0,t_0)}{\partial t} &= \frac{c_1(x_1+2)(x_1+1)}{2}P(\mathbf{x}-\mathbf{v}_1,t|\mathbf{x}_0,t_0)-\frac{c_1x_1(x_1-1)}{2}P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_2(x_2+1)P(\mathbf{x}-\mathbf{v}_2,t|\mathbf{x}_0,t_0)-c_2x_2P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_3(x_1+1)P(\mathbf{x}-\mathbf{v}_3,t|\mathbf{x}_0,t_0)-c_3x_1P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_4(x_2+1)P(\mathbf{x}-\mathbf{v}_4,t|\mathbf{x}_0,t_0)-c_4x_2P(\mathbf{x},t|\mathbf{x}_0,t_0).\end{aligned} \tag{4.1.1}$$

Implementation of the slow scale SSA and the nested SSA involves finding a probability distribution for the fast species under only the fast reactions. Reducing the system to just the fast reactions, the chemical master equation for the virtual fast process is

$$\begin{aligned}\frac{\partial P(\mathbf{x}^f,\mathbf{x}^s,t|\mathbf{x}_0,t_0)}{\partial t} &= \frac{c_1(x_1+2)(x_1+1)}{2}P(\mathbf{x}^f-\mathbf{v}_1^f,x^s,t|\mathbf{x}_0,t_0)\\ &- \frac{c_1x_1(x_1-1)}{2}P(\mathbf{x}^f,\mathbf{x}^s,t|\mathbf{x}_0,t_0)+c_2(x_2+1)P(\mathbf{x}^f-\mathbf{v}_2^f,x_0^s,t|\mathbf{x}_0,t_0)\\ &- c_2x_2P(\mathbf{x}^f,\mathbf{x}^s,t|\mathbf{x}_0,t_0).\end{aligned} \tag{4.1.2}$$

Summing over all possible values for $\mathbf{x}^s$, (4.1.2) becomes

$$\begin{aligned}\frac{\partial P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)}{\partial t} &= \frac{c_1(x_1+2)(x_1+1)}{2}P(\mathbf{x}^f-\mathbf{v}_1^f,t|\mathbf{x}_0,t_0)-\frac{c_1x_1(x_1-1)}{2}P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)\\ &+ c_2(x_2+1)P(\mathbf{x}^f-\mathbf{v}_2^f,t|\mathbf{x}_0,t_0)-c_2x_2P(\mathbf{x}^f,t|\mathbf{x}_0,t_0).\end{aligned} \tag{4.1.3}$$

There is a conservation in the fast system such that $X_T = X_1(t) + 2X_2(t)$, where $X_T = x_T$ is a constant on the fast time scale. With the conservation in the fast system (4.1.3) can be rewritten as

$$\begin{aligned}\frac{\partial Q(x_2,t|\mathbf{x}_0,t_0)}{\partial t} &= \frac{c_1(x_T-2x_2+2)(x_T-2x_2+1)}{2}Q(x_2-1,t|\mathbf{x}_0,t_0)\\ &- \frac{c_1(x_T-2x_2)(x_T-2x_2-1)}{2}Q(x_2,t|\mathbf{x}_0,t_0)\\ &+ c_2(x_2+1)Q(x_2+1,t|\mathbf{x}_0,t_0)-c_2x_2Q(x_2,t|\mathbf{x}_0,t_0)\end{aligned} \tag{4.1.4}$$

where $Q(x_2,t|\mathbf{x}_0,t_0) = P(x_T-2x_2,x_2,t|\mathbf{x}_0,t_0)$. (4.1.4) describes a birth and death process for x_2, where the birth rate is

$$b(x_2) = \frac{c_1(x_T-2x_2)(x_T-2x_2-1)}{2}$$

and the death rate is

$$d(x_2) = c_2x_2.$$

The QSSA algorithm, slow scale SSA and the nested SSA can only be used when the fast species reach a steady state under the fast reactions. In such cases the first and second moment will be in steady state as well. We consider the time evolution of the first and second moments to see how they relate to the approximation methods and the SSA. Our theoretical descriptions of these moments will provide a basis for evaluating the quality of the approximation methods illustrated in Chapter 4. The time evolution of the first moment is

$$\begin{aligned}\frac{d\langle X_2(t)\rangle}{dt} &= \langle b(X_2(t))\rangle - \langle d(X_2(t))\rangle \\ &= \frac{c_1(X_T^2 - X_T) + 2\langle X_2(t)\rangle(c_1 - 2c_1X_T - c_2) + 4c_1\langle X_2^2(t)\rangle}{2} \quad (4.1.5)\end{aligned}$$

In our implementation of the QSSA algorithm, $\langle X_2(t)\rangle$ under the fast reactions is just the steady state solution of X_2's deterministic equation in the fast system; the species $X_2(t)$ is assumed to have deterministic behaviour [17]. In general, the time evolution of the second moment is (see [11])

$$\begin{aligned}\frac{d\langle X_2^2(t)\rangle}{dt} &= \langle b(X_2(t))\rangle + \langle d(X_2(t))\rangle \\ &+ 2\langle X_2(t)b(X_2(t))\rangle - 2\langle X_2(t)d(X_2(t))\rangle \quad (4.1.6) \\ &= \frac{d\langle X_2(t)\rangle}{dt} + (X_T^2 - X_T + 2c_2)\langle X_2(t)\rangle + 2(c_1 - 2c_1X_T - c_2)\langle X_2^2(t)\rangle \\ &+ 4c_1\langle X_2^3(t)\rangle.\end{aligned}$$

Notice that the evolution of each moment depends on a higher moment. To arrive at a closed system, we must eliminate the higher moment by making an assumption. We assume that the distribution is Gaussian, i.e. the third central moment is zero. We write the third moment of $X_2(t)$ as

$$\begin{aligned}\langle X_2^3(t)\rangle &= \langle([X_2(t) - \langle X_2(t)\rangle] + \langle X_2(t)\rangle)^3\rangle \\ &= \langle(X_2(t) - \langle X_2(t)\rangle)^3\rangle + 3\langle(X_2(t) \\ &-\langle X_2(t)\rangle)^2\rangle\langle X_2(t)\rangle + 3\langle X_2(t) \\ &-\langle X_2(t)\rangle\rangle\langle X_2(t)\rangle^2 + \langle X_2(t)\rangle^3\end{aligned}$$

Using the assumption that the distribution of $X_2(t)$ is Gaussian, implying $\langle(X_2(t) - \langle X_2(t)\rangle)^3\rangle = 0$, then the third moment for $X_2(t)$ becomes

$$\langle X_2^3(t)\rangle = 3\langle X_2^2(t)\rangle\langle X_2(t)\rangle - 2\langle X_2(t)\rangle^3 \quad (4.1.7)$$

Substituting (4.1.7) into (4.1.6) the time evolution of the second moment is approximated by

$$\begin{aligned}\frac{d\langle X_2^2(t)\rangle}{dt} &\approx \frac{d\langle X_2(t)\rangle}{dt} + (c_1(X_T^2 - X_T) + 2c_2)\langle X_2(t)\rangle + 2(c_1 - 2c_1X_T - c_2)\langle X_2^2(t)\rangle \\ &+ 12c_1\langle X_2^2(t)\rangle\langle X_2(t)\rangle - 6c_1\langle X_2(t)\rangle^3 \quad (4.1.8)\end{aligned}$$

To find the equilibrium of the moment equations we will solve

$$c_1(X_T^2 - X_T) + 2\langle X_2(t)\rangle(c_1 - 2c_1X_T - c_2) + 4c_1\langle X_2^2(t)\rangle = 0 \quad (4.1.9)$$

$$\begin{aligned}&(c_1(X_T^2 - X_T) + 2c_2)\langle X_2(t)\rangle + 2(c_1 - 2c_1X_T - c_2)\langle X_2^2(t)\rangle \\ &+12c_1\langle X_2^2(t)\rangle\langle X_2(t)\rangle - 6c_1\langle X_2(t)\rangle^3 = 0 \end{aligned} \quad (4.1.10)$$

From (4.1.9)

$$\langle X_2^2(t)\rangle = \frac{c_1(X_T^2 - X_T) + 2\langle X_2(t)\rangle(c_1 - 2c_1X_T - c_2)}{4c_1} \quad (4.1.11)$$

Substituting (4.1.11) into (4.1.10), the equation to find the first moment is

$$\begin{aligned} & (c_1(X_T^2 - X_T) + 2c_2)\langle X_2(t)\rangle \\ + \;& 2(c_1 - 2c_1X_T - c_2)\frac{c_1(X_T^2 - X_T) + 2\langle X_2(t)\rangle(c_1 - 2c_1X_T - c_2)}{4c_1} \\ + \;& 3\big(c_1(X_T^2 - X_T) + 2\langle X_2(t)\rangle + (c_1 - 2c_1X_T - c_2)\big)\langle X_2(t)\rangle \\ - \;& 6c_1\langle X_2(t)\rangle^3 = 0, \end{aligned} \quad (4.1.12)$$

a cubic equation that does not admit a convenient analytical solution. The mean for X_1 is

$$\langle X_1(t)\rangle = X_T - 2\langle X_2(t)\rangle$$

The variance for X_1 is

$$\langle\langle X_1(t)\rangle\rangle = 4\langle\langle X_2(t)\rangle\rangle.$$

With these equations for the moments of X_1 and X_2, one could use these to sample from a Gaussian distribution to approximate the virtual fast process of the slow scale SSA. We will see later on in this example that there is an alternative way to find the mean and variance for our Gaussian distribution. As mentioned previously in this paper, a chemical system can also be represented by a chemical Langevin equation. The implicit tau leaping method approximates a solution to the chemical Langevin equation. The Chemical Langevin Equation for this system is

$$\begin{aligned} \frac{dX_1(t)}{dt} &= -c_1X_1(t)(X_1(t) - 1) + 2c_2X_2(t) - c_3X_1(t) \quad (4.1.13)\\ &- \sqrt{2c_1X_1(t)(X_1(t) - 1)}\Gamma_1(t) + 2\sqrt{c_2X_2(t)}\mathcal{N}_2(t) - \sqrt{c_3X_1(t)}\Gamma_3(t) \\ \frac{dX_2(t)}{dt} &= c_1X_1(t)(X_1(t) - 1)/2 - c_2X_2(t) - c_4X_2(t) \quad (4.1.14)\\ &+ \sqrt{c_1X_1(t)(X_1(t) - 1)/2}\Gamma_1(t) - \sqrt{c_2X_2(t)}\mathcal{N}_2(t) - \sqrt{c_4X_2(t)}\Gamma_4(t) \\ \frac{dX_3(t)}{dt} &= c_4X_2(t) - \sqrt{c_4X_2(t)}\Gamma_4(t) \quad (4.1.15)\end{aligned}$$

where $\Gamma_i(t)$ are temporally uncorrelated white noise [27]. With the knowledge of the theoretical representation of this system, next we will consider the details of

implementing each algorithm.

4.2 QSSA algorithm

The QSSA algorithm approximates the distribution of the fast species as the delta distribution at the steady state of the fast species under the fast reactions in the deterministic model. The deterministic equations for this system are

$$\frac{dX_1}{dt} = -c_1 X_1^2 + 2c_2 X_2 - c_3 X_1 \tag{4.2.1}$$

$$\frac{dX_2}{dt} = \frac{c_1 X_1^2}{2} - c_2 X_2 - c_4 X_2 \tag{4.2.2}$$

$$\frac{dX_3}{dt} = c_4 X_2. \tag{4.2.3}$$

The species X_1 and X_2 are fast with a conservation $X_T = X_1 + 2X_2$ in the fast reactions only. Ignoring all the slow reactions, the equation to be solved is $c_1 X_1^2 = 2c_2 X_2$. Using the conservation relation the equation is $c_1(X_T - 2X_2)^2 = 2c_2 X_2$. The approximation for X_2 is

$$X_2 = \frac{2c_1 X_T + c_2 - \sqrt{c_2^2 + 4c_1 c_2 X_T}}{4c_1}. \tag{4.2.4}$$

Using the conservation relation $X_T = X_1 + 2X_2$, the approximation for X_1 is

$$X_1 = X_T - 2X_2 = \frac{-c_2 + \sqrt{c_2^2 + 4c_1 c_2 X_T}}{2c_1}. \tag{4.2.5}$$

With the approximation of the virtual fast processes the fast time scale is removed and the following slow reactions remain to be simulated

$$\begin{aligned} S_1 &\xrightarrow{c_3} \emptyset \\ S_2 &\xrightarrow{c_4} S_3. \end{aligned}$$

4.3 Slow Scale SSA

For this example we will take a normal distribution to approximate the behaviour of the virtual fast process in the slow scale SSA. We will find the mean of the virtual fast process using the birth and death formulas from section 3.3. The relative maximum of $P(\hat{x_2}^f, \infty)$ can be computed as the greatest integer of the down going root of $b(x_2-1)-d(x_2) = 0$ [17] where $b(x_2) = c_1(4x_2^2 - 2x_2(2x_T-1) + x_T(x_T-1))/2$,

$d(x_2) = c_2 x_2$ and $x_T = x_1 + 2x_2$. Solving for x_2,

$$x_2 = \frac{2c_1 x_T + 3c_1 + c_2 - \sqrt{(2c_1 x_T + 3c_1 + c_2)^2 - 4c_1^2(x_t + 1)(x_t + 2)}}{4c_1} \tag{4.3.1}$$

Denote the greatest integer function by $x_2^{INT} = \lceil x_2 \rceil$. x_2^{INT} is the stable state of $\hat{X}_2^f(\infty)$ for the birth and death process [17]. The variance that is used for the normal distribution of $\hat{X}_2^f(\infty)$ is

$$\langle\langle \hat{X}_2^f(\infty) \rangle\rangle = \frac{c_2 x_2^{INT}}{-4c_1 x_2^{INT} + c_1(2x_T + 3) + c_2}, \tag{4.3.2}$$

which is from (3.3.14). When we sample from the normal distribution for this example we must make sure that $X_2 \geq 0$ and that $X_2 \leq X_T/2$ since $X_1 + 2X_2 = X_T$ [17].

4.4 Nested SSA

We begin by addressing which choice of design parameters will produce the best results for the nested SSA. The parameters are: n, the number of inner SSA's we run between the outer SSA steps; the amount of time to run the inner SSA, $T_f + T_0$, and the length of the transient T_0. We will test $T_f = \{0.005, 0.01, 0.015, 0.02, 0.025\}$, $T_0 = \{0.005, 0.01, 0.015, 0.02\}$ and $n = \{1, 2\}$, chosen as reasonable values after extensive testing. Our first criterion will be that the choice of parameters must provide for a CPU time less than that of the SSA. The initial number of species for the simulation is

$$\mathbf{X}(0) = (540, 730, 0).$$

The simulations were run for 400 time units. The reaction rates are

$$c_1 = 1, \quad c_2 = 200, \quad c_3 = 0.02, \quad c_4 = 0.004.$$

To run a single trajectory of 400 time units, the CPU time for the SSA was approximately 135.62 seconds. Table 4.4.1 indicates parameter values that result in a CPU time less than that of the SSA. Table 4.4.1 is for a single run of the inner loop, i.e. $n = 1$. We call a run *consistent* if the nested SSA's approximation was good enough so no significant problems occurred before the end of the simulation. We found that the fastest most consistent combination of T_f and T_0 for this example was $T_f = 0.02$ and $T_0 = 0.02$. We will also consider $T_f = 0.025$ and $T_0 = 0.02$ (with $n = 1$) because a larger T_f value will improve the approximation of the virtual fast process.

Table 4.4.2 shows results for two runs of the inner loop, i.e. $n = 2$. We will only consider parameters that produce a CPU time faster than 44.48 seconds, the best time for $n = 1$. We found by running simulations for the different combina-

$T_f + T_0$	CPU Time (sec.)
0.01	11.56
0.015	16.73
0.02	21.29
0.025	26.99
0.03	33.13
0.035	38.36
0.04	44.48
0.045	50.83

Table 4.4.1: Fast Reversible Dimerization: CPU times for the nested SSA for $n = 1$, for a single sample path. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2.

$T_f + T_0$	CPU Time (sec.)
0.01	21.99
0.015	31.52
0.02	41.00
0.025	51.82

Table 4.4.2: Fast Reversible Dimerization: CPU times for the nested SSA for $n = 2$, for a single sample path. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2.

tions of T_f and T_0 that the fastest most consistent combination of T_f and T_0 for $N = 2$ was $T_f = 0.005$ and $T_0 = 0.005$. For the case of $n = 2$ we will also consider $T_f = 0.01$ and $T_0 = 0.005$. We will compare the results of these four cases to see which one is the most accurate.

To calculate the moments of each species we took an ensemble of 250 simulations. Each parameter set that we are taking for the nested SSA was able to produce a distribution similar to that of the SSA for both of the virtual fast processes. When we are computing the error of the mean we take the base ten logarithm of the ratio of the mean produced by the approximation method over the mean produced by the SSA simulations (The moments for the results of the SSA were computed from an ensemble of 250 simulations). If the value of the error is negative we know that the approximation method under-approximated the value of the SSA, where a positive value in the error implies that the value of the SSA was over-approximated. We take a similar approach to address the error of the standard deviation. In Figure 4.4.1 we see the error of the mean of X_1 for each of the nested SSA parameter sets; in Figure 4.4.2 we see the error of the standard deviation of X_1. We see from Figure 4.4.1 and Figure 4.4.2 that the two cases for $N = 2$ give poor approximations to the system for both the mean and standard deviation. The two cases for $n = 1$ approximate the standard deviation of X_1 well. However, in Figure 4.4.3 we see that the two cases for $n = 1$ give a poor approximation near the end of the simulation for the standard deviation of X_3, which is due to the approximation of the modified slow rates. The same thing can be said for the two cases for $n = 1$ for the mean of X_3 (not shown). The optimal parameters for the nested SSA from the parameters we have tested are $n = 1$, $T_f = 0.02$ and $T_0 = 0.02$, since they produced the most accurate data for the species overall and resulted in faster computations than $n = 1$ and $T_f = 0.025$. $n = 1$ and $T_f = 0.025$, produced results similar to $n = 1$ and $T_f = 0.02$ but was not chosen was due to the longer CPU time than $T_f = 0.02$.

4.5 Implicit Tau Method

In an implementation of the implicit tau leaping method, there are three different approaches we will consider: a random time jump that is on the slow scale; a fixed time jump; and a large time jump followed by a sequence of small time jumps. Recall the CPU time for the SSA was 135.62 seconds over 400 time units, for a single sample path. Using the time jump on the slow scale the implicit tau had a CPU time of approximately 8.31 seconds. Using a fixed time jump of 0.2250 the CPU time was around 9.92 seconds. We choose the time jump of 0.2250 because it is the average of the random time jumps on the slow scale over the entire simulation. Using a slightly higher fixed time jump of 0.5, the CPU time was approximately 4.53 seconds. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors)

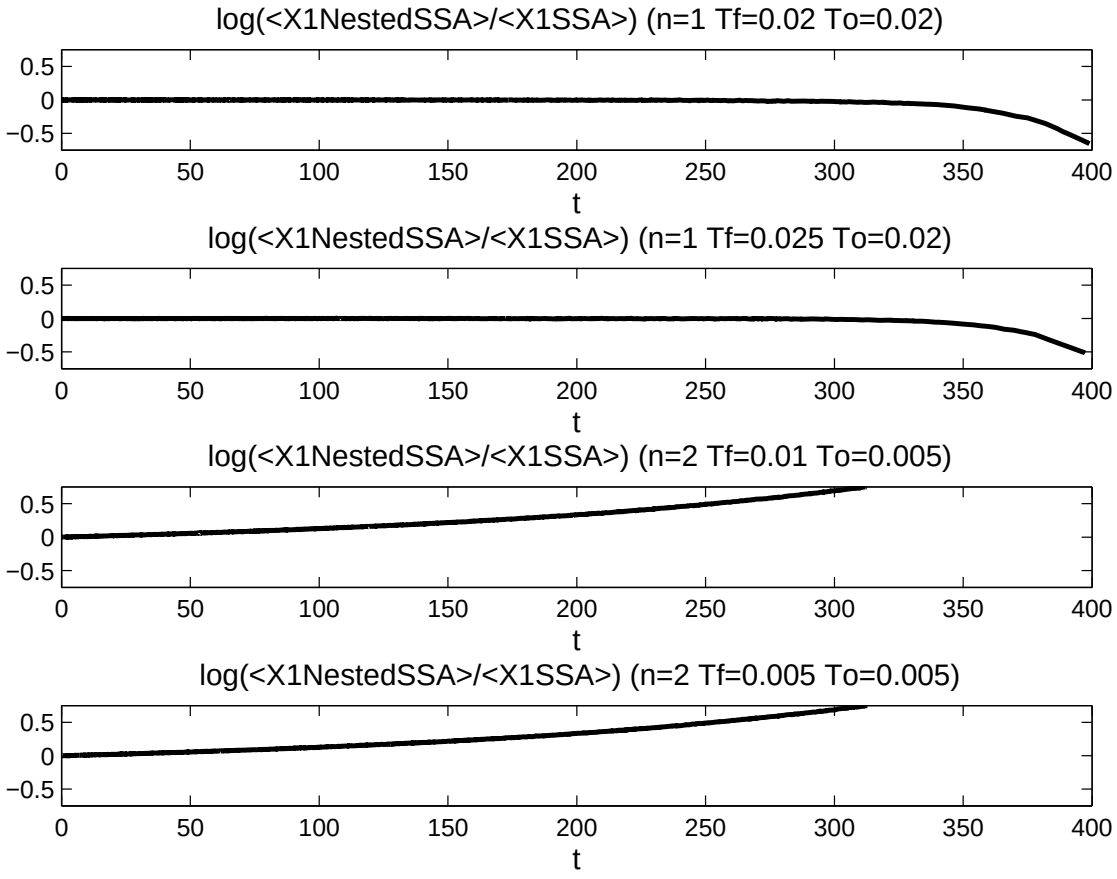

Figure 4.4.1: Fast Reversible Dimerization: The base ten logarithm of the approximation of X_1's mean by the nested SSA approximations over the approximation of X_1's mean by the SSA. Ensemble size of 250 simulations.

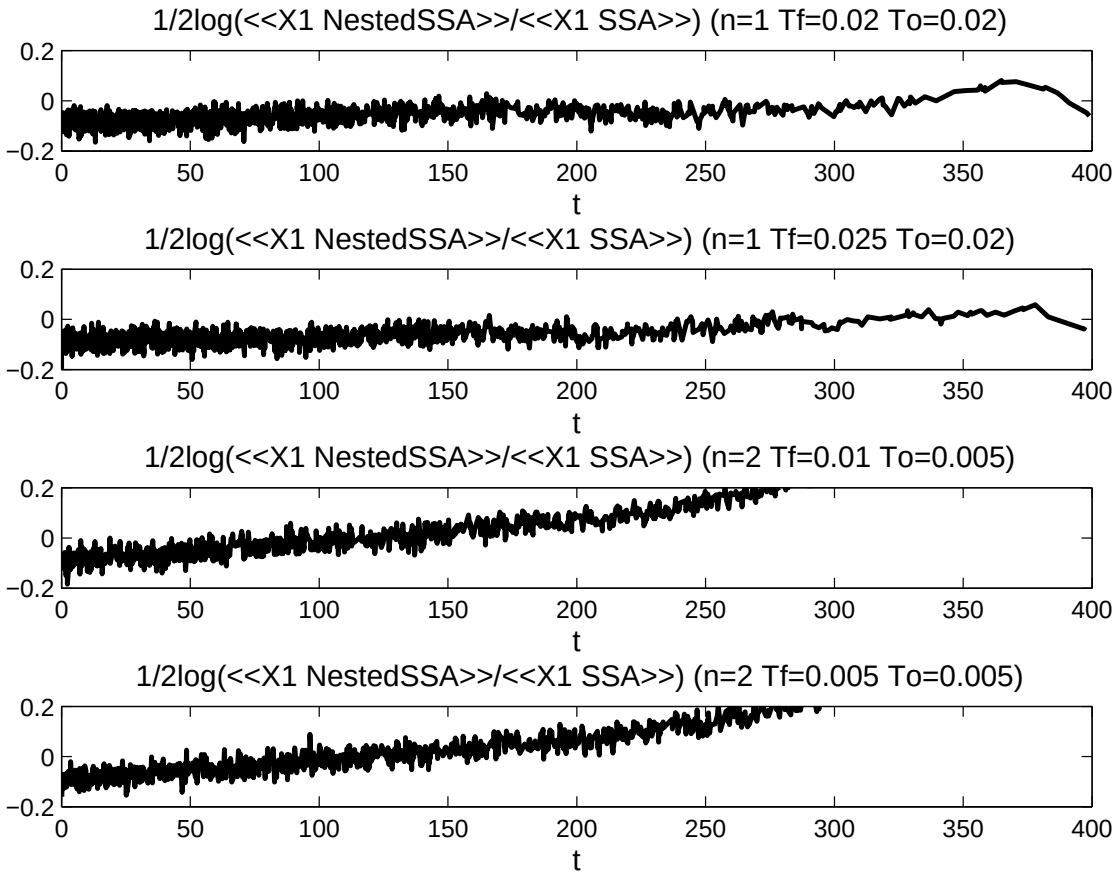

Figure 4.4.2: Fast Reversible Dimerization: The base ten logarithm of the approximation of X_1's standard deviation by the nested SSA approximations over the approximation of X_1's standard deviation by the SSA. Ensemble size of 250 simulations.

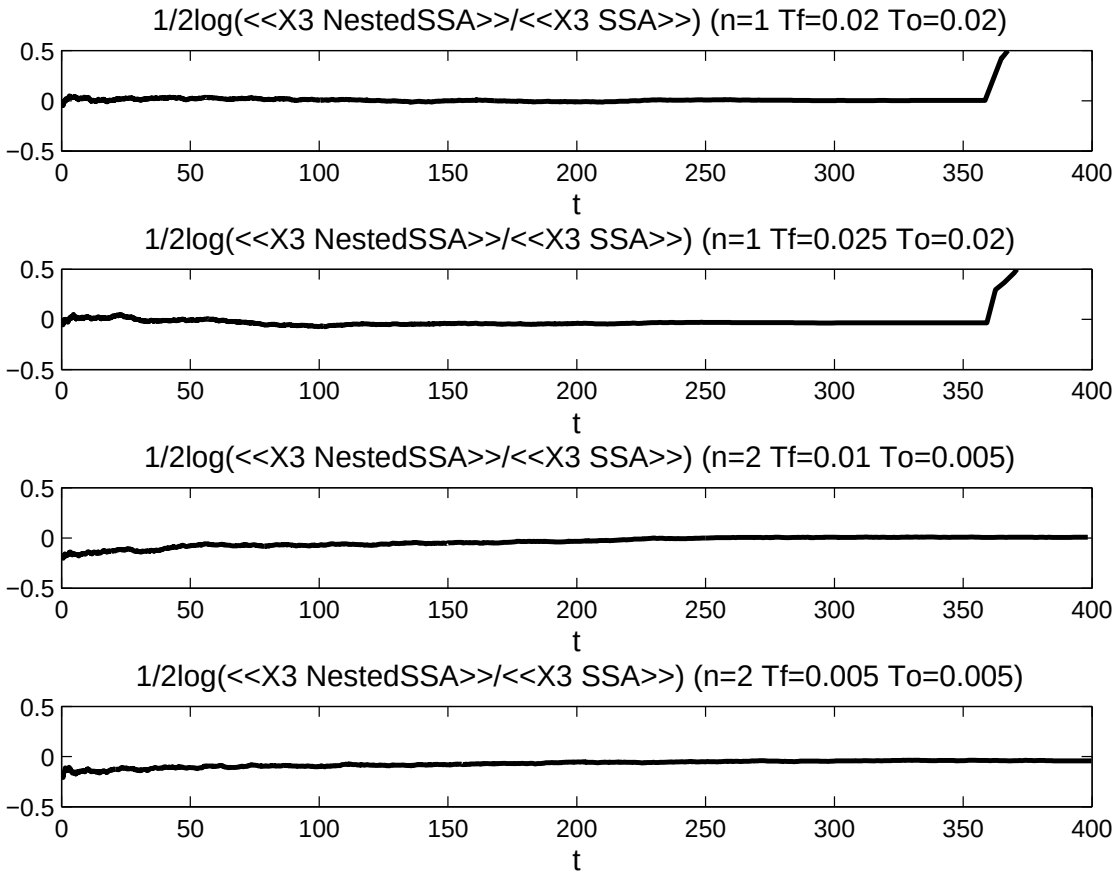

Figure 4.4.3: Fast Reversible Dimerization: The base ten logarithm of the approximation of X_3's standard deviation by the nested SSA approximations over the approximation of X_3's standard deviation by the SSA. Ensemble size of 250 simulations.

at 2.6GHz running OpenSuSE 10.2. The choice of a time jump of 0.5 produces a much coarser plot than the other approaches. Using the third approach we took one large jump and then a sequence of 4 small jumps; the CPU time was 49.77 seconds. For the small jumps we took them to be

$$\tau = \frac{-\ln(r_1)}{\sum_{i=1}^{4} a_i(\mathbf{x})}$$

and for the bigger jump we took them to be the on the slow time scale

$$\tau = \frac{-\ln(r_1)}{c_3 x_1 + c_4 x_2},$$

where r_1 is a unit interval uniform random variable. In Figure 4.5.1 we see the error of the mean for X_1; the sequence of small jumps does not approximate the mean very well near the end of the simulation. However, the sequence of small jumps is the best approach to use for approximating the mean of X_2, as seen in Figure 4.5.2. All of the approaches for the implicit tau were able to accurately approximate the mean of X_3 (not shown). In Figure 4.5.3 we see the standard deviation of X_2 is best approximated by the sequence of small jumps. The standard deviation of the other species was well approximated by all of the approaches to the implicit tau (not shown). The most accurate approach for the implicit tau is through the sequence of small jumps. However, the approach of using large fixed jumps gave just as good an approximation for X_3 (not shown) as the sequence of small jumps and slightly better approximations for X_1, Figure 4.5.1. The error of the large fixed time jumps for the approximation of X_2 was similar to the error of the approximation of the mean of X_1 that the sequence of small time jumps produced. The large fixed jump approach is much faster than the sequence of small jumps. Taking into account accuracy and CPU time the approach of the large fixed time jumps is the one we will use for the implicit tau leaping method.

4.6 Comparison of the Methods

The first thing that we will be looking at is the CPU time it takes to run each of the methods. We next will compare the results of the various algorithms against the SSA, using the SSA as the "gold standard". In each case we took an ensemble of 250 simulations. From Table 4.6.1, we see that the slow scale SSA and QSSA algorithm are the two fastest approximation methods.

How the QSSA algorithm, nested SSA and slow scale SSA approximate the fast species is important in how the three methods approximate the entire system. Overall, we found that the nested SSA, QSSA algorithm, slow scale SSA were able to produce a distribution for the virtual fast process similar to the results of the SSA. To find the delta peak of the QSSA algorithm we used (4.2.4) to approximate

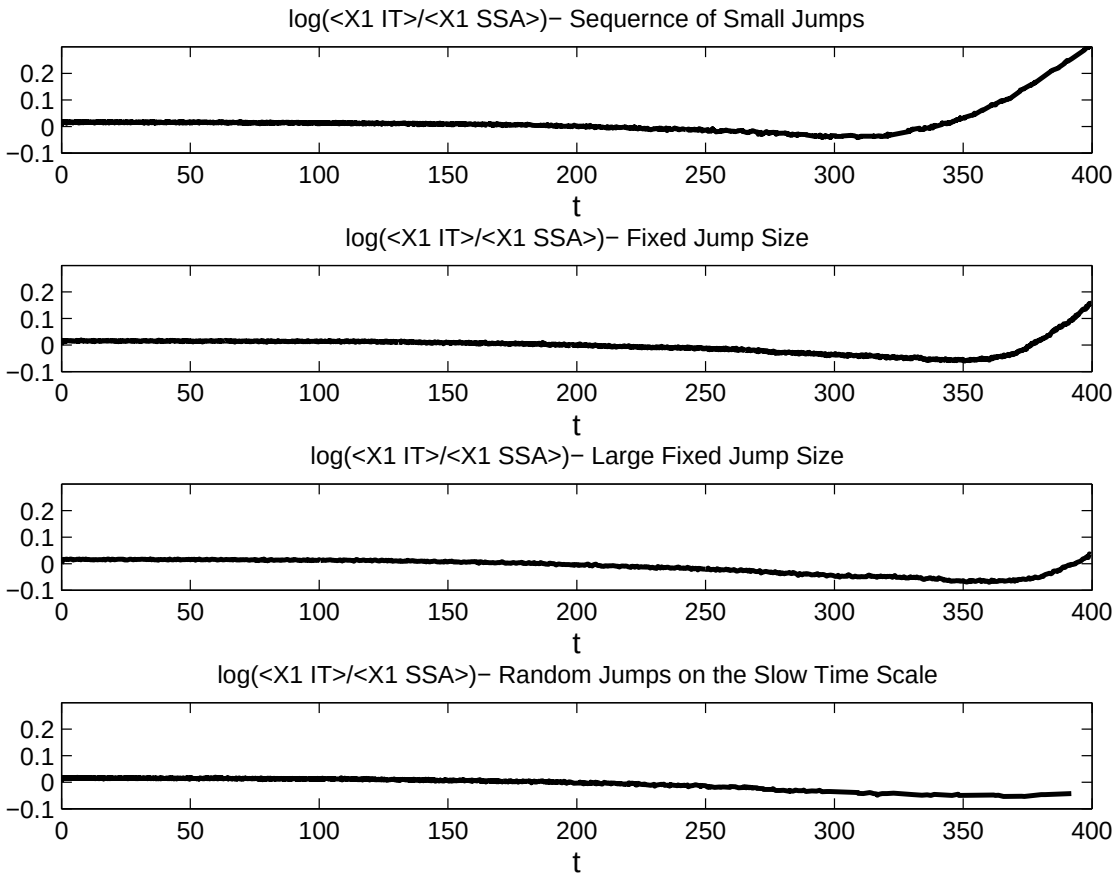

Figure 4.5.1: Fast Reversible Dimerization: The base ten logarithm of the approximation of X_1's mean by the implicit tau leaping approximations over the approximation of X_1's mean by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method)

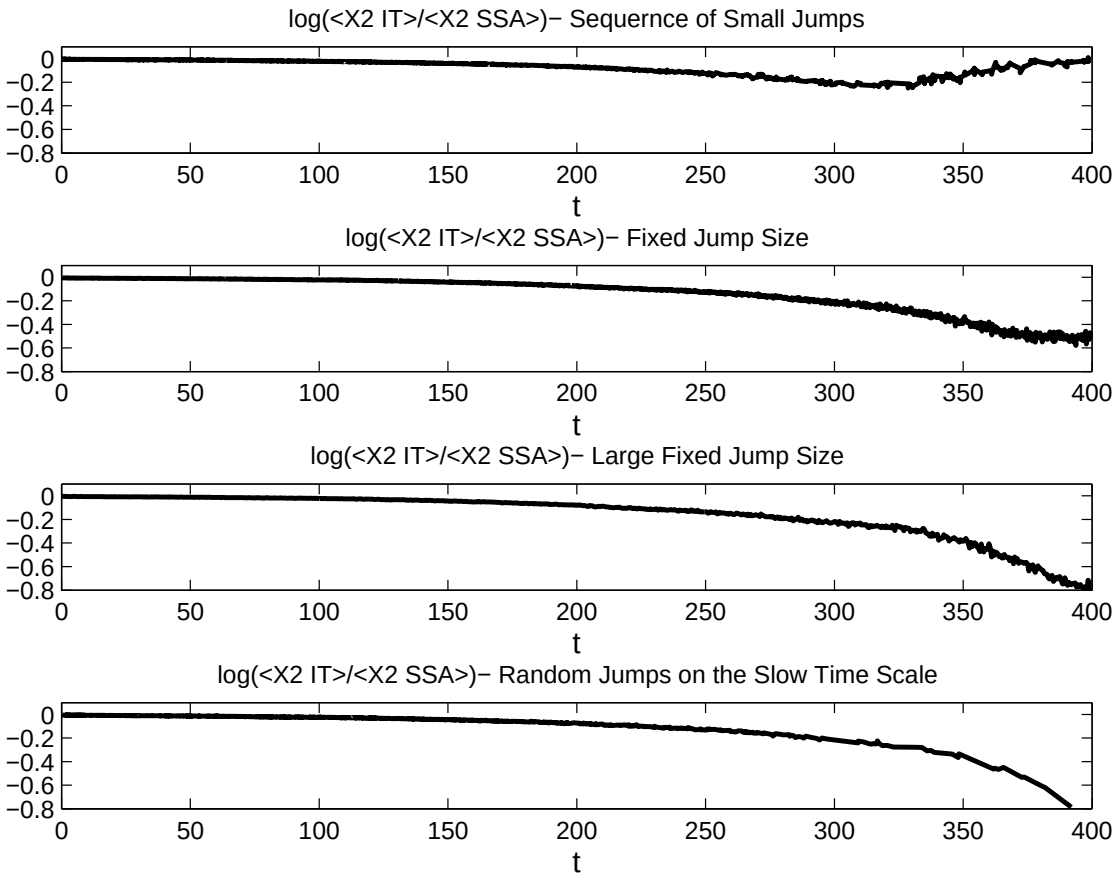

Figure 4.5.2: Fast Reversible Dimerization: The base ten logarithm of the approximation of X_2's mean by the implicit tau leaping approximations over the approximation of X_2's mean by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method)

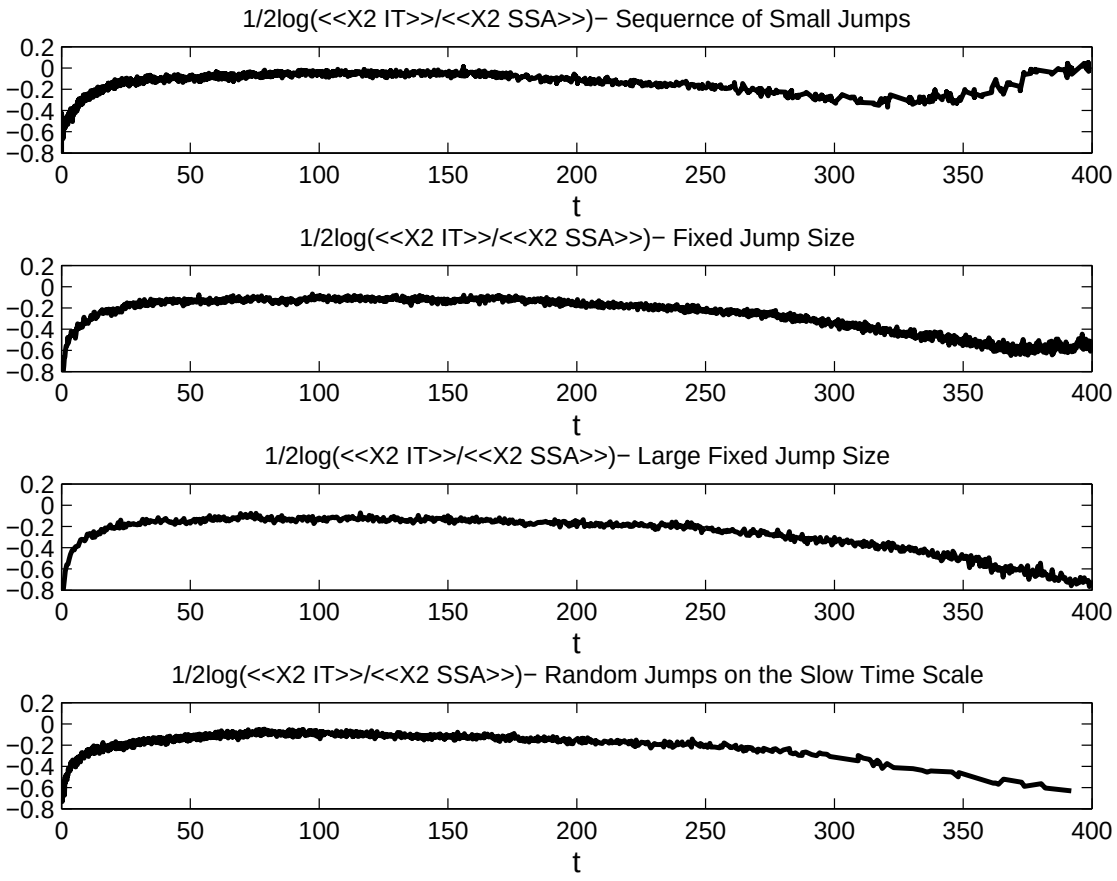

Figure 4.5.3: Fast Reversible Dimerization: The base ten logarithm of the approximation of X_2's standard deviation by the implicit tau leaping approximations over the approximation of X_2's standard deviation by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method)

Method	CPU Time (sec.)
SSA	135.62
QSSA algorithm	0.02
Nested SSA	44.48
Slow Scale SSA	0.02
Implicit Tau	4.53

Table 4.6.1: Fast Reversible Dimerization: CPU times for all the methods for 400 time units, for a single sample path. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2.

X_2 and (4.2.5) to approximate X_1. The QSSA algorithm approximations of X_1 and X_2 were located in the central part of the SSA distribution of the corresponding fast species. In implementing the slow scale SSA, the normal approximation using the mean and variance from section 4.3 was used as a distribution for X_2. Corresponding values of X_1 were derived from the conservation $X_1 = X_T - 2X_2$.

Considering the overall system behaviour, it seems that the approximation methods approximated the mean of each species very well, with the exception of the nested SSA and the species X_3. In Figure 4.6.1, we see that the nested SSA does an extremely poor job off approximating X_3 at the end of the simulation, similarly the nested SSA approximates the variance of the species X_3 poorly at the end (not shown). Looking closer at the means of each species, we notice that the nested SSA does a poor job in approximating the mean of X_1, Figure 4.6.2 , where both the QSSA algorithm and the slow scale SSA do a very good job. In Figure 4.6.3 we see that the nested SSA is the best approximation for the mean of X_2 as the slow scale SSA tails off at the end. In Figure 4.6.4 we see that the nested SSA and the slow scale SSA approximate the standard deviation of X_1 very well. The results of the implicit tau and the QSSA algorithm do not even resemble the standard deviation of X_1 produced by the SSA. The QSSA algorithm and implicit tau method also under-approximate the standard deviation of X_2 (not shown). However, this time the QSSA algorithm and implicit tau method were able to reproduce a standard deviation of X_2 similar to the SSA. Figure 4.6.5 shows a histogram of X_2 at 300 time units. We see that every method besides the implicit tau method was able to represent the distribution of the species X_2. The same results were found for the other species histograms at 300 time units (not shown). (All the values of the species are not in integer bins because we had to interpolate the data to 300 time units.)

The QSSA algorithm provides a poor approximation of the standard deviation of the fast species. The nested SSA was much slower than the slow scale SSA and poorly approximated X_3 at the end of the simulation. Because of how the modified

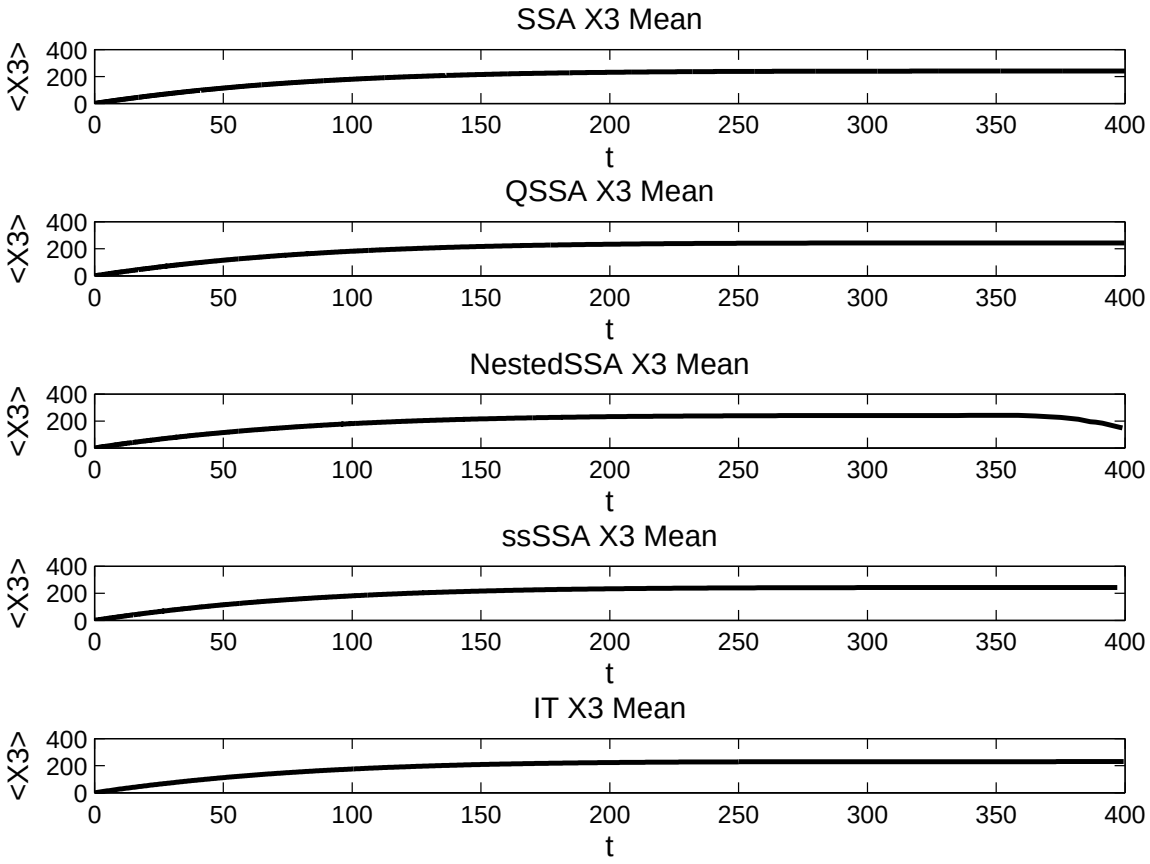

Figure 4.6.1: Fast Reversible Dimerization: The approximation of X_3's mean made by all the methods. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

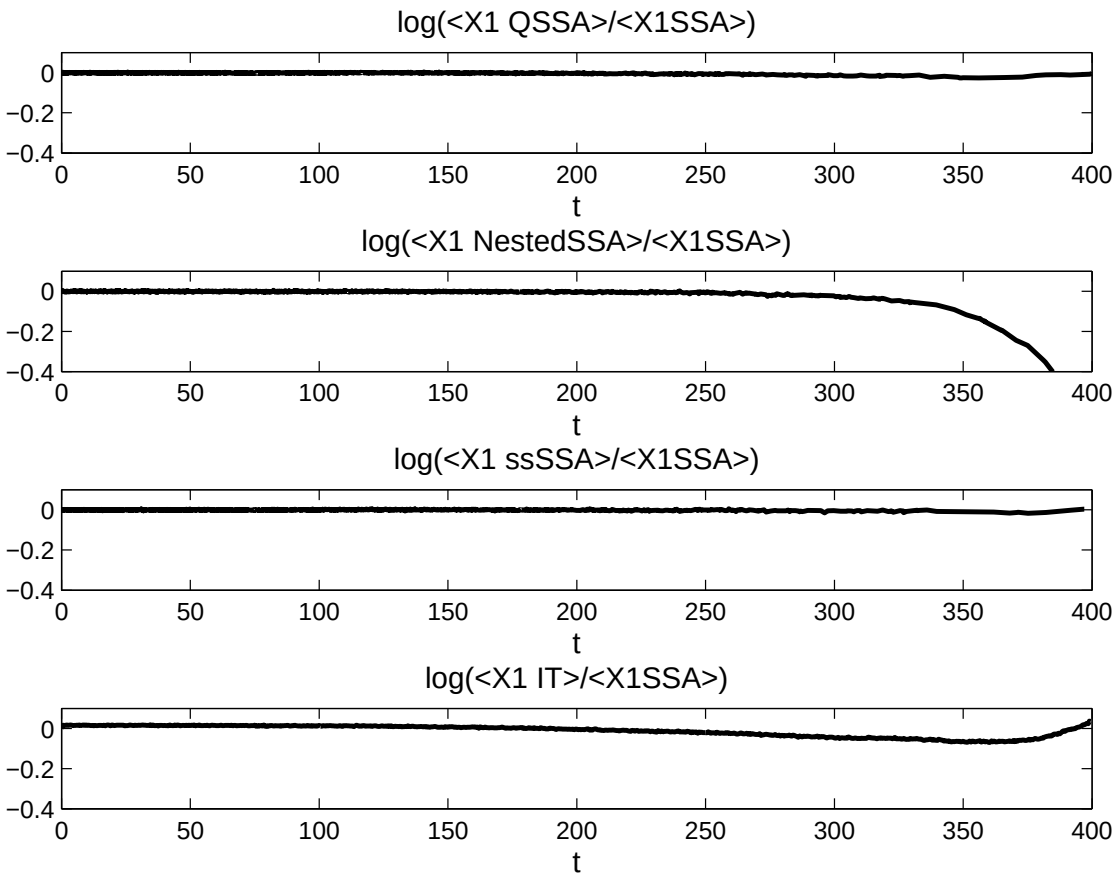

Figure 4.6.2: Fast Reversible Dimerization: The base ten logarithm of the approximation of X_1's mean by the approximation methods over the approximation of X_1's mean by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

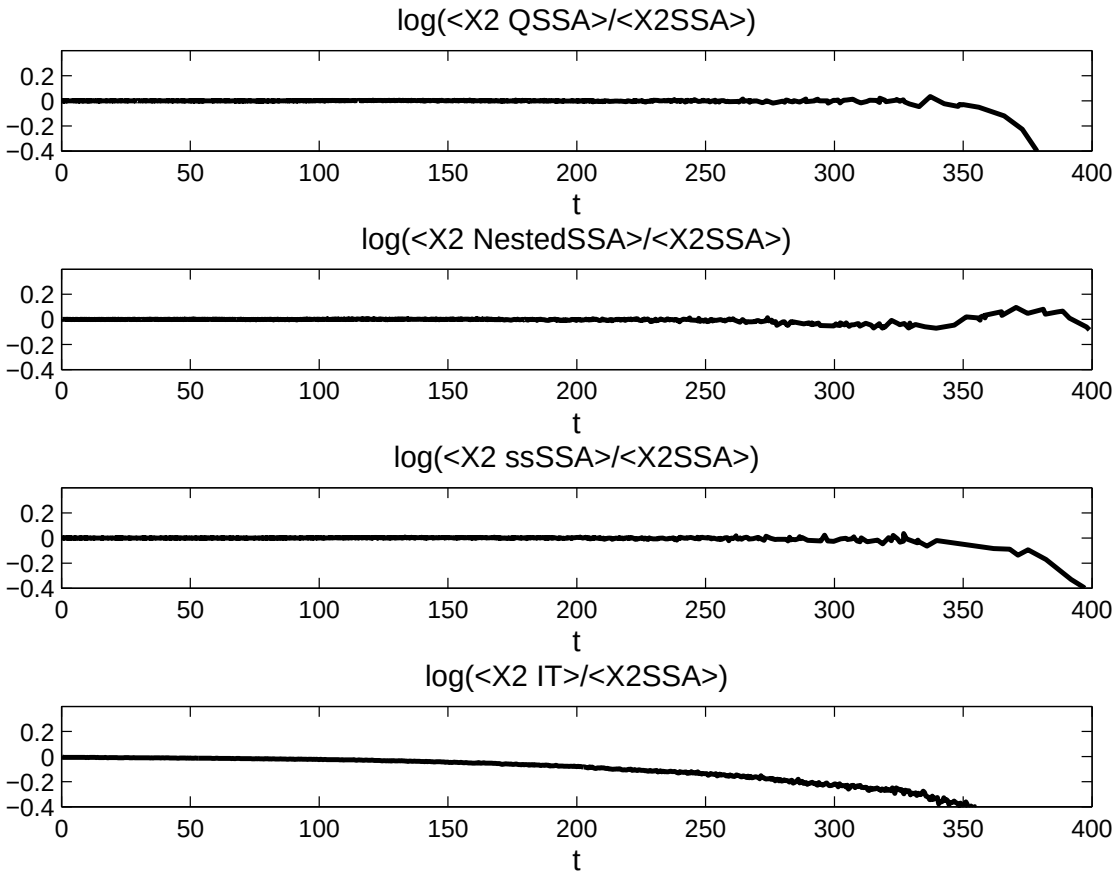

Figure 4.6.3: Fast Reversible Dimerization: The base ten logarithm of the approximation of X_2's mean by the approximation methods over the approximation of X_2's mean by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

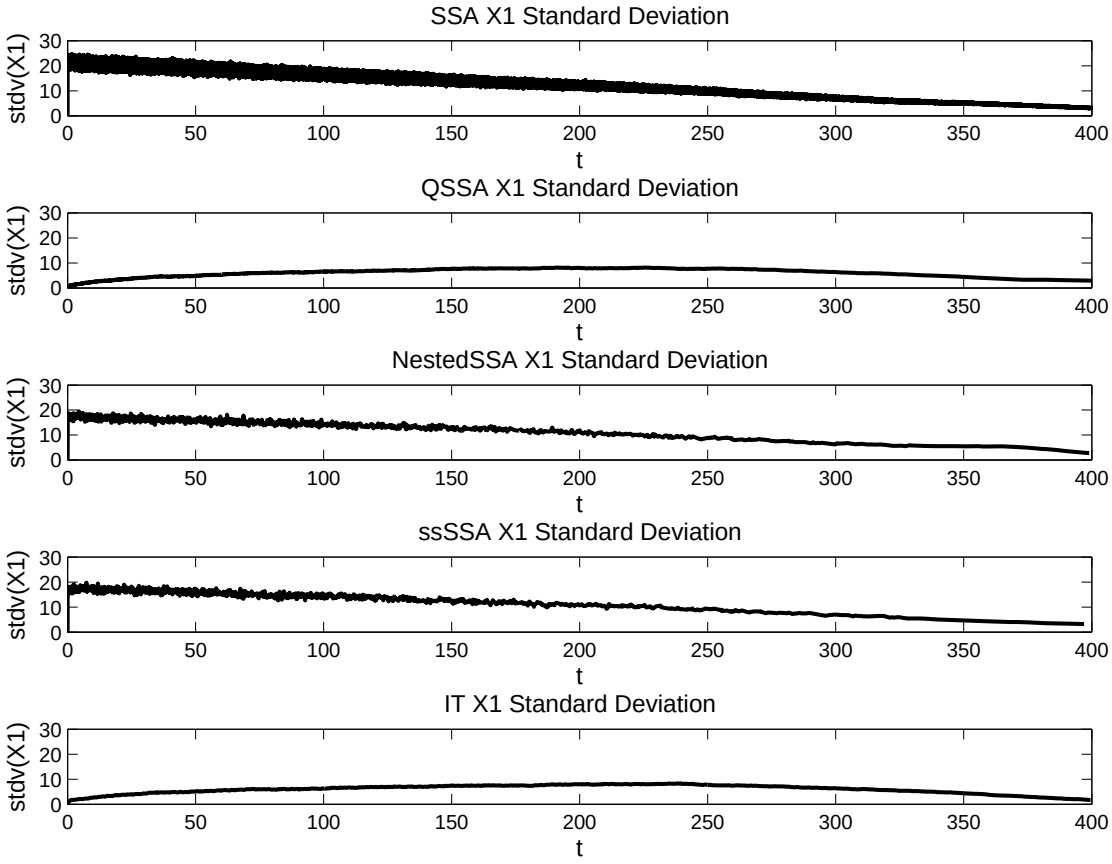

Figure 4.6.4: Fast Reversible Dimerization: The approximation of X_1's standard deviation made by all the methods. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

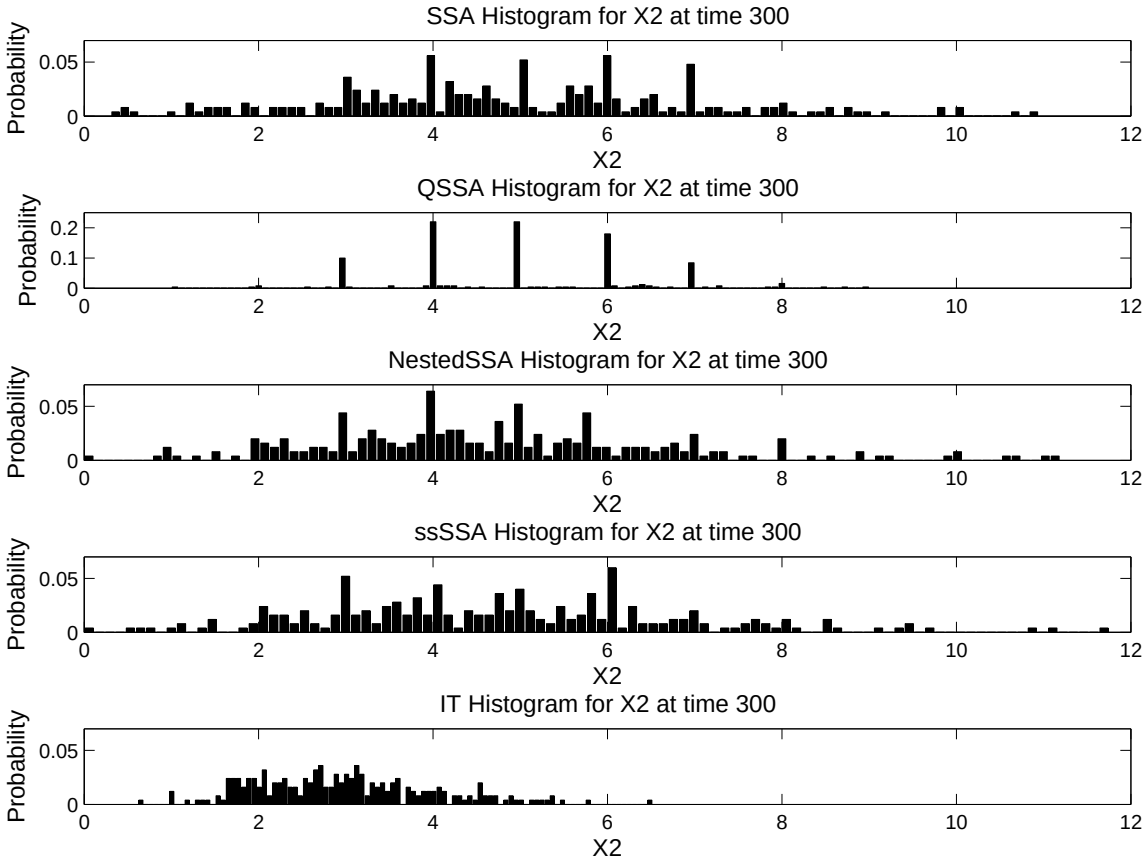

Figure 4.6.5: Fast Reversible Dimerization: The histogram of X_2 for all the methods at 300 time units. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

slow propensities were calculated: near the end of the simulation the values for X_1 and X_2 decrease, causing the tau, the time to the next fast reaction in the inner SSA to become much larger than previously in the simulation. This larger tau causes the approximation of the Reimann sum to become worse. The approximation of the integral could be improved by the use of a better quadrature method. In addition the modified slow propensities become inaccurate at the end of the inner SSA since the fast species are sometimes have a zero concentration, but the modified slow rates may not be zero. This causes the nested SSA to produce negative concentrations for a species and then we must set the species concentration to zero. From these results we can conclude that the slow scale SSA is the best approximation method to use for this example. The slow scale SSA was able to capture the characteristics of the SSA for the mean, standard deviation, majority of the skewness (not shown) and the distribution of each species.

Chapter 5

Example 2: Network of Isomerizations

This example from [18] involves four species in a network of isomerizations involving six reactions:

$$\begin{aligned} S_1 &\underset{c_2}{\overset{c_1}{\rightleftharpoons}} S_2 \\ S_2 &\underset{c_4}{\overset{c_3}{\rightleftharpoons}} S_3 \\ S_3 &\underset{c_6}{\overset{c_5}{\rightleftharpoons}} S_4. \end{aligned}$$

The propensity functions for this system are

$$\mathbf{a}(\mathbf{x}) = (c_1x_1, c_2x_2, c_3x_2, c_4, x_3, c_5, x_3, c_6x_4).$$

The stoichiometry vectors are

$$\begin{aligned} &\mathbf{v}_1 = (-1,1,0,0), \quad \mathbf{v}_2 = (1,-1,0,0), \quad \mathbf{v}_3 = (0,-1,1,0), \\ &\mathbf{v}_4 = (0,1,-1,0), \quad \mathbf{v}_5 = (0,0,-1,1), \quad \mathbf{v}_6 = (0,0,1,-1). \end{aligned}$$

Take

$$\begin{aligned} S_1 &\underset{c_2}{\overset{c_1}{\rightleftharpoons}} S_2 \\ S_3 &\underset{c_6}{\overset{c_5}{\rightleftharpoons}} S_4 \end{aligned}$$

as the fast reactions. As a result there are no slow species in the system. The fast stoichiometry vectors are $\mathbf{v}_1^f = (-1,1,0,0)$, $\mathbf{v}_2^f = (1,-1,0,0)$, $\mathbf{v}_3^f = (0,0,-1,1)$ and $\mathbf{v}_4^f = (0,0,1,-1)$. There are two independent fast systems:

$$S_1 \underset{c_2}{\overset{c_1}{\rightleftharpoons}} S_2$$

and

$$S_3 \underset{c_6}{\overset{c_5}{\rightleftharpoons}} S_4.$$

This system differs considerably from the previous example; there are no slow species, and there are two independent fast subsystems coupled through slow reactions, complicating the approximation of the behaviour of the virtual fast process. The behaviour of this system is that X_1 and X_2 both decay over time as the species X_3 and X_4 both grow over time.

5.1 Theoretical Analysis of the System

The chemical master equation for this example is

$$\begin{aligned}\frac{\partial P(\mathbf{x},t|\mathbf{x}_0,t_0)}{\partial t} &= c_1x_1P(\mathbf{x}-\mathbf{v}_1,t|\mathbf{x}_0,t_0)-c_1x_1P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_2x_2P(\mathbf{x}-\mathbf{v}_2,t|\mathbf{x}_0,t_0)-c_2x_2P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_3x_2P(\mathbf{x}-\mathbf{v}_3,t|\mathbf{x}_0,t_0)-c_3x_2P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_4x_3P(\mathbf{x}-\mathbf{v}_4,t|\mathbf{x}_0,t_0)-c_4x_3P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_5x_3P(\mathbf{x}-\mathbf{v}_5,t|\mathbf{x}_0,t_0)-c_5x_3P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_6x_4P(\mathbf{x}-\mathbf{v}_6,t|\mathbf{x}_0,t_0)-c_6x_4P(\mathbf{x},t|\mathbf{x}_0,t_0).\end{aligned} \tag{5.1.1}$$

The chemical master equation for the virtual fast process is

$$\begin{aligned}\frac{\partial P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)}{\partial t} &= c_1x_1P(\mathbf{x}^f-\mathbf{v}_1^f,t|\mathbf{x}_0,t_0)-c_1x_1P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)\\ &+ c_2x_2P(\mathbf{x}^f-\mathbf{v}_2^f,t|\mathbf{x}_0,t_0)-c_2x_2P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)\\ &+ c_5x_3P(\mathbf{x}^f-\mathbf{v}_5^f,t|\mathbf{x}_0,t_0)-c_5x_3P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)\\ &+ c_6x_4P(\mathbf{x}^f-\mathbf{v}_6^f,t|\mathbf{x}_0,t_0)-c_6x_4P(\mathbf{x}^f,t|\mathbf{x}_0,t_0).\end{aligned} \tag{5.1.2}$$

Since there are two independent fast systems, we write (5.1.2) as two separate systems.

$$\begin{aligned}\frac{\partial P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)}{\partial t} &= c_1x_1P(\mathbf{x}^f-\mathbf{v}_1^f,t|\mathbf{x}_0,t_0)-c_1x_1P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)\\ &+ c_2x_2P(\mathbf{x}^f-\mathbf{v}_2^f,t|\mathbf{x}_0,t_0)-c_2x_2P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)\end{aligned} \tag{5.1.3}$$

$$\begin{aligned}\frac{\partial P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)}{\partial t} &= c_5x_3P(\mathbf{x}^f-\mathbf{v}_5^f,t|\mathbf{x}_0,t_0)-c_5x_3P(\mathbf{x}^f,t|\mathbf{x}_0,t_0)\\ &+c_6x_4P(\mathbf{x}^f-\mathbf{v}_6^f,t|\mathbf{x}_0,t_0)-c_6x_4P(\mathbf{x}^f,t|\mathbf{x}_0,t_0).\end{aligned} \tag{5.1.4}$$

In (5.1.3), summing over x_3 and x_4 we have

$$\begin{aligned}\frac{\partial P(x_1^f, x_2^f, t|\mathbf{x}_0, t_0)}{\partial t} &= c_1 x_1 P(x_1^f + 1, x_2^f - 1, t|\mathbf{x}_0, t_0) \\ &- c_1 x_1 P(x_1^f, x_2^f, t|\mathbf{x}_0, t_0) + c_2 x_2 P(x_1^f - 1, x_2^f + 1, t|\mathbf{x}_0, t_0) \\ &- c_2 x_2 P(x_1^f, x_2^f, t|\mathbf{x}_0, t_0).\end{aligned} \tag{5.1.5}$$

There is also a conservation in the system such that $X_{T_1} = X_1(t) + X_2(t)$, where $X_{T_1} = x_{T_1}$ is constant on the fast time scale. Using the conservation, (5.1.3) is rewritten as

$$\begin{aligned}\frac{\partial Q(x_1^f, t|\mathbf{x}_0, t_0)}{\partial t} &= c_1 x_1 Q(x_1^f + 1, t|\mathbf{x}_0, t_0) - c_1 x_1 Q(x_1^f, t|\mathbf{x}_0, t_0) \\ &+ c_2(x_{T_1} - x_1) Q(x_1^f - 1, t|\mathbf{x}_0, t_0) - c_2(x_{T_1} - x_1) Q(x_1^f, t|\mathbf{x}_0, t_0),\end{aligned} \tag{5.1.6}$$

where $Q(x_1^f, t|\mathbf{x}_0, t_0) = P(x_1^f, x_{T_1} - x_1, t|\mathbf{x}_0, t_0)$. Equation (5.1.6) describes a birth and death process for X_1 with birth rate

$$b(x_1) = c_2(x_{T_1} - x_1)$$

and death rate

$$d(x_1) = c_1 x_1.$$

The time evolution of the first moment of $X_1(t)$ is

$$\frac{d\langle X_1(t)\rangle}{dt} = c_2 X_{T_1} - (c_1 + c_2)\langle X_1(t)\rangle. \tag{5.1.7}$$

The time evolution of the second moment is

$$\frac{d\langle X_1^2(t)\rangle}{dt} = \frac{d\langle X_1(t)\rangle}{dt} + 2(c_2 X_{T_1} + c_1)\langle X_1(t)\rangle - 2(c_1 + c_2)\langle X_1^2(t)\rangle. \tag{5.1.8}$$

To find the steady state solution of (5.1.7) and (5.1.8) we solve

$$c_2 X_{T_1} - (c_1 + c_2)\langle X_1(t)\rangle = 0 \tag{5.1.9}$$

$$(c_2 X_{T_1} + c_1)\langle X_1(t)\rangle - (c_1 + c_2)\langle X_1^2(t)\rangle = 0 \tag{5.1.10}$$

The solution for (5.1.9) is

$$\langle X_1(t)\rangle = \frac{c_2 X_{T_1}}{c_1 + c_2}. \tag{5.1.11}$$

Using (5.1.11) in (5.1.10), the steady state solution for the second moment is

$$\langle X_1^2(t)\rangle = \frac{c_2^2 X_{T_1}^2 + c_1 c_2 X_{T_1}}{(c_1 + c_2)^2}.$$

The variance for $X_1(t)$ is

$$\langle\langle X_1(t)\rangle\rangle = \langle X_1^2(t)\rangle - \langle X_1(t)\rangle^2 = \frac{c_1 c_2 X_{T_1}}{(c_1+c_2)^2}. \tag{5.1.12}$$

Using the conservation $X_{T_1} = X_1(t) + X_2(t)$, we have that

$$\langle X_2(t)\rangle = X_{T_1} - \langle X_1(t)\rangle = \frac{c_1 X_{T_1}}{c_1+c_2} \tag{5.1.13}$$

$$\langle\langle X_2(t)\rangle\rangle = \langle\langle X_{T_1} - X_1(t)\rangle\rangle = \frac{c_1 c_2 X_{T_1}}{(c_1+c_2)^2}. \tag{5.1.14}$$

Similarly, using the conservation relation in the other fast system $X_{T_2} = X_3(t) + X_4(t)$, where $X_{T_2} = x_{T_2}$ is constant, we have that

$$\langle X_3(t)\rangle = \frac{c_5 X_{T_2}}{c_5+c_6}, \tag{5.1.15}$$

$$\langle\langle X_3(t)\rangle\rangle = \frac{c_5 c_6 X_{T_2}}{(c_5+c_6)^2}, \tag{5.1.16}$$

$$\langle X_4(t)\rangle = \frac{c_6 X_{T_2}}{c_5+c_6}, \tag{5.1.17}$$

$$\langle\langle X_4(t)\rangle\rangle = \frac{c_5 c_6 X_{T_2}}{(c_5+c_6)^2}. \tag{5.1.18}$$

One could use these moment equations to produce a normal approximation to the virtual fast process of these four species in the slow scale SSA. However, later on in this section we will see that there is an alternative distribution for the virtual fast process.

Now we will show the chemical Langevin equation for this system:

$$\begin{aligned}\frac{dX_1(t)}{dt} &= -c_1X_1(t) + c_2X_2(t) \\ &- \sqrt{c_1X_1(t)}\Gamma_1(t) + \sqrt{c_2X_2(t)}\Gamma_2(t),\end{aligned} \tag{5.1.19}$$

$$\begin{aligned}\frac{dX_2(t)}{dt} &= c_1X_1(t) - c_2X_2(t) - c_3X_2(t) + c_4X_3(t) \\ &+ \sqrt{c_1X_1(t)}\Gamma_1(t) - \sqrt{c_2X_2(t)}\Gamma_2(t) - \sqrt{c_3X_2(t)}\Gamma_3(t) + \sqrt{c_4X_3(t)}\Gamma_4(t),\end{aligned} \tag{5.1.20}$$

$$\begin{aligned}\frac{dX_3(t)}{dt} &= c_3X_2(t) - c_4X_3(t) - c_5X_3(t) + c_6X_4(t) \\ &+ \sqrt{c_3X_2(t)}\Gamma_3(t) - \sqrt{c_4X_3(t)}\Gamma_4(t) - \sqrt{c_5X_3(t)}\Gamma_5(t) + \sqrt{c_6X_4(t)}\Gamma_6(t),\end{aligned} \tag{5.1.21}$$

$$\begin{aligned}\frac{dX_4(t)}{dt} &= c_5X_3(t) - c_6X_4(t) \\ &+ \sqrt{c_5X_3(t)}\Gamma_5(t) - \sqrt{c_6X_4}\Gamma_6(t).\end{aligned} \tag{5.1.22}$$

These equations will be used by the implicit tau leaping method to approximate the behaviour of the chemical system. With the knowledge of the theoretical representation of this system, next we will consider the details of implementing each algorithm.

5.2 QSSA algorithm

The deterministic equations for the system are

$$\begin{aligned}
\frac{dX_1}{dt} &= -c_1X_1 + c_2X_2 \\
\frac{dX_2}{dt} &= c_1X_1 - c_2X_2 - c_3X_2 + c_4X_3 \\
\frac{dX_3}{dt} &= c_3X_2 - c_4X_3 - c_5X_3 + c_6X_4 \\
\frac{dX_4}{dt} &= c_5X_3 - c_6X_4
\end{aligned}$$

Reactions 1, 2, 5 and 6 are fast. This means that there are two independent fast systems with the conservations $X_{T_1} = X_1 + X_2$ and $X_{T_2} = X_3 + X_4$. The equations to solve for the fast systems are

$$c_1X_1 = c_2X_2 = c_2(X_{T_1} - X_1)$$

$$c_5X_3 = c_6X_4 = c_6(X_{T_2} - X_3)$$

The steady states for the fast species are

$$X_1 = \frac{c_2X_{T_1}}{c_1 + c_2} \tag{5.2.1}$$

$$X_2 = \frac{c_1X_{T_1}}{c_1 + c_2} \tag{5.2.2}$$

$$X_3 = \frac{c_6X_{T_2}}{c_5 + c_6} \tag{5.2.3}$$

$$X_4 = \frac{c_5X_{T_2}}{c_5 + c_6} \tag{5.2.4}$$

With the approximation of the virtual fast processes the fast time scale is removed and the following slow reactions remain to be simulated

$$S_2 \underset{c_4}{\overset{c_3}{\rightleftharpoons}} S_3$$

5.3 Slow Scale SSA

The fast system is made up of two independent isomerizations. Consider the isomerization of X_1 and X_2; the other fast system is similar. Recall, the conservation $X_{T_1} = X_1(t) + X_2(t)$, where X_{T_1} is constant in the fast system. The isomerization can be written in terms of a birth and death process in $X_1(t)$, where the birth rate is

$$b(x_1) = c_2(x_{T_1} - x_1)$$

and the death rate is

$$d(x_1) = c_1 x_1.$$

Using the recursion formula from [17]

$$P(x_1, \infty) = \begin{cases} \frac{d(x_1+1)P(x_1+1,\infty)}{b(x_1)} & x_1 = -1, 0 \\ \frac{b(x_1-1)P(x_1-1,\infty)}{d(x_1)} & x_1 = 1, 2, \cdots, x_{T_1} \end{cases}$$

the probability distribution of $X_1(t)$ under the fast reactions is

$$P(x_1', \infty | x_1, x_2) = \begin{pmatrix} x_{T_1} \\ x_1' \end{pmatrix} (\frac{c_2}{c_1 + c_2})^{x_1'} (\frac{c_1}{c_1 + c_2})^{x_{T_1} - x_1'}, \tag{5.3.1}$$

where $x_1' = 0, 1, 2, \cdots, x_{T_1}$. The probability distribution in (5.3.1) is a binomial distribution.

5.4 Nested SSA

In [33], the parameter values used for the nested SSA were $T_f = 1.1 \times 10^{-5}$ with $T_0 = 0$ and $N = 1$. Since the authors of [33] were comparing the slow scale SSA to the nested SSA, we will use these parameters when comparing the nested SSA to the other approximation methods in this paper.

5.5 Implicit Tau Leaping

The reaction rates are

$$c_1 = 3 \times 10^4,\ c_2 = 6 \times 10^4,\ c_3 = 1,$$

$$c_4 = 1,\ c_5 = 6 \times 10^4,\ c_6 = 3 \times 10^4.$$

The initial value for each species was

$$\mathbf{X}(0) = (1200, 600, 0, 0).$$

Method	CPU Time (sec.)
SSA	2641.9
QSSA algorithm	0.04
Nested SSA	12.33
Slow Scale SSA	0.61
Implicit Tau	3.27

Table 5.6.1: Network of Isomerizations: CPU times for all the methods for 3 time units, for a single sample path. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2.

Each simulation was run for 3 time units. To calculate the moments we used an ensemble of 250 simulations. We consider four approaches to implementation of the implicit tau leaping algorithm: a fixed time step of 0.0016; a larger fixed time step of 0.008; jumps on the slow time scale; and a large jump on the slow time scale followed by a sequence of 5 small jumps on the fast time scale. The CPU times for the fixed time step of 0.0016 is approximately 11.68 seconds, the larger fixed time step of 0.008 is 2.55 seconds, the sequence of small jumps had a CPU time of 48.57 seconds and on the slow time scale it was 11.34 seconds. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2 Figure 5.5.1 shows the error of the mean of X_3, indicating there is not much difference between these different approaches in approximating the mean of X_3. The same thing can be said for all the other species in the system (not shown). Figure 5.5.2 shows the error of the approximation of the standard deviation for X_2, indicating that there is no difference in the standard deviations generated by these different approaches. Similar results were found for all other species in the system (not shown). Since all of the methods are extremely close to one another in approximations, we will use the fastest approach- the large fixed time jump of 0.008- in the comparison to follow.

5.6 Comparison of the Methods

To compute the moments and the histograms of the species at the end of the simulation we used an ensemble of 250 simulations. The CPU times for each method for this example are given in Table 5.6.1, where we see that the QSSA algorithm is the fastest approximation method. The approach the nested SSA and slow scale SSA take in dealing with the stiffness is important to how well they approximate the behaviour of the system. For the histograms of the virtual fast processes, we used an ensemble of 1000 simulations. To begin the comparison of the results,

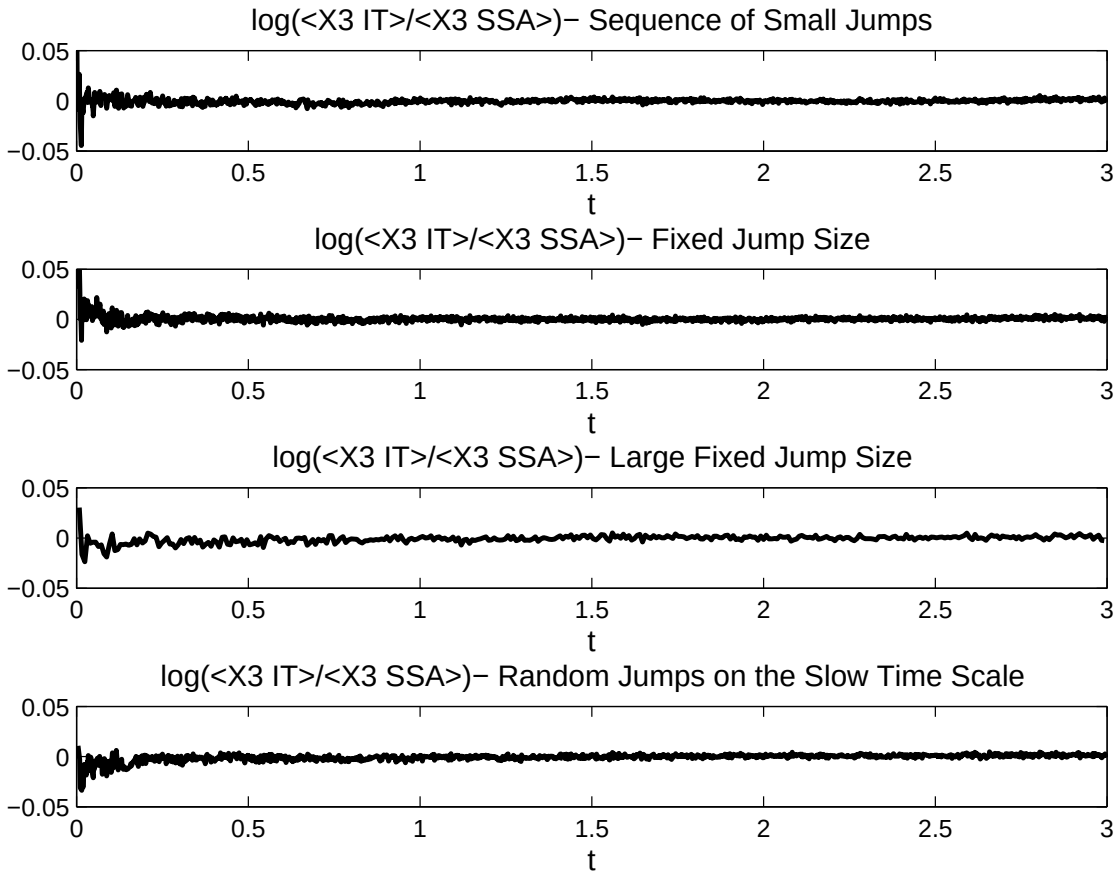

Figure 5.5.1: Network of Isomerizations: The base ten logarithm of the approximation of X_3's mean by the implicit tau leaping approximations over the approximation of X_3's mean by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method)

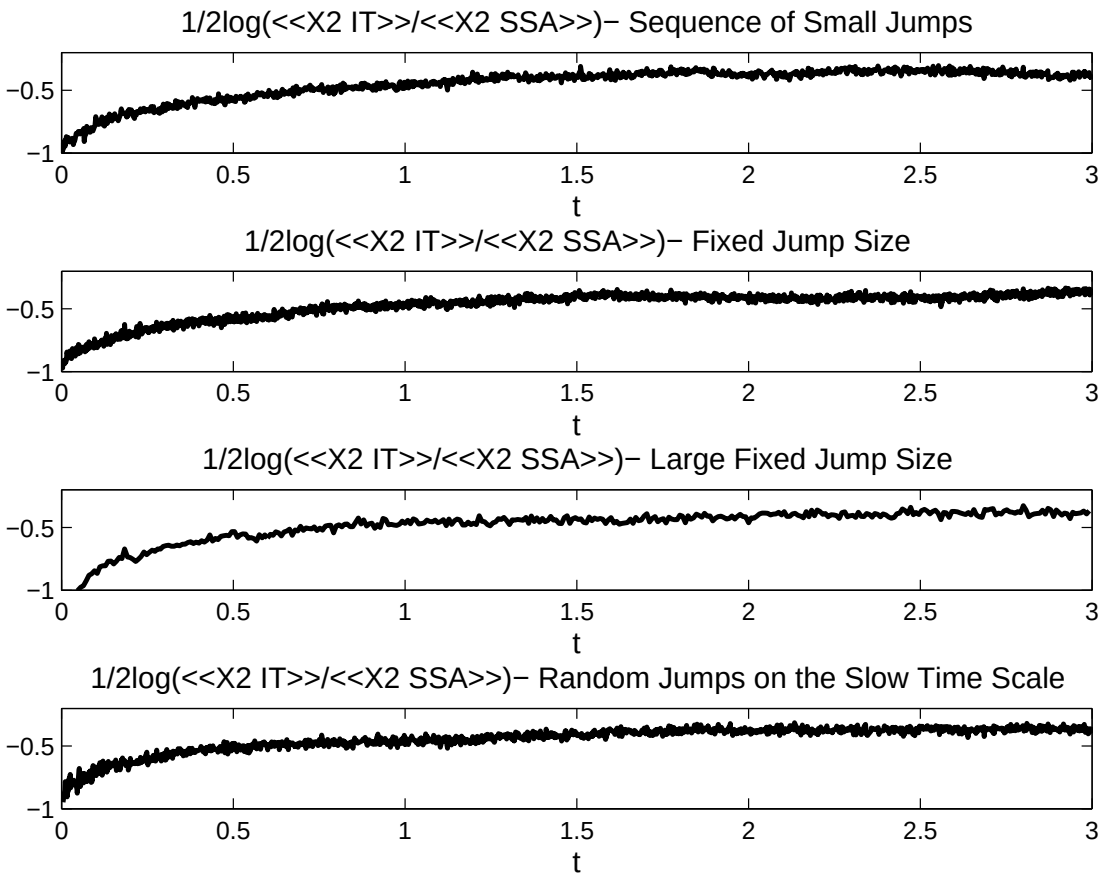

Figure 5.5.2: Network of Isomerizations: The base ten logarithm of the approximation of X_2's standard deviation by the implicit tau leaping approximations over the approximation of X_2's standard deviation by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method)

we will first focus on how the SSA, nested SSA, slow scale SSA and QSSA algorithm distribute the virtual fast processes. We will compare those results with the theoretically-determined normal approximation of the fast species distribution using the mean and variance equations from (5.1.11) to (5.1.18) for each individual species. Since there is a conservation in each subsystem, we will address only X_2 and X_4. Considering the state $X_1 = 950$, $X_2 = 425$, $X_3 = 125$ and $X_4 = 250$, the QSSA algorithm approximates the virtual fast process by the delta distribution with

$$X_2 = 458.33 \approx 458 \quad \text{and} \quad X_4 = 250$$

using (5.2.2) and (5.2.4) respectively. Figures 5.6.1 shows the distribution of the virtual fast process for $X_{T_1} = 1375$, indicating that the QSSA algorithm's approximation of X_2 is very good since it lies in the center of the SSA's distribution for X_2. Similar results for the QSSA algorithm were found for the distribution of X_4 (not shown). We see in Figure 5.6.1 that the nested SSA's distribution does not match that of the SSA's. The nested SSA's distribution for X_2 is not accurate since the inner SSA was not run long enough for X_2 to reach its steady state. We can see in Figure 5.6.2 that when we run the inner SSA for a longer period of time we have the nested SSA approximating the distribution of X_2 appropriately. The downside is the CPU time is even longer than that shown in Table 5.6.1. The nested SSA produced a distribution for the virtual fast process that resembled the distribution produced by the SSA for some time points in the simulation (not shown). The slow scale SSA reproduced he distribution of the virtual fast process of the SSA through the different states in the system (not shown). We can conclude that the slow scale SSA provides the best approximation of the distribution of the virtual fast process.

Considering the overall behaviour of the system, all of the approximation methods give an accurate approximation of the mean of each species in the system, although initial approximations of the mean of X_3 were not good. In Figure 5.6.3 we see the error of the approximation of the mean of X_3, indicating the poor approximation during the initial part of the simulation. The slow scale SSA and implicit tau leaping method do not give as poor approximation in the initial part of the simulation as the other two methods. One reason the approximation of the mean of X_3 was not as accurate as that of X_2 is due to poor approximation of the transient. With the initial conditions of $X_3 = 0$ and $X_4 = 0$ the only way that X_3 can first be produced is through X_2 and X_4 can only be produced through X_3. It will take a short time into the simulation before the approximation methods surpass the transient and produce a more accurate approximation for X_3 and X_4, as seen in Figure 5.6.3. Figure 5.6.4 shows the standard deviation of X_1, indicating that the nested SSA and the slow scale SSA were able to produce the standard deviation of X_1 in agreement with the SSA. For the standard deviation of X_3 and X_4, all the methods have the same shape for the standard deviation, however the value of the standard deviation for the QSSA algorithm and the implicit tau method are smaller than that of the SSA (not shown). From what we have seen so far in this example, the nested SSA and the slow scale SSA are the two methods that best

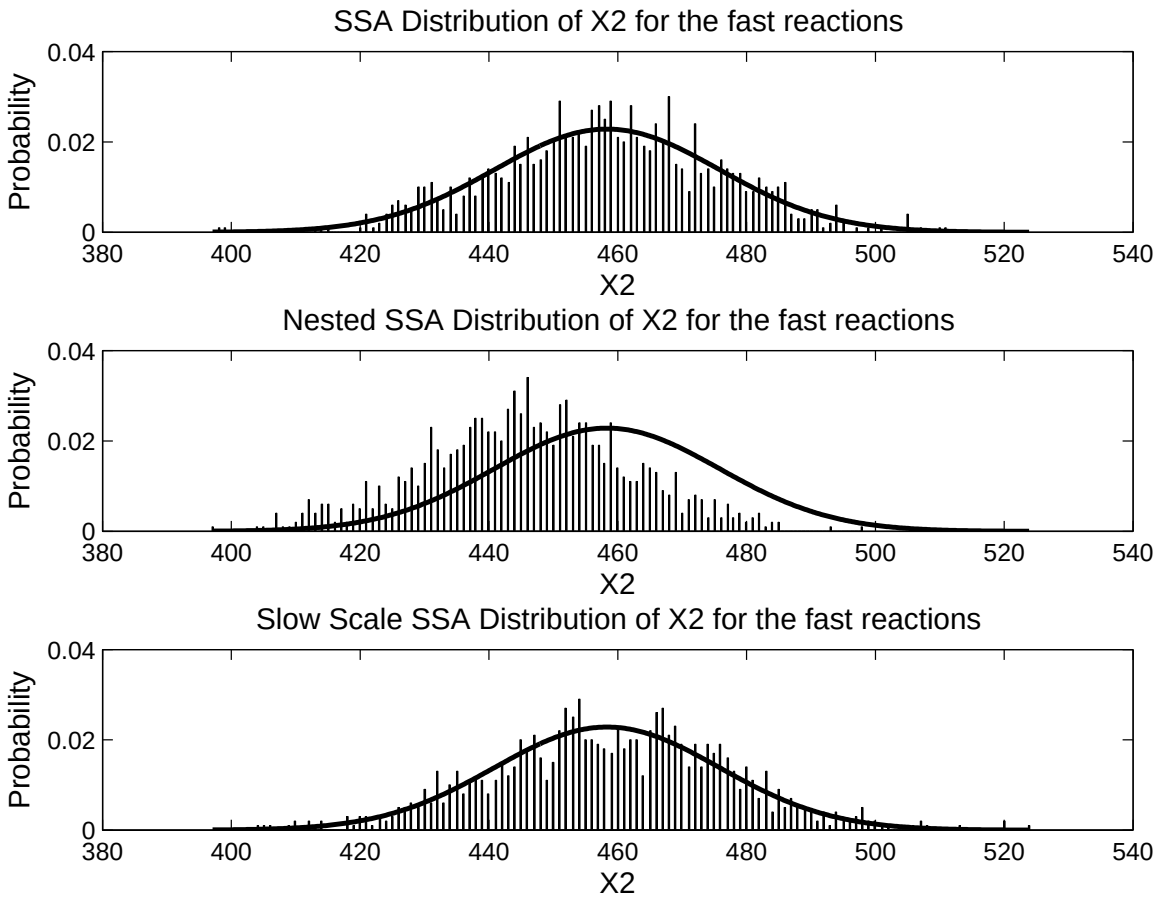

Figure 5.6.1: Network of Isomerizations: The distribution of the virtual fast process for X_2 for the SSA, nested SSA, slow scale SSA. The curve is the theoretical normal. $X_1 = 950$, $X_2 = 425$ and $X_{T_1} = 1375$. Ensemble size of 1000 simulations. (ssSSA refers to the slow scale SSA)

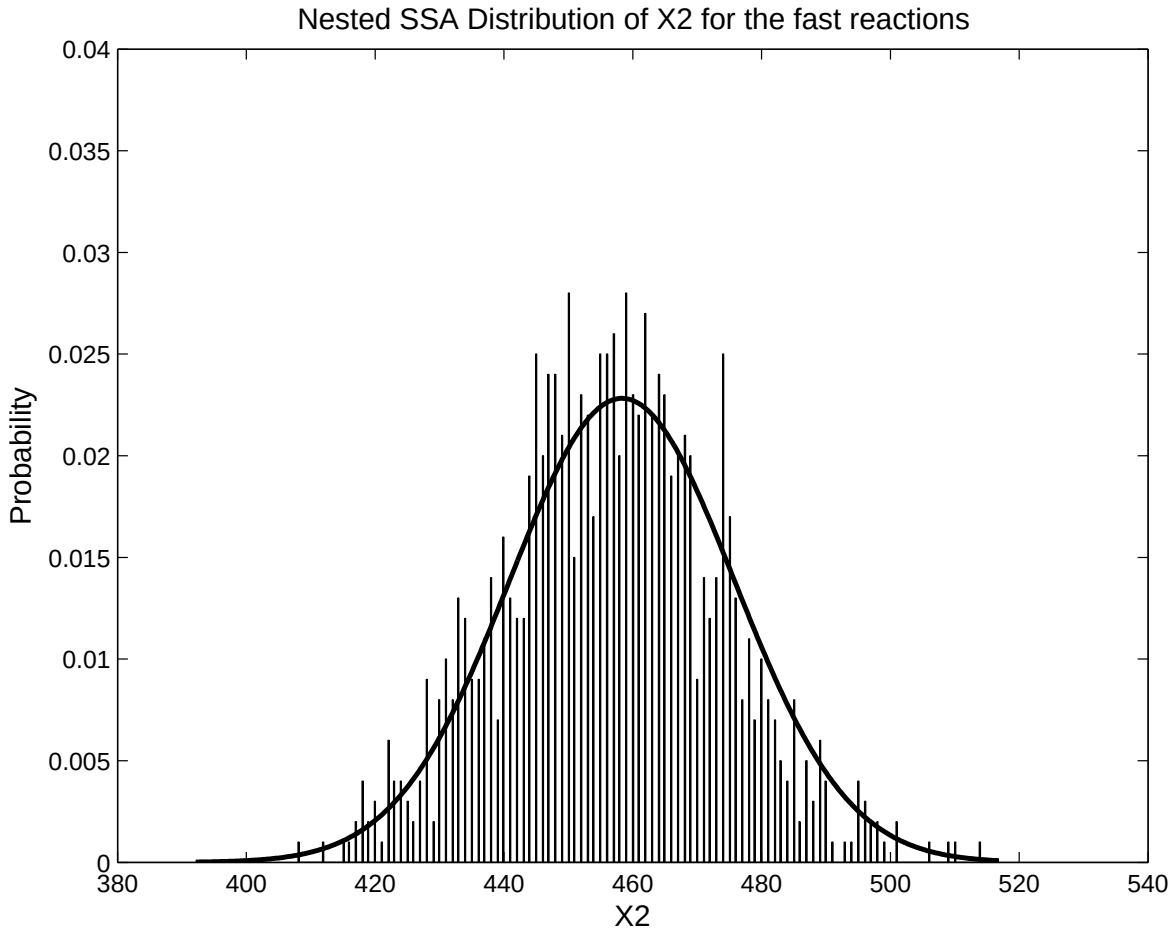

Figure 5.6.2: Network of Isomerizations: The distribution of the virtual fast process for X_2 for the nested SSA, where the inner SSA was ran for a longer period of time. The curve is the theoretical normal. $X_1 = 950$, $X_2 = 425$ and $X_{T_1} = 1375$. Ensemble size of 1000 simulations.

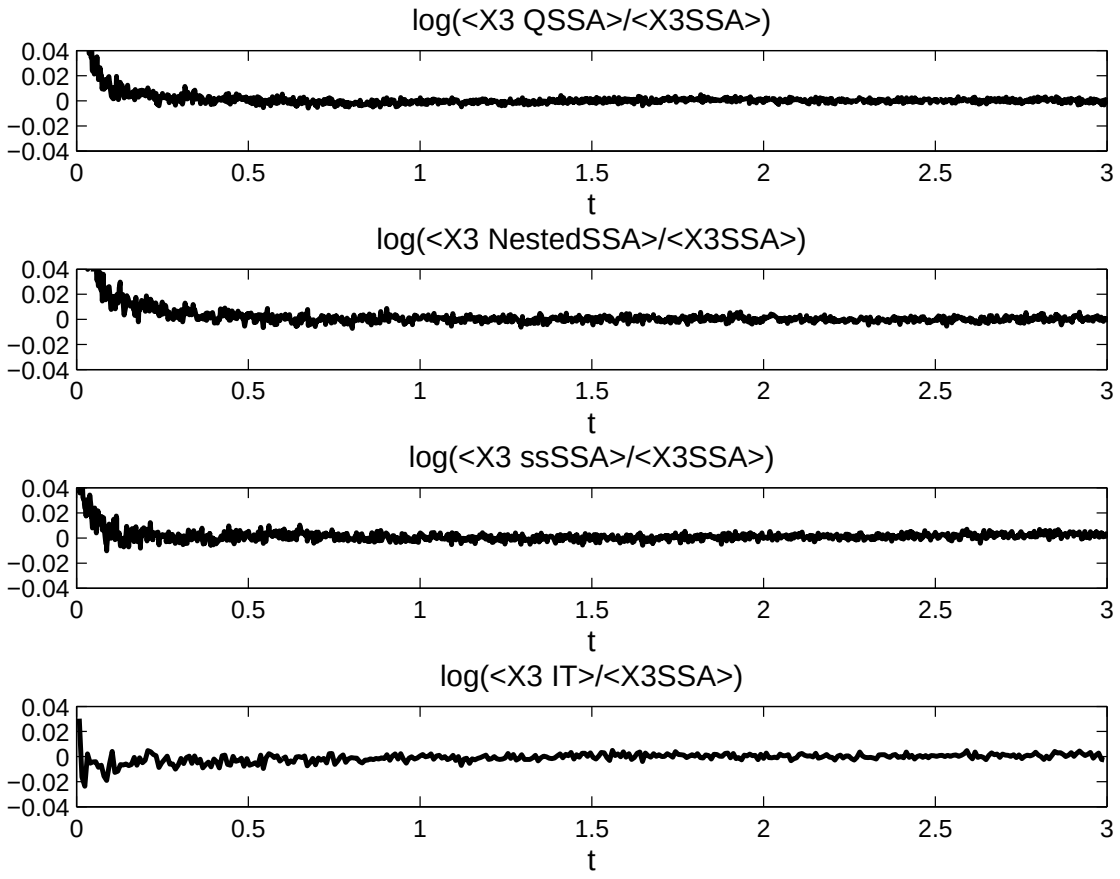

Figure 5.6.3: Network of Isomerizations: The base ten logarithm of the approximation of X_3's mean by the approximation methods over the approximation of X_3's mean by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

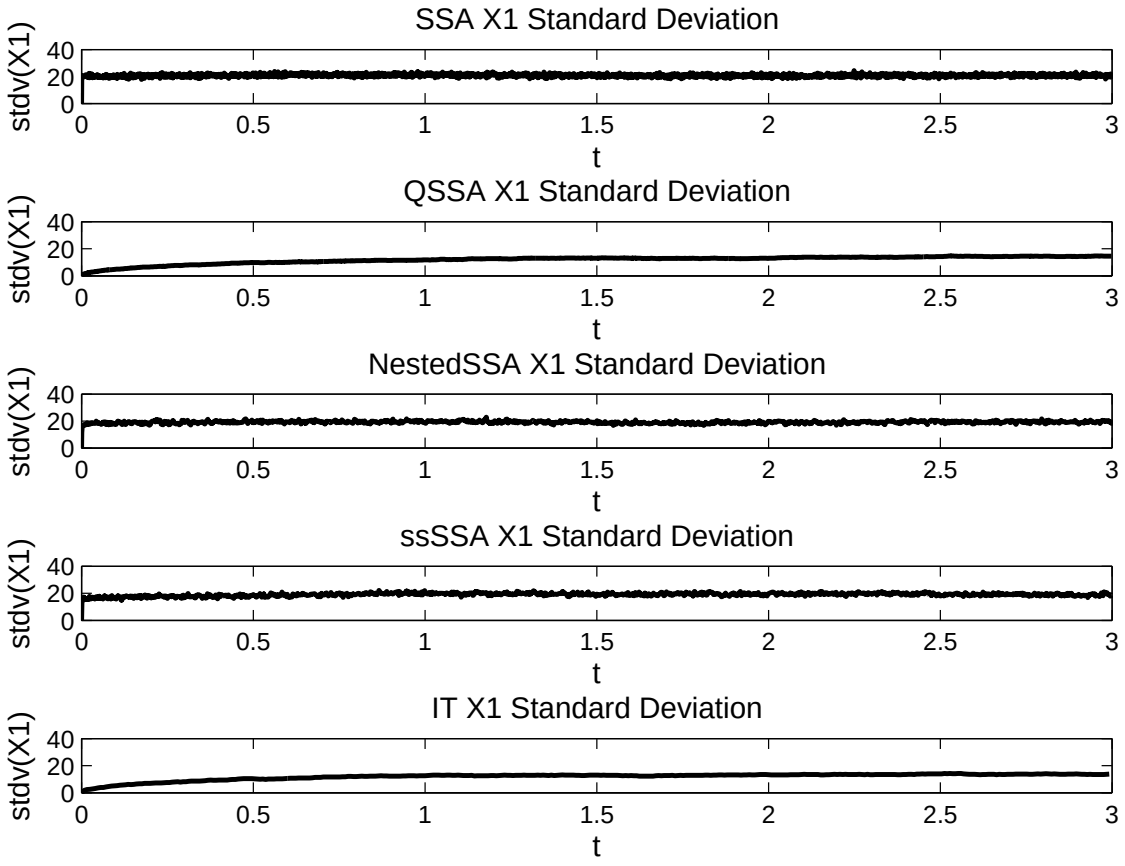

Figure 5.6.4: Network of Isomerizations: The approximation of X_1's standard deviation made by all of the methods. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

approximate the SSA. When we take a closer look at the ratio of the standard deviations (not shown), it is still hard to tell whether the nested SSA or the slow scale SSA is the best approximation. The histograms at 3 time units (not shown) and skewness (not shown) of the species also do not give us any insight into the comparison. From the results we have found, the inaccuracy of the nested SSA in determining the distribution of the virtual fast process had very little effect on the overall accuracy of the nested SSA. However, the slow scale SSA is the best approximation to use for this example, since it is able to approximate the fast species under the fast reactions consistently throughout the entire simulation, where the nested SSA is not. The slow scale SSA is also the best method to use as it was almost the fastest approximation method, second only to the QSSA algorithm. In terms of accuracy of the overall behaviour of the system the nested SSA and slow scale SSA are equivalent. One of the reasons why the QSSA algorithm provided poor results is that near the beginning of the simulation its approximation of the variance for each of the species was inaccurate until a slow reaction occurred. Until a slow reaction occurs in this system the QSSA algorithm will not have a value for the variance, we will see this more clearly in the next example. However, further into the simulation the QSSA algorithm's approximation of the variance improved.

Chapter 6

Example 3: Fast Species Acting as a Catalyst

This example is from [20], and involves three species and three reactions:

$$\begin{aligned} S_1 &\underset{c_2}{\overset{c_1}{\rightleftharpoons}} S_3 \\ S_1 + S_2 &\xrightarrow{c_3} S_1. \end{aligned}$$

The propensity functions for this system are

$$\mathbf{a}(\mathbf{x}) = (c_1 x_1, c_2 x_3, c_3 x_1 x_2).$$

The stoichiometry vectors for the system are $\mathbf{v}_1 = (-1, 0, 1)$, $\mathbf{v}_2 = (1, 0, -1)$ and $\mathbf{v}_3 = (0, -1, 0)$. We will take the fast reactions in the system to be

$$S_1 \underset{c_2}{\overset{c_1}{\rightleftharpoons}} S_3,$$

so the stoichiometry vectors for the fast system are $\mathbf{v}_1^f = (-1, 1)$, $\mathbf{v}_2^f = (1, -1)$. There is a conservation in the fast system, $X_T = X_1(t) + X_3(t)$ where X_T is constant on the fast time scale. The interesting aspect of this system is that the slow reaction has no effect on the fast species. In the two previous examples a slow reaction affected one or more of the fast species. Here the fact that the fast system remains unchanged by the slow system causes some interesting behaviour for the QSSA algorithm. In this system the two fast species remain constant, while the slow species decays over time until the slow species has depleted.

6.1 Theoretical Analysis

The chemical master equation that the SSA, QSSA algorithm, slow scale SSA and the nested SSA approximate is

$$\begin{aligned}\frac{\partial P(\mathbf{x},t|\mathbf{x}_0,t_0)}{\partial t} &= c_1x_1P(\mathbf{x}-\mathbf{v}_1,t|\mathbf{x}_0,t_0)-c_1x_1P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_2x_3P(\mathbf{x}-\mathbf{v}_2,t|\mathbf{x}_0,t_0)-c_2x_3P(\mathbf{x},t|\mathbf{x}_0,t_0)\\ &+ c_3x_1x_2P(\mathbf{x}-\mathbf{v}_3,t|\mathbf{x}_0,t_0)-c_3x_1x_2P(\mathbf{x},t|\mathbf{x}_0,t_0).\end{aligned} \tag{6.1.1}$$

Focusing on only the fast system, the chemical master equation is similar to (5.1.3) after summing over all possible values of the slow species X_2. Using the conservation $X_T = X_1(t) + X_3(t)$ in the chemical master equation we have the new chemical master equation, similar to (5.1.6). Similarly to that in section 5.1 we have that

$$\langle X_1(t)\rangle = \frac{c_2X_T}{c_1+c_2}, \tag{6.1.2}$$

$$\langle\langle X_1(t)\rangle\rangle = \frac{c_1c_2X_T}{(c_1+c_2)^2}, \tag{6.1.3}$$

$$\begin{aligned}\langle X_3(t)\rangle &= \frac{c_1X_T}{c_1+c_2},\\ \langle\langle X_3(t)\rangle\rangle &= \frac{c_1c_2X_T}{(c_1+c_2)^2}.\end{aligned}$$

These can be used by the slow scale SSA for the normal approximation to sample the virtual fast processes. However, later in this section we will see that an alternative distribution can be used. The chemical Langevin equation for this system is

$$\begin{aligned}\frac{dX_1(t)}{dt} &= -c_1X_1(t)+c_2X_3(t) \qquad (6.1.4)\\ &- \sqrt{c_1X_1(t)}\Gamma_1(t)+\sqrt{c_2X_3(t)}\Gamma_2(t)\\ \frac{dX_2(t)}{dt} &= -c_3X_1(t)X_2(t)-\sqrt{c_3X_1(t)X_2(t)}\Gamma_3(t) \qquad (6.1.5)\\ \frac{dX_3(t)}{dt} &= c_1X_1(t)-c_2X_3(t) \qquad (6.1.6)\\ &+ \sqrt{c_1X_1(t)}\Gamma_1(t)-\sqrt{c_2X_3(t)}\Gamma_2(t) \qquad (6.1.7)\end{aligned}$$

6.2 QSSA algorithm

The deterministic equations for the system are

$$\begin{aligned}\frac{dX_1}{dt} &= -c_1X_1 + c_2X_3\\ \frac{dX_2}{dt} &= -c_3X_1X_2\\ \frac{dX_3}{dt} &= c_1X_1 - c_2X_3\end{aligned} \tag{6.2.1}$$

Applying the QSSA algorithm, we will solve $-c_1X_1 + c_2X_3 = 0$ using the conservation, $X_T = X_1(t) + X_3(t)$. This means that

$$X_1 = \frac{c_2X_T}{c_1 + c_2} \tag{6.2.2}$$

and

$$X_3 = \frac{c_1X_T}{c_1 + c_2}. \tag{6.2.3}$$

With the approximation of the virtual fast processes the fast time scale is removed and the following slow reaction remains to be simulated

$$S_1 + S_2 \xrightarrow{c_3} S_1.$$

6.3 Slow Scale SSA

As in section 5.3 the distribution of the fast system for the slow scale SSA is a binomial distribution.

6.4 Nested SSA

We took an approach similar to that in [33] in which the sum of the constant fast reaction rates is inverted to determine the value for T_f. Since the fast system is starting in equilibrium, we took $T_0 = 0$. We used only one inner loop to reduce the computational time of the nested SSA.

6.5 Implicit Tau Leaping Method

With the implicit tau leaping method we used the approach of taking one large jump and a sequence of 5 small jumps. This is similar to the approach taken in [20] for this example.

Method	CPU Time (sec.)
SSA	1832.20
QSSA algorithm	0.00003
Nested SSA	2.92
Slow Scale SSA	0.05
Implicit Tau	1.38

Table 6.6.1: Fast Species Acting as a Catalyst: CPU times for all of the methods for 0.1 time units, for a single sample path. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2.

6.6 Comparison of the Methods

The reaction rates are

$$c_1 = 10^5,\ c_2 = 10^5,\ c_3 = 0.0005.$$

The initial value for each species was

$$\mathbf{X}(0) = (10000, 100, 10000).$$

Each simulation was run for 0.1 time units. The CPU times for each method for this example are given in Table 6.6.1, where we see that the QSSA algorithm is the fastest approximation method. To begin our comparison, we look at the mean of X_1 in Figure 6.6.1, and see that all the methods agree with the SSA. The most significant difference is that the mean for X_1 produced by the SSA is noisy compared to the results of the approximation methods. With the approximation of X_2, all of the methods produce similar results to that of the SSA, as shown in Figure 6.6.2 which shows the error of the mean for X_2. Although, in Figure 6.6.2, the implicit tau leaping method has a slightly better approximation of the mean of X_2 than the the other approximation methods. Figure 6.6.3 shows the standard deviation of X_2, and indicates that the standard deviation of X_2 is approximated very well by all of the approximation methods, although the QSSA algorithm's is the weakest, as shown in the error of the approximation of the standard deviation of X_2 in Figure 6.6.4. The implicit tau leaping method has some trouble at the beginning of the simulation. In Figure 6.6.5, which shows the standard deviation of X_1, we can see that the implicit tau leaping method and the QSSA algorithm give poor approximations of the standard deviation for the fast species X_1. It is understandable that the QSSA algorithm's approximation of X_1 is poor because the QSSA algorithm has X_1 remaining constant throughout the entire simulation. This can be seen in Figure 6.6.1, where the QSSA algorithm describes a constant behaviour. Similarly, the implicit tau method generates a somewhat constant mean,

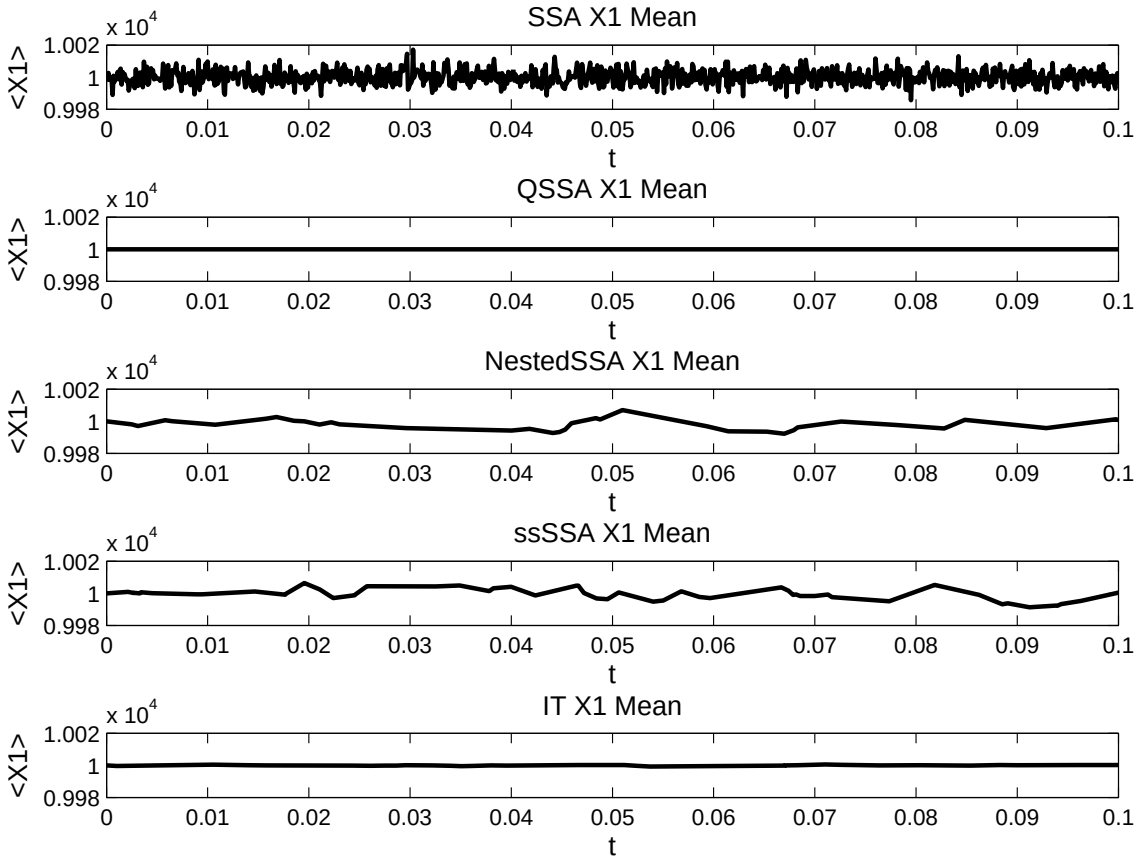

Figure 6.6.1: Fast Species Acting as a Catalyst: The approximation of X_1's mean made by all of the methods. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

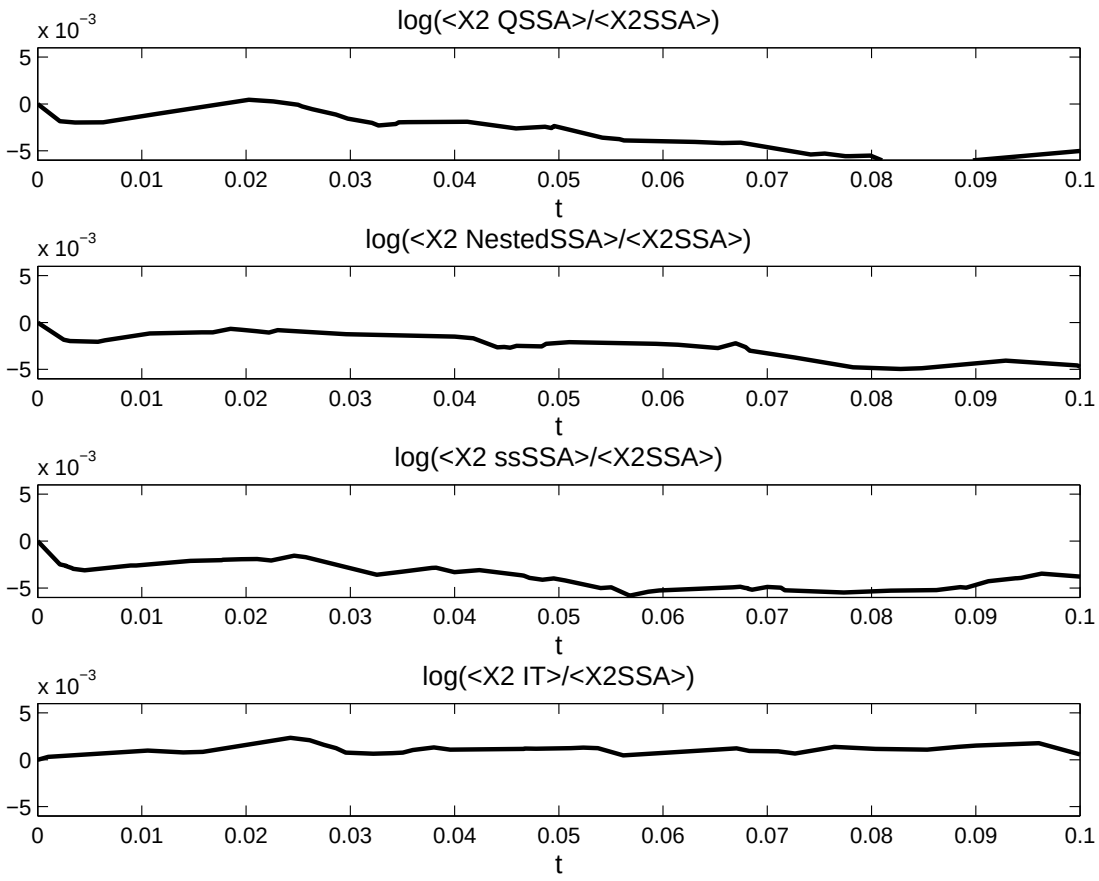

Figure 6.6.2: Fast Species Acting as a Catalyst: The base ten logarithm of the approximation of X_2's mean by the approximation methods over the approximation of X_2's mean by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

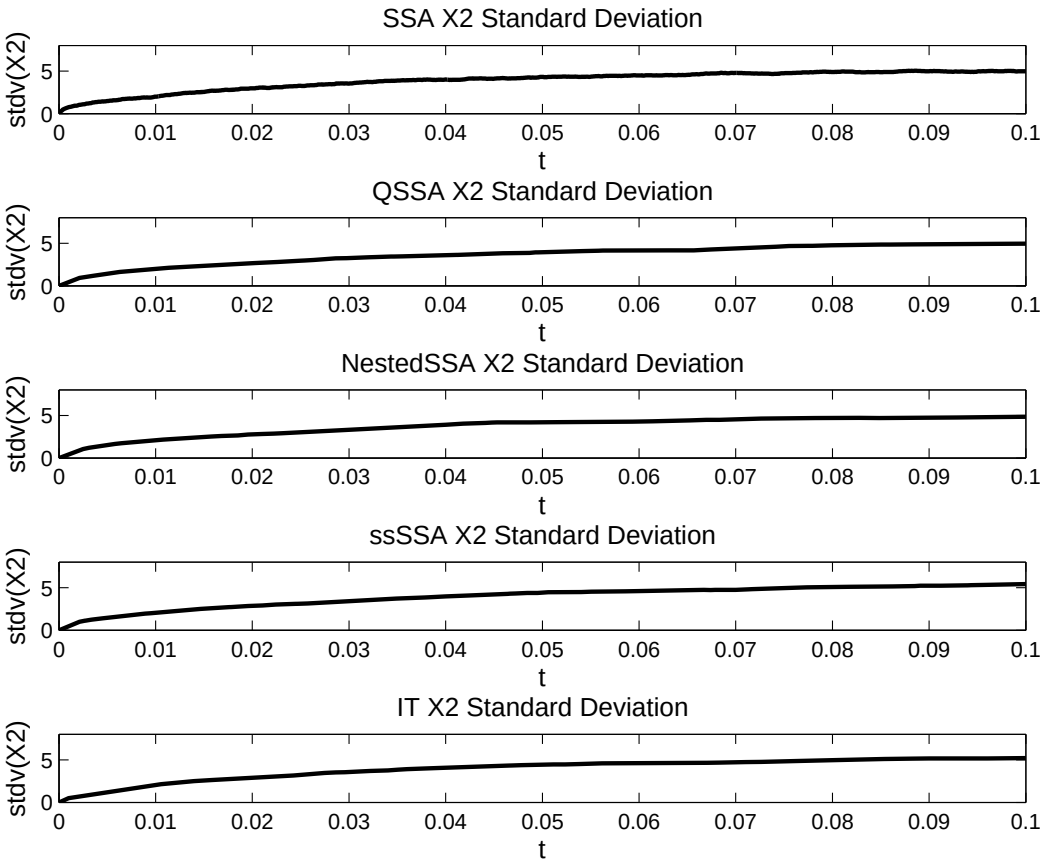

Figure 6.6.3: Fast Species Acting as a Catalyst: The approximation of X_2's standard deviation made by all of the methods. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

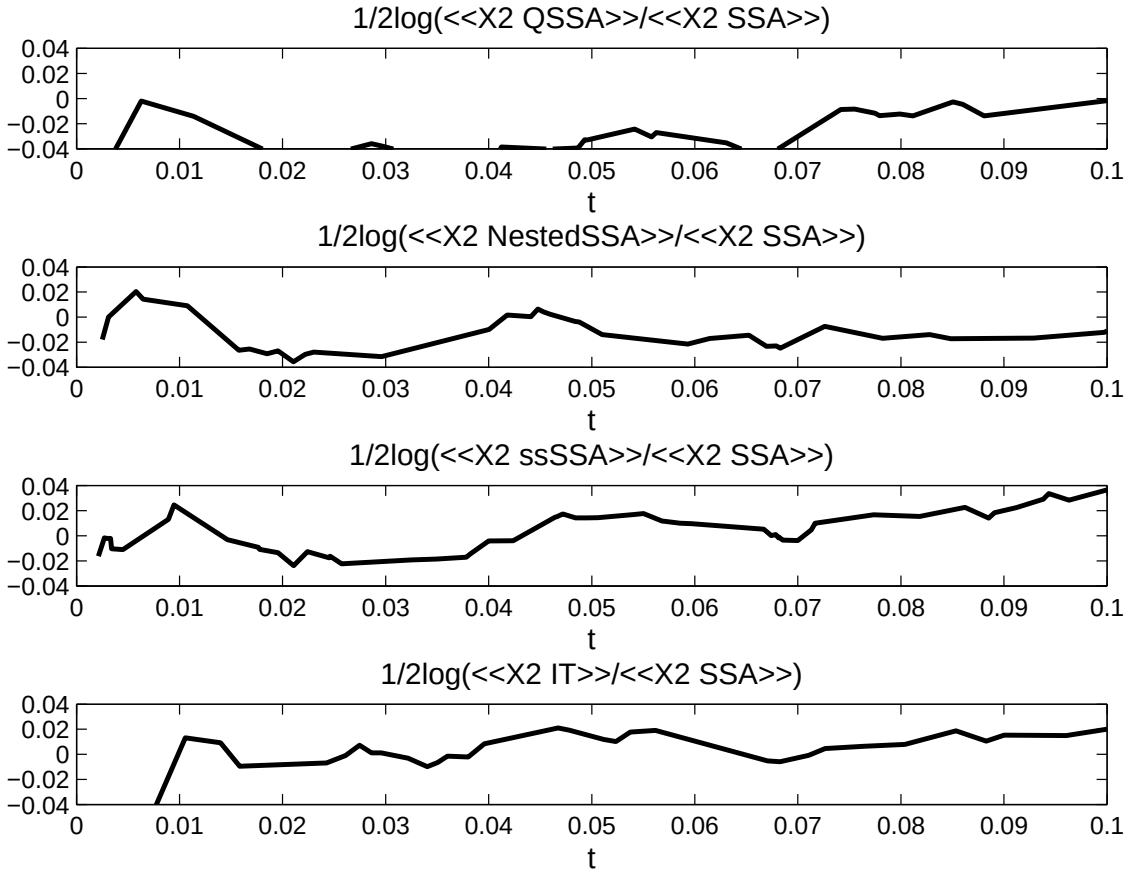

Figure 6.6.4: Fast Species Acting as a Catalyst: The base ten logarithm of the approximation of X_2's standard deviation by the approximation methods over the approximation of X_2's standard deviation by the SSA. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

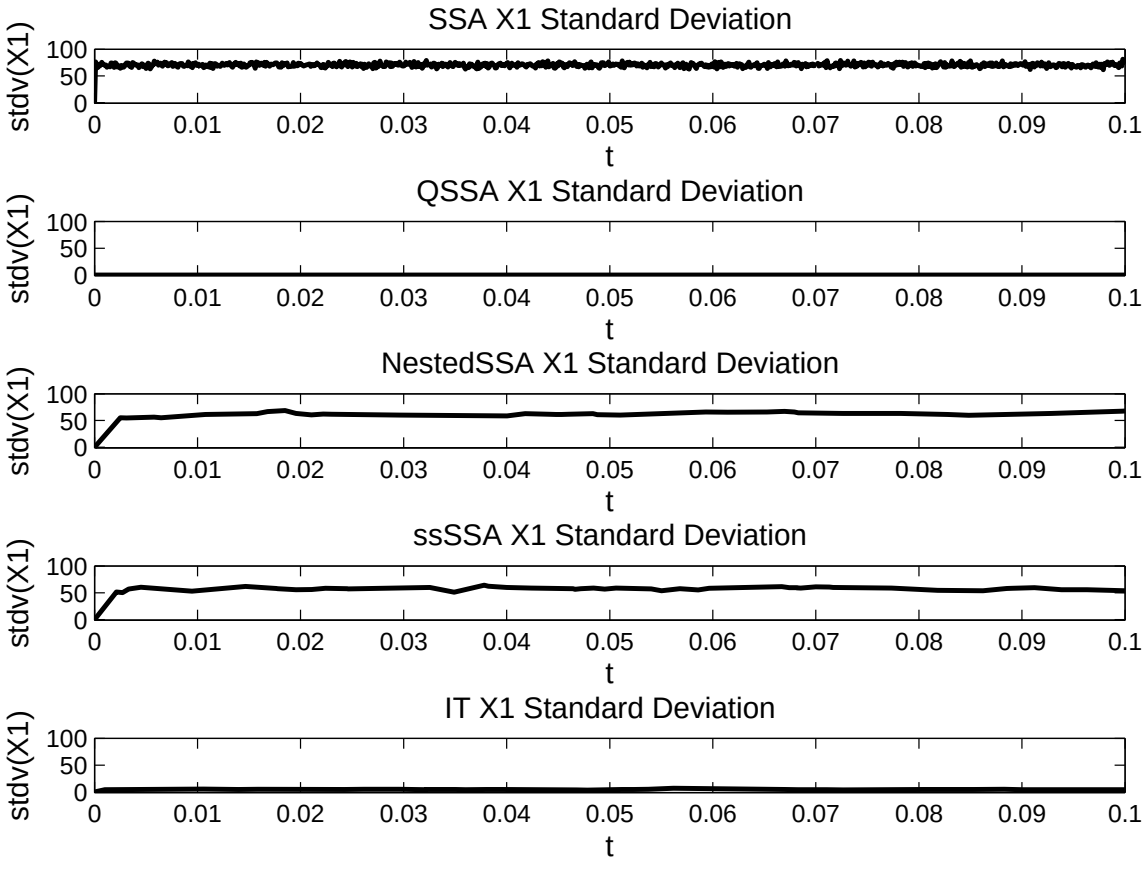

Figure 6.6.5: Fast Species Acting as a Catalyst: The approximation of X_1's standard deviation made by all of the methods. Ensemble size of 250 simulations. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

Figure 6.6.1, reflected by the variance. Even the slow scale SSA and the nested SSA were not able to reproduce the standard deviation of X_1, as the two methods under-approximated the value. Even with a closer look at the errors of the standard deviation of the nested SSA and the slow scale SSA, the two are still fairly similar to each other. All of the methods are similar to the SSA when we look at the skewness of X_2 after the time point of 0.01, but for the implicit tau method prior to 0.01 the skewness is decreasing in value while the SSA is increasing in value (not shown).

The slow scale SSA is again the best approach to use for this example, as it was able to reproduce results of the mean, variance and skewness similar to the SSA. Histograms (not shown) of the slow scale SSA shared similarities to that of the SSA as did all the other approximation methods. Another reason the slow scale SSA was the best method is that its CPU time was one of the fastest, second only to the QSSA algorithm. The QSSA algorithm was not considered to be one of the best approximation methods because of its poor ability to approximate the behaviour of the fast species. The reason the nested SSA was not chosen was because of its CPU time was much slower than the slow scale SSA's.

Chapter 7

Example 4: Oscillatory System

We consider an example of a system that exhibits sustained limit cycle oscillations. The chemical system is composed of three species involved in seven reactions:

$$\begin{aligned} S_3 &\xrightarrow{c_1} S_1 \\ S_3 &\xrightarrow{c_2} S_2 \\ S_1 &\xrightarrow{c_3} 2S_1 \\ S_1 + S_2 &\xrightarrow{c_4} 2S_2 \\ S_2 &\xrightarrow{c_5} \emptyset \\ \emptyset &\xrightarrow{c_6} S_3 \\ S_3 + S_2 &\xrightarrow{c_7} S_2. \end{aligned}$$

This system is similar to the Lotka reaction system, i.e. the predator-prey model. We created a fast system that reached a steady state dependent on the state of the slow species X_2. We also included two slow reactions where X_3 produced X_1 and X_2, so that the slow species would not go extinct. The propensities for this system are

$$\mathbf{a}(\mathbf{x}) = (c_1x_3, c_2x_3, c_3x_1, c_4x_1x_2, c_5x_2, c_6, c_7x_3x_2).$$

The stoichiometry vectors for this system are $\mathbf{v}_1 = (1, 0, -1)$, $\mathbf{v}_2 = (0, 1, -1)$, $\mathbf{v}_3 = (1, 0, 0)$, $\mathbf{v}_4 = (-1, 1, 0)$, $\mathbf{v}_5 = (0, -1, 0)$, $\mathbf{v}_6 = (0, 0, 1)$ and $\mathbf{v}_7 = (0, 0, -1)$. The fast system for this example is taken to be

$$\begin{aligned} \emptyset &\xrightarrow{c_6} S_3 \\ S_3 + S_2 &\xrightarrow{c_7} S_2. \end{aligned}$$

The fast stoichiometry values are $v_1^f = (1)$ and $v_2^f = (-1)$. From the previous examples we have yet to see a slow species have an effect on the behaviour of the fast system. As well, these approximation methods have yet to be tested against

an oscillating system.

7.1 Slow Scale SSA

Here we will show that the use of a normal distribution is sufficient for the approximation of the fast species for this system. In [17], the authors explain how to find the probability distribution of the state of a birth and death process. For this oscillatory example the birth rate of the species X_3 is $b(x_3) = c_6 B$ and the death rate is $d(x_3) = c_7 x_3 x_2$, where X_2 is a slow species and is considered fixed during the fast reactions. Using the recursive formula from [17], the probability distribution function is

$$P(x_3, \infty) = \begin{cases} \frac{c_7 x_2 (x_3+1) P(x_3+1, \infty)}{c_6} & x_3 = -1, 0 \\ \frac{c_6 P(x_3-1, \infty)}{c_7 x_2 x_3} & x_3 = 1, \cdots, L \end{cases} \tag{7.1.1}$$

where L is the upper limit of the species X_3. We start by looking at $x_3 = -1$ and see that $P(x_3 = -1, \infty) = 0$. Next we look at $x_3 = 0$ and see that

$$P(x_3 = 0, \infty) = \frac{c_7 x_2}{c_6} P(x_3 = 1, \infty)$$

or

$$P(x_3 = 1, \infty) = \frac{c_6}{c_7 x_2} P(x_3 = 0, \infty). \tag{7.1.2}$$

Using (7.1.2), and the recursive formula for the probability we found that for $x_3 \geq 1$

$$P(x_3, \infty) = \big(\frac{c_6}{c_7 x_2}\big)^{x_3} \frac{P(x_3 = 0, \infty)}{x_3!}. \tag{7.1.3}$$

Will will assume that $L = \infty$ to determine the probability distribution of X_3. Using a normalizing condition we have that

$$\sum_{x_3=0}^{L} P(x_3, \infty) = 1$$

which means using (7.1.3) and $L = \infty$ we have

$$P(x_3 = 0, \infty) \sum_{x_3=0}^{\infty} \big(\frac{c_6}{c_7 x_2}\big)^{x_3} \frac{1}{x_3!} = 1 \tag{7.1.4}$$

Solving (7.1.4), we have that

$$P(x_3 = 0, \infty) = \exp^{-\frac{c_6}{c_7 x_2}}.$$

With that we have

$$P(x_3, \infty) = \big(\frac{c_6}{c_7 x_2}\big)^{x_3} \frac{\exp^{-\frac{c_6}{c_7 x_2}}}{x_3!}.$$

This implies that the probability distribution is a Poisson, with $\lambda = \frac{c_6}{c_7 x_2}$. However, with the upper limit not truly capable of reaching an extremely large value the chance that the distribution is a Poisson is small. Since a Poisson (for the case that X_2 actually reaches a small value) can be approximated by a normal distribution [20] and it is suggested in [17] that the normal approximation be used when the distribution can not be determined, we will approximate the fast species of this system using a normal distribution. The moments for the normal distribution can be found using the stationary moment equations, from which we have

$$\begin{aligned} \langle X_3(\infty)\rangle &= \frac{c_6 B}{c_7 X_2(t)} \\ \langle\langle X_3(\infty)\rangle\rangle &= \frac{c_6 B}{c_7 X_2(t)}. \end{aligned}$$

7.2 Nested SSA and Implicit Tau Method

We chose

$$T_f = \max_{X_2(t)} \Big\{ \frac{1}{c_6 + c_7 X_2(t)} \Big\} \approx 1.0001 \times 10^{-6},$$

$T_0 = 0$ and $N = 1$ for the nested SSA. The fast system is similar to the fast system of the example in chapter 6, so we took a similar approach to determine our parameters. For the implicit tau leaping method we chose our jumps to occur on the slow time scale.

7.3 Comparison of Methods

The reaction rates are taken as

$$\begin{aligned} &c_1 = 1, \quad c_2 = 1, \quad c_3 = 10, \quad c_4 = 0.01, \\ &c_5 = 10, \quad c_6 = 1000000, \quad c_7 = 1 \end{aligned}$$

and the initial values are chosen

$$\mathbf{X}(0) = (1000, 1000, 1000).$$

We ran the simulations for 1000 time units. We focus attention on the asymptotic behaviour, we also truncated the first 100 time units to eliminate any transient. In Figures 7.3.1 and Figure 7.3.2 we see that each method generates oscillations. To address the accuracy of the approximations, we compare the average of the spectra of the sample paths. We took the fast Fourier transform (FFT) of an interval of 50 time units, from our single sample path, starting at 100 time units. This gave us 18 FFT's to average. To reduce any error caused at the ends of our intervals we implemented a Hann window. Refer to the appendix for further information on

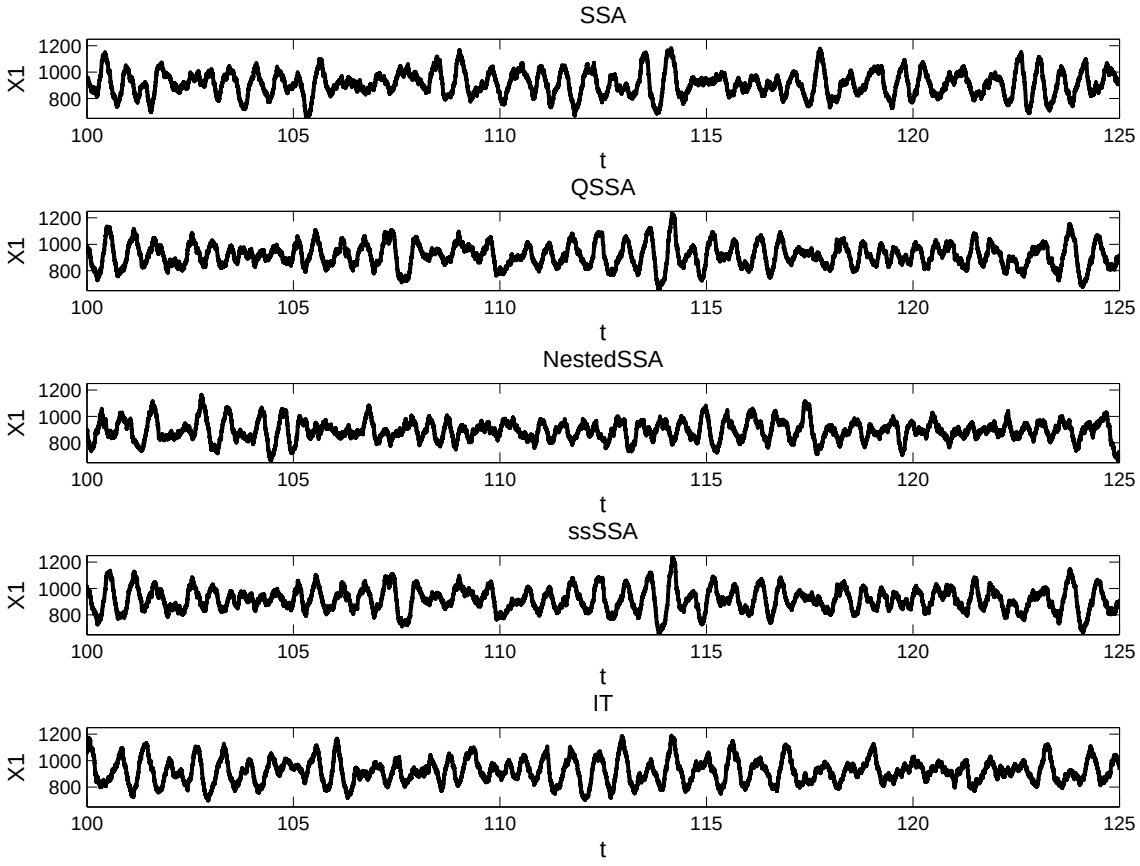

Figure 7.3.1: Oscillatory System: The plot for X_1 during a portion of the simulation, $t \in [100, 125]$. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

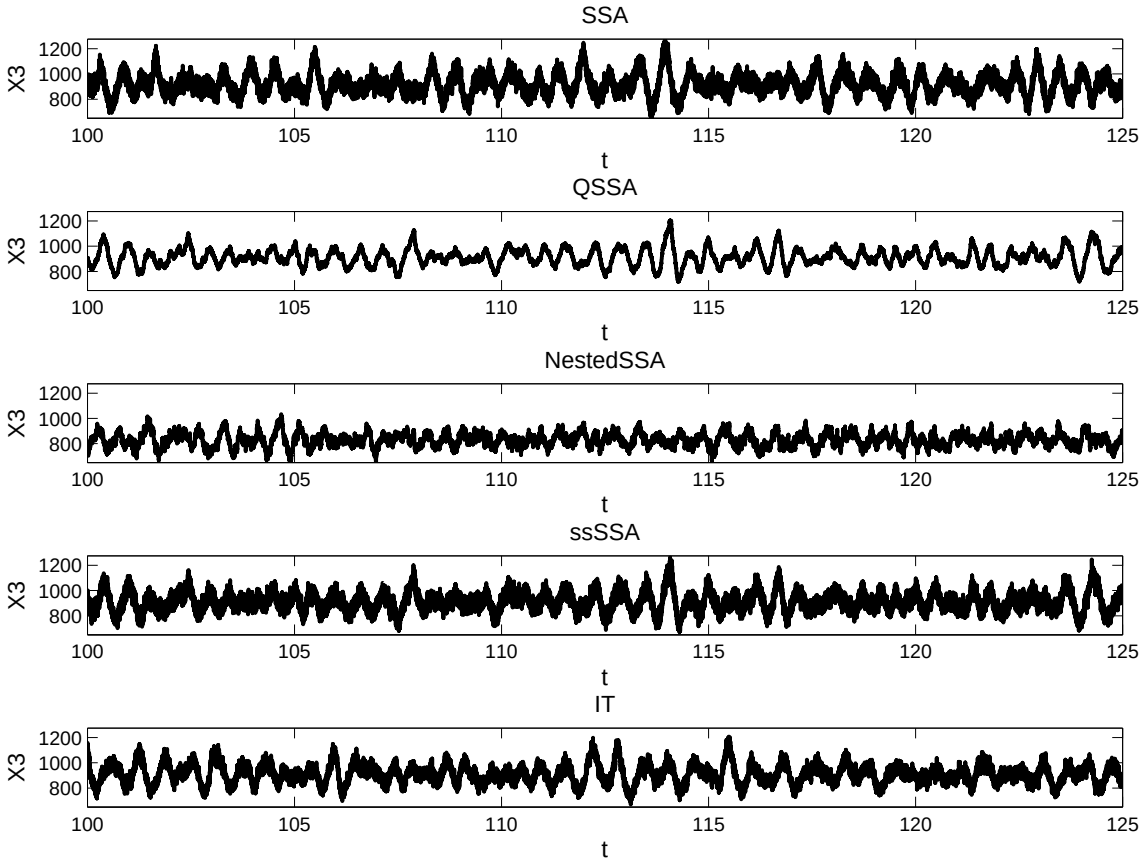

Figure 7.3.2: Oscillatory System: The plot for X_3 during a portion of the simulation, $t \in [100, 125]$. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

Method	$\langle X_1 \rangle$	$\langle X_2 \rangle$	$\langle X_3 \rangle$	$\langle\langle X_1 \rangle\rangle$	$\langle\langle X_2 \rangle\rangle$	$\langle\langle X_3 \rangle\rangle$
SSA	916.58	1099	914.27	8236.9	7953.9	6140.3
QSSA algorithm	916.01	1100	915.09	8171	7980	5564.2
Nested SSA	889.38	1140.3	835.514	6716	6823.2	3544.1
Slow Scale SSA	916.10	1099.1	915.5	7645.7	7449.9	5649.7
Implicit Tau	917.03	1099.7	913.45	7980.3	7690.6	5743.8

Table 7.3.1: Oscillatory System: The mean and variance for each species for the entire simulation produced by each method

Hann windows. Looking at the average frequency in Figure 7.3.3 we see that all the methods roughly have the same frequency as the SSA.

Next, we will look at the mean and variance of the species throughout the entire simulation to see which method best compares to the SSA. We see from the Table 7.3.1 that the QSSA algorithm is the best method to approximate both the mean and the variance of the all the species. The slow scale SSA and implicit tau method do a good job as well approximating the mean, however their approximation of the variance is slightly under the value of the SSA. The nested SSA does not approximate the mean or the variance very well at all, although this could be improved by a different choice of design parameters.

Finally we will look at Figure 7.3.4 to Figure 7.3.6 to give some insight into the phase of the oscillations. If the values in the plot for an approximation method look similar to that of the SSA we can conclude the approximation method generates a similar phase as the SSA. Refer to the appendix for a more detailed description of this analysis approach for the phase of the oscillations. In Figure 7.3.4, we see that the nested SSA best approximates the SSA, at least for when $X_2 \in (900, 1400)$. The slow scale SSA and the QSSA algorithm are somewhat similar to the SSA, except that the slow scale SSA and QSSA algorithm have a slightly higher mean for $X_2 \in (900, 1400)$. Also in Figure 7.3.4, we see that given $X_3 \in (800, 1250)$ the slow scale SSA and the implicit tau method most resemble the SSA. The QSSA algorithm does well for a short interval, $X_3 \in (800, 1000)$, but after $X_3 = 1000$ the mean of X_1 starts to increase. Before we make a conclusion about the phase of X_1 we must look at the standard deviation to see if each method varies in value similar to the SSA. As for the variance, in Figure 7.3.5 the QSSA algorithm and slow scale SSA are the methods best able to reproduce the standard deviation of X_1, given $X_2 \in (900, 1300)$. The results from the nested SSA and implicit tau are similar to the SSA but for a shorter interval of X_2; thus the nested SSA and implicit tau are considered less accurate. Given $X_3 \in (750, 1250)$ we see in Figure 7.3.5 that the slow scale SSA results best resemble those of the SSA. Given this information about X_1, we conclude that the method that best captures the phase of X_1 is the slow scale SSA. In Figure 7.3.6, we see that the results from the slow scale SSA best resemble the standard deviation of X_3 given X_1 or X_2.

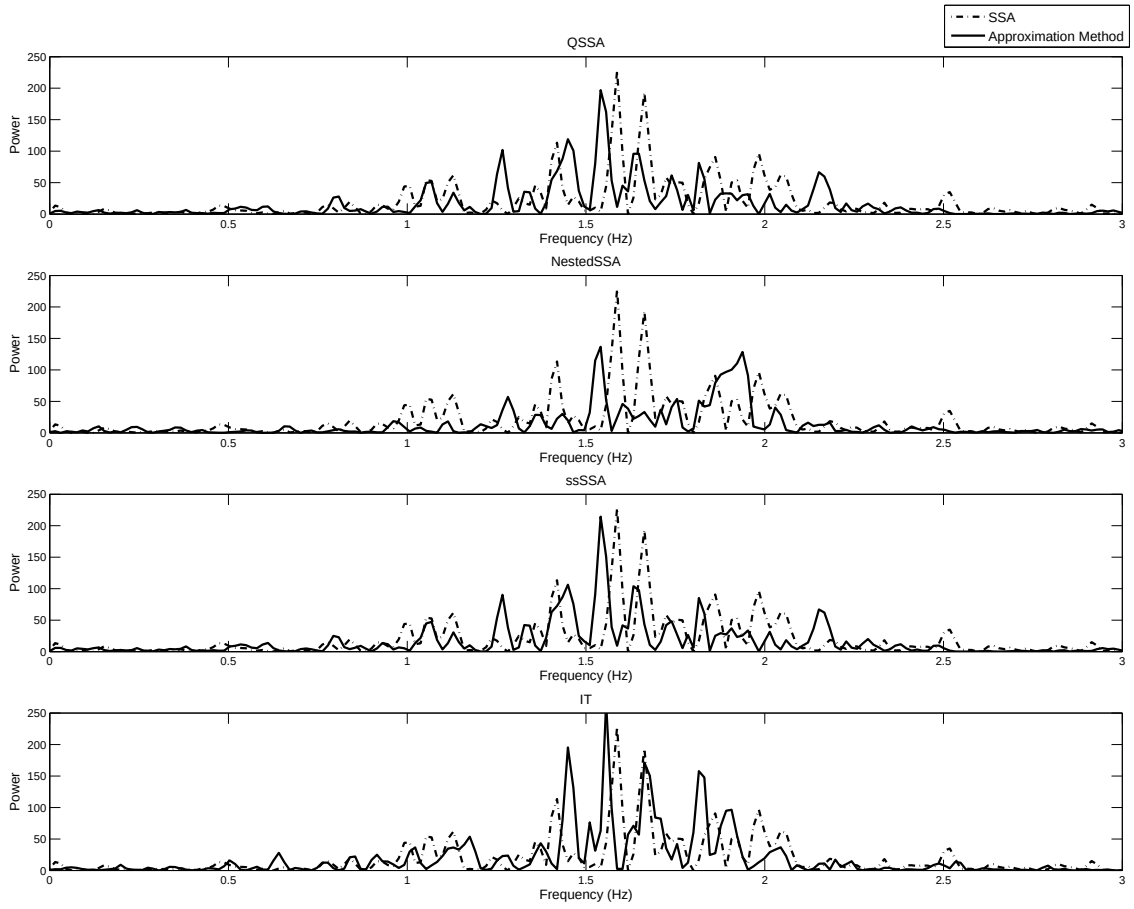

Figure 7.3.3: Oscillatory System: The average power of X_1 with the application of a Hann windows applied to each 50 time length interval for all the methods. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

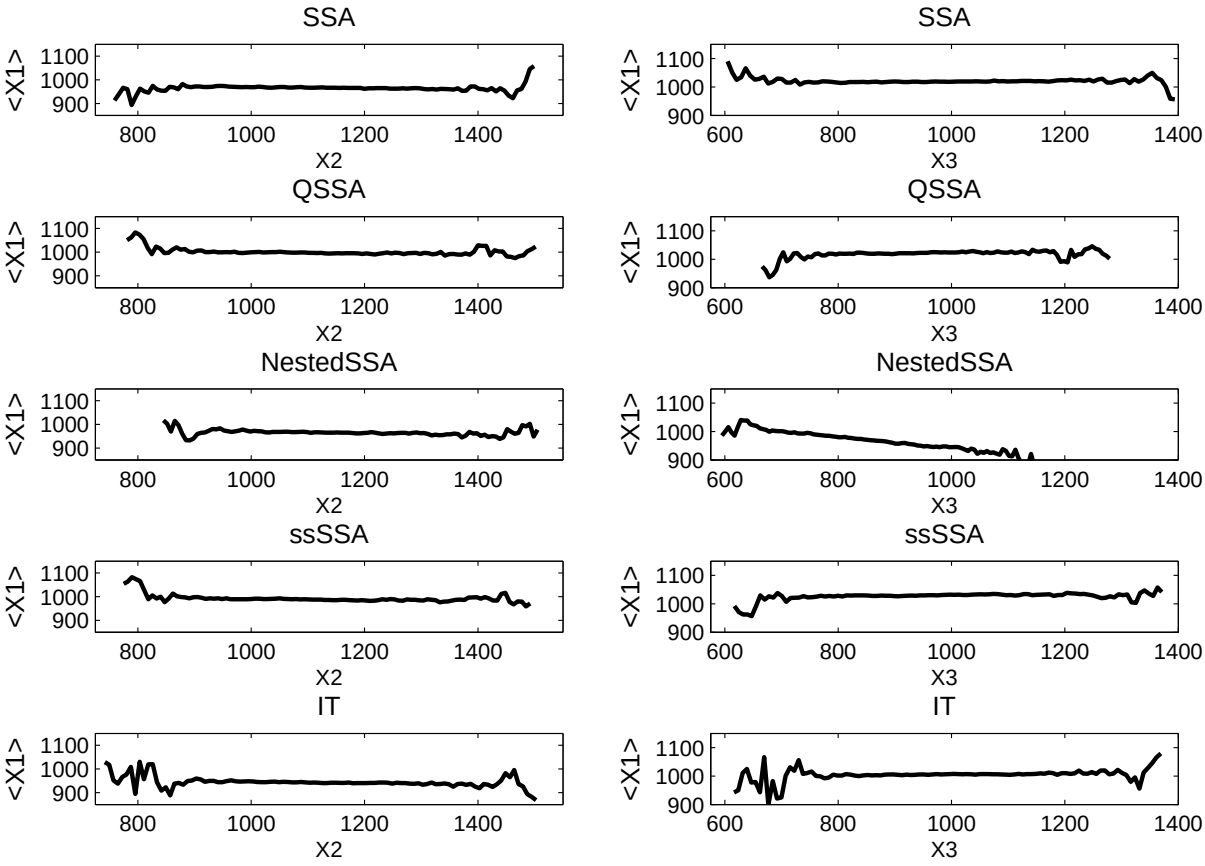

Figure 7.3.4: Oscillatory System: The phase analysis for X_1, showing the mean of X_1 given X_2 and X_3 produced by each method. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

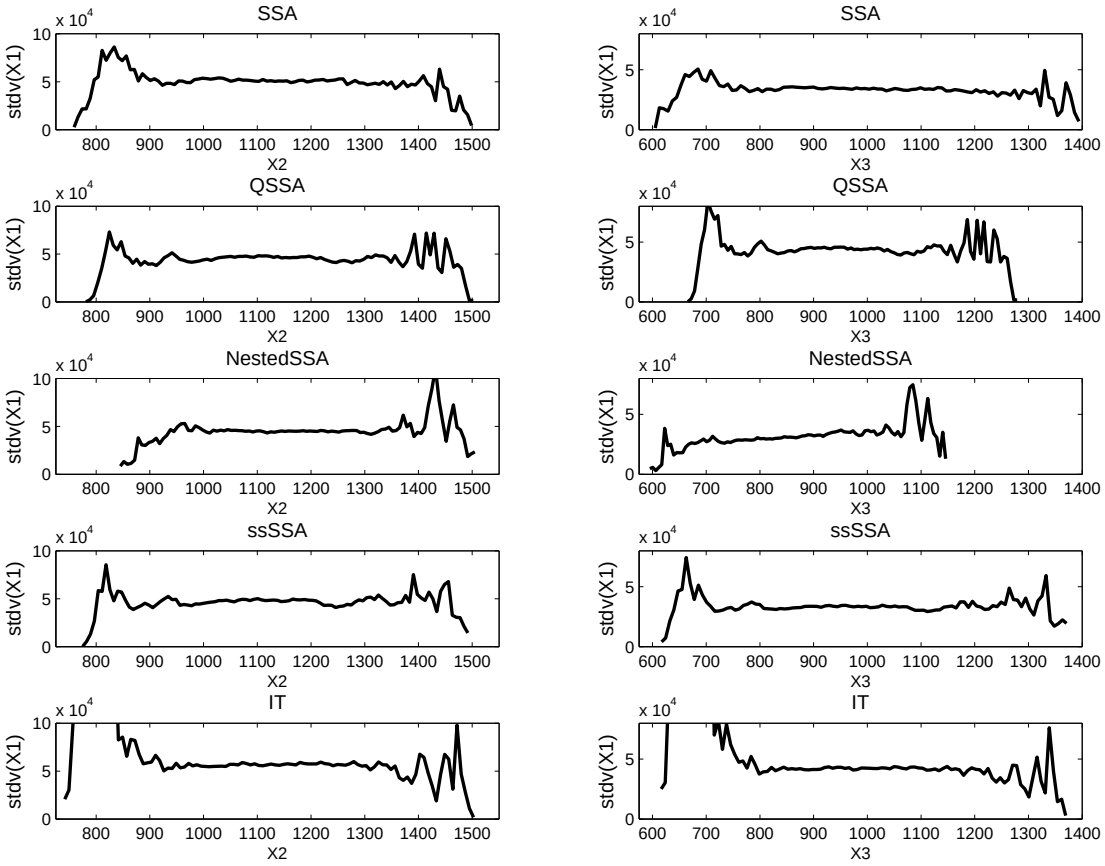

Figure 7.3.5: Oscillatory System: The phase and amplitude analysis for X_1, showing the standard deviation of X_1 given X_2 and X_3. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

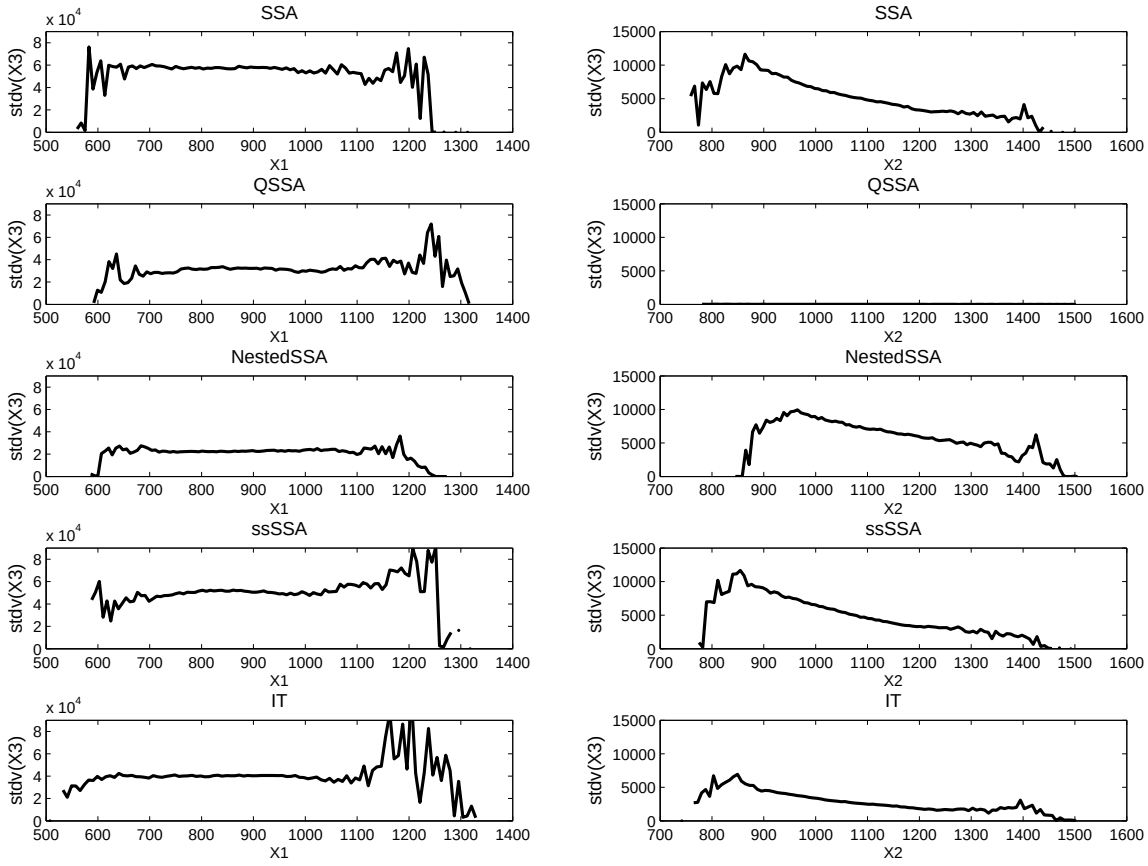

Figure 7.3.6: Oscillatory System: The phase and amplitude analysis for X_3, showing the standard deviation of X_3 given X_1 and X_2. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

The implicit tau results also resemble the SSA results but under-approximate the value of the standard deviation of X_3. The QSSA algorithm does not produce as much noise for the fast species as the other methods, as seen in Figure 7.3.2, so its approximation of the standard deviation of X_3 given X_1 or X_2 is too low. The slow scale SSA is the closest method in resembling the SSA for the phase of X_3.

From the analysis, we can conclude that the slow scale SSA and QSSA algorithm are the best methods to approximate the SSA for this oscillating system. We have found that the slow scale SSA excels where the QSSA algorithm does not and the QSSA algorithm excels where the slow scale SSA does not. The slow scale SSA's approximation of the frequency was slightly under the value of the SSA and the variance of the species was also slightly under approximated. The QSSA algorithm was not able to produce the noise as the other methods did in the fast system. This resulted in the standard deviation of X_3 given either X_1 and X_2 to be under approximated compared to the values produced by the SSA. The QSSA algorithm was able to reproduce the frequency of the system slightly better than that of the slow scale SSA. As well the QSSA algorithm was able to reproduce the variance of a species similar to the SSA slightly better than the slow scale SSA.

We will focus on the CPU time of each method to see if the QSSA algorithm or slow scale SSA was faster. We ran the system for 100 time units, for a single sample path; the QSSA algorithm had a CPU time of 83.45 seconds and the slow scale SSA had a CPU time of 96.06 seconds. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2. For this example it is difficult to come to a conclusion to determine whether the QSSA algorithm or slow scale SSA is the better approximation method. The slow scale SSA was not that far off in any of its approximations of the system but the QSSA algorithm was able to approximate those areas better than the slow scale SSA. The slow scale SSA accounted for the noise in the fast system as where the QSSA algorithm did not. The noise in the fast system can play an important role as it may slightly change the phase or even frequency of the oscillations. If the QSSA algorithm is unable to account for the noise in the fast system then at times the QSSA algorithm is not approximating the SSA very well. On the other hand, the QSSA algorithm CPU time is much faster than that of the slow scale SSA for longer simulation times.

Chapter 8

Example 5: Bistable System

This is a bistable system based on a model in [34]. The chemical system has two species that are involved in seven reactions:

$$\begin{aligned} 2S_1 + S_2 &\underset{c_2}{\overset{c_1}{\rightleftharpoons}} 3S_1 \\ S_1 &\underset{c_4}{\overset{c_3}{\rightleftharpoons}} S_2 \\ \emptyset &\xrightarrow{c_5} S_2 \\ S_1 &\xrightarrow{c_6} S_1 + S_2 \\ S_2 &\xrightarrow{c_7} \emptyset. \end{aligned}$$

The stoichiometry vectors are $\mathbf{v}_1 = (1, -1)$, $\mathbf{v}_2 = (-1, 1)$, $\mathbf{v}_3 = (-1, 1)$, $\mathbf{v}_4 = (1, -1)$, $\mathbf{v}_5 = (0, 1)$, $\mathbf{v}_6 = (0, 1)$ and $\mathbf{v}_7 = (0, -1)$. The propensity functions for this system are

$$\mathbf{a}(\mathbf{x}) = (c_1 x_1(x_1 - 1)x_2/2,\, c_2 x_1(x_1 - 1)(x_1 - 2)/6,\, c_3 x_1,\, c_4 x_2,\, c_5,\, c_6 x_1,\, c_7 x_2).$$

The fast reactions for this system are taken as

$$\begin{aligned} \emptyset &\xrightarrow{c_5} S_2 \\ {}_1 &\xrightarrow{c_6} S_1 + S_2 \\ S_2 &\xrightarrow{c_6} \emptyset. \end{aligned}$$

This implies that the fast species is S_2. The fast stoichiometry vectors are $\mathbf{v}_1^f = (1)$, $\mathbf{v}_2^f = (1)$ and $\mathbf{v}_3^f = (-1)$. This system is similar to the oscillating system, where the equilibrium of the virtual fast process depends on the state of the slow species.

8.1 Nested SSA and Implicit Tau Method

We chose

$$T_f = \max_{X_1(t)} \{ \frac{1}{c_5 + c_6 X_1(t) + c_7} \} = \frac{1}{c_5 + c_7},$$

$T_0 = T_f \times 10^{-3}$ and $N = 1$ for the nested SSA. The fast system is similar to the fast system of the oscillatory system in section 7, so we took a similar approach to determine our parameters. For the implicit tau leaping method we chose the method of a large jump then a sequence of small jumps. The size of our large jump was fixed at 50 time units. We then took a sequence of 5 small jumps fixed at 0.0001 time units.

8.2 Comparison of Methods

The reaction rates for the system are

$$c_1 = 0.001,\ c_2 = 1,\ c_3 = \frac{1}{250000},\ c_4 = 0.1,$$
$$c_5 = 3000000,\ c_6 = 40000,\ c_7 = 1000.$$

The initial values for the system are

$$S_1 = 0,\ S_2 = 3000$$

and the simulation was run for 100000 time units. Figure 8.2.1 shows a single trajectory for the both species simulated by the SSA. This shows us that the system is bistable. We will not be comparing the approximation methods to the SSA as the SSA took too long to complete the full run of 10^5 time units making it difficult to do any statistical analysis. The reason the SSA was taking so long to simulate this system was because of the fast propensity values are so much larger than the slow propensity values.

Figure 8.2.2 shows a single trajectory of species X_1 for the system and Figure 8.2.3 shows a single trajectory of species X_2, both figures produced by the approximation methods. We see from both of the figures that the implicit tau leaping method was unable to make the transition to the "high" state, indicating that it is not a good approach for this system. The implicit tau leaping method was unable to reach the high state due to the fact that the method was unable to produce enough noise to force it out of the lower state. This can be due to our approach in the selection of tau. In Figure 8.2.3, we see that the approximation of the nested SSA is not comparable to either the slow scale SSA or the QSSA algorithm's approximation of the fast species. From our knowledge from the previous examples, the choice of different parameters could possibly improve the nested SSA's approximation. Figure 8.2.4 shows the base ten logarithm of the distribution of the fast species X_2

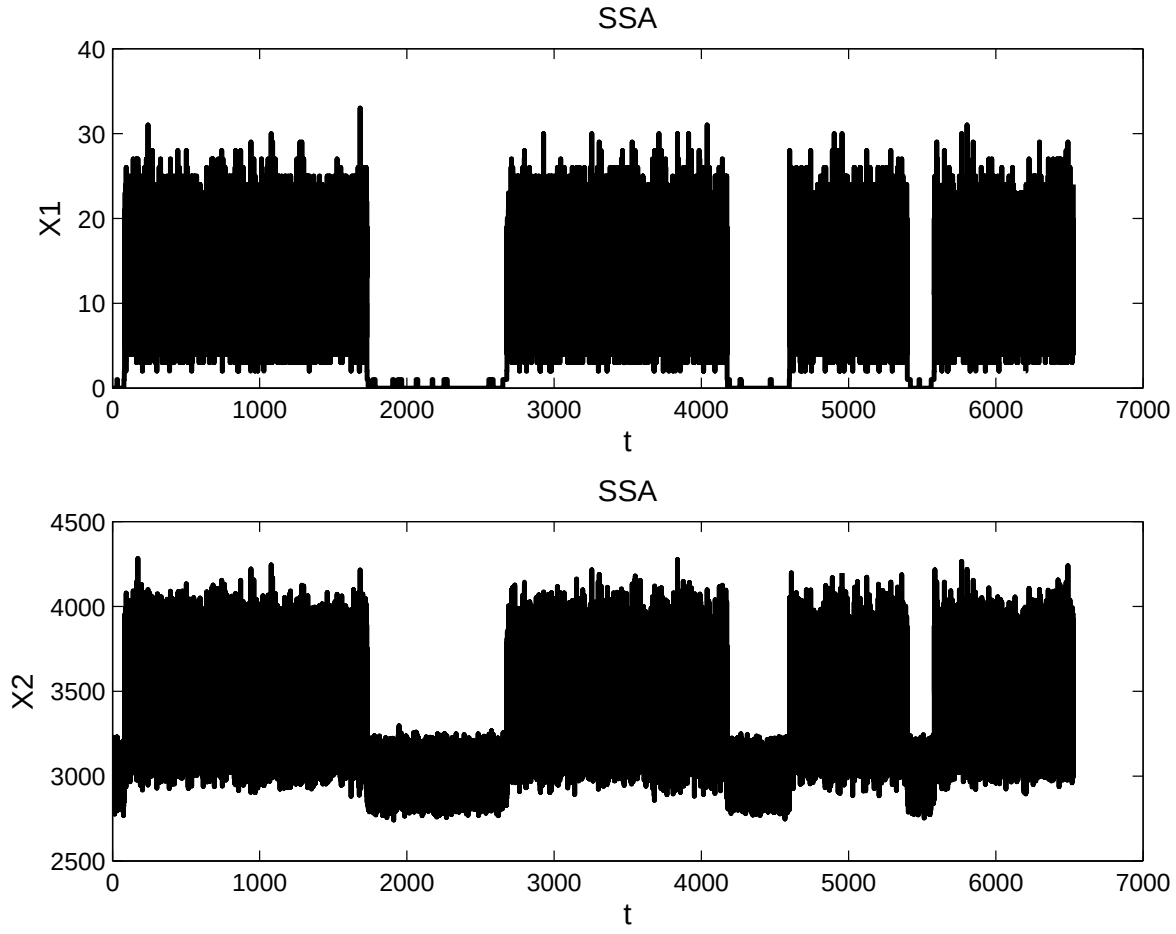

Figure 8.2.1: Bistable System: The trajectory of X_1 and X_2 produced by the SSA.

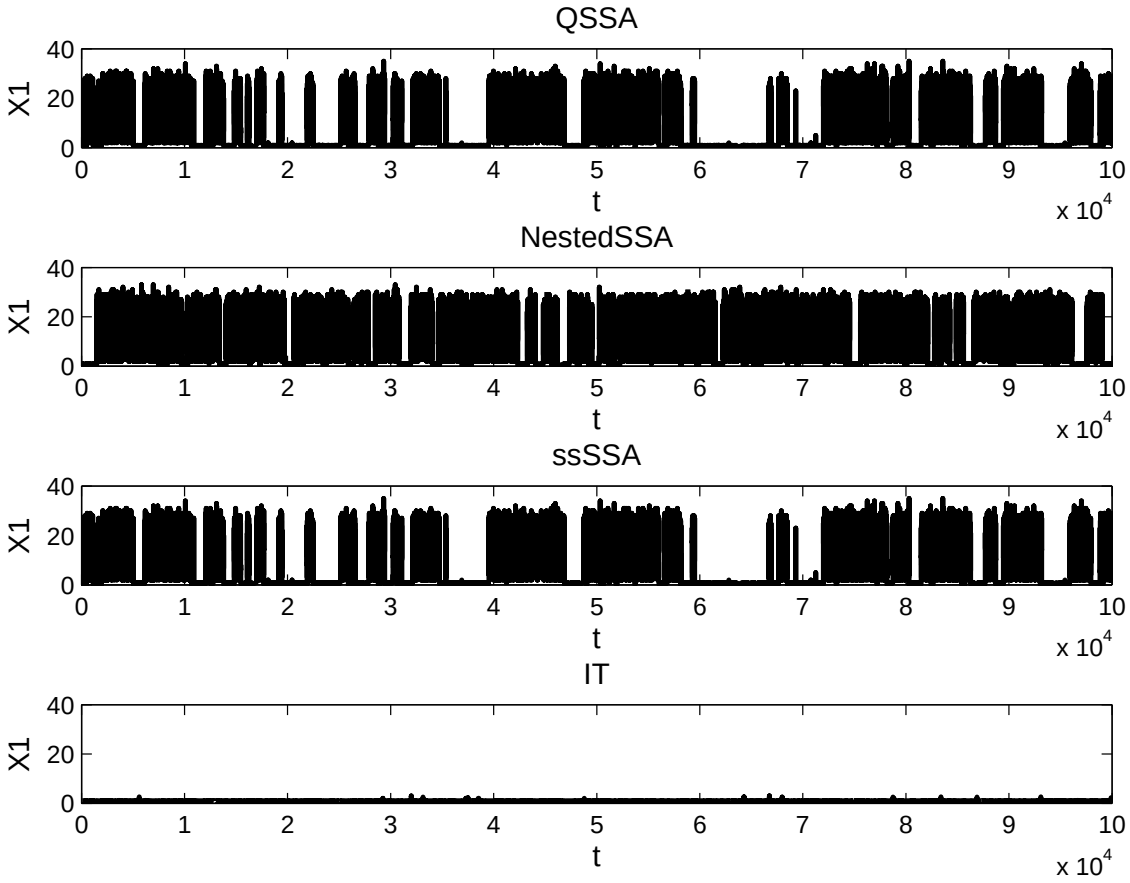

Figure 8.2.2: Bistable System: The trajectory of X_1 produced by the approximation methods. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

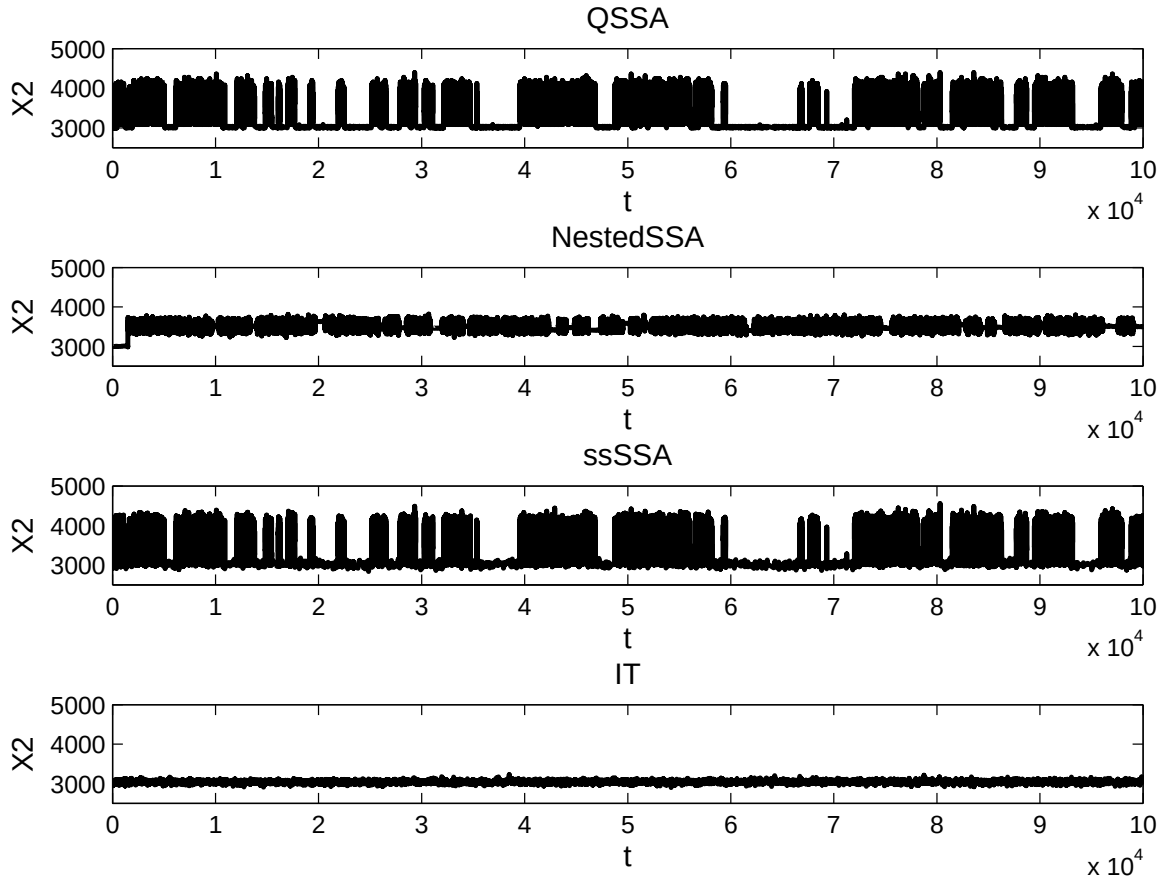

Figure 8.2.3: Bistable System: The trajectory of X_1 produced by the approximation methods. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

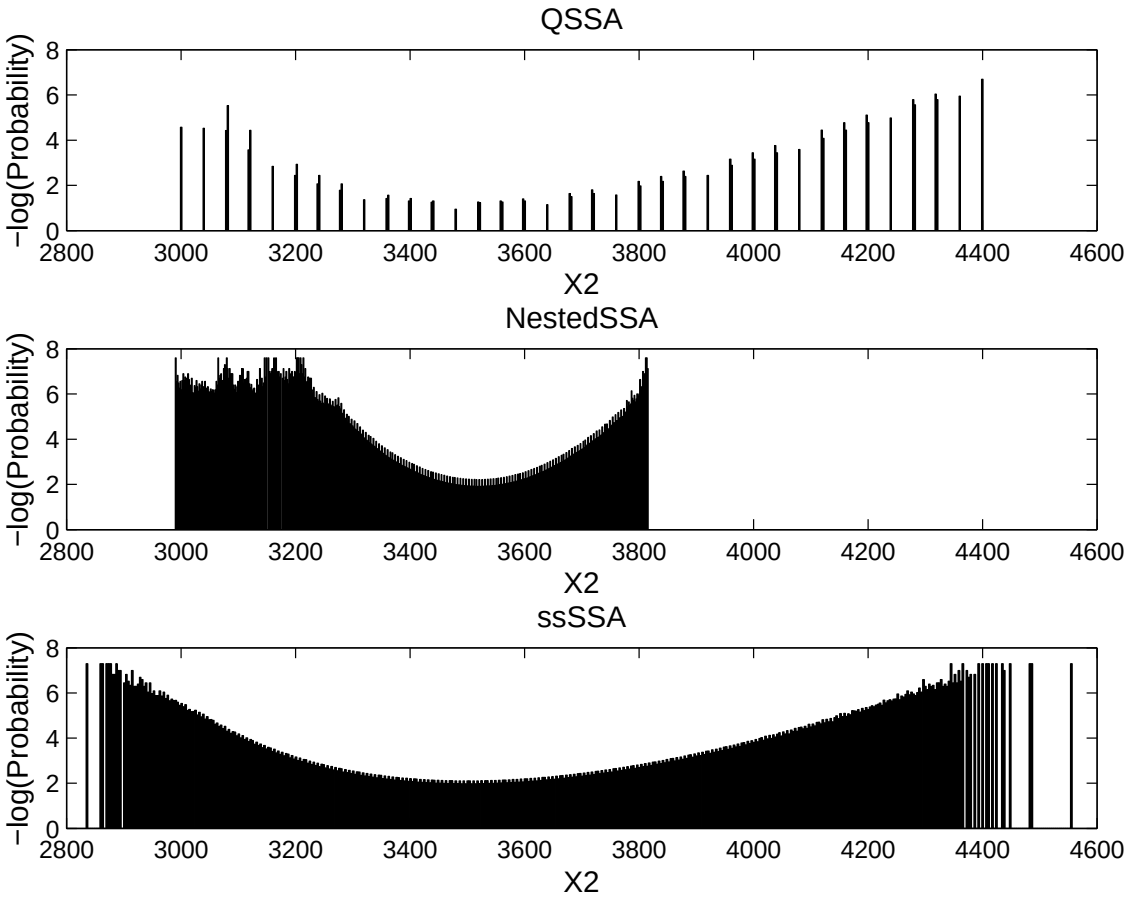

Figure 8.2.4: Bistable System: The negative base ten logarithm of X_2's distribution produced by the QSSA algorithm, slow scale SSA and the nested SSA. (ssSSA refers to the slow scale SSA)

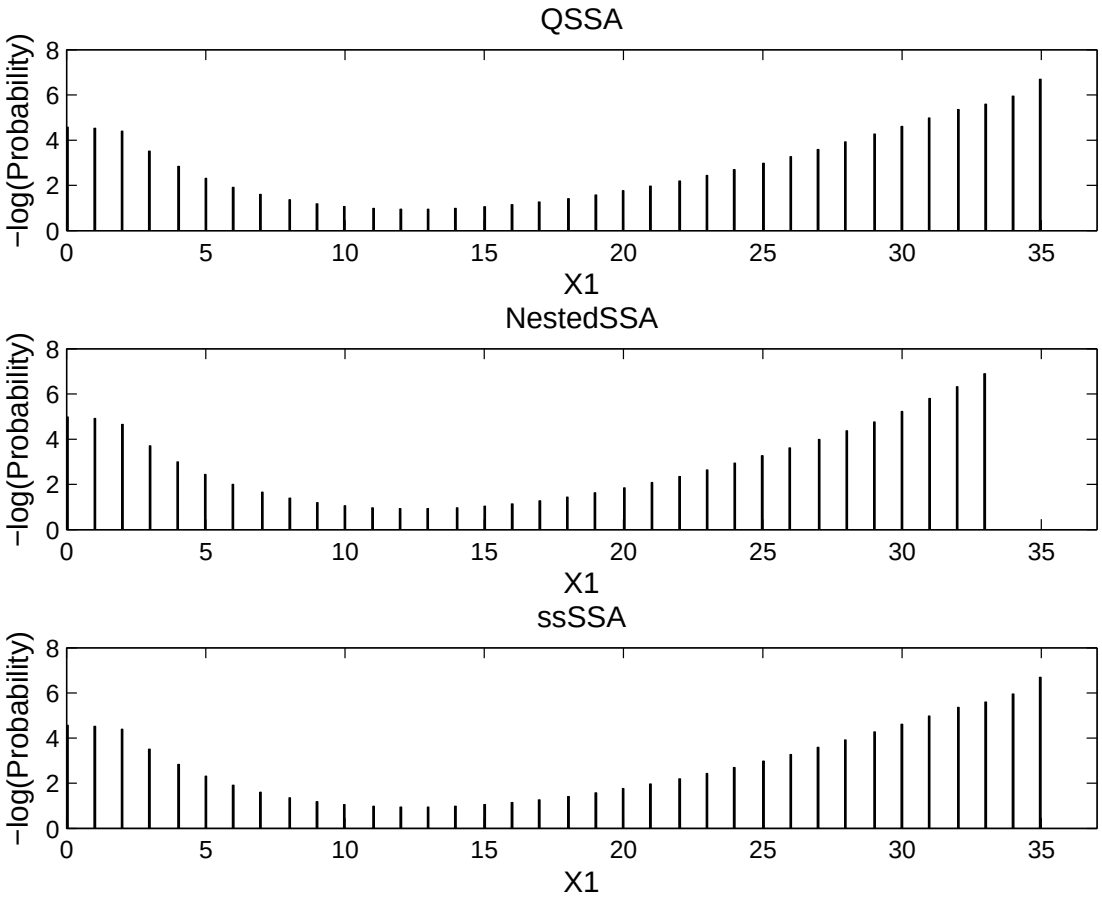

Figure 8.2.5: Bistable System: The negative base ten logarithm of X_1's distribution produced by the QSSA algorithm, slow scale SSA and the nested SSA. (ssSSA refers to the slow scale SSA)

Method	Average Time Spent in Lower State for X_1	Average Time Spent in Upper State for X_1
QSSA algorithm	62.96	73.51
Nested SSA	20.90	87.03
Slow Scale SSA	62.96	73.51

Table 8.2.1: Bistable System: The average time X_1 spent in the upper and lower state, approximated by the QSSA algorithm, slow scale SSA and the nested SSA.

throughout the entire simulation. For the QSSA algorithm, there are gaps between the bins because the steady state of the fast species depends on the slow species, implying that only certain values can be obtained through the approximation of the QSSA algorithm. From Figure 8.2.5 we see that the QSSA algorithm, the slow scale SSA and the nested SSA give a similar distribution to the slow species, X_1.

In Table 8.2, we took the lower state to be when $X_1 \leq 2$. The nested SSA's values for the average time spent in either state is very different from the other two approximation methods. From this information we can make an educated guess that the slow scale SSA would give the best approximation of the system because we believe that the QSSA algorithm did not capture the true distribution of the fast species.

Chapter 9

Marginally Stiff Systems

In previous examples there was a clear separation of the time scales in the system. We will now test the four approximation methods to see how well they compare to the SSA when the slow and fast time scales are less distinct. For all of the following examples the SSA is computationally fast, however there is still a small magnitude of stiffness occurring in the systems. Our goal with these following examples is to see which methods fail and succeed under these weak stiffness conditions. We aim to have a definitive decision on what approximation method operates best under these conditions.

9.1 Fast Reversible Dimerization

Using the system from the Example 1 (chapter 4) we changed the reaction rates to make the system less stiff. The new reaction rates are

$$c_1 = 10^{-3},\ c_2 = 0.2,\ c_3 = 0.02,\ c_4 = 0.004.$$

Here the system is stiff for the approximately half of the simulation as the species pass through a transient. We will use the approach of a large fixed tau, $\tau = 80$, for the implicit tau leaping method, to ensure that the implicit tau leaping method has a faster CPU time than the SSA. For the nested SSA we took $N = 1$ and $T_0 = 0.01$ and $T_f = 0.01$. The CPU times for this example are in Table 9.1.1. We see in Table 9.1.1 that the QSSA algorithm is the fastest approximation method for this example. The nested SSA and the implicit tau method were unable to produce accurate results of the dynamics of the system as seen in Figure 9.1.1, which shows the mean of X_1. If we were to change the parameter values of the nested SSA to have better results this would result in a higher CPU time than that of the SSA. As a result the nested SSA and the implicit tau are unable to handle a system where the time scales are so close. In Figure 9.1.2, showing the error of the approximation of the mean for X_2 and Figure 9.1.3, showing the mean of X_3, we see that the QSSA algorithm and slow scale SSA did not change much compared to the stiffer system

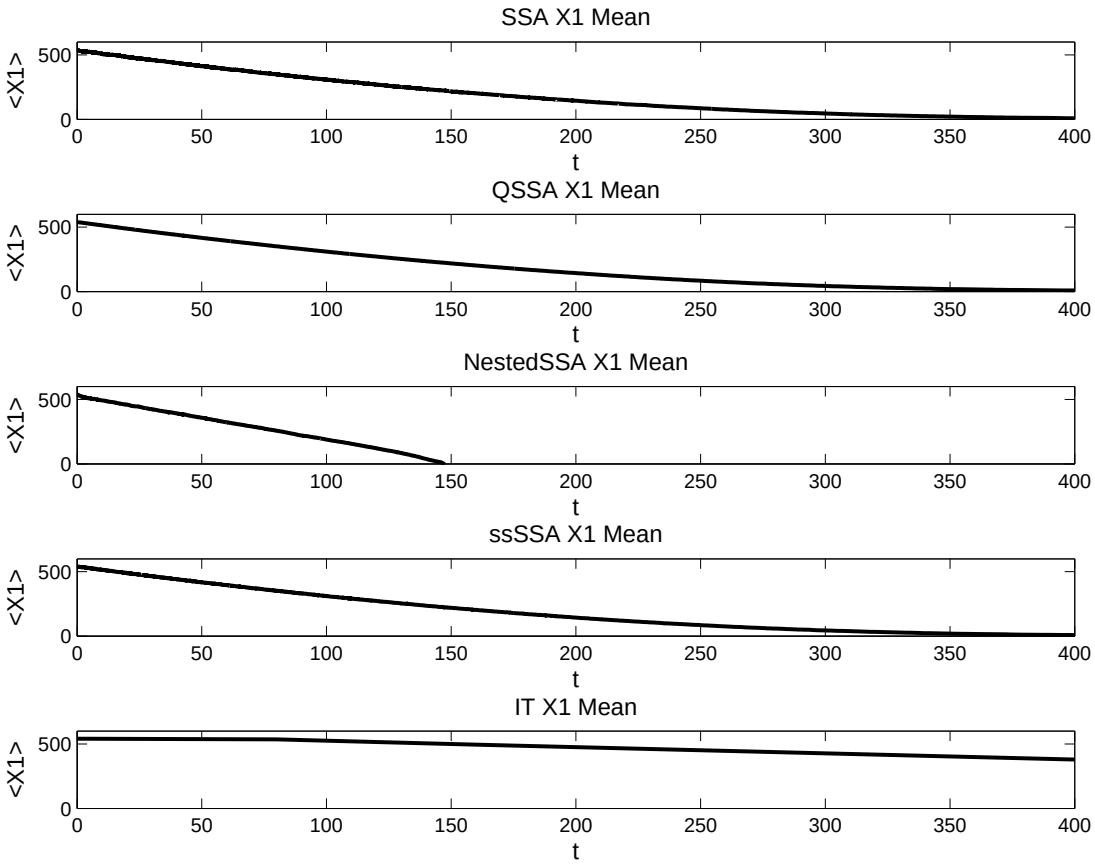

Figure 9.1.1: Marginally Stiff Fast Reversible Dimerization: The approximation of X_1's mean made by all the methods. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

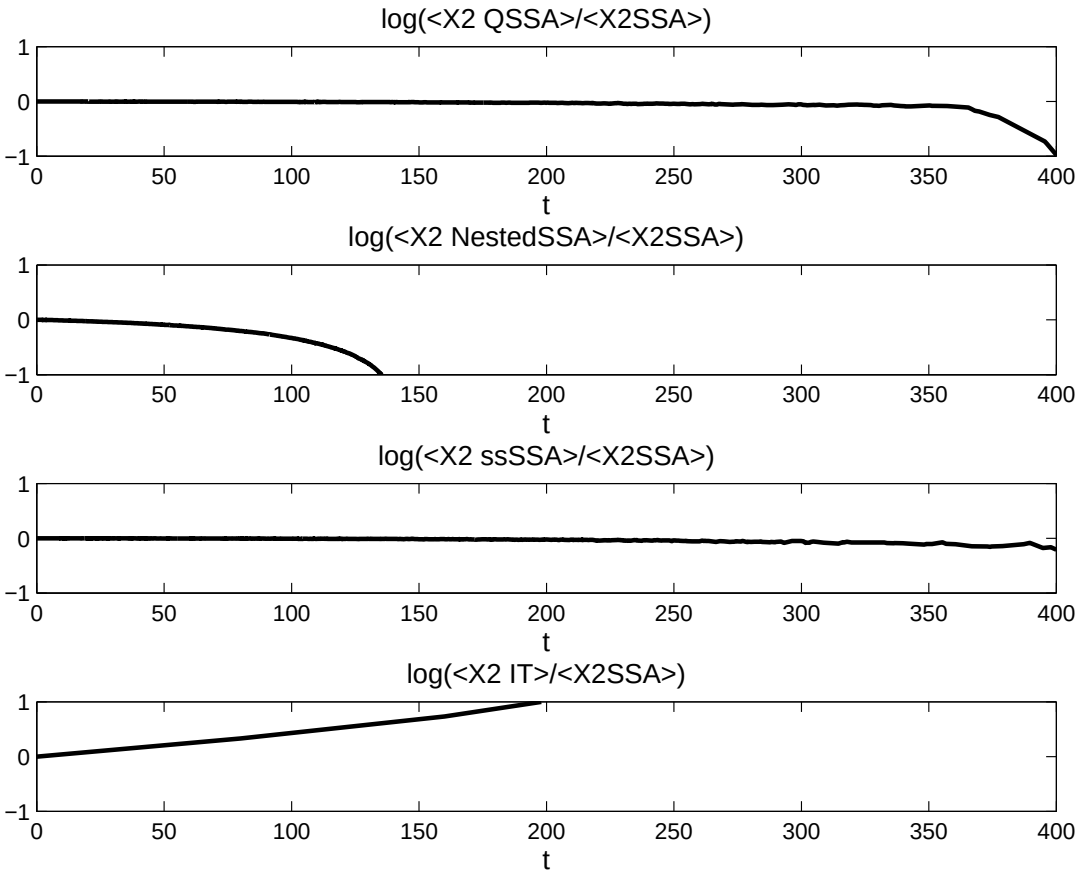

Figure 9.1.2: Marginally Stiff Fast Reversible Dimerization: The base ten logarithm of the approximation of X_2's mean by the approximation methods over the approximation of X_2's mean by the SSA. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

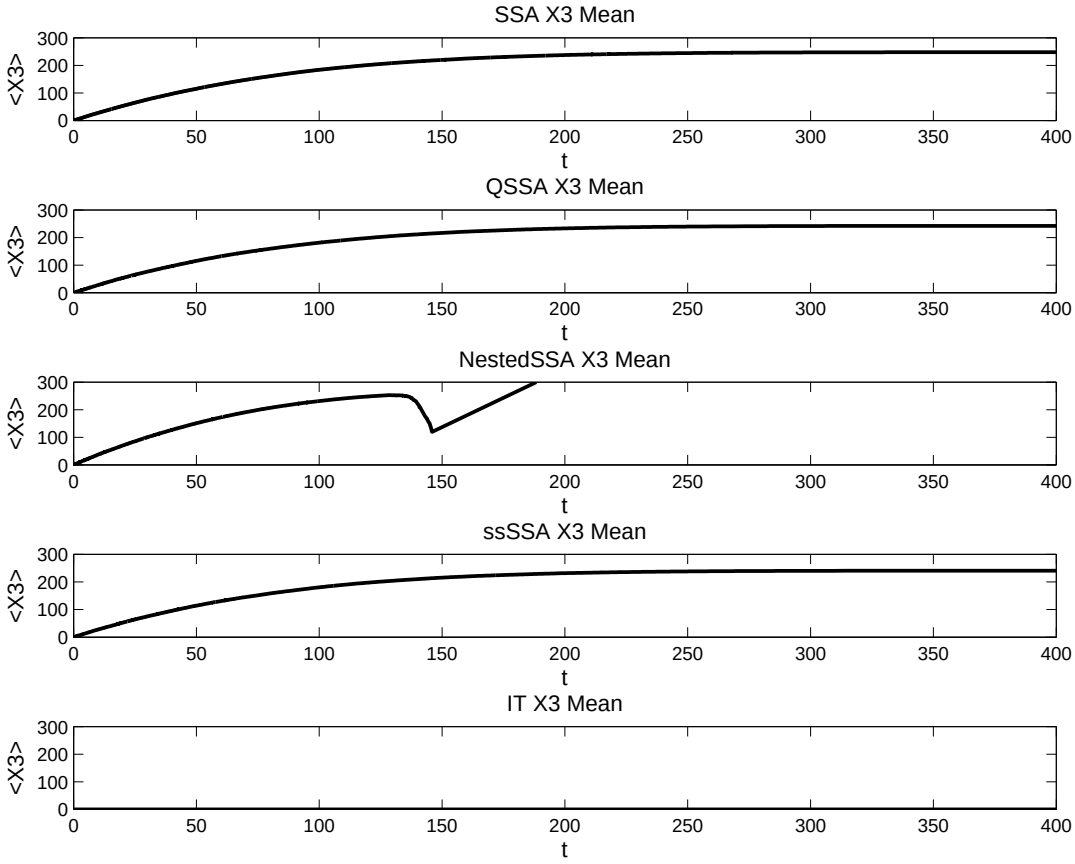

Figure 9.1.3: Marginally Stiff Fast Reversible Dimerization: The approximation of X_3's mean made by all the methods. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

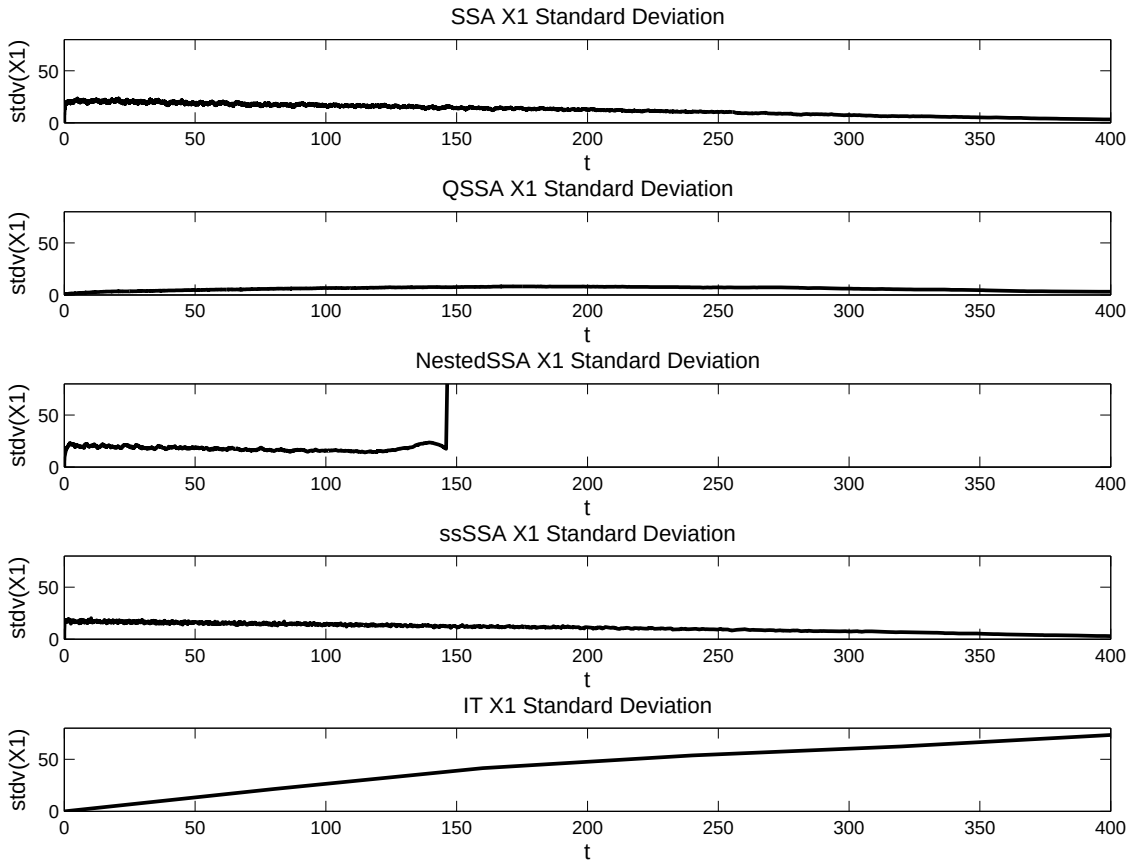

Figure 9.1.4: Marginally Stiff Fast Reversible Dimerization: The approximation of X_1's standard deviation made by all the methods. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

Method	CPU Time (sec.)
SSA	0.3281
QSSA algorithm	0.0156
Nested SSA	0.2031
Slow Scale SSA	0.0313
Implicit Tau	0.2813

Table 9.1.1: Marginally Stiff Fast Reversible Dimerization: CPU times for the all the methods for 400 time units, for a single sample path. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2.

Method	CPU Time (sec.)
SSA	0.4063
QSSA algorithm	0.0313
Nested SSA	0.1719
Slow Scale SSA	0.4062
Implicit Tau	0.3438

Table 9.2.1: Marginally Stiff Network of Isomerizations: CPU times for all the methods for 3 time units, for a single sample path. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2.

in their approximation of the less stiff system. In Figure 9.1.4, showing the standard deviation of X_1, we notice that the QSSA algorithm was unable to approximate the variance of the fast species correctly, similar to the results of the stiffer system. We conclude that the slow scale SSA is still the best approach to use for this example as an approximation method.

9.2 Network of Isomerizations

The new reaction rates for Example 2 (chapter 5) are

$$c_1 = 2.5,\, c_2 = 5\, c_3 = 1,$$

$$c_4 = 1,\, c_5 = 5\, c_6 = 2.5.$$

For the nested SSA we took $N = 1$, $T_f = 1/7500$ and $T_0 = 0$. For the implicit tau leaping method we took $\tau = 0.1$. We see in Table 9.2.1 that the QSSA algorithm is the fastest approximation method for this example. From Table 9.2.1 we see that

Method	CPU Time (sec.)
SSA	0.0313
QSSA algorithm	0.00004
Nested SSA	0.0156
Slow Scale SSA	0.0468
Implicit Tau	0.0156

Table 9.3.1: Marginally Stiff Fast Species Acting as a Catalyst: CPU times for all the methods for 0.1 time units, for a single sample path. The type of machine that was used for these simulations was a sun fire x4600 with 32G of RAM and 4 dual-core AMD Operton chips (8 processors) at 2.6GHz running OpenSuSE 10.2

the slow scale SSA has a similar CPU time to that of the SSA. In Figure 9.2.1, which shows the error of the approximation of the mean of X_4, we see that the implicit tau method is inaccurate in approximating the mean of X_4. The nested SSA approximates the distribution of the species X_1 the best, as seen in Figure 9.2.2. In Figure 9.2.3, showing the error in the standard deviation of X_2, we see that the QSSA algorithm does not approximate the standard deviation of X_2 very well compared to the other approximation methods. Since the slow scale SSA had a much slower CPU time than that of the nested SSA, the best method to approximate this system would be the nested SSA as it was able to produce accurate results in the fastest time.

9.3 Fast Species Acting as a Catalyst

The new reaction rates example 3 (chapter 6) of a fast species acting as a catalyst are

$$c_1 = .25,\ c_2 = .25\, c_3 = 0.0005.$$

For the nested SSA we took $N = 1$, $T_f = 0.002$ and $T_0 = 0$. For the implicit tau leaping method we took $\tau = -50\frac{\log(r_1)}{c_3 x_1 x_2}$. For this example the system is stiff throughout the entire simulation. We see in Table 9.3.1 that the QSSA algorithm is the fastest approximation method for this example. From Table 9.3.1 we notice that the slow scale SSA was not able to run faster than the SSA at times and there is no way to improve the slow scale SSA's CPU time. Figure 9.3.1 shows the standard deviation of X_1, indicating that the slow scale SSA was not able to approximate the standard deviation of the fast species very well, unlike in the more stiff example in chapter 6. The QSSA algorithm still gives a poor approximation of the standard deviation for the fast species for this system. The implicit tau slightly improves in its approximation of the standard deviation. However, the implicit tau's approximation of the standard deviation looks linear compared to that of the SSA's. Figure 9.3.2 shows us the distribution of X_2 at 0.1 time units, which shows

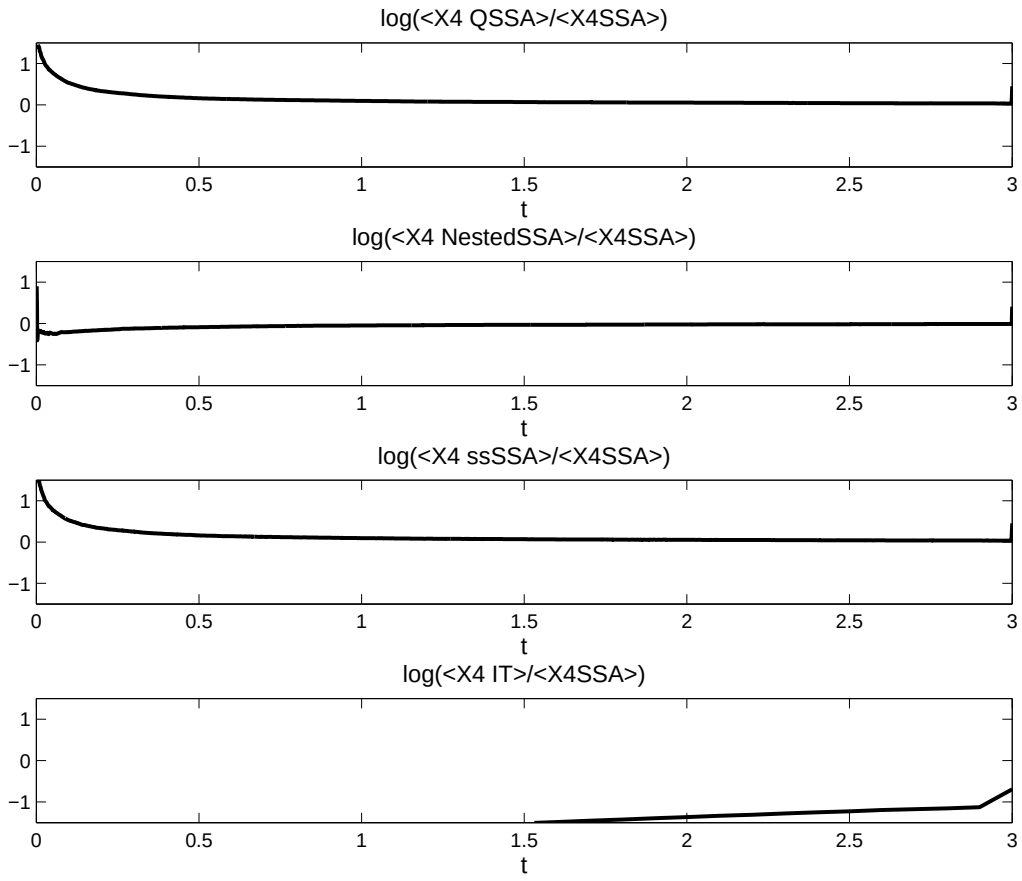

Figure 9.2.1: Marginally Stiff Network of Isomerizations: The base ten logarithm of the approximation of X_4's mean by the approximation methods over the approximation of X_4's mean by the SSA. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

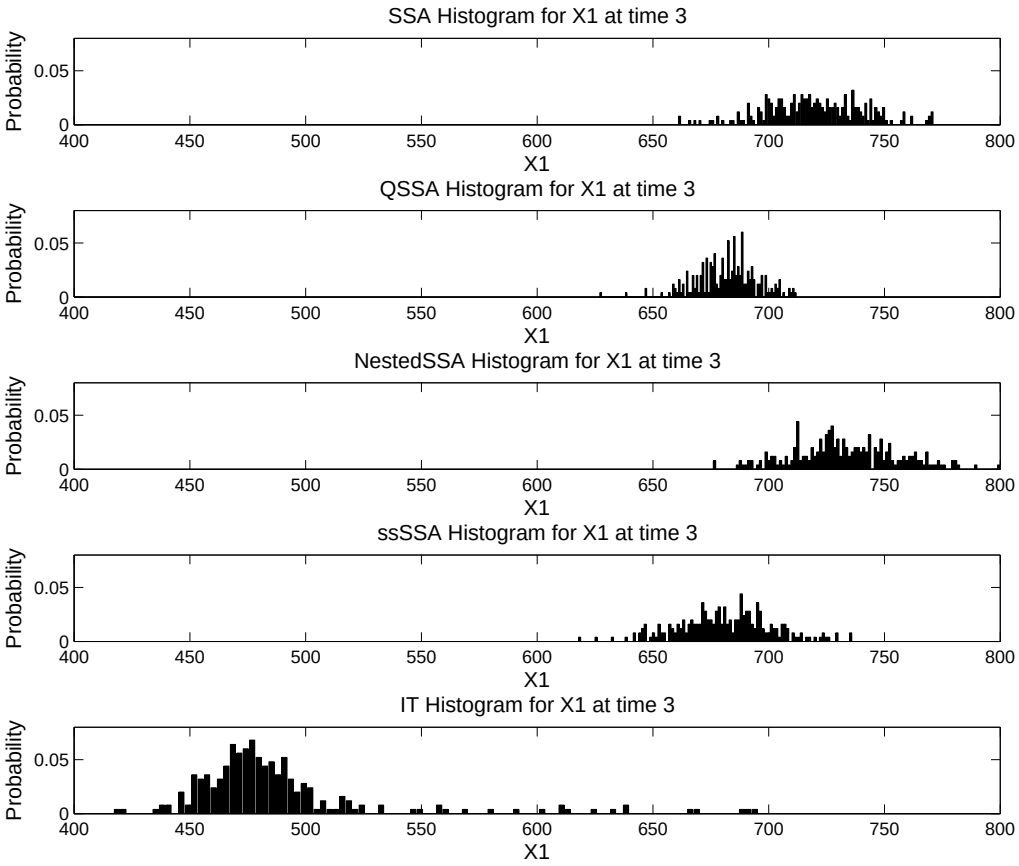

Figure 9.2.2: Marginally Stiff Network of Isomerizations: The histogram of X_1 for all the methods at 3 time units. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

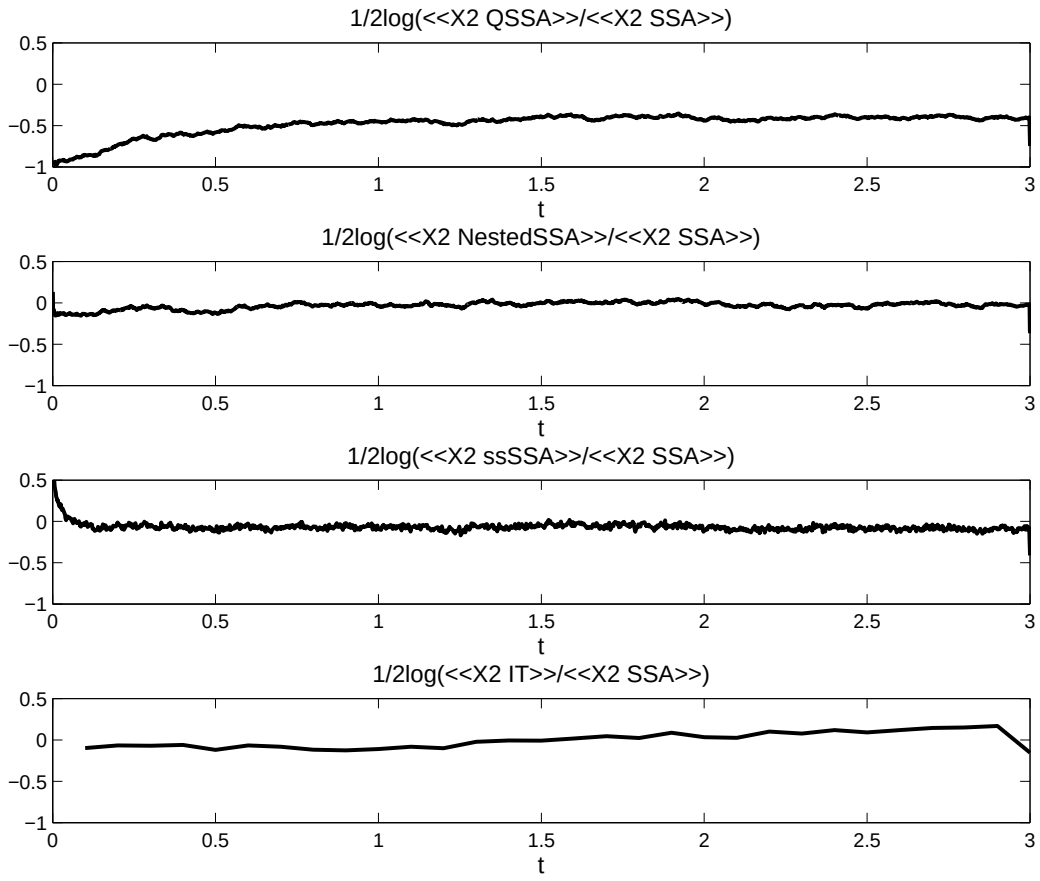

Figure 9.2.3: Marginally Stiff Network of Isomerizations: The base ten logarithm of the approximation of X_2's standard deviation by the approximation methods over the approximation of X_2's standard deviation by the SSA. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

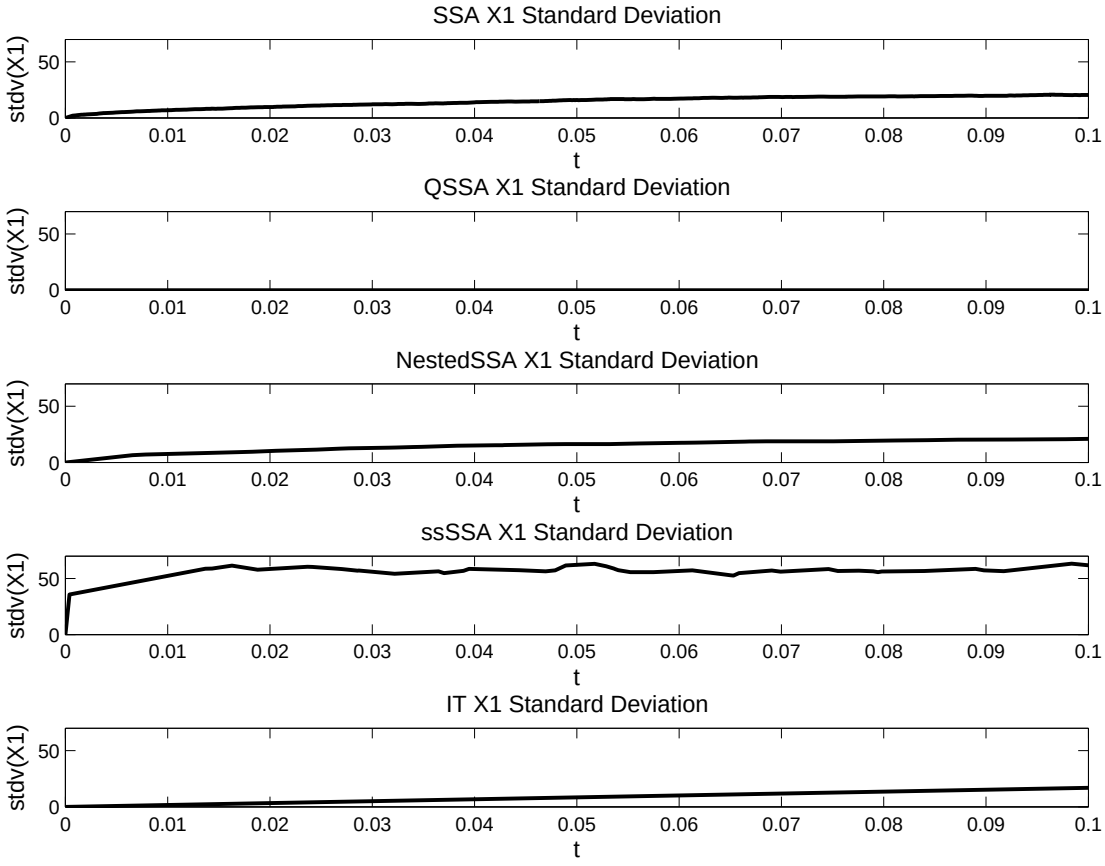

Figure 9.3.1: Marginally Stiff Fast Species Acting as a Catalyst: The approximation of X_1's standard deviation made by all the methods. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

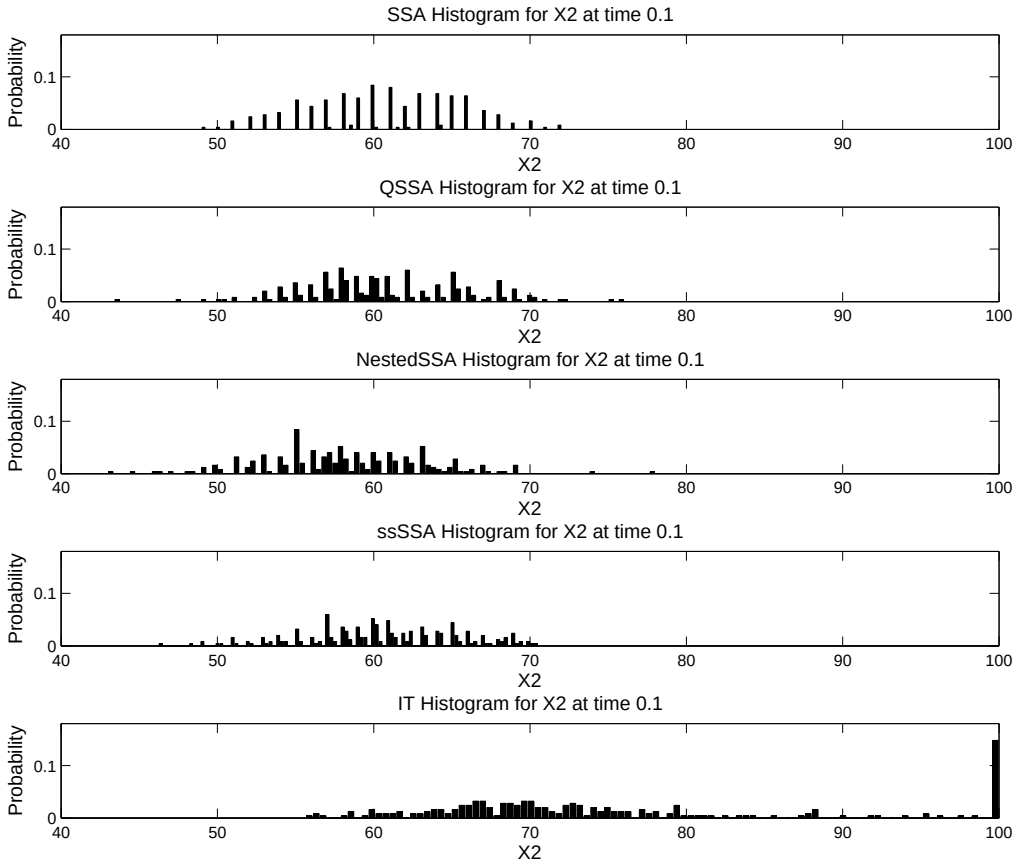

Figure 9.3.2: Marginally Stiff Fast Species Acting as a Catalyst: The histogram of X_2 for all the methods at 3 time units. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

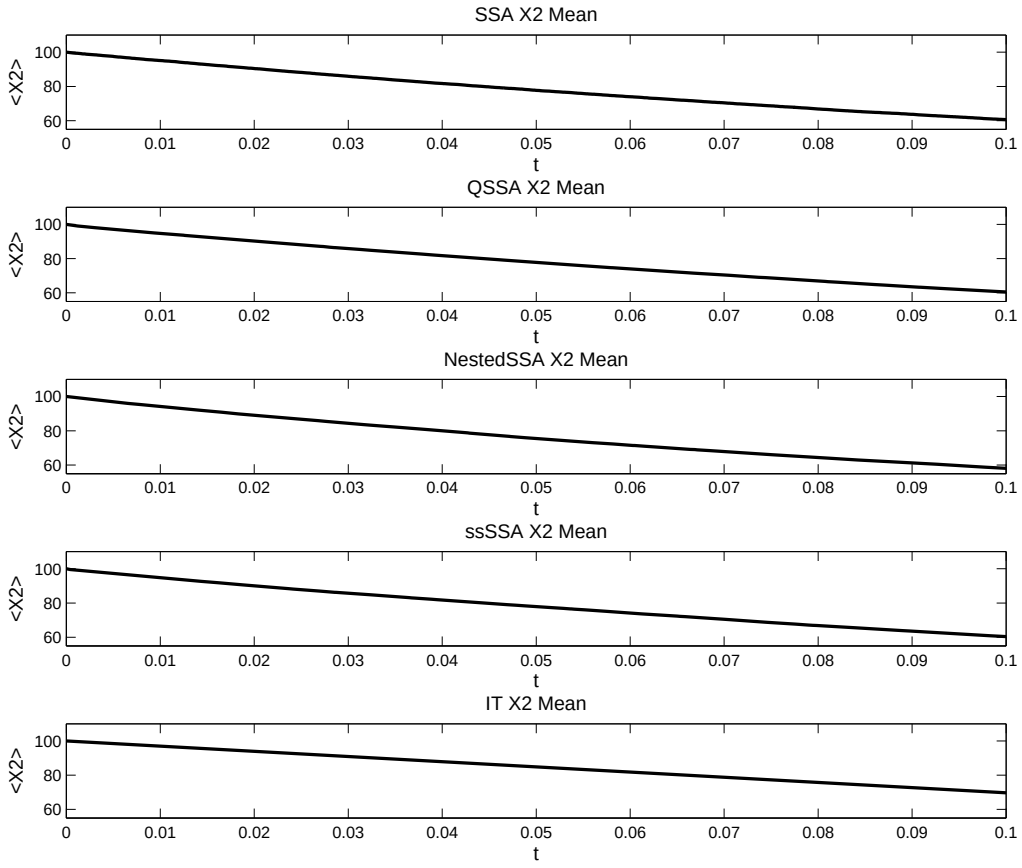

Figure 9.3.3: Marginally Stiff Fast Species Acting as a Catalyst: The approximation of X_2's mean made by all the methods. (IT refers to the implicit tau leaping method and ssSSA refers to the slow scale SSA)

that the QSSA algorithm gives the best approximation to the distribution. However, in Figure 9.3.3, which shows the mean of X_2, we see that the nested SSA's approximation of X_2 is as good as the QSSA algorithm's approximation. We conclude that the nested SSA is the best approximation method for this example, since the nested SSA's approximation of the mean is similar to the QSSA algorithm's, which best matched that of the SSA's, and the nested SSA was the best method to approximate the fast species standard deviation.

Chapter 10

Conclusion and Comments

The ensembles were chosen sufficiently large to show the behavior of the methods without making the simulations unnecessarily burdensome. We did some test (not shown) and found that the behaviour of the mean and variance did not change after running larger ensemble sizes. Since all of our results for each algorithm were from the ensemble size of 250, the mean and variance of each algorithm was computed on an even playing field.

From the results of the stiff examples it is evident that the slow scale SSA is typically the best approximation method to use. The slow scale SSA is highly accurate and extremely fast in comparison with the other methods. The other approximation methods were as accurate as the slow scale SSA but this was for only a few examples. The QSSA algorithm was as accurate as the slow scale SSA for example 1, example 4, and example 5. The nested SSA was as accurate as the slow scale SSA for example 2 and example 3. The implicit tau was accurate as the slow scale SSA for example 4. The slow scale SSA did fall behind the QSSA algorithm for the oscillatory example but the slow scale SSA still produced reliable results.

However, from the marginally stiff examples we saw that the slow scale SSA's CPU time for two of three examples was similar to the SSA's CPU time. In terms of accuracy, the slow scale SSA did not change compared to the stiff examples, with the exception of the slow scale SSA's over approximation of a species standard deviation in section 9.3. From the results of the marginally stiff systems we found that the nested SSA was typically the best approximation method to use. In the next few paragraphs we will summarize what we have learned about these different approximation methods for stiff chemical systems.

The nested SSA was not as successful as the slow scale SSA due to the parameter choices for each example: it is difficult to find parameters to produce the best results. In example 1, the choice of parameters had a large impact on the CPU time and the results. As we saw in the examples, taking only one inner loop is sufficient as multiple loops add to the CPU time, without significant improvement to the

results. As for finding a suitable time to truncate for the transient the question becomes: where does one draw the line? The truncation can be as straightforward as setting $T_0 = 0$ or as intricate as making T_0 a function of the system's time scales to determine the proper transient time. As for the value of T_f, if the value is too short, the approximation is terrible as seen in [33] and in section 4.4. If the value is taken to be longer than what is sufficient then it just adds to the CPU time of the nested SSA, as mentioned in section 4.4.

In example 2 we saw that for a given T_f the one fast system was unable to consistently reach equilibrium. The point is that with the nested SSA there are so many possible combinations of parameters one is unsure before the simulation how good the approximation is going to be. However, there is a benefit to parameter choices as it allows one to control the accuracy and CPU time of the nested SSA. On the other hand, if the information is accurate but the CPU time for the nested SSA is longer than the SSA one should just use the SSA as it is an exact method. The approach that the nested SSA has taken is good because it is fairly easy to implement, but not knowing the optimal choice of parameters is a significant problem. To improve the nested SSA one may want to look into ways of determining appropriate values for the parameters of the inner SSA to produce the best results.

The QSSA algorithm was fairly easy to implement as well because the majority of the work was determining the steady state of the virtual fast process. The approach of using the delta distribution for the fast species approximation in the fast system decreased the CPU time immensely. The QSSA algorithm had CPU times similar to and faster than that of the slow scale SSA. At times the QSSA algorithm was as accurate as the slow scale SSA, as in the fast reversible dimerization example and parts of the other examples, but failed in other aspects when it came to other examples such as the fast species acting as a catalyst example as well as parts of the network of isomerizations. The main reason it failed was due to the approximation of the species variance because the variance plays an important role when one looks at a single trajectory of the system. If the variance differs from that of the SSA then for that single trajectory one is bound to have a different outcome for the QSSA algorithm. In [13], the authors do take a probabilistic approach to the QSSA algorithm to fix this problem with the variance. The only reason we did not look at this probabilistic case is because the probabilistic approach to the QSSA algorithm is fairly similar if not identical to that of the slow scale SSA. There are a few minor differences in assumptions in [13] as compared to [17] but in the end the assumptions state the same thing: the fast system must reach a steady state.

The implicit tau leaping method allows many different ways to select the size of tau. There have been recent papers discussing more efficient ways of selecting this tau, however in this paper we decided to focus on the approaches taken in the paper [20] along with a selection of tau similar to the QSSA algorithm and slow scale SSA. Using the choice of a large fixed tau produced the best results as well as the best CPU time for the examples of the fast reversible dimerization and the network of

isomerizations. Since there may be better ways of selecting tau, we cannot conclude that the implicit tau leaping method is inaccurate for all of our test results. If we truly wanted to see how accurate the implicit tau leaping method is we would have to compare all of the different tau leaping methods and ways of selecting tau. This in itself could be another study.

Another challenging aspect of the implicit tau leaping method is the fact that one has to use a numerical solver to solve the implicit set of equations, leading to a much higher CPU time in some cases. Another slight problem that we encountered with the implicit tau leaping method was that when the concentration of a species was near zero the numerical solver occasionally produced negative values. This is a problem with the choice of numerical solver, not the implicit tau leaping method. We used the numerical solver *fsolve* in MATLAB and to work around this problem of negative values we used a transformation when solving. The transformation we used was $Z_i^2 = X_i$, for all species i. Overall, we found that on occasion the implicit tau leaping method was as accurate as the slow scale SSA. However, with improved ways of selecting tau one may produce more accurate results that are consistently reliable for all test cases. However, I believe that there is no way to decrease the CPU time of the implicit tau method to be at the same level as the slow scale SSA.

The slow scale SSA produced consistently reliable results that resembled the SSA for all of the stiff examples. In the oscillatory example, there was a slight difference in the variance throughout the entire simulation but the slow scale SSA made up for it in reproducing the phase, mean and frequency of the SSA. The toughest aspect about implementing the slow scale SSA is finding the distribution of the fast species. Solving the moment equations to find the moments for the normal approximation requires a similar effort to finding the steady state in the QSSA algorithm. However, at times one has to make an assumption about certain moments to have moment closure, as seen in the example of the fast reversible dimerization. As we saw in the example of the fast reversible dimerization the assumption that the third central moment was zero did not have any effect on the final outcome of the approximation. Overall the slow scale SSA was fairly straight forward to implement once the distribution was found and the slow scale SSA produced reliable consistent results for our test examples.

Appendix

Appendix A

Fast Fourier Transform

The fast Fourier transform (FFT), is a computational method that efficiently computes the discrete Fourier transform (DFT) of a series of data samples. The discovery of the FFT produced major changes in the computational aspect of spectral analysis and other fields [35]. We used the FFT for the spectral analysis of the oscillatory systems produced by our approximation methods. The DFT is defined as

$$A_n = \sum_{k=0}^{N-1} X_k \exp^{-2\pi i n k/N},$$

where N is the number of data points, $n = 0, \cdots, N-1$. X_k is the k^{th} data point and $i = \sqrt{-1}$. A_n represents the n^{th} coefficient of the DFT. The FFT computes the DFT by sequentially combining progressively larger weighted sums of data. The FFT computes the DFT in two ways: decimation in time and decimation in frequency. The decimation of time approach begins by splitting the data points into odd and even data points, i.e. $Y_n = X_{2n}$ and $Z_n = X_{2n+1}$. Then one splits the Fourier transform into the first $N/2$ data points and the last $N/2$ data points and the DFT can be computed since X_k can be obtained from the DFT of Y_k and Z_k. The decimation of frequency begins with the division of the data points into two separate groups, the first $N/2$ data points and the last $N/2$ data points, i.e. Y_k is the set of first $N/2$ data points and Z_k is the set of last $N/2$ data points. Then one looks at the even and odd number points of the transform, i.e. A_n where we look at the separate cases of n being odd or even. The DFT for an even number of transform points is determined by an $N/2$ point DFT and a combination of the Y_k and Z_k. For an odd number of transform points the transform is determined by another $N/2$ point DFT with a different combination of Y_k and Z_k. For a more in depth derivation of the two ways of applying the FFT refer to [35]. The FFT not only decreases the computational time, it also reduces any round off errors that are associated with the computation of the DFT [35].

Appendix B

Hann Window

To reduce the error, due to discontinuities of periodic extensions, of calculating the spectrum using the FFT the application of a Hann window to the data set is beneficial [36]. The Hann window, $H(x,t)$ is defined [37] as

$$H(x,t) = \begin{cases} 1/2 + 1/2\cos(\pi x/t) & |x| < t \\ 0 & |x| \geq t \end{cases}$$

We chose the Hann window because it best handles background noise when one is trying to calculate the frequency spectrum [38]. *Leakage* is when a non integer number of periods is acquired. Two advantages of the Hann Window is that it has little leakage and it does not effect the accuracy of the frequency spectrum. The downfall of the Hann Window is it does cause some amplitude error. [38]. A Hann Window is shown in Figure B.0.1. In Figure B.0.2 we see a sine wave and then a sine wave with the application of a Hann Window.

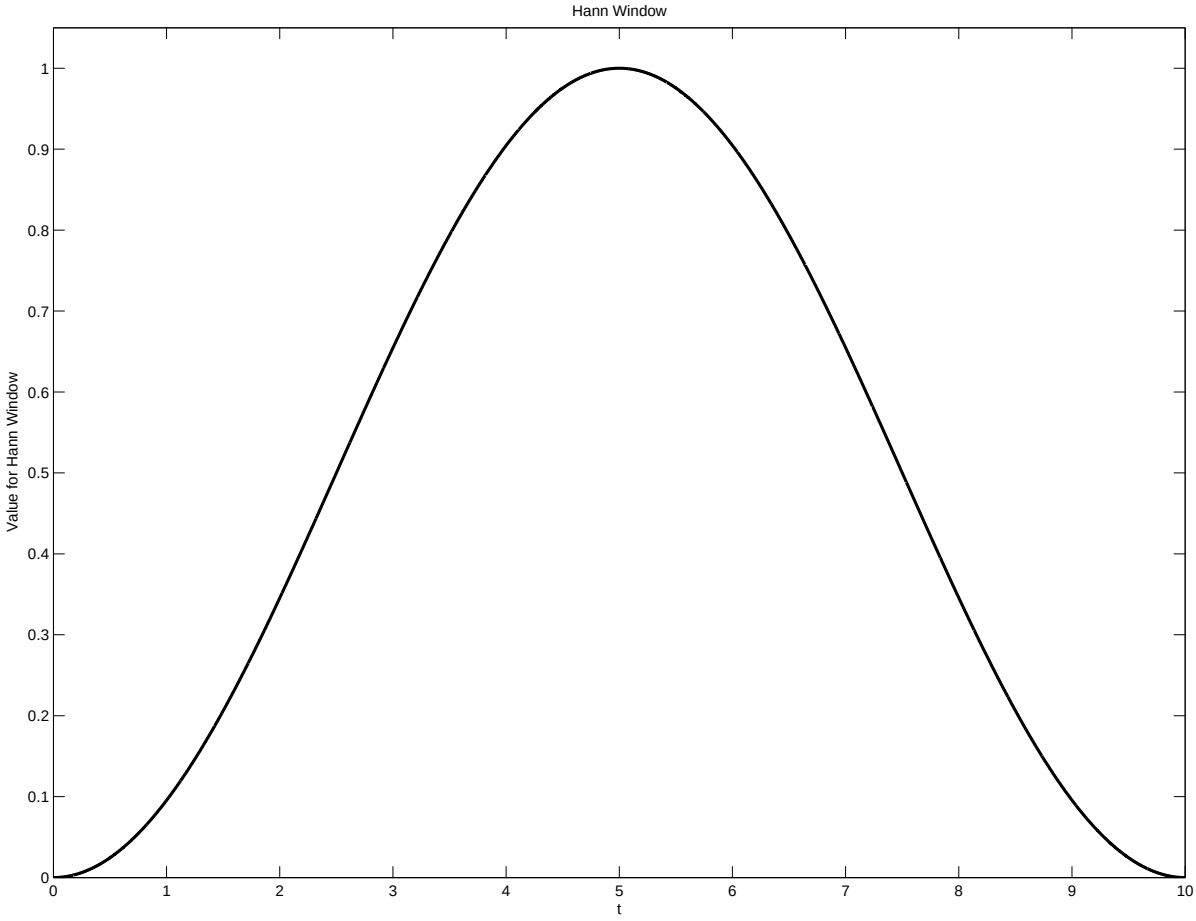

Figure B.0.1: Hann Window: A Hann window for the time interval for $0 \leq t \leq 10$.

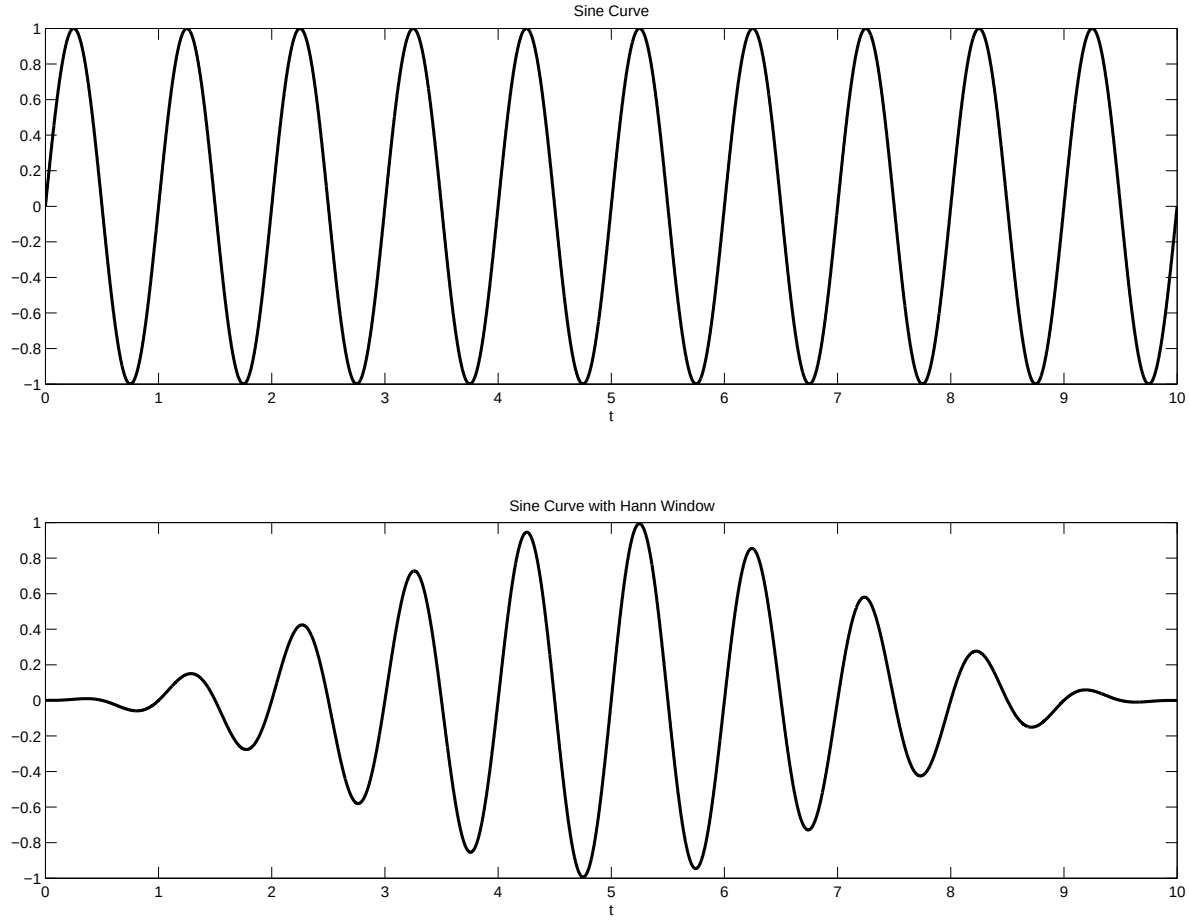

Figure B.0.2: Hann Window: A Hann window applied to the function $\sin(2\pi t)$ for the time interval for $0 \leq t \leq 10$.

Appendix C

Analysis of Oscillations

The approach we have taken for the phase analysis is to consider the mean and variance of a species conditional on the abundance of another species. If an approximation method and the SSA agree in this measure then the two methods are in relatively the same phase. The conditional variance will give us information about the amplitude of the oscillations and whether the methods produce roughly the same amplitude. We will look at a few examples to fully understand the analysis.

C.1 Example 1

Let $t \in [0, 10]$, and $X_1(t) = 2\sin(\pi t)$ while $X_2(t) = \sin(\pi t)$ compared against $\tilde{X}_1(t) = \sin(\pi t)$ and $\tilde{X}_2(t) = \cos(\pi t)$. We see in Figure C.1.1 that conditional on X_2, the two sets of curves give the same mean. When we look at Figure C.1.2 we see that the variances differ because the amplitudes differ.

C.2 Example 2

Since the approximation methods we deal with are stochastic, noise from the system will change the phase of the oscillations. Let $t \in [0, 10]$ and take

$$X_1(t) = \begin{cases} \sin(\pi t) & t \in [0, 3/\pi] \\ \sin(3) & t \in [3/\pi, 1] \\ \sin(\pi t - 3) & t \in [1, 18/\pi] \\ \sin(15) & t \in [18/\pi, 6] \\ \sin(\pi t + 15) & t \in [6, 27/\pi] \\ \sin(42) & t \in [27/\pi, 9] \\ \sin(\pi t - 42) & t \in [9, 10] \end{cases}$$

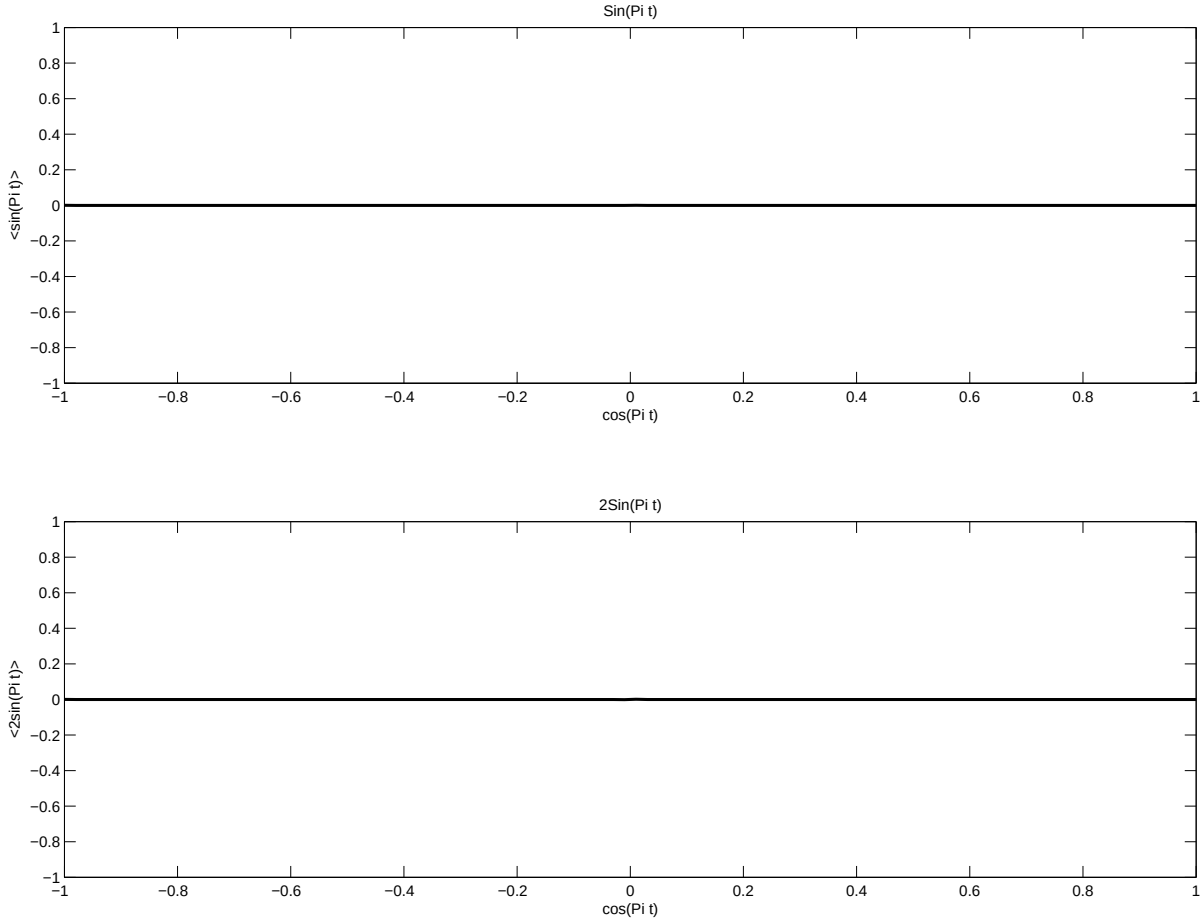

Figure C.1.1: Analysis of Oscillations (Example 1): Oscillation analysis of the mean for given values of a species.

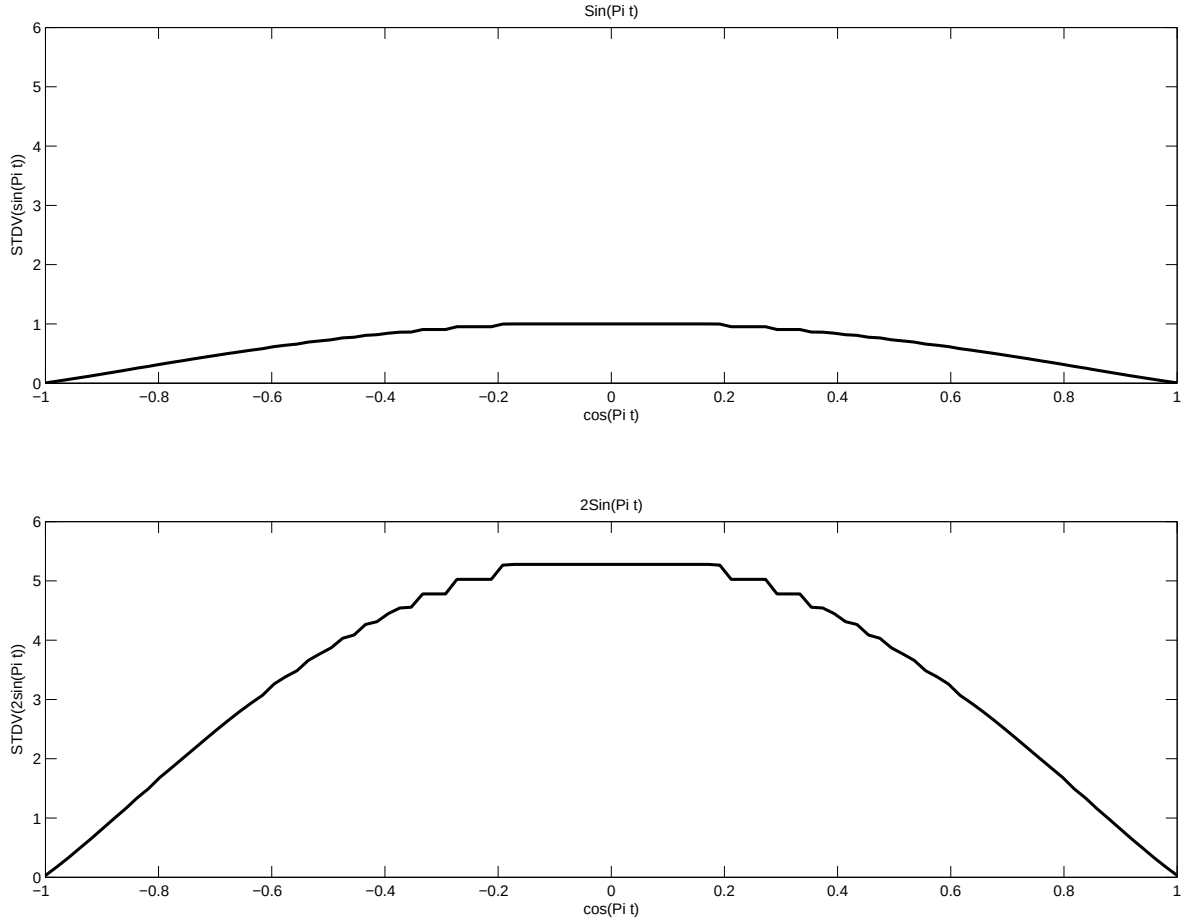

Figure C.1.2: Analysis of Oscillations (Example 1): Oscillation analysis of the standard deviation for given values of a species.

and

$$X_2(t) = \begin{cases} \cos(\pi t) & t \in [0, 3/\pi] \\ \cos(3) & t \in [3/\pi, 1] \\ -\cos(\pi t + 3) & t \in [1, 18/\pi] \\ -\cos(21) & t \in [18/\pi, 6] \\ -\cos(\pi t + 21) & t \in [6, 27/\pi] \\ -\cos(48) & t \in [27/\pi, 9] \\ \cos(\pi t + 48) & t \in [9, 10] \end{cases} .$$

Compare with $\tilde{X}_1 = \sin(\pi t)$ and $\tilde{X}_2 = \cos(\pi t)$. We see in Figure C.2.1 that the mean is different from that of the other pair the "phase" has changed. In Figure C.2.2, we see that the standard deviation is slightly different when the phases have changed.

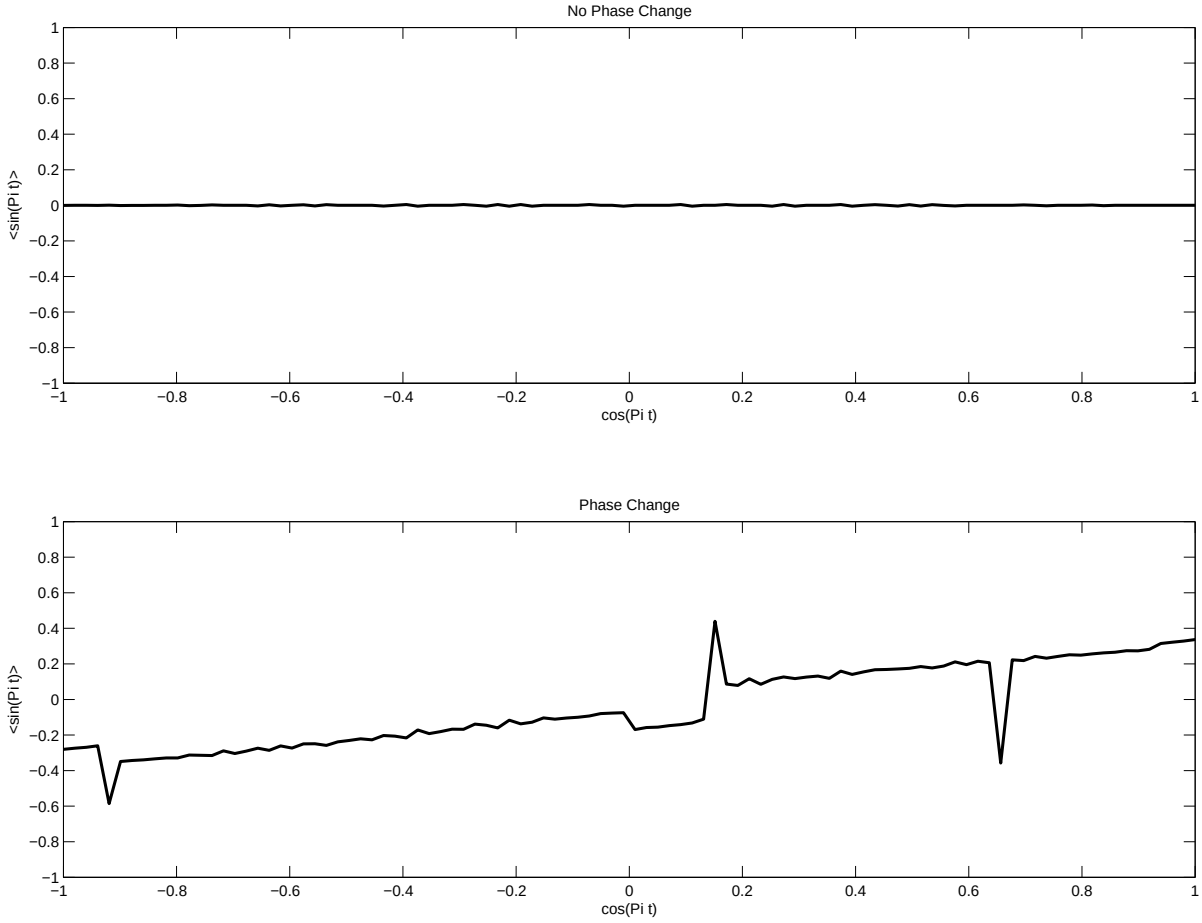

Figure C.2.1: Analysis of Oscillations (Example 2): Oscillation analysis of the mean for given values of a species.

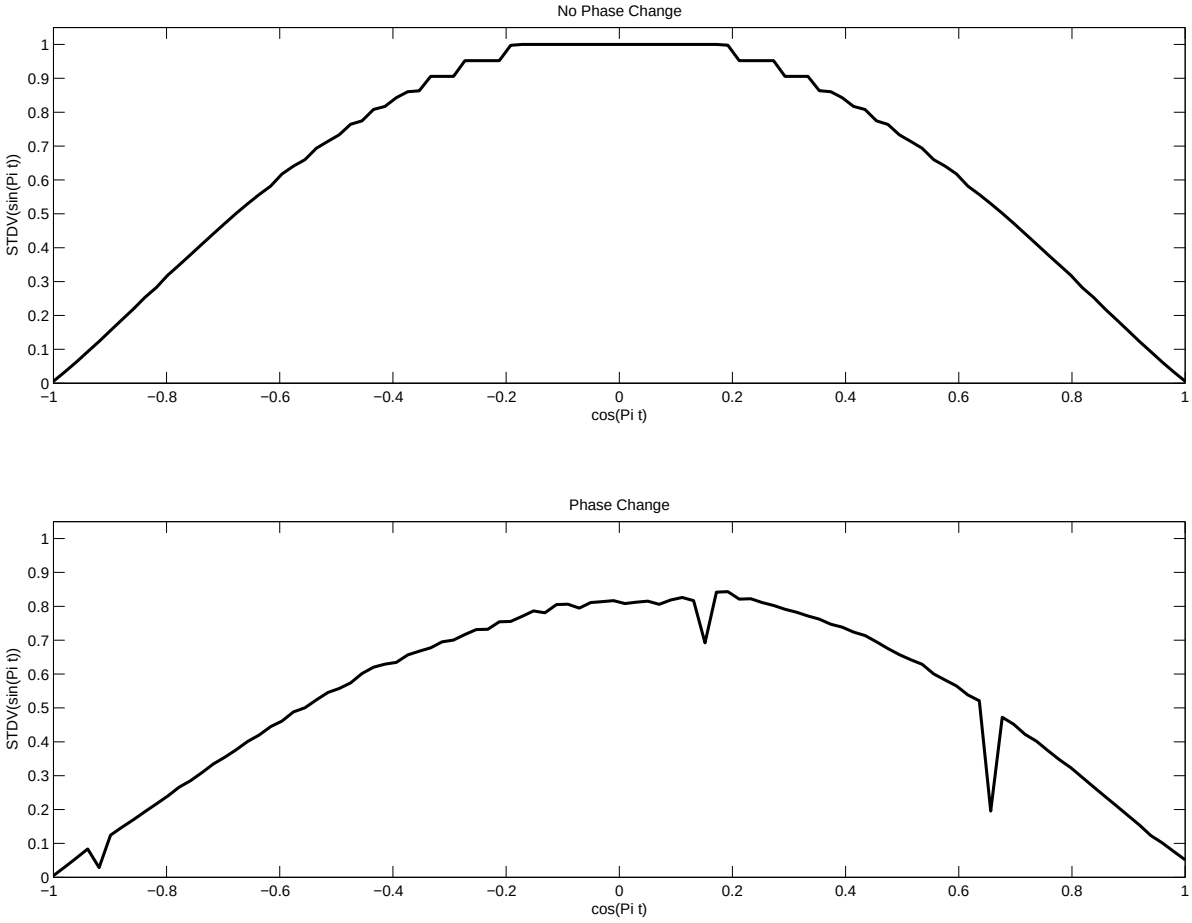

Figure C.2.2: Analysis of Oscillations (Example 2): Oscillation analysis of the standard deviation for given values of a species.

References

[1] A. Alfonsi, E. Cances, G. Turinici, B. Di Ventura and W. Huisinga. Adaptive simulation of hybrid stochastic and deterministic models for biochemical systems. *ESAIM: Proceedings*, 14:1–13, 2005.

[2] N. Johnson and J. Mueller. Updating the accounts: Global mortality of the 1918-1920 Spanish influenza pandemic. *Bulletin of the History of Medicine*, 76:105–115, 2002.

[3] E. L. Haseltine and J. B. Rawlings. Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. *Journal of Chemical Physics*, 117:6959–6969, 2002.

[4] H. De Jong. Modeling and simulation of genetic regulatory systems: A literature review. *Journal Of Computational Biology*, 9:67–103, 2002.

[5] A. Pettinen, T. Aho, O. Smolander, T. Manninen, A. Saarinen, K. Taattola, O. Yli-Harja and M. Linne. Simulation tools for biochemical networks: evolution of performance and usability. *Bioinformatics*, 21:357–363, 2005.

[6] J. Pahle. Biochemical simulations: Stochastic, approximate stochastic and hybrid methods. *Briefings in Bioinformatics*, pages 1–12, 2009.

[7] T. Tian and K. Burrage. Binomial leap methods for simulating stochastic chemical kinetics. *Journal of Chemical Physics*, 121:10356–10364, 2004.

[8] D. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81:2340–2361, 1977.

[9] R. Metzler. The future is noisy: The role of spatial fluctuations in genetic switching. *The American Physical Society*, 87:068103, 2001.

[10] D. Adalsteinsson, David McMillen and Timothy C Elston. Biochemical network stochastic simulator (bionets): software for stochastic modeling of biochemical networks. *BMC Bioinformatics*, 5, 2004.

[11] Z. Miskovic. Stochastic processes in the physical sciences: Course notes for AMATH 777. 2007.

[12] D. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *Journal of Chemical Physics*, 115:716–1733, 2001.

[13] C. V. Rao and A. P. Arkin. Stochastic chemical kinetics and the quasi-steady-state assumption: Application to the Gillespie algorithm. *Journal of Chemical Physics*, 118, 4999-5010, 2003.

[14] X. Peng, W. Zhou and Y. Wang. Efficient binomial leap method for simulating chemical kinetics. *Journal of Chemical Physics*, 126:224109, 2007.

[15] M Pettigrew and H. Resat. Multinomial tau leaping method for stochastic kinetic simulations. *Journal of Chemical Physics*, 126:084101, 2007.

[16] Y. Hu and T. Li. Highly accurate tau-leaping methods with random corrections. *Journal of Chemical Physics*, 130:124109, 2009.

[17] Y. Cao, D. Gillespie and L. Petzold. The slow-scale simulation algorithm. *Journal of Chemical Physics*, 122:014116, 2005.

[18] W. E, D. Liu and E. Vanden-Eijnden. Nested stochastic simulation algorithms for chemical kinetic systems with multiple time scales. *Journal of Computational Physics*, 221:158–180, 2007.

[19] E. Vanden-Eijnden W. E, D. Liu. Nested stochastic simulation algorithm for chemical kinetic systems with disparate rates. *Journal of Chemical Physics*, 123:194107, 2005.

[20] M. Rathinam, L. Petzold, Y. Cao and D. Gillespie. Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method. *Journal of Chemical Physics*, 119:12784–12794, 2003.

[21] H. Schmidt, Mads F. Madsen, Sune Dano and Gunnar Cederson. Complexity reduction of biochemical rate expressions. *Bioinformatics*, 24:848–854, 2008.

[22] J. Guckenheimer S. P. Ellner. *Dynamic Models in Biology*. Princeton University Press, Princeton, New Jersey, 2006.

[23] D. Gillespie. Stochastic chemical kinetics. *Springer, S. Yip, Handbook of Materials Modeling*, pages 1735–1752, 2005.

[24] D. Gillespie. The mathematics of Brownian motion and Johnson noise. *American Journal of Physics*, 64:225–240, 1996.

[25] A. Nordsieck , W. E. Lamb and G. E. Uhlenbeck. On the theory of cosmic-ray showers i the furry model and the fluctuation problem. *Physica*, 7:344–360, 1940.

[26] M. Kac. Foundations of kinetic theory. *Proceedings of the Third Mathematical Statistics and Probability*, 3:171–197, 1956.

[27] D. Gillespie. Chemical Langevin equation. *Journal of Chemical Physics*, 113:297–306, 2000.

[28] R. Goodman. *Introduction to Stochastic Models, Second Edition*. Dover Publications Inc., Mineola, New York, 2006.

[29] D. Gillespie. Approximating the master equation by Fokker-Planck type equations for single variable chemical systems. *Journal of Chemical Physics*, 72:5363–5370, 1980.

[30] D. Gillespie. The multivariate Langevin and Fokker-Planck equations. *American Journal of Physics*, 64:1246–1257, 1996.

[31] D. Gillespie. A rigorous derivation of the chemical master equation. *Physica A*, 188:404–425, 1992.

[32] M. Ullah and O. Wolkenhauer. Family tree of Markov models in systems biology. *IET Systems Biology*, 1:247–254, 2007.

[33] L. Petzold, Y. Cao and D. Gillespie. Comment on nested stochastic simulation algorithm for chemical kinetic systems with disparate rates. *Journal of Chemical Physics*, 126:137101, 2007.

[34] P. Hangii, H. Grabert, P. Talkner and H. Thomas. Bistable systems: Master equation vs Fokker-Planck equation. *Physical Review A*, 29:371–378, 1984.

[35] W.T Cochran and J. W. Cooley. What is the fast Fourier transform. *Proceedings of the IEEE*, 55:1664–1674, 1967.

[36] F. J. Harris. On the use of windows for harmonic analysis with the discrete Fourier transform. *Proceedings of the IEEE*, 66:51–83, 1978.

[37] T. Theussl. On windowing for gradient estimation in volume visualization. *Proceedings of Central European Seminar on Computer Graphics*, 1999.

[38] P. Wickramarchi. Effects of windowing on the spectral content of a signal. *Sound and Vibration, 10-11*, 2003.