Published in final edited form as:

Methods Enzymol. 2009 ; 454: 115–140. doi:10.1016/S0076-6879(08)03805-6.

Discrete Stochastic Simulation Methods for Chemically Reacting Systems

Yang Cao¹ and David C. Samuels²

David C. Samuels: dsamuels@vbi.vt.edu

²Virginia Bioinformatics Institute, Virginia Tech, Blacksburg, VA 24061; Phone: (540) 231-8999

Abstract

Discrete stochastic chemical kinetics describe the time evolution of a chemically reacting system by taking into account the fact that in reality chemical species are present with integer populations and exhibit some degree of randomness in their dynamical behavior. In recent years, with the development of new techniques to study biochemistry dynamics in a single cell, there are increasing studies using this approach to chemical kinetics in cellular systems, where the small copy number of some reactant species in the cell may lead to deviations from the predictions of the deterministic differential equations of classical chemical kinetics.

This chapter reviews the fundamental theory related to stochastic chemical kinetics and several simulation methods that are based on that theory. We focus on non-stiff biochemical systems and the two most important discrete stochastic simulation methods: Gillespie's Stochastic Simulation Algorithm (SSA) and the tau-leaping method. Different implementation strategies of these two methods are discussed. Then we recommend a relatively simple and efficient strategy that combines the strengths of the two methods: the hybrid SSA/tau-leaping method. The implementation details of the hybrid strategy are given here and a related software package is introduced. Finally, the hybrid method is applied to simple biochemical systems as a demonstration of its application.

1 Introduction

Biochemical systems have traditionally been modeled by a set of ordinary differential equations (ODEs). The general form of the Reaction Rate Equations (RREs) in that approach can be formulated as

$$\frac{dx_i}{dt} = f_i(x_1, \dots, x_n), \quad (1)$$

for $i = 1, \dots, n$, where the state variables x_i represent the concentrations of involved species, and functions f_i are inferred from the various chemical reactions of the system. This set of ODEs is usually solved with numerical methods packages such as DASSL and DASPK (Brenan et al. 1996), ODEPACK (Hindmarsh, 1983), or CVODE (Cohen and Hindmarsh, 1996) for example. An important feature of the equations in this approach is that the system is deterministic and continuous. For systems where all chemical species are present in large copy numbers, it is reasonable to model every species by its concentration and the traditional ODE approaches seem to work very well. However, if the system is small enough that the molecular populations of some of the reactant species are small, from one to thousands,

¹Author to whom correspondence should be addressed. Department of Computer Science, Virginia Tech, Blacksburg, VA 24061; Phone: (540) 231-1417; ycao@cs.vt.edu.

discreteness and stochasticity may play important roles in the dynamics of the system. Such a case occurs often in cellular systems (McAdams and Arkin, 1997, Arkin et al., 1998, Fedoroff and Fontana, 2002), which typically involve copy numbers of one or two for the number of genes of a given protein, on the order of tens to hundreds for the corresponding RNAs and on the order of thousands for regulatory proteins and enzymes. In that case, Equation (1) cannot accurately describe the system's true dynamic behavior. Thus new modeling and simulation methods are needed to reflect the discrete and stochastic features of biochemical systems on a cellular scale.

To include discreteness, the most accurate way to simulate the time evolution of a system of chemically reacting molecules is to do a molecular dynamics simulation tracking the positions and velocities of all the molecules and the occurrence of all chemical reactions when molecules physically collide with each other. But molecular dynamics simulations are generally too expensive to be practical except in the case of a relatively small number of molecules and even then only for very short time scales. Instead we consider the case where the dynamics of biochemical systems can be approximated by assuming that the reactant molecules are “well-stirred” such that their positions become randomized and need not be tracked in detail. When that is true, the state of the system can be defined simply by the instantaneous molecular populations of the various chemical species. The chemical reactions can be defined as events that change the state of the system following biochemical rules, changing the molecular populations by integer numbers.

This chapter discusses numerical methods that can be applied to simulate such systems, taking into account discreteness and stochasticity at the molecular level. The focus will be on two major simulation methods: Gillespie's stochastic simulation algorithm (SSA) and the tau-leaping method. Finally, a merger of these two methods, the hybrid SSA/tau-leaping method will be described.

2 The Chemical Master Equation

Let us consider a system of N molecular species $\{S_1, \dots, S_N\}$ interacting through M elemental chemical reaction channels $\{R_1, \dots, R_M\}$. We assume that the system is confined to a constant volume and is well-stirred, or in other words is in thermal (but not chemical) equilibrium at a constant temperature. Under these assumptions, the state of the system can be represented by the populations of the species involved. We denote these populations by $\mathbf{X}(t) \equiv (X_1(t), \dots, X_N(t))$, where $X_i(t)$ is the number of molecules of species S_i in the system at time t . The well-stirred condition is crucial. When this condition is broken, the spatial information of each species becomes important and the population information for the species will not be enough alone to determine the system dynamics. In cases where the well-stirred condition does not hold, the required simulation techniques will be different from what we discuss in this chapter. The so-called elemental reactions only include unimolecular and bimolecular reactions. Generalizations can be made to include more complicated reaction types such as the commonly used Michaelis-Menten reaction (Rao and Arkin, 2003, Cao et al. 2005B). We note that modeling these higher-order reaction types using discrete stochastic methods are still under research and are not the focus of this review.

For a well-stirred system, each reaction channel R_j can be characterized by a *propensity function* a_j and a *state change vector* $\mathbf{v}_j \equiv (v_{1j}, \dots, v_{Nj})$. The propensity function is defined by the statement:

$a_j(\mathbf{x}) dt$ is the probability, given $\mathbf{X}(t) = \mathbf{x}$; that one R_j reaction will occur in the next infinitesimal time interval $[t; t + dt)$.

ν_{ij} is the change in the molecular population S_i induced by one reaction R_j . The matrix $\mathbf{v}$ is known as the stoichiometric matrix. The propensity function $a_j(x)$ reflects the fundamental characteristics of the stochastic chemical kinetics. Its value depends on the populations of the reactant populations and a reaction probability rate constant c_j , which is defined so that

$c_j dt$ is the probability that a randomly chosen combination of R_j reactant molecules will react in the next infinitesimal time dt :

Then a_j is the product of c_j and the number of all possible combinations of R_j reactant molecules.

The following are three simple examples of basic reactions and their propensity functions and state change vectors.

$$\text{For } S_1 \xrightarrow{c_1} S_2, \quad a_j(x) = c_1 x_1, \quad \text{and } \nu_j = (-1, 1, 0, \dots, 0). \quad (2)$$

$$\text{For } S_1 + S_2 \xrightarrow{c_1} S_3, \quad a_j(x) = c_1 x_1 x_2, \quad \text{and } \nu_j = (-1, -1, 1, 0, \dots, 0). \quad (3)$$

$$\text{For } S_1 + S_1 \xrightarrow{c_1} S_2, \quad a_j(x) = \frac{1}{2} c_1 x_1 (x_1 - 1), \quad \text{and } \nu_j = (-2, -1, 0, \dots, 0). \quad (4)$$

It is easy to see that the form of the propensity function is similar to the mass action terms in the deterministic RREs. The value of c_j is similar to its counterpart: the reaction rate k_j in the RREs. Indeed there is a connection between c_j and k_j depending on the reaction type. For a unimolecular reaction such as in the example of equation (2), $c_1 = k_1$. For a bimolecular reaction between different species such as in (3), $c_1 = k_1 / A \Omega$, where A is the Avogadro number and Ω is the constant volume. For a bimolecular reaction between the same species, the forms of the propensity function and the reaction rate function have a slight difference, but when x_1 is large the difference will be negligibly small and we will have $c_1 \approx 2k_1 / A \Omega$.

Once the propensity functions and stoichiometric matrix are determined, the dynamics of the system obey the chemical master equation (CME):

$$\frac{\partial P(x, t | x_0, t_0)}{\partial t} = \sum_{j=1}^M \left[a_j(x - \nu_j) P(x - \nu_j, t | x_0, t_0) - a_j(x) P(x, t | x_0, t_0) \right], \quad (5)$$

where $P(x; t | x_0, t_0)$ denotes the probability that $X(t)$ will be x given that $X(t_0) = x_0$. In principle, the CME completely determines the dynamics of $P(x; t | x_0, t_0)$ but the CME is essentially an ODE whose dimension is given by the number of all possible combinations of states of x . Consider the example of a small reaction network of 5 species and assume that the population of each species is in the range 1 to 100. The dimension of the corresponding CME will then be $100^5 = 10^{10}$. As the number of species increases, the dimension of the corresponding CME increases exponentially, a problem known as the “curse of dimension”. It is easy to see that the CME is both theoretically and computationally intractable for all but the simplest models. In recent years there are some interesting research (Munsky and Khammash, 2006, Zhang and Watson, 2007) trying to reduce the dimension of the CME or to provide an approximate numerical solution of the CME. Progress has been made but so far these methods still can only be practically applied to simple models.

3 The Stochastic Simulation Algorithm

Another way to study the dynamics of the reaction system is to construct realizations of $X(t)$ through numerical simulation. In the numerical simulation, the key is not to get the probabilities $P(\mathbf{x}; t / \mathbf{x}_0, t_0)$ but to generate a single trajectory (a realization) that the system may undergo. The most important simulation method for this is Gillespie's stochastic simulation algorithm (SSA) (Gillespie, 1976, 1977). Instead of following the time evolution of the probabilities, the SSA generates a trajectory of the system step by step. In each step the SSA starts from a current state $\mathbf{x}(t) = \mathbf{x}$ and asks two questions:

When will the next reaction occur? We denote this time interval by τ .

When the next reaction occurs, which reaction will it be? We denote the chosen reaction by the index j .

To answer the above questions, one needs to study the joint probability density function $p(\tau, j / \mathbf{x}; t)$, which is defined by

$$p(\tau, j / \mathbf{x}, t) d\tau = \text{the probability, given } X(t) = \mathbf{x}, \text{ that the next reaction will occur in the infinitesimal time interval } [t + \tau, t + \tau + d\tau), \text{ and will be an } R_j \text{ reaction} \quad (6)$$

It can be derived (Gillespie, 1976, 1977) that

$$p(\tau, j / \mathbf{x}, t) = a_j(\mathbf{x}) \exp(-a_0(\mathbf{x}) \tau) \quad (7)$$

Where $a_0(\mathbf{x}) \equiv \sum_{j=1}^M a_j(\mathbf{x})$. Equation (7) is the theoretical foundation for the SSA. It implies that the time τ to the next occurring reaction is an exponentially distributed random variable with mean value $1 / a_0(\mathbf{x})$, and that the index j of that reaction is the integer random variable with point probability $a_j(\mathbf{x}) / a_0(\mathbf{x})$. To advance the system from state $\mathbf{x}$ at time t , the SSA generates two random numbers r_1 and r_2 uniformly over the unit interval, and then takes the time of the next reaction to be $t + \tau$ where

$$\tau = \frac{1}{a_0(\mathbf{x})} \ln \left(\frac{1}{r_1} \right) \quad (8)$$

and the index for the next reaction to be the smallest integer j satisfying

$$\sum_{j'=1}^j a_{j'}(\mathbf{x}) > r_2 a_0(\mathbf{x}) \quad (9)$$

The system state is then updated according to $X(t + \tau) = \mathbf{x} + \mathbf{v}_j$, and this process is repeated until the simulation final time or some other terminating condition is reached. The algorithm is listed below:

Algorithm 1: The Basic SSA Method

Starting from initial condition $t = t_0$ and $\mathbf{x} = \mathbf{x}_0$,

Step 1: With $\mathbf{x}(t) = \mathbf{x}$, calculate all $a_j(\mathbf{x})$ and $a_0(\mathbf{x})$.

Step 2: If $a_0(\mathbf{x}) = 0$, terminate the simulation. Otherwise generate two uniform random numbers r_1 and r_2 . Calculate τ and j according to (8) and (9) respectively.

Step 3: Update the system by $t = t + \tau$ and $\mathbf{x} = \mathbf{x} + \mathbf{v}_j$.

Step 4: If t reaches the end time, stop. Otherwise, go to step 1.

The SSA is exact in the sense that the sample paths it generates are precisely distributed according to the solution of the CME, which makes it one of the most fundamental simulation methods for discrete stochastic biochemical systems. Although this algorithm looks quite simple, due to its importance there are several different implementation strategies proposed in the literature for the SSA. They are the direct method (DM) (Gillespie, 1977), the first reaction method (FRM) (Gillespie, 1977), the next reaction method (NRM) (Gibson and Bruck, 2000), the optimized direct method (ODM) (Cao et al., 2004), the sorted direct method (SDM) (McCollum et al., 2006), and the Logarithmic Direct Method (LDM) (Li and Petzold, 2006). Below we give a brief review for these implementation strategies.

The Direct Method is exactly the Algorithm 1 we have just given. The First Reaction Method is theoretically equivalent to the Direct Method but is quite different in the implementation details. The FRM generates a potential reaction time for each reaction and chooses the “first” reaction channel that has the earliest firing time to occur. In the FRM implementation, one generates M uniform random numbers $r_1, \dots, r_M$ in every step and calculates a time τ_k for each reaction channel R_k by

$$\tau_k = \frac{1}{a_j(\mathbf{x})} \ln \left(\frac{1}{r_k} \right). \quad (10)$$

Then τ and j are given by

$$\begin{aligned} \tau &= \min_{1 \leq k \leq M} (\tau_k) \\ j &= \text{the index for the smallest } \tau_k. \end{aligned} \quad (11)$$

It can be proved that the τ and j generated from (11) follow the same distributions as in (8) and (9). Thus the DM and the FRM are statistically equivalent. However, in every step the FRM generates M τ_k values but uses only one of them. Thus the FRM is much less efficient than the DM.

Gibson and Bruck (Gibson and Bruck, 2000) have made remarkable progress improving the implementation efficiency of the FRM. Their method is the Next Reaction Method (NRM). The NRM uses a dependent graph to record the influence of each reaction channel on the other reaction channels. It records the absolute time $t + \tau_k$ as the expected firing time for the R_k reaction. If the firing of one reaction channel does not change the propensity of another reaction channel, the expected firing time for the latter reaction remains the same. In this way the NRM avoids unnecessary updates of the propensity function and expected firing time. For a reaction channel R_k whose reactants have been changed by the firing reaction, the NRM uses a cleverly designed formula to reuse the uniform random number r_k generated in the previous step. As a result, in every step there is only one uniform random number generated. The NRM turns out to be much more efficient than the FRM. However, using a detailed numerical analysis, it has been shown (Cao et al. 2004) that the NRM still has a higher computational cost than the Direct Method except for simple systems where the reactions are almost totally independent of each other.

To decrease the computational cost the Optimized Direct Method (ODM) (Cao et al. 2004) adopts the dependent graph to avoid the unnecessary recalculation of propensity functions and rearranges the index of the reaction channels so that the more frequent reaction channels are always indexed before the less frequent ones. With these two improvements over the DM, the ODM becomes one of the most efficient SSA implementation strategies currently in use.

The re-index technique of the ODM requires one or a few sample runs using the SSA to collect the necessary information. This is not convenient in many applications. In order to dynamically adjust the index of the reaction channels the Sorted Direct Method (SDM) was proposed (McCollum et al., 2006). In the SDM, a bubble-up sorting method was applied to the index of reaction channels. In the simulation, every time when one reaction occurs, its reaction index decreases by one so that in the next step it is more quickly found. Then, after a certain initial simulation time the index list will be sorted into a form very close to the optimal one. The SDM is a little less efficient than the ODM but its adaptive feature makes it a very good strategy, particular in simulation of oscillation systems where a fast reaction in one time period may become slow in another time period. In that case, the dynamic indexing of this method is very useful.

Recently the Logarithmic Direct Method (LDM) was proposed (Li and Petzold, 2006), which applies a binary search method to the direct method. When the number of reaction channels, M , is large the LDM can complete the search for the index j within $O(\log(M))$ time. Thus the LDM has advantages for large biochemical system.

We note that for all the above implementation strategies, the differences among them (except for the case of the FRM) in computation time are usually less than 20%. This is far from the computation speed needed in many applications. As the SSA is a procedure simulating every reaction event one-at-a-time, the computational cost is inevitably high. Thus people have to consider alternative methods to gain efficiency by sacrificing exactness, as long as the approximation accuracy is kept under control.

4 The Tau-Leaping Method

The tau-leaping method (Gillespie, 2001) was designed to speed up a stochastic simulation by leaping over many reactions in one time step. This idea is illustrated in Figure 1. The tau-leaping method makes the leap by answering the following question: How often does each reaction channel fire in the next specified time interval τ ? More precisely, let

$$K_j(\tau; \mathbf{x}, t) \triangleq \text{the number of times given } X(t) = \mathbf{x}, \text{ that reaction channel } \mathbf{R}_j \text{ will fire in the time interval } [t, t + \tau), \quad (j=1, \dots, M). \quad (12)$$

For arbitrary values of τ it will be about as difficult to compute $K_j(\tau; \mathbf{x}, t)$ as to solve the CME. The tau-leaping method chooses a small τ value to satisfy the following **Leap Condition**: *For the current state x , require τ to be small enough that the change in the state during $[t; t + \tau)$ will be so small that no propensity function will suffer an appreciable change in its value.* Under the Leap Condition, a good approximation to $K_j(\tau; \mathbf{x}, t)$ will be provided by $P(a_j(\mathbf{x}), \tau)$, the Poisson random variable with mean (and variance) $a_j(\mathbf{x}) \tau$. So if $X(t) = \mathbf{x}$ and we choose τ to satisfy the Leap Condition, we can update the state to time $t + \tau$ according to the approximate formula

$$X(t+\tau) \doteq x + \sum_{j=1}^M \nu_j P(a_j(x)\tau) \quad (13)$$

where $P(a_j(x)\tau)$ for each $j = 1, \dots, M$ denotes an independent sample of the Poisson random variable with mean and variance $a_j(x)\tau$. This computational procedure is the *tau-leaping approximation*.

The tau-leaping method makes a natural connection between the SSA and the deterministic RREs. When τ is chosen very small such that in every time step there is at most one reaction occurring, the tau-leaping method reduces to a linear approximation of the SSA. When τ is allowed to be large such that

$$a_j(x)\tau \gg 1, \text{ for all } j=1, \dots, M, \quad (14)$$

the Poisson random number $P(a_j(x)\tau)$ can be approximated by the Normal random number with mean and variance $a_j(x)\tau$, denoted by $N(a_j(x)\tau, a_j(x)\tau)$. Then the formula (13) reduces to the forward Euler method for the chemical Langevin equation (CLE) (Gillespie, 2001). Moreover, when the values $a_j(x)\tau$ for all $j = 1, \dots, M$ are even larger, the standard deviation is then negligible compared to the mean value. The Poisson random number $P(a_j(x)\tau)$ can then be simply replaced by its mean value $a_j(x)\tau$. Then the equation (13) becomes

$$X(t+\tau) \doteq x + \sum_{j=1}^M \nu_j a_j(x)\tau \quad (15)$$

which is the forward Euler method for the corresponding RREs. Note that here the merger of the tau-leaping method into the forward Euler method is seamless. One does not need to check the condition (14) for all j 's. The idea of using the normal random number or just the mean value to approximate the Poisson random number can be applied for any individual j . This procedure can be wrapped in a Poisson random number approximation procedure. Choose two threshold values: M_1 , for which the Poisson random number $P(M_1)$ can be safely approximated by a normal random number $N(M_1, M_1)$, and M_2 , for which $P(M_2)$ can be safely approximated by M_2 . We then have the following algorithm.

Algorithm 2: The Poisson random number approximation procedure

Given the mean and variance value m , we follow the following procedure to generate an approximation to the Poisson random number $P(m)$:

Case 1: If $m < M_1$, return a Poisson random number $P(m)$.

Case 2: If $m \geq M_1$ and $m < M_2$, return a Normal random number $N(m, m)$.

Case 3: If $m \geq M_2$, return m .

We denote the approximation function generated by Algorithm 2 as $\rho(m)$. Then equation (13) becomes

$$X(t+\tau) \doteq x + \sum_{j=1}^M \nu_j \rho(a_j(x)\tau) \quad (16)$$

There are two practical problems to be addressed before the tau-leaping method can be applied to realistic applications. First, we need a procedure to quickly determine the largest value of τ that is compatible with the Leap Condition. Second, we need to develop a method to avoid possible negative populations that could result from two reasons:

- a. The Poisson random number (or its approximation) is unbounded. There is a small possibility that a large random number may exceed the number of some reactants and causes negative populations to occur.
- b. There are multiple reaction channels consuming the same reactant. When they fire at the same time, even though neither of them separately exhausts the number of that reactant, their overall effect may do so.

In the following section we will discuss the details on how to solve these two practical problems. As negative populations present the more serious problem, we discuss the corresponding solution first. Then we review several τ -selection formulas and present a simple and efficient one.

4.1 The Hybrid SSA/Tau-Leaping Strategy

Negative populations resulting from the original tau-leaping method have been found to happen in the simulation of certain systems in which some consumed reactant species are present in small numbers. It was believed that the reason for this error was mostly due to the fact that the Poisson random variable is unbounded such that the Poisson approximation to $K_j(\tau; x, t)$ in equation (13) might result in reaction channel R_j firing so many times that the population of one of its reactant species would be driven negative. To resolve this problem, the *binomial* tau-leaping method (Tian and Burrage, 2004, Chatterjee et al., 2005) was proposed, in which bounded binomial random variables replace the unbounded Poisson random variables. However, recently we have developed an understanding that the negative population problem arises more often from multiple reaction channels consuming the same reactant, than from the unbounded Poisson random variable. The binomial tau-leaping method becomes complicated when dealing with this case. Cao et al. (2005C) made an observation that most negative populations in both cases were related to species with a low population. This may seem to be a rather obvious observation, but it does point us toward a method of dealing with this problem. Based on this observation, an adaptive hybrid SSA/tau-leaping method was proposed, which seems to resolve the negativity problem satisfactorily.

The hybrid SSA/tau-leaping algorithm (Cao et al., 2005C) is based on the fact that negative populations typically arise from multiple firings of reactions that are only a few reaction events away from consuming all the molecules of one of their reactants. To focus on those reaction channels, the hybrid SSA/tau-leaping algorithm introduces a second control parameter n_c , a positive integer that is usually set somewhere between 2 and 20. Any reaction channel with a positive propensity function that is currently within n_c firings of exhausting one of its reactants is classified as a *critical reaction*. The hybrid algorithm chooses τ in such a way that no more than one firing of *all* the critical reactions can occur during the leap. Essentially, the algorithm simulates the critical reactions using an adapted (and thus not quite exact) version of the SSA, and the remaining non-critical reactions using the Poisson tau-leaping method. Since no more than one firing of a critical reaction can occur during a leap, the probability of producing a negative population is reduced to nearly zero. On those rare occasions when a negative population does arise (from firings of some non-critical reaction), that step can simply be rejected and repeated with τ reduced by half, or else the simulation can be started over using a larger value for n_c .

It can be shown (Cao et al., 2005C) that the hybrid SSA/tau-leaping procedure becomes identical to the SSA if n_c is chosen so large that *every* reaction channel is critical, and becomes identical to the tau-leaping method if $n_c = 0$ so that *none* of the reaction channels is critical. Thus, the hybrid SSA/tau-leaping algorithm is not only more robust, but also potentially more accurate, than the earlier tau-leaping algorithm.

There are some details left to discuss before giving the full description of the hybrid SSA/Tau-leaping method. First, how do we decide whether or not a reaction is *critical* with the parameter n_c ? This is done by first estimating for each reaction R_j with $a_j(\mathbf{x}) > 0$ the maximum number of times L_j that R_j can fire before exhausting one of its reactants (Tian and Burrage, 2004, Chatterjee et al., 2005):

$$L_j = \min_{i \in [1, N]; v_{ij} < 0} \left\lceil \frac{x_i}{|v_{ij}|} \right\rceil \quad (17)$$

Here the minimum is taken over only those index values i for which $v_{ij} < 0$, and the brackets denote “greatest-integer-in”. For example, for a reaction

L_j is the population of S_1 . For a reaction

L_j takes the smaller value between the populations of S_1 and S_2 . For a reaction

L_j takes the integer part of one half of the population of S_1 .

After L_j is calculated, it is compared with n_c . If $L_j < n_c$, R_j is considered as a critical reaction and should be simulated by the adapted SSA part. Otherwise, R_j is noncritical and can be simulated by the tau-leaping part.

The next step is to decide how to implement the SSA part and the tau-leaping part together. To solve this problem, in every simulation step we first generate a τ' from a τ -selection procedure and a τ'' from the SSA part. If τ' is even smaller than a few fold of the expected step-size of a pure SSA method, $a_0(\mathbf{x})$, we will stick with the pure SSA method. Otherwise, we use the tau-leaping method to simulate the non-critical reactions and the SSA method to simulate the critical reactions. The real simulation time step τ is chosen to be the smaller value between τ' and τ'' . If τ'' is smaller, the critical reaction fires. Otherwise, no critical reaction should fire before τ . In both cases, the numbers of noncritical reaction firings are calculated using the Poisson tau-leaping method. The τ'' for the SSA part can simply follow the SSA procedure limited to only critical reactions. The τ' for the tau-leaping part will be discussed in the next subsection.

4.2 The Tau-Selection Formula

The simulation formula for the tau-leaping method is quite simple. The key point is how to select the τ value so that the Leap Condition is satisfied. There have been several tau selection formulae proposed in the literature. Gillespie (Gillespie, 2001) originally proposed that the Leap Condition could be considered satisfied if the expected change in each propensity function $a_j(x)$ during the leap were bounded by $\varepsilon a_0(x)$, where ε is an error control parameter ($0 < \varepsilon \ll 1$). Later, this condition is refined by Gillespie and Petzold (Gillespie and Petzold, 2003). They showed that the largest value of ε that satisfies this requirement can be estimated as follows: First compute the $M^2 + 2M$ auxiliary quantities

$$f_{jj'}(x) \equiv \sum_{i=1}^N \frac{\partial a_j(x)}{\partial x_i} \nu_{ij'}, \quad j, j' = 1, \dots, M, \quad (18)$$

$$\mu_j(x) \equiv \sum_{i=1}^N f_{ji'}(x) a_{j'}(x), \quad j, j' = 1, \dots, M, \quad (19)$$

$$\sigma_j^2(x) \equiv \sum_{i=1}^N f_{ji'}^2(x) a_{j'}(x), \quad j, j' = 1, \dots, M;$$

then take

$$\tau = \min_{j \in [1, M]} \left(\frac{\varepsilon a_0(x)}{|\mu_j(x)|}, \frac{(\varepsilon a_0(x))^2}{\sigma_j^2(x)} \right). \quad (20)$$

The derivation of these formulas (Gillespie and Petzold, 2003) shows that $\mu_j(x)$ estimates the *mean* of the expected change in $a_j(x)$ in time τ , $\sqrt{\sigma_j^2(x)} \tau$ estimates the *standard deviation* of the expected change in $a_j(x)$ in time τ , and formula (20) essentially requires that both of those quantities be bounded by $\varepsilon a_0(x)$ for all j . We should note that Gillespie's original τ -selection formula (Gillespie, 2001) was deficient in that it lacked the σ_j^2 argument in Eq. (20).

The tau-selection procedure (20) seeks to set a bound on the change in each propensity function $a_j(x)$ during a time step τ by a small fraction ε of the *sum* $a_0(x)$ of all the propensity functions. Denoting the change in propensity function a_j from time t to time $t + \tau$, given $\mathbf{X}(t) = \mathbf{x}$, by $\Delta_\tau a_j(x)$, this requirement can be stated as

$$|\Delta_\tau a_j(x)| \leq \varepsilon a_0(x), \quad j=1, \dots, M. \quad (21)$$

This bound is explicitly reflected in the numerators of the two fractions in the τ -selection formula (20). Although this strategy does indeed limit the changes in the propensities during a leap as required, it does not fully accomplish the task with a proper scaling. The Leap Condition requires that every propensity function remains “practically constant” during a τ time period, since that is what allows the number of reaction events R_j during τ to be accurately approximated by a statistically independent Poisson random variable with mean $a_j(x) \tau$. If $a_j(x)$ for R_j reaction happens to be very small compared to $a_k(x)$ for R_k reaction,

$a_j(\mathbf{x})$ will then be much smaller than $a_0(\mathbf{x})$. The condition (21) may allow a large *relative* change in $a_j(\mathbf{x})$, and that could result in simulation inaccuracies.

To allow the formula for the Leap Condition to reflect the relative scales, we change the condition (21) by

$$|\Delta_\tau a_j(\mathbf{x})| \leq \varepsilon a_j(\mathbf{x}), \quad j=1, \dots, M. \quad (22)$$

But doing this can lead to difficulties if $a_j(\mathbf{x})$ happens to approach zero; because then condition (22) will force τ to approach zero, effectively bringing the tau-leaping process to a halt. Thus we have to make a simple modification to avoid this problem. The limit procedure described above is implicitly based on treating the propensity functions as continuous functions. Actually, the propensity functions change as reactions occur by discrete amounts, and for every propensity function $a_j(\mathbf{x})$ there will always be a minimum amount by which it can change. For example, if R_j is the unimolecular reaction with propensity function $a_j(\mathbf{x}) = c_j x_j$, then the minimum (positive) amount by which $a_j(\mathbf{x})$ can change will obviously be c_j . It is not hard to show that if the propensity function of any bimolecular or trimolecular reaction R_j changes at all, it must do so by an amount greater than or equal to c_j . Since it is therefore unreasonable to require any propensity function $a_j(\mathbf{x})$ to change by less than c_j , we should replace the bound on the right-hand side of condition (22) with the *larger* of $\varepsilon a_j(\mathbf{x})$ and c_j .

$$\Delta_\tau a_j(\mathbf{x}) \leq \max \{ \varepsilon a_j(\mathbf{x}), c_j \}, \quad j=1, \dots, M. \quad (23)$$

To apply the new condition (23), we can simply replace formula (20) with

$$\tau' = \min_{j \in [1, M]} \left(\frac{\max \{ \varepsilon a_j(\mathbf{x}), c_j \}}{|\mu_j(\mathbf{x})|}, \frac{(\max \{ \varepsilon a_j(\mathbf{x}), c_j \})^2}{\sigma_j^2(\mathbf{x})} \right), \quad (24)$$

where the definitions of μ_j and σ_j remain the same as in (19).

Although τ -selection using formula (24) results in a more accurate simulation than τ -selection using formula (20), the evaluation of the functions $\mu_j(\mathbf{x})$ and $\sigma_j^2(\mathbf{x})$ in Eqs. (18) and (19) prior to each leap tends to be very time-consuming, especially if both M and N are large. A new τ -selection formula (Cao et al. 2006) was then proposed to avoid this computational burden. Here we introduce a simplified version of this new formula. The underlying strategy of this new τ -selection procedure is to bound the relative changes in the molecular populations by a specified value ε ($0 < \varepsilon \ll 1$). Let

$$\Delta_\tau X_i \equiv \Delta_\tau X_i(\mathbf{x}) \triangleq X_i(t+\tau) - x_i, \quad \text{given } \mathbf{X}(t) = \mathbf{x}. \quad (25)$$

Instead of basing the τ -selection on condition (23), we base it on the condition

$$\Delta_\tau X_i \leq \max \{ \varepsilon x_i, 1 \}, \quad \forall i \in I_{rs}, \quad (26)$$

where I_{rs} denotes the set of indices of all reactant species (so $i \in I_{rs}$ if and only if x_i is an argument of at least one propensity function). Condition (26) evidently requires the relative

change in X_i to be bounded by ε , except that X_i will never be required to change by an amount less than 1.

Recalling the tau-leaping formula (16), we see that the quantity defined in (25) will essentially be given by

$$\Delta_\tau X_i = \sum_{j=1}^M v_{ij} \rho(a_j(x) \tau), \quad \forall i \in I_{rs} \quad (27)$$

Since the Poisson random variables (or the corresponding approximations) $\rho(a_j(x) \tau)$ on the right-hand side of Eq.(27) are statistically independent and have means and variances $a_j(x) \tau$, the mean and variance of that linear combination can be straightforwardly computed:

$$\langle \Delta_\tau X_i \rangle = \sum_{j=1}^M v_{ij} [a_j(x) \tau], \quad \forall i \in I_{rs} \quad (28)$$

$$\text{var} \{ \Delta_\tau X_i \} = \sum_{j=1}^M v_{ij}^2 [a_j(x) \tau], \quad \forall i \in I_{rs}$$

Using the same reasoning that was used in deriving the Gillespie-Petzold τ -selection procedure (Gillespie and Petzold, 2003), we may consider the bound (26) on $\Delta_\tau X_i$ to be “substantially satisfied” if it is simultaneously satisfied by the absolute mean and the standard deviation of $\Delta_\tau X_i$:

$$|\langle \Delta_\tau X_i \rangle| \leq \max \{ \varepsilon x_i, 1 \}, \quad \sqrt{\text{var} \{ \Delta_\tau X_i \}} \leq \max \{ \varepsilon x_i, 1 \}, \quad \forall i \in I_{rs} \quad (29)$$

Substituting formulas (28) into conditions (29), we obtain the following bounds on τ :

$$\tau \leq \frac{\max \{ \varepsilon x_i, 1 \}}{\left| \sum_{j=1}^M v_{ij} a_j(x) \right|}, \quad \tau \leq \frac{\max \{ \varepsilon x_i, 1 \}^2}{\sum_{j=1}^M v_{ij}^2 a_j(x)}, \quad \forall i \in I_{rs}. \quad (30)$$

$$\widehat{\mu}_i(x) \triangleq \sum_{j=1}^M v_{ij} a_j(x), \quad \forall i \in I_{rs}, \quad (31)$$

$$\widehat{\sigma}_i^2(x) \triangleq \sum_{j=1}^M v_{ij}^2 a_j(x), \quad \forall i \in I_{rs},$$

where I_{rs} is the set of indices of all reactant species, and then taking

$$\tau = \min_{i \in I_{rs}} \left(\frac{\max\{\varepsilon x_i, 1\}}{|\hat{\mu}_i(\mathbf{x})|}, \frac{\max\{\varepsilon x_i, 1\}^2}{\hat{\sigma}_i^2(\mathbf{x})} \right), \quad (32)$$

The τ -selection procedure of formulas (31) and (32) will obviously be simpler to program and faster to execute than the τ -selection procedure of formulas (18), (19) and (24). Note in particular that the required number of computational operations increases quadratically with the number of reaction channels in the old formulas, but only linearly with the number of species in the new formulas. Since τ -selection has to be performed prior to every tau-leap, using these new formulas leads to substantially faster simulations when the system has many reactions and species.

The formulas (31) and (32) are for the original tau-leaping method. In order to apply them to the hybrid SSA/tau-leaping method, they need a little modification. The calculation should not be extended to critical reactions since they are handled by the adapted SSA part. Thus we let J_{ncr} denote the set of indices of the non-critical reactions. If J_{ncr} is empty (i.e., there are no non-critical reactions), we simply take $\tau = \infty$ (practically this can be a large step size, for example the whole simulation time interval). Otherwise, the $\hat{\mu}_i$ and $\hat{\sigma}_i$ are calculated with the following formula:

$$\hat{\mu}_i(\mathbf{x}) \triangleq \sum_{j \in J_{ncr}} v_{ij} a_j(\mathbf{x}), \quad \forall i \in I_{rs}, \quad (33)$$

$$\hat{\sigma}_i^2(\mathbf{x}) \triangleq \sum_{j \in J_{ncr}} v_{ij}^2 a_j(\mathbf{x}), \quad \forall i \in I_{rs},$$

The formula of τ remains the same as in (32) but the calculation of $\hat{\mu}_i$ and $\hat{\sigma}_i$ are replaced by (33). Notice that the difference between (31) and (33) is that in (33) only non-critical reactions are considered, while in (31) all reactions are included.

The full description of the hybrid SSA/Tau-leaping method is given as follows.

Algorithm 3: The Hybrid SSA/Tau-Leaping Method

1. In state $\mathbf{x}$ at time t , identify the currently critical reactions. We calculate L_j according to the formula (17). Any reaction R_j with $a_j(\mathbf{x}) > 0$ is deemed *critical* if $L_j < n_c$. Otherwise, it is *non-critical*. (We normally take $n_c = 10$ as a practical value.)
2. Let J_{ncr} denote the set of indices of the non-critical reactions. If J_{ncr} is empty, we take $\tau' = \infty$ (or the final simulation time). Otherwise, with a value chosen for ε (we normally take $\varepsilon = 0.03$), compute a candidate time leap τ' from the τ -selection formula (32) and (33). Thus τ' tentatively estimates the time to the next non-critical reaction.
3. If τ' is less than some small multiple (which we usually take to be 10) of $1/a_0(\mathbf{x})$, abandon tau-leaping temporarily, execute some modest number (which we usually take to be 100) of single-reaction SSA steps, and return to step 1. Otherwise, proceed to step 4.
4. Compute the sum $a_0(\mathbf{x})$ of the propensity functions of all the *critical* reactions. Generate a second candidate time leap τ'' as a sample of the exponential random

variable with mean $1/a_j^c(x)$. As thus computed, τ'' tentatively estimates the time to the next critical reaction.

5. Take the actual time leap τ to be the smaller of τ' and τ'' , and set the number of firings k_j of each reaction R_j accordingly:
 - a. If $\tau' < \tau''$, take $\tau = \tau'$. For all critical reactions R_j set $k_j = 0$ (no critical reactions will occur during this leap). For all non-critical reactions R_j , generate k_j as a sample of the Poisson random variable with mean $a_j(x)\tau$.
 - b. If $\tau'' < \tau'$, take $\tau = \tau''$. Generate j_c as a sample of the integer random variable with point probabilities $a_j(x)/a_j^c(x)$, where j runs over the index values of the *critical* reactions only. (The value of j_c identifies the next critical reaction, the *only* critical reaction that will occur in this leap.) Set $k_{j_c} = 1$, and for all other critical reactions R_j set $k_j = 0$. For all the non-critical reactions R_j , generate k_j as a sample of the Poisson random variable with mean $a_j(x)\tau$.
6. If there is a negative component in $\mathbf{x} + \sum_j k_j \mathbf{v}_j$, reduce τ' by half, and return to step three. Otherwise, leap by replacing $t \leftarrow t + \tau$ and $\mathbf{x} \leftarrow \mathbf{x} + \sum_j k_j \mathbf{v}_j$; then return to step one, or else stop.

5 Measurement of Simulation Error

Now the only question left in the implementation of the hybrid SSA/tau-leaping method is how to select a proper error control parameter ϵ . From our experience we recommend $\epsilon = 0.03$. We should always keep in mind that the tau-leaping method is just an approximation to the SSA method. Thus there will always be some numerical error, which depends on the error control parameter ϵ . The smaller ϵ is, the more accurate the result will be and the more time it will take to run the simulation. Thus the choice of ϵ is really a balance between accuracy and efficiency. We want an ϵ value small enough so that the overall error is acceptable. But how do we know that a particular ϵ is enough? There is no solid answer. When τ is fixed, it has been proved (Rathinam et al., 2005) that the errors of the mean and variance for the tau-leaping method are linear with τ . But Algorithm 3 is an adaptive method, in the sense that the τ value varies in every simulation step and is directly connected to the choice of ϵ . Our intuition is that the errors should be linear with the value ϵ . But there is no proof for that result. All we can do is to run simulations on many test problems and measure the simulation errors with respect to different values of ϵ . The usual measurement for errors is the difference of the mean and variance between ensembles resulted from the SSA and the hybrid SSA/tau-leaping method. However, for some systems (see the Schlögl example below), the mean and variance do not have a physical meaning. We are more interested in the error of the histogram of certain system properties (this could be the population of some species, or a derived function, such as the cell cycle time, from the simulation trajectory). Thus the histogram distance is introduced to measure the simulation error of the distribution of a scalar random variable.

For a scalar random variable X , the probability density function (pdf) is defined as

$$p_X(x)dx = P(x \leq X < x+dx), \quad (34)$$

for a continuous distribution. For a discrete distribution, the pdf is defined as the δ -function given by

$$p_X(x) = \sum_x P(X=x) \delta(X-x), \quad (35)$$

To measure the error, we need to define a distance between two probability distributions. Suppose X and Y have probability density functions p_X and p_Y . We define the density distance between X and Y as

$$D(X, Y) = \int |p_X(s) - p_Y(s)| ds. \quad (36)$$

When X and Y are integers, (36) becomes

$$D(X, Y) = \sum_n (|P(X=n) - P(Y=n)|). \quad (37)$$

In many practical problems, it is difficult or impossible to obtain an analytic distribution. Instead we obtain samples from Monte Carlo simulations or observations. With those samples, the histogram of the observations is used to estimate the pdf. Let $x_1, x_2, \dots, x_N$ be independent random variables each having the same distribution as X . Defining the sign function

$$\kappa(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}, \quad (38)$$

Suppose that all the sample values are bounded in the interval $I = [x_{\min}, x_{\max}]$. Let $L = x_{\max} - x_{\min}$. Divide the interval I into K subintervals and denote the subintervals by $I_i = [x_{\min} + (i-1)L/K, x_{\min} + iL/K)$. We define the characteristic function $\chi(x, I_i)$ as

$$\chi(x, I_i) = \begin{cases} 1 & \text{if } x \in I_i \\ 0 & \text{otherwise} \end{cases}, \quad (39)$$

Then the pdf p_X can be approximated by the histogram function h_X computed from

$$h_X(I_i) = \frac{K}{NL} \sum_{j=1}^N \chi(x_j, I_i). \quad (40)$$

The sum in (40) gives the number of points falling into the interval I_i . When that sum is divided by N we get the fraction of the points inside that interval, which approximates the probability of a sample point lying inside that interval. We divide this by the interval length, L/K , to approximate the probability density. Thus, $h_X(I_i)$ measures the average density function of X in the interval I_i . When K tends to infinity, the length of I_i reduces to 0. Then I_i is close to a point and h_X is close to p_X at that point.

For two groups of samples X_i and Y_j , we have the *histogram distance*

$$D_K(X, Y) = \sum_{i=1}^K \frac{|h_X(I_i) - h_Y(I_i)| L}{K}. \quad (41)$$

Substituting (40) into (41), we obtain

$$D_K(X, Y) = \sum_{i=1}^K \left| \frac{\sum_{j=1}^N \chi(x_j, I_i)}{N} - \frac{\sum_{j=1}^M \chi(y_j, I_i)}{M} \right|. \quad (42)$$

$D_K(X, Y)$ varies depending on the value of K . When $K = 1$ there is only one subinterval and we cannot tell the difference between X and Y . When K becomes larger we obtain more detailed information about the difference, and $D_K(X, Y)$ will increase. When K is very large we must generate a large number of samples, otherwise there will not be enough data falling into each subinterval and there will be a large measurement error. When K , N and M are sufficiently large, the histogram distance $D_K(X, Y)$ is close to the density distance $D(X, Y)$,

$$D_K(X, Y) \rightarrow D(X, Y) \text{ as } K, N, M \rightarrow \infty. \quad (43)$$

In our numerical experiments, we use the histogram distance to measure the simulation errors.

6 Software and Two Numerical Experiments

In this section we demonstrate the application of the SSA and the hybrid SSA/tau-leaping method to two biochemical systems; a relatively simple toy model, the Schlögl model, and a realistic and far more complex model, the LacZ/LacY model. These comparison tests were performed with the software StochKit on a 1.4Ghz Pentium IV Linux workstation.

6.1 StochKit: a *Stochastic Simulation toolkit*

The above Algorithm 3 has been fully implemented in the package StochKit, a software tool kit for discrete stochastic and multiscale simulation of chemically reacting systems. StochKit is an efficient, extensible stochastic simulation toolkit developed in C++ that aims to make state of the art stochastic simulation algorithms accessible to biologists and chemists, while remaining open to extension via new stochastic and multiscale algorithms. StochKit consists of a suite of software applications for stochastic simulation. The StochKit core implements the simulation algorithms. Additional tools are provided for the convenience of simulation and analysis. A typical simulation process of StochKit is shown in Figure 6.1. A more detailed introduction to StochKit is given in Reference (Li et al. 2008). The StochKit package is freely available for download at www.engr.ucsb.edu/~cse. The User's Guide is also available from that link.

There are several other algorithms in StochKit, such as implicit tau-leaping method (Rathinam et al. 2003), trapezoidal tau-leaping method (Cao and Petzold, 2005), the slow scale SSA (Cao et al., 2005A, 2005D, 2007 and its practical implementation as the multiscale SSA (Cao et al. 2005B). These algorithms are still under development, while the implementations of Gillespie's SSA and the hybrid SSA/tau-leaping method are now mature enough for direct applications to biological systems. To select these two algorithms, configure the solver option in the driver file before calling the solver. There are two preset options: 1 is for the SSA and 0 is for the hybrid SSA/tau-leaping method. The code looks like the following:

```
SolverOptions opt = ConfigStochRxn(1);
```

where the argument 1 selects the SSA, and 0 selects the hybrid SSA/tau-leaping method. To demonstrate the application of the SSA and the hybrid SSA/tau-leaping methods, we apply

both methods to the Schlögl model (Gillespie, 1992) and LacZ/LacY model (Kierzek, 2002, Tian and Burrage, 2004).

6.2 The Schlögl Model

This model is famous for its bistable steady-state distribution. The reactions are

where B_1 and B_2 denote buffered species whose respective molecular populations N_1 and N_2 are assumed to remain essentially constant over the time interval of interest. There is only one time-varying species, X ; the state change vectors are $\nu_1 = \nu_3 = 1$, $\nu_2 = \nu_4 = -1$ and the propensity functions are

$$\begin{aligned} a_1(x) &= \frac{c_1}{2} N_1 x(x-1), \\ a_2(x) &= \frac{c_2}{2} x(x-1)(x-2), \\ a_3(x) &= c_3 N_2, \\ a_4(x) &= c_4 x. \end{aligned} \quad (45)$$

For some values of the parameters this model has two stable states, and that is the case for the parameter values we have chosen here:

$$\begin{aligned} c_1 &= 3 \times 10^{-7}, \quad c_2 = 10^{-4}, \quad c_3 = 10^{-3}, \quad c_4 = 3.5, \\ N_1 &= 10^5, \quad N_2 = 2 \times 10^5. \end{aligned} \quad (46)$$

We made ensembles of 10^5 simulation runs from the initial state $X(0) = 250$ to time $t = 4$ using the SSA and the hybrid SSA/tau-leaping method, the latter for a range of ϵ -values. Fig. 3 shows the histogram distance or “error” between the SSA ensemble and the tau-leaping ensembles as a function of ϵ . We can see that the errors increase roughly linearly with ϵ .

6.3 The LacZ/LacY Model

This model was first proposed by Kierzek (Kierzek, 2002) and later used for an efficiency test in (Tian and Burrage, 2004). This model has 22 reactions, 19 species, and an extremely multiscale nature. A detailed description of this model is omitted here. Interested readers can refer to the two references above and a list of the reaction channels and reaction rates of this model are given in Table 1. It was reported in Tian and Burrage, 2004 that negative populations were observed many times in their simulation using the original tau-leaping method. In our numerical experiments for this model, a single simulation from $t = 0$ to $t = 2100$ by SSA took 3,359 seconds CPU time. With an error tolerance of $\epsilon = 0.03$, a single simulation by the hybrid SSA/tau-leaping method took 113.77 seconds CPU time with no negative population observed during the simulation.

Since a single SSA simulation from $t = 0$ to $t = 2100$ took about an hour on our computer, obtaining a large number of SSA samples posed a challenge. We ran the SSA from time $t = 0$ to time $t = 1000$ to obtain an “initial” state; then we made 10^5 SSA runs from time $t = 1000$ to time $t = 1001$ (which required about 3.5 hours of computer time) and histogrammed the resulting populations. Finally, we made the same number of the SSA/tau-leaping runs

over the same time interval for a range of values for ϵ . Fig. 4 shows the plot of histogram distance or “error” as a function of ϵ . We note again that the error increases roughly linearly with ϵ .

7 Conclusion

In this chapter we have given a review of the two most important simulation algorithms for discrete stochastic systems. Gillespie's SSA has the distinct advantage that it is an exact simulation method but it can be slow for many practical systems. The tau-leaping method gives an approximation of the SSA that is a natural bridge connecting the discrete stochastic SSA regime at one extreme of the approximation to the continuous deterministic RRE regime at the other extreme. However, the approximations of the tau-leaping method can sometimes cause unphysical results such as negative numbers of molecules. The hybrid SSA/tau-leaping method is a practical implementation strategy for the tau-leaping method. Though it is more complex than either the SSA or the tau-leaping method, the hybrid approach combines the strengths of each method, while also avoiding the major pitfalls of each method. The three algorithms detailed in this chapter provide a recipe for implementing each of these three methods. Most importantly, the method of measuring the simulation error is given in detail. The complexity of this measure often leads to its neglect in many applications, a serious oversight for any numerical method.

Acknowledgments

This work was supported by the National Science Foundation under award CCF-0726763, and the National Institutes of Health under award GM073744.

References

- Arkin A, Ross J, McAdams H. Stochastic kinetic analysis of developmental pathway bifurcation in phage λ -infected E. Coli cells. *Genetics*. 1998; 149:1633–1648. [PubMed: 9691025]
- Brenan, KE.; Campbell, SL.; Petzold, LR. Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations. SIAM; Philadelphia, PA: 1996.
- Cao Y, Li H, Petzold L. Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *J Chem Phys*. 2004; 121:4059–4067. [PubMed: 15332951]
- Cao Y, Petzold L. Trapezoidal tau-leaping formula for the stochastic simulation of chemically reacting systems. *Proceedings of Foundations of Systems Biology in Engineering (FOSBE 2005)*. 2005:149–152.
- Cao Y, Gillespie D, Petzold L. The slow-scale stochastic simulation algorithm. *J Chem Phys*. 2005A; 122:014116.
- Cao Y, Gillespie D, Petzold L. Multiscale stochastic simulation algorithm with stochastic partial equilibrium assumption for chemically reacting systems. *J Comput Phys*. 2005B; 206:395–411.
- Cao Y, Gillespie D, Petzold L. Avoiding negative populations in explicit tau leaping. *J Chem Phys*. 2005C; 123:054104. [PubMed: 16108628]
- Cao Y, Gillespie D, Petzold L. Accelerated Stochastic Simulation of the Sti_ Enzyme-Substrate Reaction. *J Chem Phys*. 2005D; 123:144917. [PubMed: 16238434]
- Cao Y, Gillespie D, Petzold L. Efficient Stepsize Selection for the Tau-Leaping Method. *J Chem Phys*. 2006; 124:044109. [PubMed: 16460151]
- Chatterjee A, Vlachos D, Katsoulakis M. Binomial distribution based tauleap accelerated stochastic simulation. *J Chem Phys*. 2005; 122:024112. [PubMed: 15638577]
- Cohen S, Hindmarsh A. CVODE, A Stiff/Nonstiff ODE Solver in C. *Computers in Physics*. 1996; 10:138–143.
- Fedoroff N, Fontana W. Small numbers of big molecules. *Science*. 2002; 297:1129–1131. [PubMed: 12183614]

- Gibson M, Bruck J. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J Phys Chem A*. 2000; 104:1876.
- Gillespie D. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J Comput Phys*. 1976; 22:403–434.
- Gillespie D. Exact stochastic simulation of coupled chemical reactions. *J Phys Chem*. 1977; 81:2340–2361.
- Gillespie, D. *Markov Processes: An Introduction for Physical Scientists*. Academic Press; 1992.
- Gillespie D. Approximate accelerated stochastic simulation of chemically reacting systems. *J Chem Phys*. 2001; 115:1716.
- Gillespie D, Petzold L. Improved leap-size selection for accelerated stochastic simulation. *J Chem Phys*. 2003; 119:8229–8234.
- Gillespie D, Petzold L, Cao Y. Comment on nested stochastic simulation algorithm for chemical kinetic systems with disparate rates [*J. Chem. Phys.* 123, 194107 (2005)]. *J Chem Phys*. 2007; 126:137101. [PubMed: 17430072]
- Hindmarsh, A. ODEPACK, A Systematized Collection of ODE Solvers. In: Stepleman, RS., et al., editors. *Scientific Computing*. Vol. 1. IMACS Transactions on Scientific Computation; 1983. p. 55–64.1983
- Kierzek A. STOCKS: STOChastic Kinetic Simulations of biochemical systems with Gillespie algorithm. *Bioinformatics*. 2002; 18:470–481. [PubMed: 11934747]
- Li H, Petzold L. Logarithmic Direct Method for Discrete Stochastic Simulation of Chemically Reacting Systems. technical report. 2006
- Li H, Cao Y, Petzold L, Gillespie D. Algorithms and software for stochastic simulation of biochemical reacting systems. *Biotechnology Progress*. 2008; 24:56–61. [PubMed: 17894470]
- McAdams H, Arkin A. Stochastic mechanisms in gene expression. *Proc Natl Acad Sci USA*. 1997; 94:814–819. [PubMed: 9023339]
- McCollum JM, Peterson GD, Cox CD, Simpson ML, Samatova NF. The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior. *Computational Biology and Chemistry*. 2006; 30:39–49. [PubMed: 16321569]
- Munsky B, Khammash M. The finite state projection algorithm for the solution of the chemical master equation. *J Chem Phys*. 2006; 124:044101. [PubMed: 16460143]
- Rao C, Arkin A. Stochastic chemical Kinetics and the quasi steady-state assumption: application to the Gillespie algorithm. *J Chem Phys*. 2003; 118:4999–5010.
- Rathinam M, Petzold L, Cao Y, Gillespie D. Stiffness in stochastic chemically reacting systems: the implicit tau-leaping method. *J Chem Phys*. 2003; 119:12784–12794.
- Rathinam M, Petzold L, Cao Y, Gillespie D. Consistency and stability of tau leaping schemes for chemical reaction systems. *SIAM Multiscale Modeling*. 2005; 4:867–895.
- Tian T, Burrage K. Binomial leap methods for simulating stochastic chemical kinetics. *J Chem Phys*. 2004; 121:10356–10364. [PubMed: 15549913]
- Zhang, J.; Watson, L. A modified uniformization method for the chemical master equation. *Proc. 7th IEEE Internat. Conf. on Bioinformatics and Bioengineering*; Boston, MA. 2007. p. 1429–1433.

Figure 1.

The comparison between the SSA and the tau-leaping method. The tau-leaping method leaps over many reactions in one time step.

Figure 2. Simulation Process of StochKit

Figure 3.

Plot of histogram distance errors corresponding to different ϵ values for the Schlögl model. Histogram distance errors are measured by 10^5 samples generated from the SSA method and the hybrid SSA/tau-leaping method using different τ -selection formulas.

Figure 4.

Plot of histogram distance errors corresponding to different ϵ values for the LacZ/LacY model. Histogram distance errors are measured between the population distributions of LacZlactose in 10^5 runs of the SSA and the hybrid SSA/tau-leaping method using different τ -selection formulas.

Table 1

A full list of reaction channels and deterministic reaction rates for LacY/LacZ model.

Reaction Channel	Reaction rate
PLac+RNAP $\rightarrow$ PLacRNAP	0.17
PLacRNAP $\rightarrow$ PLac+RNAP	10
PLacRNAP $\rightarrow$ TrLacZ1	1
TrLacZ1 $\rightarrow$ RbsLacZ+PLac+TrLacZ2	1
TrLacZ2 $\rightarrow$ TrLacY1	0.015
TrLacY1 $\rightarrow$ RbsLacY+TrLacY2	1
TrLacY2 $\rightarrow$ RNAP	0.36
Ribosome+RbsLacZ $\rightarrow$ RbsRibosomeLacZ	0.17
Ribosome+RbsLacY $\rightarrow$ RbsRibosomeLacY	0.17
RbsRibosomeLacZ $\rightarrow$ Ribosome+RbsLacZ	0.45
RbsRibosomeLacY $\rightarrow$ Ribosome+RbsLacY	0.45
RbsRibosomeLacZ $\rightarrow$ TrRbsLacZ+RbsLacZ	0.4
RbsRibosomeLacY $\rightarrow$ TrRbsLacY+RbsLacY	0.4
TrRbsLacZ $\rightarrow$ LacZ	0.015
TrRbsLacY $\rightarrow$ LacY	0.036
LacZ $\rightarrow$ dgrLacZ	6.42×10^{-5}
LacY $\rightarrow$ dgrLacY	6.42×10^{-5}
RbsLacZ $\rightarrow$ dgrRbsLacZ	0.3
RbsLacY $\rightarrow$ dgrRbsLacY	0.3
LacZ+lactose $\rightarrow$ LacZlactose	9.52×10^{-5}
LacZlactose $\rightarrow$ product+LacZ	431
LacY $\rightarrow$ lactose+LacY	14