Modeling and Simulating Chemical Reactions*

Desmond J. Higham†

10th January 2007

Abstract

Many students are familiar with the idea of modeling chemical reactions in terms of ordinary differential equations. However, these deterministic *reaction rate equations* are really a certain large-scale limit of a sequence of finer-scale probabilistic models. In studying this hierarchy of models, students can be exposed to a range of modern ideas in applied and computational mathematics. This article introduces some of the basic concepts in an accessible manner, and points to some challenges that currently occupy researchers in this area. Short, downloadable MATLAB codes are listed and described.

1 Motivation

Our aim here is to use chemical reaction modeling as a means to illustrate three relevant and widely applicable ideas.

1 Mathematical models must rely on *modeling assumptions* and the appropriateness of a model depends entirely on the appropriateness of these assumptions.

2 Complex systems are typically too expensive to simulate in complete detail. Hence, there is a need for *multi-scale* models and corresponding computational algorithms that, to maximise efficiency, can adaptively operate at the most coarse-grained level possible.

*This manuscript appears as University of Strathclyde Mathematics Research Report 02 (2007). A revised version is to appear in SIAM Review Education section.

†Department of Mathematics, University of Strathclyde, Glasgow G1 1XH, UK. Supported by EPSRC grant GR/S62383/01.

3 Continuous valued, deterministic differential equations generally arise from discrete (particle-based) probabilistic models, and moving between these two extremes is often a subtle and poorly-understood business.

We also believe in a fourth principle.

4 The best way to learn about a computational algorithm is to understand and experiment with a simple code.

Hence, focusing on a concrete example, we list short MATLAB programs that illustrate the three main modeling regimes in chemical kinetics. These may be downloaded from

`http://www.maths.strath.ac.uk/~aas96106/algfiles.html`

Slightly longer versions of the programs that produce the figures in this article are also available there.

This treatment should be accessible to students who have been exposed to standard first or second year undergraduate classes on calculus and numerical methods. We also assume a familiarity with basic concepts from probability; in particular, expected value, variance, density functions, and uniformly and normally distributed random variables. Since typical readers are likely to be less familiar with probability and stochastic simulation than with applied mathematics and differential equations, we have added an appendix that expands on some of this material.

The topics that we cover could be incorporated in lectures on differential equations, stochastic simulation, mathematical modeling, computational biology or scientific computing, and could also be the basis of a self-study project.

This article is organised as follows. Section 2 gives a general, equation-free overview of the issues involved. In section 3 we set up some mathematical details and show how the Chemical Master Equation arises. The Stochastic Simulation Algorithm, a computational tool for sampling from the Chemical Master Equation, is described in section 4. In section 5 we introduce tau-leaping as a way to speed up simulations, and in section 6 we show how this modification leads naturally to the Chemical Langevin Equation. The classic Reaction Rate Equation is described in section 7, and shown to be essentially equivalent to the Chemical Langevin Equation when fluctuations are ignored. Using the Michaelis–Menten system, in section 8 we give MATLAB codes and computational results for the three main modeling regimes.

Let us emphasize that this introductory article has been kept as tightly focused as possible and does not contain any original results. In section 9 we point the reader towards more detailed references and mention recent developments, especially in the field of computational cell biology. At this stage, however, it is appropriate to mention that this research area owes a great debt to the pioneering work of Gillespie [10, 11] and that this article was heavily influenced by the expository treatments in [14] and [26].

2 Introduction

In this section we summarize key concepts, without writing down any equations. Our intention is to focus the reader on the big picture. Subsequent sections then flesh out the details.

We are concerned with a process that involves N different types of molecules, or *chemical species*. These molecules may take place in one or more of M types of chemical reactions; for example we may know that "a molecule of species A and a molecule of species B can combine to create a molecule of species C." In principle, we could start with a position and a velocity for each molecule, and let the system evolve under appropriate laws of physics, keeping track of collisions between molecules and the resulting interactions. However, this *molecular dynamics* approach is typically too expensive, computationally, when the overall number of molecules is large or the dynamics over a long time interval are of interest [18]. Instead our approach is to ignore spatial information and simply keep track of the *number* of molecules of each type. This simplification is valid if we have a *well-stirred* system, so that molecules of each type are spread uniformly throughout the spatial domain. We also assume that the system is in thermal equilibrium and that the volume of the spatial domain is constant.

Suppose that initially, at time $t = 0$, we know how many molecules of each species are present and our aim is to describe how these numbers evolve as time increases. We therefore think of a *state vector*,

$$\mathbf{X}(t) = \begin{bmatrix} X_1(t) \\ X_2(t) \\ \vdots \\ X_N(t) \end{bmatrix},$$

where $X_i(t)$ is a non-negative integer that records how many molecules of species i are present at time t.

The state vector $\mathbf{X}(t)$ can change whenever one of the M types of reaction takes

place. In this set-up, because we do not have spatial information, we think in terms of the probability of a reaction taking place, based on the current state of the system. This is analogous to modeling a coin toss—for practical purposes, rather than attempting to follow the dynamics of the coin it is usually acceptable to treat the outcome as a random variable. It is therefore natural to talk about the probability of the system being in a particular state at time t, and to describe the evolution of these probabilities. This leads to the *Chemical Master Equation* (CME), a set of ordinary differential equations (ODEs); one ODE for each possible state of the system. At time t the kth equation gives the probability of the system being in the kth state. The important point here is that the dimension of the ODE is not given by the number of species, N, but by the number of possible states of the system. The latter quantity depends upon the total number of molecules present and the precise form of the chemical reactions, and is usually very large. For example, suppose there are $N = 4$ species A, B, C, D, and $M = 2$ types of reaction

- a molecule of A and a molecule of B can combine to create a molecule of C plus a molecule of D, which we may write as $A + B \to C + D$,
- the reverse reaction, which we may write as $C + D \to A + B$.

If we start with K molecules of type A and K molecules of type B, with no molecules of C or D, so that

$$\mathbf{X}(0) = \begin{bmatrix} K \\ K \\ 0 \\ 0 \end{bmatrix},$$

then the state vector $\mathbf{X}(t)$ takes the possible values values

$$\begin{bmatrix} K \\ K \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K-1 \\ K-1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} K-2 \\ K-2 \\ 2 \\ 2 \end{bmatrix}, \ldots\ldots, \begin{bmatrix} 0 \\ 0 \\ K \\ K \end{bmatrix},$$

so there are $K + 1$ different states.

Generally, the CME has such extremely high dimension that it cannot be handled analytically or computationally.

The *stochastic simulation algorithm* (SSA), also called *Gillespie's algorithm*, gives one approach to computing indirectly with the CME. Here, rather than solving the full set of ODEs to get a probability distribution over all possible states

for each time t, we compute samples from these distributions, that is, compute realisations of the state vector $\{t, \mathbf{X}(t)\}$ in such a way that the chance of a particular realisation being computed reflects the corresponding probability given by the CME.

Although straightforward to implement, the SSA can be impractically slow when reactions occur frequently. We can try to speed up SSA by "lumping together" reactions and only updating the state vector after many reactions have fired. This *tau-leaping* approximation introduces errors that will be small as long as the state vector updates are relatively small. Pushing the approximation further leads to the *Chemical Langevin Equation* (CLE). Here, we have a set of N stochastic differential equations (SDEs), with one component for each chemical species. The state vector, which we write as $\mathbf{Y}(t)$, is now a continuous time, real-valued stochastic process—at each time t, $Y_i(t)$ is a real-valued random variable representing the number of molecules of the ith species. We emphasize that in moving from the CME to the CLE we have

- reduced the dimension of the system from the number of possible states to the number of chemical species,
- relaxed from integers to real values in describing the numbers of molecules for each species,
- changed from a probability distribution over the large but discrete set of states to a continuous probability distribution for each of the N chemical species.

Simplifying from the CME to the CLE is attractive from a computational point of view. It is relatively cheap to simulate the N-dimensional SDE; that is, to compute approximate trajectories $\{t, \mathbf{Y}(t)\}$ whose chance of arising agree with the CLE.

As a further simplification, we can ignore fluctuations in the CLE and regard the deterministic part as our model. This produces the *Reaction Rate Equations* (RREs). This is a set of N ODEs involving a state vector that we write as $\mathbf{y}(t)$. Here, $y_i(t)$ is a real number representing the concentration of the ith species at time t. In comparison with the CME and CLE, simulating with the RRE is an extremely simple task that can be handled by any stiff ODE solving software.

In the following sidebar we summarize the high-level differences between the CME, CLE and RRE philosophies.

sidebar

Chemical Master Equation A set of linear, autonomous ODEs. One ODE for each possible state of the system. Solution of the kth equation at time t is a real number giving the probability of system being in that particular state at time t.

Chemical Langevin Equation A set of nonlinear, autonomous SDEs. One SDE for each chemical species. Solution of jth equation at time t is a real-valued random variable representing the amount of species j at time t.

Reaction Rate Equations A set of nonlinear, autonomous ODEs. One ODE for each chemical species. Solution of jth equation at time t is a real number representing the concentration of species j at time t.

3 Chemical Master Equation

We begin this section with a motivating example of a chemical system.

The names *Michaelis–Menten* are associated with a system involving four species

- a substrate, S_1,
- an enzyme, S_2,
- a complex, S_3, and
- a product, S_4.

The reactions may be written

$$S_1 + S_2 \xrightarrow{c_1} S_3 \tag{1}$$
$$S_3 \xrightarrow{c_2} S_1 + S_2 \tag{2}$$
$$S_3 \xrightarrow{c_3} S_4 + S_2 \tag{3}$$

The role of the *rate constants*, c_1, c_2, c_3, will be explained later in this section. Overall, the enzyme converts substrate into product. It does this by binding with the substrate to form the enzyme-substrate complex; reaction (1). A molecule of this complex may simply dissociate back into enzyme and product; reaction (2). Or it may dissociate into the product (an altered version of the substrate) and the enzyme; reaction (3). This model rules out the possibility of reaction (3) being reversed—product and enzyme cannot recombine to form the complex.

Although simple, the Michaelis–Menten system deals with an extremely important mechanism and hence has been very widely studied. Crampin and Schnell

[6] argue that "almost everything that happens in life boils down to enzymatic catalysis and biochemical kinetics."

We explained in section 2 that in deriving the CME we will not consider the position or velocity of individual molecules, and hence we will be content to describe the system in terms of the state vector $\mathbf{X}(t)$. In the system (1)—(3) we see that, once created, a molecule of the product S_4 is never involved in any further reactions. If we are not interested in the amount of product then we could ignore S_4 and use a state vector that only recorded the amounts of S_1, S_2 and S_3 (since any change of state only depends on these three species). Also, we could recover the state of S_4 from the states of S_1 and S_3 by noting that "$S_1 + S_3 + S_4 = \text{constant}$." However, in this illustration we will work with the full state vector $\mathbf{X}(t) \in \mathbb{R}^4$.

If reaction (1) takes place, then $X_1(t)$ and $X_2(t)$ decrease by one, and $X_3(t)$ increases by one. So $\mathbf{X}(t)$ becomes $\mathbf{X}(t) + \boldsymbol{\nu}_1$, where

$$\boldsymbol{\nu}_1 = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 0 \end{bmatrix}.$$

Similarly, for reactions (2) and (3) we introduce

$$\boldsymbol{\nu}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{\nu}_3 = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1 \end{bmatrix},$$

respectively.

Now a type (1) reaction can only arise when an S_1 molecule meets an S_2 molecule. Intuitively, if there are very few S_1 or S_2 molecules present at some time, then a reaction of this type is less likely to take place than if there are many S_1 and S_2 molecules present. Using this type of argument, we may talk about the probability of this reaction taking place, and we will take this probability to be proportional to the product of the number of S_1 and S_2 molecules present. More precisely, we assume that the probability of this reaction taking place in the infinitesimal time interval $[t, t+dt)$ is given by $c_1 X_1(t) X_2(t) dt$. Here the product $X_1(t)X_2(t)$ relates to the likelihood of two appropriate molecules coming into contact, and the constant c_1 is a scale factor that, among other things, allows for the fact that not every such collision will result in a reaction.

For the second type of reaction, (2), only S_3 has an active role. Hence, we take the corresponding probability to be $c_2 X_3(t) dt$—proportional to the amount of S_3

present. Similarly, we use the probability $c_3 X_3(t)dt$ for the third type of reaction, (3).

This set-up generalizes to any collection of *unimolecular* (one molecule on the left hand side) or *bimolecular* (two molecules on the left hand side) reactions. Generally, we have chemical species $S_1, S_2, \ldots, S_N$ that take part in M different types of chemical reaction, or *reaction channels*. For $1 \leq j \leq M$, the jth type of reaction has an associated *stoichiometric*, or *state-change* vector, $\boldsymbol{\nu}_j \in \mathbb{R}^N$, whose ith component is the change in the number of S_i molecules caused by the jth reaction. So one reaction of type j has the effect of changing the state vector from $\mathbf{X}(t)$ to $\mathbf{X}(t) + \boldsymbol{\nu}_j$.

Also associated with the jth reaction is the *propensity function*, $a_j(\mathbf{X}(t))$, which has the property that the probability of this reaction taking place in the infinitesimal time interval $[t, t+dt)$ is given by $a_j(\mathbf{X}(t))dt$. The propensity functions are constructed as follows.

Second order $S_m + S_n \xrightarrow{c_j}$ something, with $m \neq n$, has $a_j(\mathbf{X}(t)) = c_j X_m(t) X_n(t)$.

Dimerization $S_m + S_m \xrightarrow{c_j}$ something, has $a_j(\mathbf{X}(t)) = c_j \frac{1}{2} X_m(t)(X_m(t) - 1)$.

First order $S_m \xrightarrow{c_j}$ something, has $a_j(\mathbf{X}(t)) = c_j X_m(t)$.

The first and third of these were illustrated above in the Michaelis–Menten example. The second deals with the case where two molecules from the same chemical species interact. The factor $\frac{1}{2} X_m(t)(X_m(t) - 1)$ can be understood from the combinatoric fact that it represents the number of ways of choosing an (unordered) pair of objects from a total of $X_m(t)$.

The forms of the propensity functions should make intuitive sense. In fact, they can be justified rigorously from first principle physical arguments. (This contrasts with the case of the Reaction Rate Equation in section 7, where the modeling arguments are much more ad hoc.)

We are now in a position to study the quantity $P(\mathbf{x}, t)$, which we define to be the probability that $\mathbf{X}(t) = \mathbf{x}$. We assume that $\mathbf{X}(0)$ is given. Our approach is to derive a recurrence. Given that we know the probability of being in any of the possible states at time t, we will work out the probability of being in state $\mathbf{x}$ at time $t + dt$, assuming dt is so small that at most one reaction can take place over $[t, t + dt)$. The first step is to notice that to be in state $\mathbf{x}$ at time $t + dt$ there are only two basic scenarios for time t; either the the system was already in state $\mathbf{x}$ at time t and no reaction took place over $[t, t + dt)$, or for some $1 \leq j \leq M$ the system was in state $\mathbf{x} - \boldsymbol{\nu}_j$ at time t and the jth reaction fired over $[t, t + dt)$,

thereby bringing the system into state $\mathbf{x}$. We need to apply a standard result from probability theory known as the *law of total probability*. Generally, suppose A is the event of interest and suppose that the events $H_0, H_1, H_2, \ldots, H_M, H_{M+1}$ are (a) disjoint (no more than one can happen) and (b) exhaustive (one of them must happen). Then the law of total probability says that

$$\mathbb{P}(A) = \sum_{j=0}^{M+1} \mathbb{P}(A|H_j)\,\mathbb{P}(H_j)\,. \tag{4}$$

Here, $\mathbb{P}(A|H_j)$ means the probability that A happens, given that H_j happens. In our case, A is the event that the system is in state $\mathbf{x}$ at time $t+dt$. We can let H_0 be the event that the system is in state $\mathbf{x}$ at time t, let H_j for $1 \le j \le M$ be the event that the system is in state $\mathbf{x}-\boldsymbol{\nu}_j$ at time t, and let H_{M+1} be the event that the system is in any other state at time t. Now, for $1 \le j \le M$, $\mathbb{P}(A|H_j)$ is simply the probability of the jth reaction firing over $[t, t+dt)$. From the definition of the propensity functions this means

$$\mathbb{P}(A|H_j) = a_j(\mathbf{x}-\boldsymbol{\nu}_j)dt, \qquad 1 \le j \le M. \tag{5}$$

Similarly, $\mathbb{P}(A|H_0)$ is the probability of no reaction firing over $[t, t+dt)$. This must equal one minus the probability of any reaction firing, so

$$\mathbb{P}(A|H_0) = 1 - \sum_{j=1}^{M} a_j(\mathbf{x})dt. \tag{6}$$

Finally,

$$\mathbb{P}(A|H_{M+1}) = 0, \tag{7}$$

because H_{M+1} contains all the states that are more than one reaction way from $\mathbf{x}$. Using (5), (6) and (7) in (4), along with the definition of $P(\mathbf{x},t)$, we find that

$$P(\mathbf{x}, t+dt) = \left(1 - \sum_{j=1}^{M} a_j(\mathbf{x})dt\right) P(\mathbf{x},t) + \sum_{j=1}^{M} a_j(\mathbf{x}-\boldsymbol{\nu}_j)dtP(\mathbf{x}-\boldsymbol{\nu}_j, t).$$

This equation rearranges to

$$\frac{P(\mathbf{x},t+dt) - P(\mathbf{x},t)}{dt} = \sum_{j=1}^{M} \left(a_j(\mathbf{x}-\boldsymbol{\nu}_j)P(\mathbf{x}-\boldsymbol{\nu}_j,t) - a_j(\mathbf{x})P(\mathbf{x},t)\right).$$

Letting $dt \to 0$ we see that the left-hand side of this equation becomes a time derivative, leading to the Chemical Master Equation

$$\frac{dP(\mathbf{x},t)}{dt} = \sum_{j=1}^{M} \left(a_j(\mathbf{x}-\boldsymbol{\nu}_j)P(\mathbf{x}-\boldsymbol{\nu}_j,t) - a_j(\mathbf{x})P(\mathbf{x},t)\right). \tag{8}$$

We emphasize here that the state vector $\mathbf{x}$ takes a finite (but large) number of possible values, and the CME (8) is a linear ODE system with one ODE for each possible state.

4 Stochastic Simulation Algorithm

Typically, the CME is too high-dimensional to deal with computationally. The SSA gets around this issue by computing single realisations of the state vector rather than an entire probability distribution.

To derive the SSA we introduce the quantity $P_0(\tau|\mathbf{x},t)$, where

> given $\mathbf{X}(t)=\mathbf{x}$, $P_0(\tau|\mathbf{x},t)$ is the probability that no reaction takes place in the time interval $[t,t+\tau)$.

Now we consider the time interval $[t,t+\tau+d\tau)$. We assume that what happens over $[t,t+\tau)$ is independent of what happens over $[t+\tau,t+\tau+d\tau)$, so that "and" translates into "product." In words, we have

$$\begin{aligned}
\text{Prob. no reaction over } [t,t+\tau+d\tau) &= \text{Prob. no reaction over } [t,t+\tau) \\
&\quad \text{and no reaction over } [t+\tau,t+\tau+d\tau) \\
&= \text{Prob. no reaction over } [t,t+\tau) \times \\
&\quad \text{Prob. no reaction over } [t+\tau,t+\tau+d\tau) \\
&= \text{Prob. no reaction over } [t,t+\tau) \times \\
&\quad (1-\text{sum of prob. each reaction over } [t+\tau,t+\tau+d\tau))\,.
\end{aligned}$$

Using the definition of the propensity function, this may be written

$$P_0(\tau+\delta\tau|\mathbf{x},t) = P_0(\tau|\mathbf{x},t)\left(1-\sum_{k=1}^{M} a_k(\mathbf{x})d\tau\right),$$

that is,

$$\frac{P_0(\tau+\delta\tau|\mathbf{x},t)-P_0(\tau|\mathbf{x},t)}{d\tau} = -a_{\text{sum}}(\mathbf{x}), \quad \text{where } a_{\text{sum}}(\mathbf{x}) := \sum_{k=1}^{M} a_k(\mathbf{x}).$$

Passing to the limit as $d\tau \to 0$ leads to a linear scalar ODE for P_0 that is readily solved to give

$$P_0(\tau|\mathbf{x},t) = e^{-a_{\text{sum}}(\mathbf{x})\tau}. \tag{9}$$

Now, the key quantity for the SSA is $p(\tau,j|\mathbf{x},t)$, which is defined by

> given $\mathbf{X}(t)=\mathbf{x}$, $p(\tau,j|\mathbf{x},t)d\tau$ is the probability that the next reaction (a) will be the jth reaction, and (b) will occur in the time interval $[t+\tau,t+\tau+d\tau)$.

In words, with "and" becoming a product again, we have

$$\begin{aligned}\text{Prob. (a) and (b)} \;=\; &\text{Prob. no reaction took place over } [t, t+\tau) \\ &\times \text{Prob. } j\text{th reaction took place over } [t+\tau, t+\tau+d\tau).\end{aligned}$$

(Here, we assume that $d\tau$ is so small that at most one reaction can take place over that length of time.) Using the definitions of P_0 and a_j, this translates to

$$p(\tau, j|\mathbf{x}, t)d\tau = P_0(\tau|\mathbf{x}, t)a_j(\mathbf{x})d\tau,$$

so, from (9),

$$p(\tau, j|\mathbf{x}, t) = a_j(\mathbf{x})e^{-a_{\text{sum}}(\mathbf{x})\tau}.$$

This is conveniently re-written as

$$p(\tau, j|\mathbf{x}, t) = \frac{a_j(\mathbf{x})}{a_{\text{sum}}(\mathbf{x})} a_{\text{sum}}(\mathbf{x})e^{-a_{\text{sum}}(\mathbf{x})\tau}. \tag{10}$$

Formally, $p(\tau, j|\mathbf{x}, t)$ is the joint density function for two random variables

- next reaction index, and
- time until next reaction,

and (10) shows that it may be written as the product of two individual density functions.

Next reaction index $a_j(\mathbf{x})/a_{\text{sum}}(\mathbf{x})$ corresponds to a discrete random variable: pick one of the reactions with the rule that the chance of picking the jth reaction is proportional to $a_j(\mathbf{x})$.

Time until next reaction $a_{\text{sum}}(\mathbf{x})e^{-a_{\text{sum}}(\mathbf{x})\tau}$ is the density function for a continuous random variable with an *exponential distribution*. These exponential random variables arise universally in descriptions of the time elapsing between unpredictable events.

From a computational perspective this is a very important observation. It allows us to simulate independently a reaction index and a reaction time. Each can be computed via a uniform (0,1) sample. The details are spelled out in the Appendix. The resulting algorithm can be summarized very simply in the following pseudocode, where an initial state $\mathbf{X}(0)$ is given.

1. Evaluate $\{a_k(\mathbf{X}(t))\}_{k=1}^M$ and $a_{\text{sum}}(\mathbf{X}(t)) := \sum_{k=1}^M a_k(\mathbf{X}(t))$.

2. Draw two independent uniform (0,1) random numbers, ξ_1 and ξ_2.

3. Set j to be the smallest integer satisfying $\sum_{k=1}^{j} a_k(\mathbf{X}(t)) > \xi_1 a_{\text{sum}}(\mathbf{X}(t))$.

4. Set $\tau = \ln(1/\xi_2)/a_{\text{sum}}(\mathbf{X}(t))$.

5. Set $\mathbf{X}(t+\tau) = \mathbf{X}(t) + \boldsymbol{\nu}_j$.

6. Return to step 1.

In practice, of course, step 6 would also include a termination condition; for example, stop when t has passed a specified value, when some species exceeds a specified upper or lower bound, or when a specified number of iterations have been taken.

5 Tau Leaping

SSA is exact, in the sense that the statistics from the CME are reproduced precisely. However, this exactness comes at a high computational price. If there are many molecules in the system and/or some fast reactions then this can result in a huge amount of random number generation and bookkeeping—at each iteration, a reaction time and reaction index must be drawn, and then the state vector and propensity functions must be updated. It is therefore attractive to consider a fixed time interval length, τ, and to fire simultaneously all reactions that would have taken place. More precisely, given $\mathbf{X}(t)$, we could freeze the propensity functions $a_j(\mathbf{X}(t))$ and, using these values, fire an appropriate number of reactions of each type. This gives the tau-leaping method

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \sum_{j=1}^{M} \boldsymbol{\nu}_j \mathcal{P}_j(a_j(\mathbf{X}(t)), \tau), \tag{11}$$

where the random variables $\{\mathcal{P}_j(a_j(\mathbf{X}(t)), \tau)\}_{j=1}^{M}$ must now be determined.

In order for this approximation to SSA to be valid, we require that τ is sufficiently small that *relatively few* reactions take place, in the sense that the propensity functions $a_j(\mathbf{X}(t))$ would not have changed very much if we had taken the effort to update after each reaction.

Now if $a_j(\mathbf{X}(t))$ were to stay exactly constant over $[t, t+\tau)$ then the number of type j reactions to fire would be given by a *counting process*: we know that the probability of the jth reaction firing over a small time interval of length $d\tau$ is given by $a_j(\mathbf{X}(t))d\tau$, and we need to count how many of these events arise over

$[t, t+\tau)$. It follows that the number of reactions, $\mathcal{P}_j(a_j(\mathbf{X}(t)), \tau)$, will have what is known as a Poisson distribution with parameter $a_j(\mathbf{X}(t))\tau$. (Generally a Poisson random variable P with parameter $\lambda > 0$ takes possible values $\{0, 1, 2, 3, \ldots\}$ such that $\mathbb{P}(P = i) = e^{-\lambda}\frac{\lambda^i}{i!}$.)

Overall, one step of the tau-leaping method with leap time τ can be summarized as follows.

1. For $j = 1$ to M, compute a sample p_j from the distribution of the Poisson random variable $\mathcal{P}_j(a_j(\mathbf{X}(t)), \tau)$.

2. Set $\mathbf{X}(t + \tau) = \mathbf{X}(t) + \sum_{j=1}^{M} \boldsymbol{\nu}_j p_j$.

In practice, the leap time, τ, can be chosen adaptively, based on the current state vector and propensity function values.

6 Chemical Langevin Equation

In addition to being a viable computational approach, tau-leaping can also be viewed as a link that connects the CME to a coarser grained model. The Poisson random variable $\mathcal{P}_j(a_j(\mathbf{X}(t)), \tau)$ has mean $a_j(\mathbf{X}(t))\tau$, so this is the expected number of times for the jth reaction to fire over $[t, t + \tau)$. The variance of $\mathcal{P}_j(a_j(\mathbf{X}(t)), \tau)$ is also $a_j(\mathbf{X}(t))\tau$. Now suppose the leap time τ is chosen so that $a_j(\mathbf{X}(t))\tau$ is large for all $1 \leq j \leq M$. A standard result from probability theory says that a Poisson random variable with large mean is well approximated by a normal random variable with the same mean and variance. Hence, in the tau-leaping method it is reasonable to change from Poisson to normal if *every* reaction is expected to fire many times over $[t, t+\tau)$. (Recall that tau-leaping was motivated under a different assumption—that each $a_j(\mathbf{X}(t))$ will have a relatively small change over $[t, t + \tau)$. This is not incompatible with the large $a_j(\mathbf{X}(t))\tau$ assumption, and we are now hoping that both are valid.)

Replacing each $\mathcal{P}_j(a_j(\mathbf{X}(t)), \tau)$ in (11) with $a_j(\mathbf{X}(t))\tau + \sqrt{a_j(\mathbf{X}(t))\tau}Z_j$, where the Z_j are independent normal (0,1) random variables, we arrive at the recurrence

$$\mathbf{Y}(t + \tau) = \mathbf{Y}(t) + \tau \sum_{j=1}^{M} \boldsymbol{\nu}_j a_j(\mathbf{Y}(t)) + \sqrt{\tau} \sum_{j=1}^{M} \boldsymbol{\nu}_j \sqrt{a_j(\mathbf{Y}(t))} Z_j. \qquad (12)$$

Note that we have switched from integer-valued Poisson random variables to real-valued normal random variables. To mark the fact that real numbers are now

being used to describe the amount of each species present, we are using a different symbol, $\mathbf{Y}(t)$, for the state vector.

Although continuous in the state variables, the recurrence (12) runs over discrete time. It generates a sequence of random variables $\{\mathbf{Y}(0), \mathbf{Y}(\tau), \mathbf{Y}(2\tau), \ldots\}$ corresponding to the state vector at discrete times $\{0, \tau, 2\tau, \ldots\}$. Computationally, we could simulate a sample path of this process by taking steps of the following form.

1. For $j = 1$ to M, compute a sample z_j from the normal (0,1) distribution.

2. Set $\mathbf{Y}(t+\tau) = \mathbf{Y}(t) + \tau \sum_{j=1}^{M} \boldsymbol{\nu}_j a_j(\mathbf{Y}(t)) + \sqrt{\tau} \sum_{j=1}^{M} \boldsymbol{\nu}_j \sqrt{a_j(\mathbf{Y}(t))}\, z_j$.

Moreover, readers familiar with numerical methods for stochastic differential equations (SDEs) will recognize (12) as an Euler–Maruyama discretization of the continuous time problem

$$d\mathbf{Y}(t) = \sum_{j=1}^{M} \boldsymbol{\nu}_j a_j(\mathbf{Y}(t))dt + \sum_{j=1}^{M} \boldsymbol{\nu}_j \sqrt{a_j(\mathbf{Y}(t))}\, dW_j(t), \tag{13}$$

where the $W_j(t)$ are independent scalar Brownian motions. We refer to [15] for an introduction to numerical SDEs. Readers not familiar with SDEs can simply accept that in the limit $\tau \to 0$ the discrete time recurrence (12) converges to a continuous time process $\mathbf{Y}(t)$, and (13) is a way of writing the equation satisfied by this limiting process. Further, the standard way to solve (13) numerically is to use the recurrence (12) with a sufficiently small τ.

The system (13) is called the Chemical Langevin Equation. For every $t \geq 0$, its solution is a random variable with N components, describing the state of each species.

It is worthwhile to recap the the assumptions that were used to take us from the SSA, via tau-leaping, to the CLE.

1. Each propensity function $a_j(\mathbf{X}(t))$ will undergo a relatively small change over each $[t, t+\tau)$.
2. Each of the products $a_j(\mathbf{X}(t))\tau$ will be large.
3. The leap time τ will be small enough for the numerical method (12) to give a good approximation to the SDE (13).

Points 1 and 3 ask for τ to be "small enough", whereas point 2 asks for τ to be "large enough". Noting that the propensity functions depend linearly or quadratically on the state vector components, we see that to meet these three requirements simultaneously, we must have a large number of molecules present.

7 Reaction Rate Equation

If we simply ignore the stochastic part of the CLE, we arrive at the set of ODEs

$$\frac{d\mathbf{Y}(t)}{dt} = \sum_{j=1}^{M} \boldsymbol{\nu}_j a_j(\mathbf{Y}(t)). \tag{14}$$

We will show in this section that these ODEs essentially correspond to the "textbook" model for chemical kinetics.

The classical approach to chemical kinetics deals with a state vector $\mathbf{Y}(t) \in \mathbb{R}^M$, with $Y_i(t)$ a nonnegative real number representing the concentration of species S_i at time t. Concentrations are usually measured in M (moles per litre). Avagadro's constant, $n_{\mathrm{A}} \approx 6.023 \times 10^{23}$, gives the number of molecules in a mole, So a concentration $Y_i(t)$M of species S_i in a fixed volume of vol litres corresponds to $Y_i(t)n_{\mathrm{A}}$vol molecules of that species.

In this setting the concentrations are assumed to vary continuously in time, according to the reaction rate equation. An empirical rule of thumb, called the *law of mass action* is used to determine the RRE. Essentially, this law says that each reaction in the system affects the rate of change of the species involved. More precisely, the effect on the instantaneous rate of change is proportional to the product of the concentrations of the reacting species.

For the Michaelis–Menten system (1)–(3), letting the reaction constants be denoted k_1, k_2 and k_3, the RRE reads

$$\begin{aligned}
\frac{dY_1(t)}{dt} &= -k_1 Y_1(t) Y_2(t) + k_2 Y_3(t), \\
\frac{dY_2(t)}{dt} &= -k_1 Y_1(t) Y_2(t) + (k_2 + k_3) Y_3(t), \\
\frac{dY_3(t)}{dt} &= k_1 Y_1(t) Y_2(t) - (k_2 + k_3) Y_3(t), \\
\frac{dY_4(t)}{dt} &= k_3 Y_3(t).
\end{aligned}$$

For example, to obtain the second equation we note that species S_2 is involved in all three reactions. Reaction (1) decreases the amount of S_2 at a rate proportional

to the product of the two reactants involved, giving the term $-k_1 Y_1(t)Y_2(t)$. Reaction (2) increases the amount of S_2 at a rate proportional to the single reactant, giving the term $k_2 Y_3(t)$. Similarly, reaction (3) gives the term $k_3 Y_3(t)$.

Generally, in terms of the stoichiometric vectors $\{\boldsymbol{\nu}_j\}$, the RRE can be constructed as follows.

Second order $S_m + S_n \xrightarrow{k_j}$ something, with $m \neq n$, contributes $\boldsymbol{\nu}_j k_j Y_m(t) Y_n(t)$ to the RRE system.

Dimerization $S_m + S_m \xrightarrow{k_j}$ something, contributes $\boldsymbol{\nu}_j k_j Y_m(t)^2$ to the RRE system.

First order $S_m \xrightarrow{k_j}$ something, contributes $\boldsymbol{\nu}_j k_j Y_m(t)$ to the RRE system.

We may now convince ourselves that this RRE description looks very much like the CLE with the noise terms switched off, (14). To do this, we must compare the appropriate terms for each of our reaction types, remembering how to convert between concentrations and molecule counts.

Second order $S_m + S_n \rightarrow$ something, with $m \neq n$. Here the RRE law has a concentration-based rate of change of $k_j Y_m(t) Y_n(t) M s^{-1}$. With $X_m(t) = Y_m(t) n_{\mathrm{A}} \mathrm{vol}$ and $X_n(t) = Y_n(t) n_{\mathrm{A}} \mathrm{vol}$ denoting the number of molecules of S_m and S_n, this rate is equivalent to $k_j X_m(t) X_n(t) / (n_{\mathrm{A}} \mathrm{vol})$ molecules per second. Equating this with the propensity function value, $c_j X_m(t) X_n(t)$, gives the relation

$$c_j = \frac{k_j}{n_{\mathrm{A}} \mathrm{vol}}. \tag{15}$$

Dimerization $S_m + S_m \rightarrow$ something. Now the concentration-based rate $k_j Y_m(t)^2$ corresponds to a molecular rate of $2 \times k_j X_m(t)^2 / (n_{\mathrm{A}} \mathrm{vol})$. Equating this with the propensity function value, $2 \times c_j X_m(t) \left(X_m(t) - 1\right) / 2$, we arrive at the relation

$$c_j \approx \frac{2k_j}{n_{\mathrm{A}} \mathrm{vol}}. \tag{16}$$

Here, "$\approx$" appears because $X_m(t)(X_m(t) - 1)$ does not equal $X_m(t)^2$, but approximates it well when $X_m(t)$ is large.

First order $S_m \rightarrow$ something. In this case the deterministic rate $k_j Y_m(t)$ gives a molecular rate of $k_j X_m(t)$, exactly matching the rate from the propensity function, so

$$c_j = k_j. \tag{17}$$

In summary, the classical RRE model can be regarded as arising from the CLE (13) when the stochastic terms are removed, the reactions constants are converted according to (15)–(17) and, in the case of dimerization-type reactions, the "large molecule number" approximation $X_m(t)(X_m(t)-1) \approx X_m(t)^2$ is used.

It is, of course, no coincidence that the CME and CLE are related. The idea of modeling in terms of concentrations and instantaneous rates of change presupposes that large numbers of molecules are present. Formally, we can think of a *thermodynamic limit* in which the system volume, vol, and the species populations, $X_i(t)$, tend to ∞, but with the species concentrations $X_i(t)/\text{vol}$ remaining constant. In this limit, the deterministic coefficients in the CME grow like the system size, but the stochastic coefficients grow like the square root of the system size, and hence the ODE part dominates. So, in this well-defined sense, the RRE is a logical consequence of the CME model under appropriate simplifying assumptions.

Since the propensity function approach of section 3 can be justified from first principles, these arguments put the "mass action" framework on a firmer footing, but also emphasize that the RRE can only be valid when there are large numbers of molecules of each species.

An important point is that the RRE solution, in general, does not correspond to the "average" solution from the CLE or CME. So, for example, it cannot be argued that solving the RRE is equivalent to computing many solution paths of the CLE and then forming an ensemble average. The underlying reason for this mismatch is that $\mathbb{E}[X^2] \neq (\mathbb{E}[X])^2$ for a general random variable X, and $\mathbb{E}[XY] \neq \mathbb{E}[X]\mathbb{E}[Y]$ for general non-independent random variables X and Y. Hence the quadratic terms arising from second order/dimerization reactions do not succumb easily to the expectation operator. So the RRE is not computing averages, it is computing the thermodynamic limit.

8 Computational Experiment

Our purpose in this section is to give numerical simulations on the Michaelis–Menten system (1)–(3) using the SSA, CLE and RRE frameworks.

We will use initial data and rate constants from [31, Section 7.3]. Here, the

concentration-based data is

$$\begin{bmatrix} Y_1(0) \\ Y_2(0) \\ Y_3(0) \\ Y_4(0) \end{bmatrix} = \begin{bmatrix} 5 \times 10^{-7} M \\ 2 \times 10^{-7} M \\ 0M \\ 0M \end{bmatrix}, \qquad k_1 = 10^6,\ k_2 = 10^{-4},\ k_3 = 10^{-1},$$

within a volume of $\text{vol} = 10^{-15}$ litres. This corresponds to molecular data of

$$\begin{bmatrix} X_1(0) \\ X_2(0) \\ X_3(0) \\ X_4(0) \end{bmatrix} = \begin{bmatrix} \lfloor 5 \times 10^{-7} n_{\text{A}} \text{vol} \rfloor \\ \lfloor 2 \times 10^{-7} n_{\text{A}} \text{vol} \rfloor \\ 0 \\ 0 \end{bmatrix}, \qquad c_1 = \frac{10^6}{n_{\text{A}} \text{vol}},\ c_2 = 10^{-4},\ c_3 = 10^{-1},$$

where $\lfloor \cdot \rfloor$ denotes rounding to the nearest integer. We will simulate reactions until the time exceeds $t = 50$.

Since $X_1(0) = 312$ and $X_2(0) = 125$, we are in the regime of a reasonably small number of molecules, with roughly 2.5 times as many substrate molecules as enzyme molecules, initially, and no complex or product.

The code `ssa_mm` in Listing 8.1 implements SSA for this system. This is a straightforward translation into MATLAB [16] of the pseudocode summary in section 4, making use of the built-in uniform random number generator, `rand`. Perhaps the only non-transparent line of code is

```
j = min(find(rand<cumsum(a/asum)));
```

Here, `cumsum(a/asum)` is a vector whose kth component is the cumulative sum of the first k components of `a/asum`; that is

$$\left[\frac{a(1)}{\text{asum}}, \quad \frac{a(1)+a(2)}{\text{asum}}, \quad \frac{a(1)+a(2)+a(3)}{\text{asum}} \right].$$

By construction, the final component equals one. Then `find(rand<cumsum(a/asum))` is a vector giving the indices for those components of `cumsum(a/asum)` that exceed the random number. So, `min(find(rand<cumsum(a/asum)))` records the smallest such index. This is precisely the required reaction index.

Figure 1 shows the evolution of the number of molecules of substrate, `X(1)`, and product, `X(4)`. To emphasize that these are integer valued quantities changing at discrete points in time, in Figure 2 we zoom in on part of the plot. The circles and asterisks denote substrate and product counts, respectively, before and after each reaction.

```
%SSA_MM.M
%
% Simple implementation of the Stochastic Simulation Algorithm
% (or Gillespie's algorithm) on the Michaelis-Menten system.
%
% Parameters from Chapter 7 of
%      Stochastic Modelling for Systems Biology,
%      by Darren J. Wilkinson, Chapman & Hall/CRC, 2006.
%
% Downloadable from
%    http://www.maths.strath.ac.uk/~aas96106/algfiles.html
% along with an extended version that produces graphical output.

rand('state',100)

%stoichiometric matrix
V = [-1 1 0; -1 1 1; 1 -1 -1; 0 0 1];

%%%%%%%%% Parameters and Initial Conditions %%%%%%%%
nA = 6.023e23;             % Avagadro's number
vol = 1e-15;               % volume of system
X = zeros(4,1);
X(1) = round(5e-7*nA*vol); % molecules of substrate
X(2) = round(2e-7*nA*vol); % molecules of enzyme
c(1) = 1e6/(nA*vol); c(2) = 1e-4; c(3) = 0.1;

t = 0;
tfinal = 50;
while t < tfinal
     a(1) = c(1)*X(1)*X(2);
     a(2) = c(2)*X(3);
     a(3) = c(3)*X(3);
     asum = sum(a);
     j = min(find(rand<cumsum(a/asum)));
     tau = log(1/rand)/asum;
     X = X + V(:,j);
     t = t + tau;

     % Record or plot system state here if required
end
```

Listing 8.1: Listing of `ssa_mm.m`.

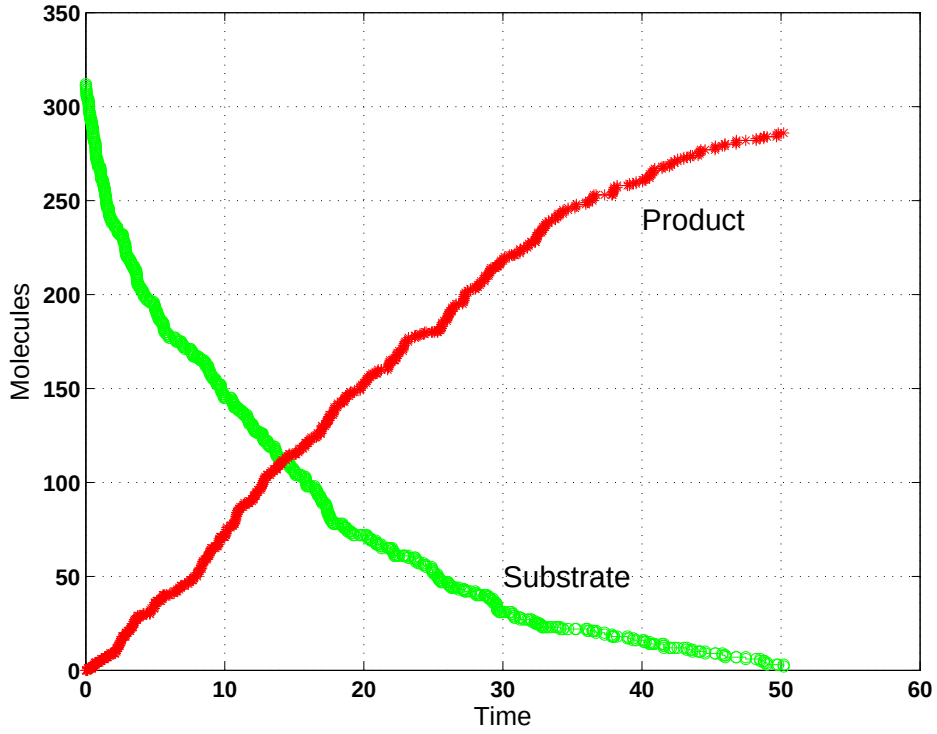

Figure 1: Molecules of substrate and product from a run of SSA on the Michaelis–Menten system.

Listing 8.2 shows `cle_mm.m`, which implements the Euler–Maruyama method (12) on the CLE for the same system, with a stepsize $\tau = 0.2$. In this case, MATLAB's normal random number generator, `randn`, is used. We note that the state vector is not guaranteed to remain non-negative in this computation, and hence we apply `abs` before taking square roots. For comparison with Figure 1, we show in Figure 3 the evolution of substrate and product. The underlying CLE produces paths in continuous time that are, with probability one, continuous but not differentiable. The Euler–Maruyama method gives a discrete time approximation to these paths. In Figure 4 we zoom in on part of Figure 3 to show some of the detail.

Finally, the function, `rre_mm.m` in Listing 8.3 applies MATLAB's built-in stiff ODE solver `ode15s` to the RRE formulation. The format is based on `rossler_ex1.m` from [16, Listing 12.1], and a nested function is used to specify the ODE system. Figure 5 shows the substrate and product curves. In this case, the underlying RRE model produces continuous time, smooth solutions, and the symbols in the plot simply indicate where `ode15s` has chosen to discretize for the purpose of numerical approximation. We note that this model uses concentrations and has output in moles per litre. The absolute size of the solution is quite small, of the

```
%CLE_MM.M
%
% Simple implementation of Euler--Maruyama to simulate the
% Chemical Langevin Equation for the Michaelis-Menten system.
%
% Parameters from Chapter 7 of
%      Stochastic Modelling for Systems Biology,
%      by Darren J. Wilkinson, Chapman & Hall/CRC, 2006.
%
% Downloadable from
%    http://www.maths.strath.ac.uk/~aas96106/algfiles.html
% along with an extended version that produces graphical output.

clf

randn('state',100)

%stoichiometric matrix
V = [-1 1 0; -1 1 1; 1 -1 -1; 0 0 1];

%%%%%%%%% Parameters and Initial Conditions %%%%%%%%%
nA = 6.023e23;             % Avagadro's number
vol = 1e-15;               % volume of system
Y = zeros(4,1);
Y(1) = round(5e-7*nA*vol); % molecules of substrate
Y(2) = round(2e-7*nA*vol); % molecules of enzyme
c(1) = 1e6/(nA*vol); c(2) = 1e-4; c(3) = 0.1;

tfinal = 50;
M = 250;
tau = tfinal/M;    % stepsize

for k = 1:M
     a(1) = c(1)*Y(1)*Y(2);
     a(2) = c(2)*Y(3);
     a(3) = c(3)*Y(3);
     d(1) = tau*a(1) + sqrt(abs(tau*a(1)))*randn;
     d(2) = tau*a(2) + sqrt(abs(tau*a(2)))*randn;
     d(3) = tau*a(3) + sqrt(abs(tau*a(3)))*randn;
     Y = Y + d(1)*V(:,1) + d(2)*V(:,2) + d(3)*V(:,3);

     % Record or plot system state here if required
end
```


Listing 8.2: Listing of `cle_mm.m`.

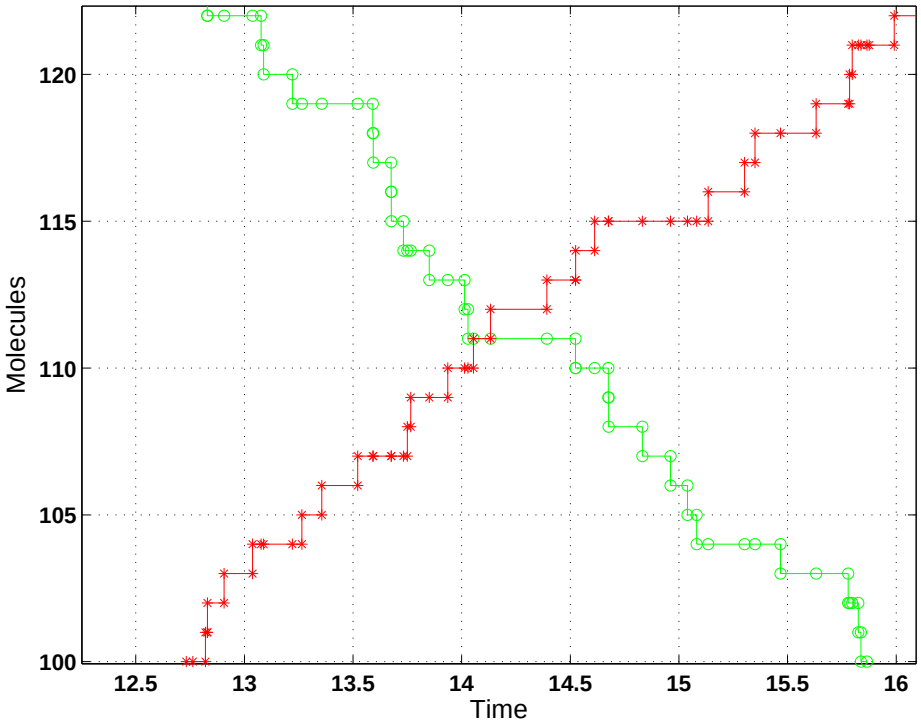

Figure 2: A zoom in on the picture in Figure 1.

order 10^{-7}, which can cause difficulties for an ODE solver. We found that using the default error tolerances in `ode15s` produced qualitatively incorrect results, hence in `rre_mm.m` we specify a more stringent absolute tolerance via `odeset`.

9 Discussion and Further Reading

For the computations in section 8, even with molecule counts as low as the hundreds, the RRE gave a reasonable match to the single paths that we drew from the CLE and SSA. In general, whether the RRE is an acceptable model depends not only on the initial data and system parameters, but also on the purpose for which the model is to be used. Biochemistry within the cell is an important example where, in many cases, the law of mass action is inappropriate. Many cellular processes involve extremely small population sizes, where it is unrealistic to think in terms of concentration, and often the system response depends critically on the precise quantitative values. A related issue is that in addition

```
function rre_mm
%
% ODE15s solution the Reaction Rate Equation for
% the Michaelis-Menten system.
%
% Parameters from Chapter 7 of
%      Stochastic Modelling for Systems Biology,
%      by Darren J. Wilkinson, Chapman & Hall/CRC, 2006.
%
% Downloadable from
%    http://www.maths.strath.ac.uk/~aas96106/algfiles.html
% along with an extended version that produces graphical output.

tspan = [0 50]; yzero = [5e-7; 2e-7; 0; 0];
options = odeset('AbsTol',1e-8);
k1 = 1e6; k2 = 1e-4; k3 = 0.1;

[t,y] = ode15s(@mm_rre,tspan,yzero,options);

% Record or plot (t,y) at this stage

      %--------------Nested function----------
      function yprime = mm_rre(t,y)
      % MM_RRE    Michaelis-Menten Reaction Rate Equation
      yprime = zeros(4,1);
      yprime(1) = -k1*y(1)*y(2) + k2*y(3);
      yprime(2) = -k1*y(1)*y(2) + (k2+k3)*y(3);
      yprime(3) =  k1*y(1)*y(2) - (k2+k3)*y(3);
      yprime(4) =  k3*y(3);
      end

end
```

Listing 8.3: Listing of `rre_mm.m`.

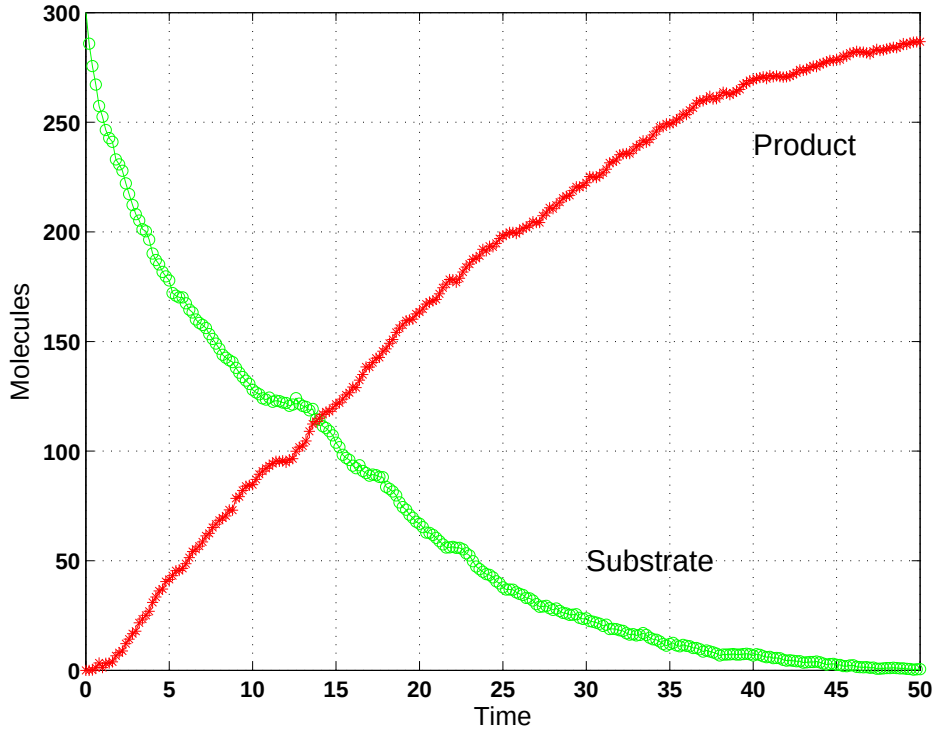

Figure 3: Molecules of substrate and product from an Euler–Maruyama simulation of the Chemical Langevin Equation for the Michaelis–Menten system.

to discreteness, stochasticity is often a vital component. For example, a biological system may exhibit *bi-stability*, switching between two distinct states. Here, the switching is driven purely by the inherent noise in the system—a sufficiently large excursion can cause a path to move from one state to another. Rather than an "ensemble average," quantitative measures such as "average time between switches" and "relative time spent in each state" are more likely to be of interest. These properties can be computed naturally from large scale stochastic simulations.

We finish with some pointers to the literature.

9.1 Classics

The stochastic simulation algorithm was invented by Dan Gillespie. The classic early references [10, 11] remain highly pertinent. Streamed video of a 2004 seminar titled *Stochastic Chemical Kinetics* given by Gillespie at the Mathematical Biosciences Institute, Ohio State University, can be downloaded from

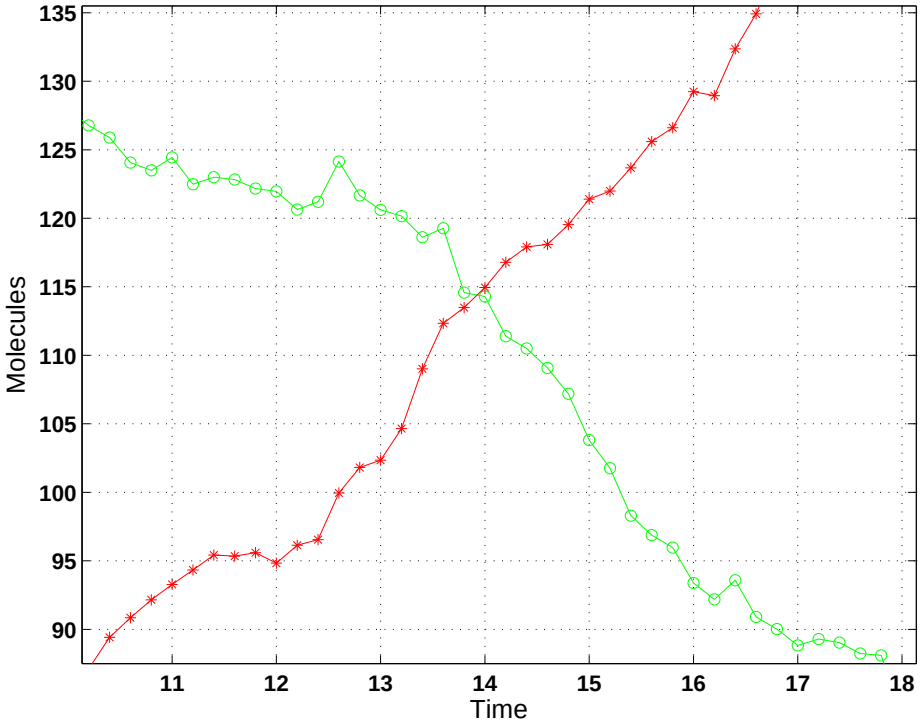

Figure 4: A zoom in on the picture in Figure 3.

`http://mbi.osu.edu/2004/ws2abstracts.html`

There are many viewpoints on the Chemical Master Equation and Chemical Langevin Equation. Good starting points are [8, 12, 30]. The original tau-leaping idea is also due to Gillespie [13].

9.2 Expository

For further expository reading we strongly recommend [14] and [26], both of which greatly influenced this article. The reference [20] provides a simple tutorial, with MATLAB code. The recent text [31] gives a general exposition from the viewpoint of systems biology and discusses many issues, including the task of infering parameters from experimental data.

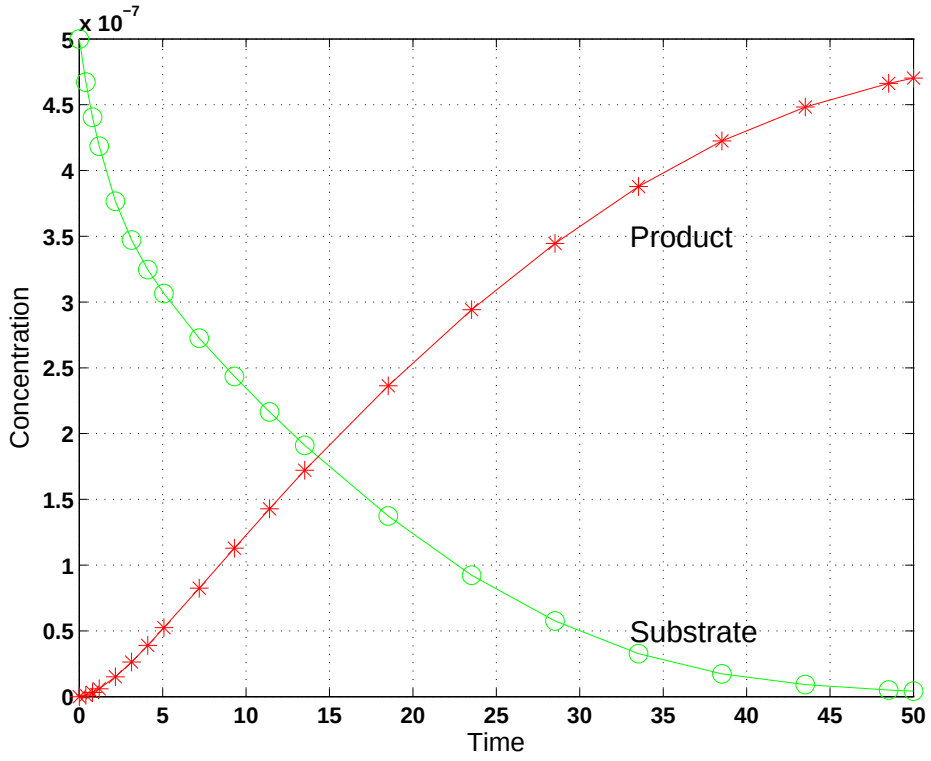

Figure 5: Concentrations of substrate and product from MATLAB's `ode15s` solver applied to the Reaction Rate Equation for the Michaelis–Menten system. Using absolute error tolerance of `AbsTol=1e-8`.

9.3 Biochemistry

Computational cell biology is perhaps the most high-profile area where the issues of discrete/stochastic versus continuous/deterministic arise. There is now a vast array of literature involving Gillespie-style simulations of cellular processes. An excellent route in to this material is [31]. A very readable overview can also be found in [28], which includes some insightful computations comparing different modeling regimes. The article [29] has a systems biology perspective and includes MATLAB examples.

There are many issues to be addressed when stochastic simulation techniques are applied in a biological context. Important examples are that cell growth and cell division give rise to time-varying volumes, [19], "reactions" may not be instantaneous, [2] and spatial effects can be significant [27].

9.4 Algorithms and Software

Software that implements stochastic simulation algorithms is described in [1, 5, 17, 25]. Also, the MATLAB SimBiology Toolbox has SSA and tau-leaping algorithms.

There is currently a great deal of algorithmic activity, both in (a) designing new variants for increased efficiency or accuracy in certain circumstances [4, 9, 21], and (b) developing *multi-scale* algorithms that work across different modeling regimes to get the benefits of stochastic simulation without the full cost [3, 7, 24].

Chapter 8 of [31] gives an accessible overview of recent developments.

9.5 Related Fields

The CME is an example of a birth-and-death process, providing links to classical population dynamics [23].

The SSA fits into the frameworks of discrete event simulation and Petri nets. The text [31] is a good place to learn more.

The type of noise arising in the CLE is called *intrinsic*, distinguishing it from *extrinsic* noise that may arise from external factors (temperature, pressure, etc.). Modeling extrinsic noise is a separate issue that gives rise to different stochastic terms that are typically derived in an ad hoc manner [22].

A Simulation and Probability Issues

This appendix fills in some of the details concerning stochastic simulation that were glossed over in the main text.

First, we very briefly explain how steps 2,3 and 4 in the pseudocode description of SSA in section 4 follow from the result (10). Our task is to compute an integer j between 1 and M for the reaction index, and a positive real number τ for the time to the next reaction.

The reaction index is easy to handle. The chance of a reaction index being chosen must be proportional to its propensity function. Conceptually, we can imagine the unit interval being divided up into M subintervals, where the jth subinterval has length $a_j(\mathbf{x})/a_{\rm sum}(\mathbf{x})$. Then if we choose a point in the unit interval uniformly

at random, the chance of this point lying in the jth subinterval will be precisely $a_j(\mathbf{x})/a_{\text{sum}}(\mathbf{x})$. Step 3 in the pseudocode does exactly this: j is taken to be the index of the subinterval in which ξ_1 lands.

For the time to the next reaction, we need to know that, in general, an *exponentially distributed random variable* with parameter $\lambda > 0$ is characterised by the density function

$$\begin{cases} \lambda e^{-\lambda x} & \text{for } x \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

Hence, if X is exponentially distributed with parameter λ then for any $0 < a < b$,

$$\mathbb{P}\left(a < X < b\right) = \int_a^b \lambda e^{-\lambda x}\, dx = e^{-\lambda a} - e^{-\lambda b}.$$

Now,

$$\mathbb{P}\left(a < X < b\right) = \mathbb{P}\left(-b < -X < -a\right) = \mathbb{P}\left(e^{-\lambda b} < e^{-\lambda X} < e^{-\lambda a}\right),$$

so we conclude that

$$\mathbb{P}\left(e^{-\lambda b} < e^{-\lambda X} < e^{-\lambda a}\right) = e^{-\lambda a} - e^{-\lambda b}.$$

This shows that the random variable $Z := e^{-\lambda X}$ has a uniform (0,1) distribution—the probability of Z lying in any subinterval of $(0, 1)$ is given by the length of that subinterval. Hence X may be written $\ln(1/Z)/\lambda$, where Z is uniform (0,1). With $\lambda = a_{\text{sum}}(\mathbf{X}(t))$, this gives step 4 of the pseudocode.

References

[1] D. Adalsteinsson, D. McMillen, and T. C. Elston, *Biochemical network stochastic simulator (BioNetS): software for stochastic modeling of biochemical networks*, BMC Bioinformatics, 5:24 (2004).

[2] D. Bratsun, D. Volfson, L. S. Tsimring, and J. Hasty, *Delay-induced stochastic oscillations in gene regulation*, Proceedings of the National Academy of Sciences of the USA, 102 (2005), pp. 14593–14598.

[3] Y. Cao, D. T. Gillespie, and L. Petzold, *The slow-scale stochastic simulation algorithm*, J. Chem. Phys., 122 (2005), p. 014116.

[4] ——, *Efficient stepsize selection for the Tau-leaping method*, J. Chem. Phys., To appear (2006).

[5] COPASI Development Team, *COPASI Documentation*, http://www.copasi.org/tiki-index.php, (2006).

[6] E. J. Crampin and S. Schnell, *New approaches to modelling and analysis of biochemical reactions, pathways and networks*, Progress in Biophysics & Molecular Biology, 86 (2004), pp. 1–4.

[7] W. E, D. Liu, and E. Vanden-Eijnden, *Nested stochastic simulation algorithms for chemical kinetic systems with multiple time scales*, J. Chem. Phys., 123 (2005), p. 194107.

[8] C. W. Gardiner, *Handbook of Stochastic Methods, for Physics, Chemistry and the Natural Sciences*, Springer-Verlag, third ed., 2004.

[9] M. Gibson and J. Bruck, *Efficient exact stochastic simulation of chemical systems with many species and many channels*, J. Phys. Chem. A, 104 (2000), pp. 1876–1889.

[10] D. T. Gillespie, *A general method for numerically simulating the stochastic time evolution of coupled chemical reactions*, J. Comp. Phys., 22 (1976), pp. 403—434.

[11] ——, *Exact stochastic simulation of coupled chemical reactions*, J. Phys. Chem., 81 (1977), pp. 2340–2361.

[12] ——, *The chemical Langevin equation*, J. Chem. Phys., 113 (2000), pp. 297–306.

[13] ——, *Approximate accelerated stochastic simulation of chemically reacting systems*, J. Chem. Phys., 115 (2001), pp. 1716–1733.

[14] D. T. Gillespie and L. Petzold, *Numerical simulation for biochemical kinetics*, in System Modelling in Cellular Biology, Z. Szallasi, J. Stelling, and V. Periwal, eds., MIT Press, 2006, p. to appear.

[15] D. J. Higham, *An algorithmic introduction to numerical simulation of stochastic differential equations*, SIAM Review, 43 (2001), pp. 525–546.

[16] D. J. Higham and N. J. Higham, *MATLAB Guide*, SIAM, second ed., 2005.

[17] N. Le Novére and T. S. Shimizu, *StochSim: modelling of stochastic biomolecular processes*, Bioinformatics, 17 (2001), pp. 575–576.

[18] B. Leimkuhler and S. Reich, *Simulating Hamiltonian Dynamics*, Cambridge University Press, 2005.

[19] T. Lu, D. Volfson, L. Tsimring, and J. Hasty, *Cellular growth and division in the Gillespie algorithm*, IEE Proc. Systems Biology, 1 (2004), pp. 121–128.

[20] J. Martínez-Urreaga, J. Mira, and C. Gonzàlez-Fernández, *Introducing the stochastic simulation of chemical reactions. Using the Gillespie algorithm and MATLAB*, Chemical Engineering Education, 37 (2003), pp. 14–19.

[21] C. V. Rao and A. P. Arkin, *Stochastic chemical kinetics and the quasi-steady-state assumption: application to the Gillespie algorithm*, J. Chem. Phys., 118 (2003), pp. 4999–5010.

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

[23] E. Renshaw, *Modelling Biological Populations in Space and Time*, Cambridge University Press, 1991.

[24] H. Salis and Y. Kaznessis, *Accurate hybrid stochastic simulation of coupled checmial or biochemical reactions*, J. Chem. Phys., 122 (2005), pp. 0541031–13.

[25] H. Salis, V. Sotiropoulos, and Y. N. Kaznessis, *Multiscale Hy3S: Hybrid stochastic simulation for supercomputers*, BMC Bioinformatics, 7 (2006), p. 93.

[26] H. E. Samad, M. Khammash, L. Petzold, and D. T. Gillespie, *Stochastic modeling of gene regulatory networks*, Int. J. Robust and Nonlinear Control, 15 (2005), pp. 691–711.

[27] S. Schnell and T. E. Turner, *Reaction kinetics in intracellular environments with macromolecular crowding: simulations and rate laws*, Progress in Biophysics & Molecular Biology, 85 (2004), pp. 235–260.

[28] T. E. Turner, S. Schnell, and K. Burrage, *Stochastic approaches for modelling in vivo reactions*, Computational Biology and Chemistry, 28 (2004), pp. 165–178.

[29] M. Ullah, H.Schmidt, K-H.Cho, and O.Wolkenhauer, *Deterministic modelling and stochastic simulation of pathways using MATLAB*, IEE Proc. Systems Biology, 153 (2006), pp. 53–60.

[30] N. van Kampen, *Stochastic Processes in Physics and Chemistry*, North Holland, third ed., 2001.

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