Multi-level Monte Carlo for stochastically modeled chemical kinetic systems

David F. Anderson[1] and Desmond J. Higham[2]

July 13, 2011

Abstract

A chemical reaction network involves multiple reactions and species. The simplest stochastic models of such networks treat the system as a continuous time Markov chain with the state being the number of molecules of each species and with reactions modeled as possible transitions of the chain. While there are methods that generate exact sample paths of the Markov chain, their computational cost scales linearly with the number of reaction events. Therefore, such methods become computationally intense for even moderately sized systems. This drawback is greatly exacerbated when such simulations are performed in conjunction with Monte Carlo techniques, as is the norm, which require the generation of many paths.

We show how to extend a recently proposed multi-level Monte Carlo approach to this stochastic chemical kinetic setting, lowering the computational complexity needed to compute expected values of functions of the state of the system to a specified accuracy. The extension is non-trivial and a novel coupling of the requisite processes is introduced that is both easy to simulate and provides a small variance for the estimator. Further, and in a stark departure from other implementations of multi-level Monte Carlo, we show how to make use of the existence of exact algorithms to produce an unbiased estimator that is significantly less computationally expensive than the usual unbiased estimator arising from exact algorithms in conjunction with crude Monte Carlo. We thereby show that the basic computational complexity of the Gillespie/stochastic simulation algorithm/tau-leaping approaches can be dramatically improved in a manner that can be quantified precisely.

Keywords: computational complexity, diffusion, Gillespie, Langevin, next reaction method, random time change, tau-leaping, variance.

1 Introduction

This paper concerns the efficient computation of expectations for stochastic models of biochemical reaction networks. A chemical reaction network is a chemical system involving multiple reactions and species. The simplest stochastic models of such networks [32, 37] treat the system as a continuous time Markov chain with the state, X, being the number of molecules of each species and with reactions modeled as possible transitions of the chain. If the abundances of the constituent molecules of a reaction network are sufficiently high then their concentrations are typically modeled by a coupled set of ordinary differential equations. If, however, the abundances are low then the standard deterministic models do not provide a good representation of the behavior of the system and stochastic models are used.

[1]Department of Mathematics, University of Wisconsin, Madison, Wi. 53706, anderson@math.wisc.edu, grant support from NSF-DMS-1009275.

[2]Department of Mathematics and Statistics, University of Strathclyde, Glasgow, G1 1XH, d.j.higham@strath.ac.uk, supported by a Fellowship from the Leverhulme Trust.

[0]AMS 2000 subject classifications: Primary 60H35, 65C99; Secondary 92C40

There is now a large literature demonstrating that the fluctuations arising from the effective randomness of molecular interactions can have significant consequences, including a randomization of phenotypic outcomes and non-genetic population heterogeneity; see, for example, [6, 9, 14, 36, 38, 39, 40]. When modeled stochastically, these systems often have a wide variation in scales in that the different species and reaction rates vary over several orders of magnitude. In this multi-scale setting it is typically an extremely difficult task to develop analytical approximations, such as those involving Langevin or law of large number type arguments, to simplify or reduce a system; although progress has been made for some problem classes [7, 31]. Therefore, computational methods oftentimes are the only reasonable means by which such models can be understood in real time.

Many methods are available to compute paths of stochastic models of reaction networks. First, as the models of interest are continuous time Markov chains, we may simulate sample paths exactly. The most commonly used exact methods are the next reaction method [16] and the stochastic simulation algorithm, which is also known as Gillespie's algorithm in the current setting [21, 22]. Both are examples of discrete event simulation [25, 45]. As the computational cost of exact algorithms scales linearly with the number of reaction events, such methods become computationally intense for even moderately sized systems. This issue looms large when many sample paths are needed in a Monte Carlo setting. To address this, approximate algorithms, and notably the class of algorithms termed "tau-leaping" methods introduced by Gillespie [23], have been developed with the explicit aim of greatly lowering the computational complexity of each path simulation while controlling the bias [2, 3, 4, 35, 41].

A common problem of interest in the stochastic chemical kinetic setting (and, in fact, in nearly all scenarios where randomness is incorporated into a model [43]), and the main focus of this paper, is to efficiently approximate $\mathbb{E}f(X(T))$, where f is a function of the state of the system which gives a measurement of interest. For example, $f(X(T))$ could be the abundance of one of the species at a particular time. A key observation for our work is that optimizing the overall expected value computation is a different, and typically more relevant, goal than optimizing along each path. Suppose we use an exact algorithm to approximate $\mathbb{E}f(X(T))$ to an order of accuracy of $O(\epsilon)$ in the sense of confidence intervals. To do so, we need to generate $n = O(\epsilon^{-2})$ paths so that the standard deviation of the usual Monte Carlo estimator,

$$\mu_n = \frac{1}{n}\sum_{j=1}^{n} f(X_{[j]}(T)),$$

where $X_{[j]}$ are independent realizations generated via an exact algorithm, is $O(\epsilon)$. If we let $\overline{N} > 0$ be the order of magnitude of the number of computations needed to produce a single sample path using an exact algorithm, then the total computational complexity becomes $O(\overline{N}\epsilon^{-2})$. (Here, and throughout, we work in terms of expected computational complexity.)

When $\overline{N} \gg 1$, which is the norm as opposed to the exception in our setting, it may be desirable to make use of an approximate algorithm. Suppose $\mathbb{E}f(X(T)) - \mathbb{E}f(Z_h(T)) = O(h)$, where Z_h is an approximate path generated from a time discretization with a magnitude of h (i.e. we have an order one method). We first make the trivial observation that the estimator

$$\mu_n = \frac{1}{n}\sum_{j=1}^{n} f(Z_{h,[j]}(T)), \tag{1}$$

where $Z_{h,[j]}$ are independent paths generated via the approximate algorithm with a step size of h, is an unbiased estimator of $\mathbb{E}f(Z_h(T))$, and not $\mathbb{E}f(X(T))$. However, noting that

$$\mathbb{E}f(X(T)) - \mu_n = \left[\mathbb{E}f(X(T)) - \mathbb{E}f(Z_h(T))\right] + \left[\mathbb{E}f(Z_h(T)) - \mu_n\right], \tag{2}$$

we see that choosing $h = O(\epsilon)$, so that the first term on the right is $O(\epsilon)$, and $n = \epsilon^{-2}$, so that the standard deviation is $O(\epsilon)$, delivers the desired accuracy. With a fixed cost per time step, the computational complexity of generating a single such path is $O(\epsilon^{-1})$ and we find that the total computational complexity is $O(\epsilon^{-3})$. Second order methods lower the computational complexity to $O(\epsilon^{-2.5})$, as h may be chosen to be $O(\epsilon^{1/2})$.

The discussion above suggests that the choice between exact or approximate path computation should depend upon whether ϵ^{-1} or $\overline{N}$ is the larger value, with an exact algorithm being beneficial when $\overline{N} < \epsilon^{-1}$. It is again worth noting, however, that the estimators built from approximate methods are biased, and while analytic bounds can be provided for that bias [3, 4, 35] these are typically neither sharp nor computable, and hence of limited practical value. The exact algorithm, on the other hand, trivially produces an *unbiased* estimator, so it may be argued that $\epsilon^{-1} \ll \overline{N}$ is necessary before it is worthwhile to switch to an approximate method.

In the diffusive setting, the so-called multi-level Monte Carlo approach of Giles [18], with related earlier work by Heinrich [26], has the remarkable property of lowering the standard $O(\epsilon^{-3})$ cost of computing an $O(\epsilon)$ accurate Monte Carlo estimate of $\mathbb{E}f(X(T))$ down to $O(\epsilon^{-2}\log(\epsilon)^2)$. Here, we are assuming that a weak order one and strong order $1/2$ discretization method, such as Euler–Maruyama, is used. Further refinements have appeared in [19, 20, 28, 30], and the same ideas have been applied to partial differential equations [8, 13, 27].

In this paper we extend the multi-level aproach to the stochastic chemical kinetic setting. The extension is non-trivial and a novel coupling of the requisite processes is introduced that is both easy to simulate and provides a very small variance for the estimator. In fact, showing the practical importance of the couplings (19) and (22), which were first presented as an analytical tool to study strong errors in [3], could be viewed as the most important contribution of this paper. Further, and in a stark departure from other implementations of multi-level Monte Carlo, we provide a second multi-level Monte Carlo algorithm which exploits the existence of exact algorithms in the current setting to produce an *unbiased* estimator giving the desired accuracy with significantly less computational complexity than an exact algorithm alone. The authors believe that this unbiased multi-level Monte Carlo will become a standard, generic algorithm for simulating stochastically modeled chemical reaction networks.

While the language of biochemistry is used throughout the paper, the mathematical models considered here, and formally developed in Section 2, are quite universal in biology. For example, many models at the level of populations satisfy equations with the same mathematical structure [42]. One of the goals of systems and evolutionary biology is to combine models from the cellular level with those at the population level. This paper directly addresses the challenging computational costs associated with their simulation. We emphasize that the gains in computational efficiency reported in this work apply to generic models, and do not rely on any specific structural properties. However, the ideas have the potential to be fine-tuned further in appropriate cases; for example by exploiting known analytical results or multi-scale partitions.

The outline of the paper is as follows. In Section 2, we introduce the basic model for stochastically modeled chemical reaction systems and, further, introduce an equivalent model which incorporates the natural temporal and other quantitative scales. Consideration of such a scaled model is critical for realistic quantitative comparisons of accuracy versus cost for computational methods, though plays no role in the actual simulations. In Section 3, we briefly review Euler's method, often called tau-leaping, in the current setting. In Section 4, we outline the original multi-Level Monte Carlo method. In Section 5, we extend multi-level Monte Carlo to the stochastic chemical kinetic setting in two different ways. In the first, exact algorithms are not used and we are led to an efficient method with a bias. In the second, exact algorithms play a key role and allow us to develop *unbiased* estimators. In both cases, we quantify precisely the generic computational efficiencies obtained, relative to standard Monte Carlo. In Section 6, we provide the delayed proofs of the main analytical results of Section 5. In Section 7, we briefly discuss some implementation issues. Finally, in Section 8, we provide a computational example demonstrating our main results.

2 The basic model

An example of a chemical reaction is

$$2S_1 + S_2 \ \to \ S_3,$$

where we would interpret the above as saying two molecules of type S_1 combine with a molecule of type S_2 to produce a molecule of type S_3. The S_i are called chemical *species*. Letting

$$\nu_1 = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}, \quad \nu_1' = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \text{and} \quad \zeta_1 = \nu_1' - \nu_1 = \begin{pmatrix} -2 \\ -1 \\ 1 \end{pmatrix},$$

we see that every instance of the reaction changes the state of the system by addition of ζ_1. Here the subscript "1" is used to denote the first (and in this case only) reaction of the system.

In the general setting we denote the number of species by d, and for $i \in \{1, \dots, d\}$ we denote the ith species by S_i. (Though we will use more descriptive notation for the specific system considered in Section 8.) We then consider a finite set of R reactions, where the model for the kth reaction is determined by

(i) a vector of inputs ν_k specifying the number of molecules of each chemical species that are consumed in the reaction,

(ii) a vector of outputs ν_k' specifying the number of molecules of each species that are created in the reaction, and

(iii) a function of the state λ_k that gives the *transition intensity* or rate at which the reaction occurs. (Note that in the biological and chemical literature, transition intensities are referred to as *propensities*.)

Specifically, if we denote the state of the system at time t by $X(t) \in \mathbb{Z}^d_{\geq 0}$, and if the kth reaction occurs at time t, we update the state by addition of the *reaction vector*

$$\xi_k \overset{\text{def}}{=} \nu_k' - \nu_k$$

and the new state becomes $X(t) = X(t-) + \xi_k$. For the standard Markov chain model, the number of times that the kth reaction occurs by time t can be represented by the counting process $R_k(t) = Y_k\left(\int_0^t \lambda_k(X(s))ds\right)$, where the Y_k are independent unit-rate Poisson processes; see, for example, [34], [15, Chapter 6], or the recent survey [5]. The state of the system may then be characterized as

$$X(t) = X(0) + \sum_k Y_k \left(\int_0^t \lambda_k(X(s))ds \right) \xi_k. \tag{3}$$

The above formulation is termed a "random time change representation" and is equivalent to the "chemical master equation representation" found in much of the biology and chemistry literature.

Remark 1. Simulation of the representation (3) is equivalent to the next reaction method. See [1] for details. Later we will present similar, albeit more complicated, representations for two coupled processes that will be simulated in a similar manner. These representations can be found in (19) and (22).

A common choice of intensity function for chemical reaction systems, and the one we adopt throughout, is that of mass action kinetics. Under mass action kinetics, the intensity function for the kth reaction is

$$\lambda_k(x) = \kappa_k \prod_{i=1}^d \frac{x_i!}{(x_i - \nu_{ki})!}, \tag{4}$$

where ν_{ki} denotes the ith component of ν_k. Implicit in the assumption of mass action kinetics is that the vessel under consideration is "well-stirred." While the natural state space of the process is the non-negative orthant, we extend λ_k to all of $\mathbb{R}^d$ by setting it to zero whenever a species which λ_k explicitly depends upon is non-positive. That is $\lambda_k(x) = 0$ whenever $x_i \leq 0$ and $\nu_{ki} \neq 0$. We note that none of the core ideas of this paper depend upon the fact that λ_k are mass-action kinetics and the assumption is made for analytical convenience and historical consistency.

This model is a continuous time Markov chain in $\mathbb{Z}^d$ with generator

$$(\mathcal{A}f)(x) = \sum_k \lambda_k(x)(f(x+\zeta_k) - f(x)),$$

where $f : \mathbb{Z}^d \to \mathbb{R}$. Kolmogorov's forward equation, termed the *chemical master equation* in much of the biology literature, for this model is

$$\frac{d}{dt}P(x,t|\pi) = \sum_k \lambda_k(x-\zeta_k)1_{\{x-\zeta_k\in\mathbb{Z}^d_{\geq 0}\}}P(x-\zeta_k,t|\pi) - \sum_k \lambda_k(x)P(x,t|\pi),$$

where for $x \in \mathbb{Z}^d_{\geq 0}$, $P(x,t|\pi)$ represents the probability that $X(t) = x$, conditioned upon the initial distribution π.

Example 1
To solidify notation, we consider the network

$$S_1 \underset{\kappa_2}{\overset{\kappa_1}{\rightleftarrows}} S_2, \qquad 2S_2 \overset{\kappa_3}{\to} S_3,$$

where we have placed the rate constants κ_k above or below their respective reactions. For this example, equation (3) *is*

$$X(t) = X(0) + Y_1\left(\int_0^t \kappa_1 X_1(s)ds\right)\begin{bmatrix}-1\\1\\0\end{bmatrix} + Y_2\left(\int_0^t \kappa_2 X_2(s)ds\right)\begin{bmatrix}1\\-1\\0\end{bmatrix}$$
$$+ Y_3\left(\int_0^t \kappa_3 X_2(s)(X_2(s)-1)ds\right)\begin{bmatrix}0\\-2\\1\end{bmatrix}.$$

Defining $\zeta_1 = [-1,1,0]^T$, $\zeta_2 = [1,-1,0]^T$, and $\zeta_3 = [0,-2,1]^T$, the generator $\mathcal{A}$ satisfies

$$(\mathcal{A}f)(x) = \kappa_1 x_1(f(x+\zeta_1) - f(x)) + \kappa_2 x_2(f(x+\zeta_2) - f(x)) + \kappa_3 x_2(x_2-1)(f(x+\zeta_3) - f(x)).$$

2.1 Scalings

We may convert from abundances to concentrations. Defining $|\nu_k| = \sum_i \nu_{ki}$ and letting N be a scaling parameter usually taken to be the volume of the system times Avogadro's number, the rate constants should be scaled so that $\lambda_k^N(x) = \hat{\kappa}_k \frac{1}{N^{|\nu_k|-1}} \prod_i \frac{x_i!}{(x_i - \nu_{ki})!}$ (see [45, Chapter 6]). When N is the volume times Avogadro's number, since x records the number of molecules of each species present, the quantity $c = N^{-1}x$ gives the concentrations in moles per unit volume. With this scaling and a large volume limit $\lambda_k^N(x) \approx N\hat{\kappa}_k \prod_i c_i^{\nu_{ki}} \stackrel{\text{def}}{=} N\hat{\lambda}_k(c)$. Since the law of large numbers for the Poisson process implies $N^{-1}Y(Nu) \approx u$, equation (3) combined with the above scaling yields

$$c(t) = N^{-1}X(t) \approx c(0) + \sum_k \int_0^t \hat{\kappa}_k \prod_i c(s)_i^{\nu_{ki}} ds\, (\nu_k' - \nu_k),$$

which in the large volume limit gives the classical *deterministic* law of mass action

$$\dot{c}(t) = \sum_k \hat{\kappa}_k c(t)^{\nu_k} (\nu_k' - \nu_k),$$

where for two vectors $u, v \in \mathbb{R}^d_{\geq 0}$ we define $u^v \equiv \prod_i u_i^{v_i}$ and adopt the convention that $0^0 = 1$. For a precise formulation of the above scaling argument, termed the "classical scaling," see [32, 33, 34].

In the biological setting it is rare that a system satisfies the classical scaling described above. For example, there can not be *many* copies of a particular gene, and it is quite common for there to be only a handful of mRNA molecules and perhaps hundreds or thousands of proteins from a particular gene at any given time. As these molecules interact with metabolites, whose abundances can be quite large, it is clear that these systems naturally have multiple temporal and other quantitative scales.

Let $N \gg 1$, where now N is simply a parameter of the system. Assume that we are given a model of the form

$$X(t) = X(0) + \sum_k Y_k \left(\int_0^t \lambda_k'(X(s)) ds \right) \zeta_k,$$

where the λ_k' are of the form

$$\lambda_k'(x) = \kappa_k' \prod_i \frac{x_i!}{(x_i - \nu_{ki})!},$$

and where we recall that $\zeta_k = \nu_k' - \nu_k$ is the reaction vector and ν_k is the source vector for the kth reaction. For each species, define the *normalized abundance* (or simply, the abundance) by

$$X_i^N(t) \stackrel{\text{def}}{=} N^{-\alpha_i} X_i(t), \tag{5}$$

where $\alpha_i \geq 0$ should be selected so that $X_i^N = O(1)$. Here X_i^N may be the species number ($\alpha_i = 0$) or the species concentration or something else. Since the rate constants may also vary over several orders of magnitude, we write $\kappa_k' = \kappa_k N^{\beta_k}$ where the β_k are selected so that $\kappa_k = O(1)$. Under the mass-action kinetics assumption, we have that $\lambda_k'(X(s)) = N^{\beta_k + \nu_k \cdot \alpha} \lambda_k(X^N(s))$, where λ_k is deterministic mass-action kinetics with parameter κ_k [31]. Our model has therefore become

$$X^N(t) = X^N(0) + \sum_k Y_k \left(\int_0^t N^{\beta_k + \nu_k \cdot \alpha} \lambda_k(X^N(s)) ds \right) \zeta_k^N, \tag{6}$$

where $\zeta_{ki}^N \stackrel{\text{def}}{=} \zeta_{ki}/N^{\alpha_i}$ (so ζ_k^N is the scaled reaction vector). Note that

$$\overline{N} \stackrel{\text{def}}{=} \sum_k N^{\beta_k + \nu_k \cdot \alpha} \tag{7}$$

is the order of magnitude of the number of steps needed to generate a single path using an exact algorithm.

For any vector $w \in \mathbb{R}^d$, define w^N to be the vector with ith component

$$w_i^N \stackrel{\text{def}}{=} \frac{w_i}{N^{\alpha_i}},$$

and define

$$\mathbb{L}_N = \left\{ x^N \mid x \in \mathbb{Z}^d \right\}.$$

By construction, the process (6) lives in $\mathbb{L}_N$, and its generator is

$$\mathcal{A}^N f(x) = \sum_k N^{\beta_k + \nu_k \cdot \alpha} \lambda_k(x)(f(x + \zeta_k^N) - f(x)).$$

Remark 2. It is worth explicitly pointing out that the models (3) and (6) are equivalent in that X^N is the scaled version of X. The scaling is essentially an analytical tool as now both X^N and $\lambda_k(X^N(\cdot))$ are $O(1)$, and in Section 6 it will be shown how the representation (6) is useful in the quantification of the behavior of different computational methods. However, we stress that the scaling itself plays *no role* in the actual simulation of the processes, with the small exception that it can inform the decision for the size of the time step of an approximate method.

To quantify the natural time-scale of the system, define $\gamma \in \mathbb{R}$ via

$$\gamma \overset{\text{def}}{=} \max_{\{i,k \,:\, \zeta_{ki}^N \neq 0\}} \{\beta_k + \nu_k \cdot \alpha - \alpha_i\},$$

where we recall that ν_k is the source vector for the kth reaction. It is worth noting that $\gamma = 0$ if one assumes the system satisfies the classical scaling discussed above. However, $\gamma = 0$ in many other settings as well. We will see that our main analytical results are most useful when $\gamma \leq 0$.

For $k \in \{1, \dots, R\}$ we define

$$c_k \overset{\text{def}}{=} \beta_k + \nu_k \cdot \alpha - \gamma. \tag{8}$$

Finally, for $i \in \{1, \dots, d\}$ and $k \in \{1, \dots, R\}$, we define

$$\rho_k \overset{\text{def}}{=} \min\{\alpha_i \;:\; \zeta_{ki}^N \neq 0\},$$

so that $O(|\zeta_k^N|)$ is equivalent to $O(N^{-\rho_k})$, and $\rho \overset{\text{def}}{=} \min\{\rho_k\}$. Note that $\rho \geq 0$, and by the choice of γ we have $c_k - \rho_k \leq 0$ for all k. Further, we point out that γ is chosen so that $c_k = 0$ for at least one k. Finally, we note that if $\|\nabla f\|_\infty$ is bounded, then

$$N^{c_k}(f(x + \zeta_k^N) - f(x)) = O(N^{c_k - \rho_k}),$$

with $c_k - \rho_k = 0$ for at least one k. Note that the classical scaling holds if and only if $c_k \equiv \rho_k \equiv 1$ and $\gamma = 0$.

Example 2

To again solidify notation, consider the reversible isometry

$$S_1 \underset{100}{\overset{100}{\rightleftarrows}} S_2$$

with $X_1(0) = X_2(0) =$ 10,000. In this case, it is natural to take $N =$ 10,000 and $\alpha_1 = \alpha_2 = 1$. As the rate constants are $100 = \sqrt{10{,}000}$, we take $\beta_1 = \beta_2 = 1/2$ and find that $\gamma = 1/2$ and $\rho_1 = \rho_1 = 1$. The normalized process X_1^N satisfies

$$X_1^N(t) = X_1^N(0) - Y_1\left(N^{1/2} N \int_0^t X_1^N(s)ds\right)\frac{1}{N} + Y_2\left(N^{1/2} N \int_0^t (2 - X_1^N(s))ds\right)\frac{1}{N},$$

where we have used that $X_1^N + X_2^N \equiv 2$.

Remark 3. While in general the functions λ_k are polynomials, and are therefore not globally Lipschitz, we note that we will be performing our analysis under the assumption that both X^N and $\lambda_k(X^N(\cdot))$, together with their approximations developed in the following section, are $O(1)$. Therefore, after possibly redefining the kinetics by multiplication with a cutoff function, see, for example, [3, 4], we may assume that each λ_k is globally Lipschitz.

3 A review of explicit Euler tau-leaping

We briefly review the most common approximation method used in the stochastic chemical kinetic setting. First, note that if $\sum_k \lambda_k(X(t)) \gg 1$, then the amount of time that an exact algorithm requires to wait before making a transition, Δ_t, which is an exponential random variable with parameter $\sum_k \lambda_k(X(t))$, satisfies

$$\mathbb{E}\Delta_t = \frac{1}{\sum_k \lambda_k(X(t))} \ll 1.$$

Therefore, the time needed to generate a single path may be prohibitive. The approximate algorithm termed *explicit tau-leaping*, or Euler tau-leaping, was developed by Gillespie in an effort to overcome this problem [23]. The basic idea of tau-leaping is to hold the intensity functions fixed over a time interval $[t_n, t_n + h]$ at the values $\lambda_k(X(t_n))$, where $X(t_n)$ is the current state of the system, and, under this assumption, compute the number of times each reaction takes place over this period. This method will potentially yield lower runtimes only if $h \gg 1/\sum_k \lambda_k(X(t_n)) \approx \Delta_t$. As the waiting times for the reactions are exponentially distributed this leads to the following algorithm, which simulates up to a time of $T > 0$. Below and in the sequel, for $x \geq 0$ we will write Poisson(x) for a Poisson random variable with parameter x.

Algorithm 1 (Euler tau-leaping)**.** Fix $h > 0$. Set $Z_h(0) = x_0$, $t_0 = 0$, $n = 0$ and repeat the following until $t_{n+1} = T$:

(i) Set $t_{n+1} = t_n + h$. If $t_{n+1} \geq T$, set $t_{n+1} = T$ and $h = T - t_n$.

(ii) For each k, let $\Lambda_k = \text{Poisson}(\lambda_k(Z_h(t_n))h)$ be independent of each other and all previous random variables.

(iii) Set $Z_h(t_{n+1}) = Z_h(t_n) + \sum_k \Lambda_k \zeta_k$.

(iv) Set $n \leftarrow n + 1$.

Several improvements and modifications have been made to the basic algorithm described above over the years. However, they are mainly concerned with how to choose the step-size adaptively [11, 24] and/or how to ensure that population values do not go negative during the course of a simulation [2, 10, 12, 44], and are not directly relevant to the current discussion.

Similar to (3), a path-wise representation of Euler tau-leaping defined for all $t \geq 0$ can be given through a random time change of Poisson processes:

$$Z_h(t) = Z_h(0) + \sum_k Y_k \left(\int_0^t \lambda_k(Z_h \circ \eta(s)) ds \right) \zeta_k, \tag{9}$$

where the Y_k are as before, and $\eta(s) \overset{\text{def}}{=} \left\lfloor \frac{s}{h} \right\rfloor h$. Thus, $Z_h \circ \eta(s) = Z_h(t_n)$ if $t_n \leq s < t_{n+1}$. Noting that $\int_0^{t_{n+1}} \lambda_k(Z_h \circ \eta(s)) ds = \sum_{i=0}^n \lambda_k(Z_h(t_i))(t_{i+1} - t_i)$ explains why this method is called Euler tau-leaping. Following (5), for each $i \in \{1, \dots, d\}$ we let $Z_{h,i}^N \overset{\text{def}}{=} N^{-\alpha_i} Z_{h,i}$, so the scaled version of (9) is

$$Z_h^N(t) = Z_h^N(0) + \sum_k Y_k \left(\int_0^t N^{\beta_k + \nu_k \cdot \alpha} \lambda_k(Z_h^N \circ \eta(s)) ds \right) \zeta_k^N, \tag{10}$$

where all other notation is as before. We again stress that the models (9) and (10) are equivalent models, with (9) giving the number of molecules of each species and (10) providing the normalized abundances.

Remark 4. Historically, the time discretization parameter for the methods described in this paper has been τ, leading to the name "τ-leaping methods." We choose to break from this tradition so as not to confuse τ with a stopping time, and we denote our time-step by h to be consistent with much of the numerical analysis literature.

4 A review multi-level Monte Carlo

Suppose we have a stochastic process of interest, $X(\cdot)$. Let f be a function of the state of the system which gives a measurement of interest. The problem is to efficiently approximate $\mathbb{E}f(X(T))$. As discussed in Section 1, using the "crude Monte Carlo" estimator (1) with a weakly first order method will provide an estimate with an accuracy of $O(\epsilon)$, in the sense of confidence intervals, at a computational cost of $O(\epsilon^{-3})$.

In multi-level Monte Carlo (MLMC) paths of varying step-sizes are generated and are coupled in an intelligent manner so that the computational complexity is reduced to $O(\epsilon^{-2}(\log\epsilon)^2)$ [18]. Sometimes even the $\log(\epsilon)$ terms can be reduced further [17]. Suppose we have an approximate method, such as Euler's method in the diffusive setting, which is known to be first order accurate in a weak sense, and 1/2 order accurate in a strong L^2 sense. The MLMC estimator is then built in the following manner. For a fixed integer M, and $\ell \in \{0, 1, \dots, L\}$, where L is to be determined, let $h_\ell = TM^{-\ell}$. Reasonable choices for M include 2, 3, and 4. We will denote Z_ℓ as the approximate process generated using a step-size of h_ℓ. Choose $L = O(\ln(\epsilon^{-1}))$, so that $h_L = O(\epsilon)$ and $\mathbb{E}f(X(T)) - \mathbb{E}f(Z_L(T)) = O(\epsilon)$, and the bias (i.e. the first term on the right hand side of (2)) is of the desired order of magnitude. We then have

$$\mathbb{E}f(Z_L(T)) = \mathbb{E}[f(Z_0(T))] + \sum_{\ell=1}^{L} \mathbb{E}[f(Z_\ell(T)) - f(Z_{\ell-1}(T))], \tag{11}$$

where the telescoping sum is the key feature to note. We will now denote the estimator of $\mathbb{E}[f(Z_0(T))]$ using n_0 paths by $\widehat{Q}_0$, and the estimator of $\mathbb{E}[f(Z_\ell(T)) - f(Z_{\ell-1}(T))]$ using n_ℓ paths as $\widehat{Q}_\ell$. That is

$$\widehat{Q}_0 \overset{\text{def}}{=} \frac{1}{n_0}\sum_{i=1}^{n_0} f(Z_{0,[i]}(T)), \quad \text{and} \quad \widehat{Q}_\ell \overset{\text{def}}{=} \frac{1}{n_\ell}\sum_{i=1}^{n_\ell}(f(Z_{\ell,[i]}(T)) - f(Z_{\ell-1,[i]}(T))), \tag{12}$$

where the important point is that both $Z_{\ell,[i]}(T)$ and $Z_{\ell-1,[i]}(T)$ are generated using the same randomness, but are constructed using different time discretizations (see [18, 29] for details on how to do this in the diffusive setting). We then let

$$\widehat{Q} \overset{\text{def}}{=} \sum_{\ell=0}^{L} \widehat{Q}_\ell, \tag{13}$$

be the unbiased estimator for $\mathbb{E}[f(Z_L(T))]$. Assuming that we can show $\mathsf{Var}(f(Z_\ell(T)) - f(Z_{\ell-1}(T))) = O(h_\ell)$, which follows if the method has a strong error of order 1/2, then a clever choice of the n_ℓ will give $\mathsf{Var}(\widehat{Q}) = O(\epsilon^2)$, but with a total computational complexity of $O(\epsilon^{-2}(\log\epsilon)^2)$, as desired (see [18] for full details). Note that it is necessary to know both the weak (for the choice of h_L) and strong (for the variance of $\widehat{Q}_\ell$) behavior of the numerical method. Note also that the estimator (13) is a biased estimator of $\mathbb{E}f(X(T))$, and the choice of L was made specifically to ensure that the bias is within the desired tolerance.

5 Multi-level Monte Carlo in the stochastic chemical kinetic setting

We now consider the problem of estimating $\mathbb{E}f(X^N(T))$, where X^N satisfies the general system (6). We stress that as X^N of (6) is equivalent to the process X of (3), efficiently approximating values of the form $\mathbb{E}f(X^N(T))$, for suitable f, is *equivalent* to efficiently approximating values of the form $\mathbb{E}g(X(T))$, for suitable functions g. The scaled systems are easier to analyze because the temporal and other quantitative scales have been made explicit.

Recall that $\overline{N} = \sum_k N^{\beta_k + \nu_k\cdot\alpha}$, as defined in (7), gives the order of magnitude of the number of steps needed to generate a single path using an exact algorithm. As discussed in Section 1, to approximate $\mathbb{E}f(X^N(T))$ to an order of accuracy of $\epsilon > 0$ using an exact algorithm (such as Gillespie's algorithm or the next reaction method) combined with the crude Monte Carlo estimator, we need to generate ϵ^{-2} paths. Thus, we have a computational complexity of $O(\overline{N}\epsilon^{-2})$.

We will now extend the core ideas of multi-level Monte Carlo as described in Section 4 to the stochastic chemical kinetics setting with Euler tau-leaping, given in (10), as our approximation method. We again fix an integer $M > 0$, and for $\ell \in \{\ell_0, \dots, L\}$, where both ℓ_0 and L are to be determined, let $h_\ell = TM^{-\ell}$. We then denote by Z_ℓ^N the approximate process (10) generated with a step-size of size h_ℓ. By [4], we know that for $f \in C^2$,

$$\mathbb{E}f(X^N(T)) - \mathbb{E}f(Z_\ell^N(T)) = O(h_\ell).$$

Choose $L = O(\ln(\epsilon^{-1}))$, so that $h_L = O(\epsilon)$ and the bias is of the desired order of magnitude. We then have

$$\mathbb{E}f(Z_L^N(T)) = \mathbb{E}[f(Z_{\ell_0}^N(T))] + \sum_{\ell=\ell_0}^{L} \mathbb{E}[f(Z_\ell^N(T)) - f(Z_{\ell-1}^N(T))], \tag{14}$$

where, again, the telescoping sum is the key feature to note. We will again denote the estimator of $\mathbb{E}[f(Z_{\ell_0}^N(T))]$ using n_0 paths by $\widehat{Q}_0$, and the estimator of $\mathbb{E}[f(Z_\ell^N(T)) - f(Z_{\ell-1}^N(T))]$ using n_ℓ paths as $\widehat{Q}_\ell$. That is

$$\widehat{Q}_0 \overset{\text{def}}{=} \frac{1}{n_0}\sum_{i=1}^{n_0} f(Z_{\ell_0,[i]}^N(T)), \quad \text{and} \quad \widehat{Q}_\ell \overset{\text{def}}{=} \frac{1}{n_\ell}\sum_{i=1}^{n_\ell}(f(Z_{\ell,[i]}^N(T)) - f(Z_{\ell-1,[i]}^N(T))), \tag{15}$$

where we hope that $Z_{\ell,[i]}^N$ and $Z_{\ell-1,[i]}^N$ can be generated in such a way that $\mathsf{Var}(\widehat{Q}_\ell)$ is small. We will then let

$$\widehat{Q} \overset{\text{def}}{=} \sum_{\ell=\ell_0}^{L} \widehat{Q}_\ell, \tag{16}$$

be the unbiased estimator for $\mathbb{E}[f(Z_L^N(T))]$. The choices for n_ℓ will depend upon the variances of $\widehat{Q}_\ell$.

The main requirements for effectively extending MLMC to the current setting have now come into focus. First, we must be able to simulate the paths Z_ℓ^N and $Z_{\ell-1}^N$ simultaneously in a manner that is efficient, and that minimizes the variances between the paths. Second, we must be able to quantify the variance between the paths so that we can get control over the variance of the associated $\widehat{Q}_\ell$ terms of (15). Both requirements demand a good coupling of the processes Z_ℓ^N and $Z_{\ell-1}^N$.

Let $a \wedge b \overset{\text{def}}{=} \min\{a, b\}$. We propose to couple the processes Z_ℓ^N and $Z_{\ell-1}^N$ in the following manner, which is similar to a coupling originally used in [3] to quantify the strong error of different approximate methods:

$$\begin{aligned} Z_\ell^N(t) = Z_\ell^N(0) + \sum_k \zeta_k^N \Bigg[& Y_{k,1}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s)) \wedge \lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s))ds\right) \\ & + Y_{k,2}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s)) - \lambda_k(Z_\ell^N\circ\eta_\ell(s)) \wedge \lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s))ds\right)\Bigg] \end{aligned} \tag{17}$$

$$\begin{aligned} Z_{\ell-1}^N(t) = Z_{\ell-1}^N(0) + \sum_k \zeta_k^N \Bigg[& Y_{k,1}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s)) \wedge \lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s))ds\right) \\ & + Y_{k,3}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s)) - \lambda_k(Z_\ell^N\circ\eta_\ell(s)) \wedge \lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s))ds\right)\Bigg], \end{aligned} \tag{18}$$

where the $Y_{k,i},\ i \in \{1, 2, 3\}$, are independent, unit-rate Poisson processes, and for each ℓ, we define $\eta_\ell(s) \overset{\text{def}}{=} \lfloor s/h_\ell \rfloor h_\ell$. These paths can easily be computed simultaneously and the distributions of the marginal processes are the same as the usual scaled Euler approximate paths (10) with similar step-sizes. To be clear, the system (17),(18) is the scaled version, and is hence equivalent to, the system

$$
\begin{aligned}
Z_\ell(t) =& Z_\ell(0) + \sum_k Y_{k,1}\left(\int_0^t \lambda_k(Z_\ell \circ \eta_\ell(s)) \wedge \lambda_k(Z_{\ell-1} \circ \eta_{\ell-1}(s))ds\right)\zeta_k \\
&+ \sum_k Y_{k,2}\left(\int_0^t \lambda_k(Z_\ell \circ \eta_\ell(s)) - \lambda_k(Z_\ell \circ \eta_\ell(s)) \wedge \lambda_k(Z_{\ell-1} \circ \eta_{\ell-1}(s))ds\right)\zeta_k \\
Z_{\ell-1}(t) =& Z_{\ell-1}(0) + \sum_k Y_{k,1}\left(\int_0^t \lambda_k(Z_\ell \circ \eta_\ell(s)) \wedge \lambda_k(Z_{\ell-1} \circ \eta_{\ell-1}(s))ds\right)\zeta_k \\
&+ \sum_k Y_{k,3}\left(\int_0^t \lambda_k(Z_{\ell-1} \circ \eta_{\ell-1}(s)) - \lambda_k(Z_\ell \circ \eta_\ell(s)) \wedge \lambda_k(Z_{\ell-1} \circ \eta_{\ell-1}(s))ds\right)\zeta_k,
\end{aligned}
\tag{19}
$$

where now the marginal processes are distributionally equivalent to the approximate processes (9) with similar step-sizes, and all notation is as before. The natural algorithm to simulate the representation (19) (and hence (17),(18)) to a time T is the following.

Algorithm 2 (Simulation of the representation (19))**.** Fix an integer $M > 0$. Fix $h_\ell > 0$ and set $h_{\ell-1} = M \times h_\ell$. Set $Z_\ell(0) = Z_{\ell-1}(0) = x_0$, $t_0 = 0$, $n = 0$. Repeat the following steps until $t_{n+1} \geq T$:

```
(i) For $j = 0, \dots, M-1$, do
    (a) Set $t_j = t_n + j \times h_\ell$ and
        • $A_{k,1} = \lambda_k(Z_\ell(t_j)) \wedge \lambda_k(Z_{\ell-1}(t_n))$.
        • $A_{k,2} = \lambda_k(Z_\ell(t_j)) - A_{k,1}$.
        • $A_{k,3} = \lambda_k(Z_{\ell-1}(t_n)) - A_{k,1}$.
    (b) For each $k$, let
        • $\Lambda_{k,1} = \text{Poisson}(A_{k,1}h_\ell)$ be independent of each other and all previous random variables.
        • $\Lambda_{k,2} = \text{Poisson}(A_{k,2}h_\ell)$ be independent of each other and all previous random variables.
        • $\Lambda_{k,3} = \text{Poisson}(A_{k,3}h_\ell)$ be independent of each other and all previous random variables.
    (c) Set
        • $Z_\ell(t_j + h_\ell) = Z_\ell(t_j) + \sum_k(\Lambda_{k,1} + \Lambda_{k,2})\zeta_k$.
        • $Z_{\ell-1}(t_j + h_\ell) = Z_{\ell-1}(t_j) + \sum_k(\Lambda_{k,1} + \Lambda_{k,3})\zeta_k$.
(ii) Set $t_{n+1} = t_n + h_{\ell-1}$.
(iii) Set $n \leftarrow n + 1$.
```

We make the following observations. First, while Algorithm 2 formally simulates the representation (19), the scaled version of the process generated via Algorithm 2 satisfies (17), (18). Second, we do not need to update either $Z_{\ell-1}$ or $\lambda_k(Z_{\ell-1})$ during the workings of the inner loop of $j = 0, \dots, M-1$. Third, at most one of A_2, A_3 will be non-zero during each step, with both being zero whenever $\lambda_k Z_\ell(t_j)) = \lambda_k(Z_{\ell-1}(t_n))$. Therefore, at most two Poisson random variables will be required per reaction channel at each step and not three. Fourth, the above algorithm, and hence the couplings (19) and/or (17),(18), is no harder to simulate, from a implementation standpoint, than is the usual Euler tau-leaping. Fifth, while two paths are being generated, it should be the case that $\max\{A_2, A_3\}$ is small for each step. Hence the work in computing the Poisson random variables will fall on $\Lambda_{k,1}$, which is the same amount of work as would be needed for the generation of a *single* path of Euler's tau-leaping.

In Section 6 we will prove the following theorem.

Theorem 1. *Suppose* $(Z_\ell^N, Z_{\ell-1}^N)$ *satisfy* (17) *and* (18) *with* $Z_\ell^N(0) = Z_{\ell-1}^N(0)$. *Then, there exist positive constants* C_1, C_2, *that do not depend on* N *but do depend upon both* T *and* γ, *such that*

$$\sup_{t\le T} \mathbb{E}|Z_\ell^N(t) - Z_{\ell-1}^N(t)|^2 \le C_1(T,\gamma)N^{-\rho}h_\ell + C_2(T,\gamma)h_\ell^2.$$

Remark 5. The specific form of $C_1(T,\gamma)$ and $C_2(T,\gamma)$ for Theorem 1 and Theorem 2 below are given in Section 6. However, we note here that if $\gamma > 0$, the terms $C_1(T,\gamma)$ and/or $C_2(T,\gamma)$ could be huge and the upper bound in Theorem 1, and in Theorem 2 below, may be large. Thus, the analytical results of this paper are most applicable when $\gamma \le 0$.

Note that Theorem 1 gives us the estimate

$$\mathsf{Var}(Z_\ell^N(t) - Z_{\ell-1}^N(t)) \le \mathbb{E}|Z_\ell^N(t) - Z_{\ell-1}^N(t)|^2 \le C_1(T,\gamma)N^{-\rho}h_\ell + C_2(T,\gamma)h_\ell^2,$$

which we will use to control the variance of $\widehat{Q}_\ell$ in (15).

Before further exploring MLMC in the current setting, we present a coupling of the exact process X^N and the approximate process Z_ℓ^N. We will later use this coupling to produce an unbiased MLMC estimator. The coupling below was originally used in [3] to quantify the strong error of different approximate methods. We define X^N and Z_ℓ^N via

$$\begin{aligned} X^N(t) =& X^N(0) + \sum_k Y_{k,1}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(X^N(s)) \wedge \lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\right)\zeta_k^N \\ &+ \sum_k Y_{k,2}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(X^N(s)) - \lambda_k(X^N(s)) \wedge \lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\right)\zeta_k^N \end{aligned} \tag{20}$$

$$\begin{aligned} Z_\ell^N(t) =& Z_\ell^N(0) + \sum_k Y_{k,1}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(X^N(s)) \wedge \lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\right)\zeta_k^N \\ &+ \sum_k Y_{k,3}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s)) - \lambda_k(X^N(s)) \wedge \lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\right)\zeta_k^N \end{aligned} \tag{21}$$

where all notation is as before. Note that the distributions of the marginal processes X^N and Z_ℓ^N are equal to those of (6) and (10). The unscaled processes satisfy

$$\begin{aligned} X(t) =& X(0) + \sum_k Y_{k,1}\left(\int_0^t \lambda_k(X(s)) \wedge \lambda_k(Z_\ell\circ\eta_\ell(s))ds\right)\zeta_k \\ &+ \sum_k Y_{k,2}\left(\int_0^t \lambda_k(X(s)) - \lambda_k(X(s)) \wedge \lambda_k(Z_\ell\circ\eta_\ell(s))ds\right)\zeta_k \\ Z_\ell(t) =& Z_\ell(0) + \sum_k Y_{k,1}\left(\int_0^t \lambda_k(X(s)) \wedge \lambda_k(Z_\ell\circ\eta_\ell(s))ds\right)\zeta_k \\ &+ \sum_k Y_{k,3}\left(\int_0^t \lambda_k(Z_\ell\circ\eta_\ell(s)) - \lambda_k(X(s)) \wedge \lambda_k(Z_\ell\circ\eta_\ell(s))ds\right)\zeta_k \end{aligned} \tag{22}$$

which is equivalent to (20) and (21), and whose marginal processes have the same distributions as (3) and (9).

The natural algorithm to simulate (22), and hence (20)–(21), is the next reaction method [1, 16], where the system is viewed as a $2 \times d$ dimensional system with state (X^N, Z_ℓ^N), and each of the “next reactions” must be calculated over the Poisson processes $Y_{k,1}, Y_{k,2}, Y_{k,3}$. See [1] for a thorough explanation of how the next reaction method is equivalent to simulating representations of the forms considered here. Below, we will denote a uniform$[0,1]$ random variable by rand$(0,1)$, and we remind the reader that if $U \sim$ rand$(0,1)$, then $\ln(1/U)$ is an exponential random variable with a parameter of one. All random variables generated are assumed to be independent of each other and all previous random variables.

Algorithm 3 (Simulation of the representation (22))**. Initialize**. Fix $h_\ell > 0$. Set $X(0) = Z_\ell(0) = x_0$ and $t = 0$. Set $\widetilde{Z}_\ell = Z_\ell(0)$. Set $T_{\text{tau}} = h_\ell$. For each k, set

- $P_{k,1} = P_{k,2} = P_{k,3} = 0$.
- $T_{k,1} = T_{k,2} = T_{k,3} = 0$.

(i) For each $k \in \{1, \dots, R\}$ and $i \in \{1,2,3\}$, set $P_{k,i} = \ln(1/r_{k,i})$, where $r_{k,i}$ is rand$(0,1)$.

(ii) For each k, set

- $A_{k,1} = \lambda_k(X(t)) \wedge \lambda_k(\widetilde{Z}_\ell)$.
- $A_{k,2} = \lambda_k(X(t)) - A_{k,1}$.
- $A_{k,3} = \lambda_k(\widetilde{Z}_\ell) - A_{k,1}$.

(iii) For each $k \in \{1, \dots, R\}$ and $i \in \{1,2,3\}$, set

$$\Delta t_{k,i} = \begin{cases} (P_{k,i} - T_{k,i})/A_{k,i}, & \text{if } A_{k,i} \neq 0 \\ \infty, & \text{if } A_{k,i} = 0 \end{cases}.$$

(iv) Set $\Delta = \min_{k,i}\{\Delta t_{k,i}\}$, and let $\mu \equiv \{k,i\}$ be the indices where the minimum is achieved.

(v) If $t + \Delta \geq T_{\text{tau}}$, then update tau-leaped process

(a) Set $\widetilde{Z}_\ell = Z_\ell(t)$.

(b) For each $k \in \{1, \dots, R\}$ and $i \in \{1,2,3\}$, set $T_{k,i} = T_{k,i} + A_{k,i} \times (T_{\text{tau}} - t)$.

(c) Set $T_{\text{tau}} = T_{\text{tau}} + h_\ell$.

(d) Set $t = T_{\text{tau}}$.

(e) Return to step (ii) or quit.

(vi) Else,

(a) Set $t = t + \Delta$.

(b) Update according to reaction ζ_μ (where minimum occurred in step (iv)).

(c) For each $k \in \{1, \dots, R\}$ and $i \in \{1,2,3\}$, set $T_{k,i} = T_{k,i} + A_{k,i} \times \Delta$.

(d) Set $P_\mu = P_\mu + \ln(1/r)$, where r is rand$(0,1)$.

(e) Return to step (ii) or quit.

The following theorem, which should be compared with Theorem 1, is proven in Section 6.

Theorem 2. *Suppose (X^N, Z_ℓ^N) satisfy* (20) *and* (21) *with $X^N(0) = Z_\ell^N(0)$. Then, there exist positive constants C_1, C_2, that do not depend on N but do depend upon both T and γ, such that*

$$\sup_{t \leq T} \mathbb{E}|X^N(t) - Z_\ell^N(t)|^2 \leq C_1(T,\gamma)N^{-\rho}h_\ell + C_2(T,\gamma)h_\ell^2.$$

We are now in a position to develop MLMC in the stochastic chemical kinetic setting. We return to the $\widehat{Q}_\ell$ terms in (15). Supposing that the test function f is Lipschitz in our domain of interest (note that this is automatic for any reasonable f in the case when mass is conserved), then for $\ell > \ell_0$

$$\begin{aligned}\mathsf{Var}(\widehat{Q}_\ell) &\le C\frac{1}{n_\ell}\mathsf{Var}(Z_\ell^N(T) - Z_{\ell-1}^N(T)) \le C\frac{1}{n_\ell}\mathbb{E}|Z_\ell^N(T) - Z_{\ell-1}^N(T)|^2 \\ &\le C\frac{1}{n_\ell}\left[C_1(T,\gamma)N^{-\rho}h_\ell + C_2(T,\gamma)h_\ell^2\right],\end{aligned}$$

where C is the Lipschitz constant of f and we have applied Theorem 1. Note that if $\rho \ge 1$, the leading order of the error is h_ℓ^2. As a heuristic argument for this behavior, note that if $\rho \ge 1$ and N is large, then the processes are nearing a scaling regime in which deterministic dynamics would be a good approximation for the model X^N. In this case, one should expect that the *squared* difference between two Euler paths should behave like the usual order one error, squared.

We may now conclude that the variance of the estimator $\widehat{Q}$ defined in (16) satisfies

$$\begin{aligned}\mathsf{Var}(\widehat{Q}) &= \mathsf{Var}(\widehat{Q}_{\ell_0}) + \sum_{\ell=\ell_0+1}^{L}\mathsf{Var}(\widehat{Q}_\ell) \\ &\le \frac{K_0}{n_0} + \sum_{\ell=\ell_0+1}^{L} C\frac{1}{n_\ell}\left[C_1(T,\gamma)N^{-\rho}h_\ell + C_2(T,\gamma)h_\ell^2\right],\end{aligned}$$

where $K_0 = \mathsf{Var}(f(Z_{\ell_0}^N(T)))$. For $h > 0$ we define

$$A(h) \overset{\text{def}}{=} \max\{N^{-\rho}h, h^2\}. \tag{23}$$

Letting $n_0 = O(\epsilon^{-2})$, and for $\ell > \ell_0$ letting

$$n_\ell = O\left(\epsilon^{-2}(L-\ell_0)A(h_\ell)\right),$$

we see that

$$\mathsf{Var}(\widehat{Q}) = O(\epsilon^2).$$

As the computational complexity of generating a single path of the coupled processes $(Z_\ell^N, Z_{\ell-1}^N)$ is $O(h_\ell^{-1})$, we see that the total computational complexity of the method with these choices of n_ℓ is of order

$$\begin{aligned}n_0h_{\ell_0}^{-1} + \sum_{\ell=\ell_0+1}^{L} n_\ell h_\ell^{-1} &= \epsilon^{-2}h_{\ell_0}^{-1} + \sum_{\ell=\ell_0+1}^{L}\epsilon^{-2}(L-\ell_0)A(h_\ell)h_\ell^{-1} \\ &= \epsilon^{-2}\left(h_{\ell_0}^{-1} + (L-\ell_0)\sum_{\ell=\ell_0+1}^{L}\max\{N^{-\rho}, h_\ell\}\right) \\ &\le \begin{cases}\epsilon^{-2}(h_{\ell_0}^{-1} + \ln(\epsilon)^2N^{-\rho}), & \text{if } h_\ell < N^{-\rho} \\ \epsilon^{-2}(h_{\ell_0}^{-1} + \ln(\epsilon^{-1})/(1-M)), & \text{if } h_\ell \ge N^{-\rho}\end{cases},\end{aligned} \tag{24}$$

where we used that

$$\sum_{\ell=\ell_0+1}^{L} h_\ell \le \sum_{\ell=1}^{\infty}\frac{1}{M^\ell} = \frac{1}{1-M}. \tag{25}$$

A more careful choice of n_ℓ can potentially reduce the $\ln(\epsilon)$ terms further, see for example [18], but in the present case, the computational complexity will be dominated by $\epsilon^{-2}h_{\ell_0}^{-1}$ in most nontrivial examples. Further, as will be discussed in Section 7, the n_ℓ can be chosen algorithmically, by optimizing for a given problem.

5.1 An unbiased MLMC

We now build an unbiased MLMC estimator for $\mathbb{E}f(X^N(T))$ in a similar manner as before with a single important difference: at the finest scale, we couple X^N with Z_L^N. That is, we use the identity

$$\mathbb{E}f(X^N(T)) = \mathbb{E}[f(X^N(T)) - f(Z_L^N(T))] + \sum_{\ell=\ell_0+1}^{L} \mathbb{E}[f(Z_\ell^N) - f(Z_{\ell-1}^N)] + \mathbb{E}f(Z_{\ell_0}^N(T)).$$

For appropiate choices of n_0, n_ℓ, and n_E, we define the estimators for the three terms above via

$$\begin{aligned}
\widehat{Q}_E &\stackrel{\text{def}}{=} \frac{1}{n_E}\sum_{i=1}^{n_E}(f(X_{[i]}^N(T) - f(Z_{L,[i]}^N(T))),\\
\widehat{Q}_\ell &\stackrel{\text{def}}{=} \frac{1}{n_\ell}\sum_{i=1}^{n_\ell}(f(Z_{\ell,[i]}^N(T)) - f(Z_{\ell-1,[i]}^N(T))), \quad \text{for } \ell \in \{\ell_0+1,\dots,L\}\\
\widehat{Q}_0 &\stackrel{\text{def}}{=} \frac{1}{n_0}\sum_{i=1}^{n_0} f(Z_{\ell_0,[i]}^N(T)),
\end{aligned}$$

and note that

$$\widehat{Q} \stackrel{\text{def}}{=} \widehat{Q}_E + \sum_{\ell=\ell_0}^{L} \widehat{Q}_\ell \tag{26}$$

is an *unbiased* estimator for $\mathbb{E}f(X^N(T))$. Applying both Theorems 1 and 2 yields

$$\begin{aligned}
\mathsf{Var}(\widehat{Q}_E) &\le K_1\frac{1}{n_E}(T,\gamma)A(h_L)\\
\mathsf{Var}(\widehat{Q}_\ell) &\le K_2(T,\gamma)\frac{1}{n_\ell}A(h_\ell), \quad \text{for } \ell \in \{\ell_0+1,\dots,L\}
\end{aligned}$$

where $K_1(T,\gamma)$ and $K_2(T,\gamma)$ are independent of N or h_ℓ, and $A(h)$ is defined in (23). We let

$$\begin{aligned}
n_E &= O(\epsilon^{-2}A(h_L))\\
n_\ell &= O\left(\epsilon^{-2}(L-\ell_0)A(h_\ell)\right), \quad \text{for } \ell \in \{\ell_0,\dots,L\}\\
n_0 &= O(\epsilon^{-2}),
\end{aligned} \tag{27}$$

which gives us

$$\begin{aligned}
\mathsf{Var}(\widehat{Q}) &= \mathsf{Var}(\widehat{Q}_E) + \sum_{\ell=\ell_0+1}^{L}\mathsf{Var}(\widehat{Q}_\ell) + \mathsf{Var}(\widehat{Q}_0)\\
&= O(\epsilon^2) + \sum_{\ell=\ell_0+1}^{L} O(\epsilon^2(L-\ell_0)^{-1}) + O(\epsilon^2)\\
&= O(\epsilon^2).
\end{aligned}$$

The computational complexity is now of order

$$\begin{aligned}
n_E\overline{N} + \sum_{\ell=\ell_0+1}^{L} n_\ell h_\ell^{-1} + n_0 h_{\ell_0}^{-1} &= \overline{N}\epsilon^{-2}A(h_L) + \sum_{\ell=\ell_0+1}^{L} \epsilon^{-2}(L-\ell_0)A(h_\ell)h_\ell^{-1} + \epsilon^{-2}h_{\ell_0}^{-1} \\
&= \epsilon^{-2}\left(\overline{N}A(h_L) + (L-\ell_0)\sum_{\ell=\ell_0}^{L}\max\{N^{-\rho}, h_\ell\} + h_{\ell_0}^{-1}\right) \\
&\le \begin{cases} \epsilon^{-2}\left(\overline{N}A(h_L) + h_{\ell_0}^{-1} + \ln(\epsilon)^2 N^{-\rho}\right), & \text{if } h_\ell < N^{-\rho} \\ \epsilon^{-2}\left(\overline{N}A(h_L) + h_{\ell_0}^{-1} + \ln(\epsilon^{-1})/(1-M)\right), & \text{if } h_\ell \ge N^{-\rho} \end{cases},
\end{aligned} \tag{28}$$

where we again made use of the inequality (25).

5.2 Some observations

A few observations are in order. First, in the above analysis of the unbiased MLMC estimator, the weak error of the process Z_h^N plays *no role*. Thus, there is no reason to choose $h_L = O(\epsilon)$ for a desired accuracy of $\epsilon > 0$. Without having to worry about the bias, we have the opportunity to simply choose h_L "small enough" for $\mathsf{Var}(X^N(\cdot) - Z_L^N(\cdot))$ to be small, which can be approximated with a few preliminary simulations before the full MLMC is carried out (see Section 7 for more implementation details).

Second, one of the main impediments to the use of tau-leaping methods has been the possibility for paths to leave the non-positive orthant. In fact, there have been multiple papers written on the subject of how to enforce non-negativity of species numbers with [2, 10, 12, 44] representing just a sample. We note that for the unbiased MLMC estimator (26) it almost does not matter how, or even if, non-negativity is enforced. So long as the processes are well defined on all of $\mathbb{Z}^d$, for example by defining the intensity functions λ_k in some reasonable way, and we can still quantify the relations given in Theorems 1 and 2, everything above still holds. The cost, to the user, of poorly defining what happens if Z_h leaves the positive orthant will simply be the need for the generation of more paths to reduce the variance of the (still unbiased) estimator.

Third, inspecting (24) and (28) shows that the unbiased MLMC estimator (26) will have an added computational complexity of $O(\overline{N}A(h_L)\epsilon^{-2})$ as compared with the biased MLMC estimator (16). The authors feel that $\overline{N}A(h_L)$ would have to be quite substantial to warrant not using the unbiased version as one can never be 100% sure that the bias falls within the target range.

Fourth, note that the following *always* holds so long as $\gamma \le 0$ (see Section 6):

$$\begin{aligned}
\text{Computational complexity of unbiased MLMC} &= O\left(\epsilon^{-2}(NA(h_L) + h_{\ell_0}^{-1} + \text{log term})\right) \\
&\ll O\left(\epsilon^{-2}\overline{N}\right) \\
&= \begin{array}{l}\text{Computational complexity of exact algorithm} \\ \text{with crude Monte Carlo}\end{array}.
\end{aligned} \tag{29}$$

Thus, so long as $\gamma \le 0$, the unbiased MLMC estimator should be the method of choice over using an exact algorithm alone together with crude Monte Carlo, which is by far the most popular method today. For example, in the case when the system satisfies the classical scaling detailed in Section 2.1, we have that $\rho = 1$, and $\beta_k + \nu_k \cdot \alpha \equiv 1$. Therefore, $\overline{N} = N$ and, as there is little reason to use an approximate method with a time step that is smaller than the order of magnitude of the wait time between jumps for an exact method, we may assume that $h_L > 1/N = N^{-\rho}$. Therefore, in this specific case, $A(h_L) = h_L^2$ and the computational speedup predicted by (28) and/or (29) is of the order

$$\text{Speed-up factor} \approx \frac{\epsilon^{-2}N}{\epsilon^{-2}(Nh_L^2 + h_{\ell_0}^{-1} + \log(\epsilon))} = \frac{N}{Nh_L^2 + h_{\ell_0}^{-1} + \log(\epsilon)}.$$

Thus we have

$$\text{Speed-up factor} \gtrapprox \min\left(h_L^{-2}, Nh_{\ell_0}\right).$$

Therefore, even though the method is unbiased, the computational burden has been shifted from the exact process to that of an approximate process with a crude time-step. This behavior is demonstrated in the example found in Section 8, though on a system not satisfying the classical scaling.

Note also that (29) holds even if $\overline{N}$, the approximate cost of computing a single path, is not extremely large. For example, even if the cost is only in the hundreds, or maybe thousands, of steps per exact path, the above analysis points out that if great accuracy is required (so that ϵ^{-2} is very large), the unbiased MLMC estimator will still decrease the computational complexity substantially. It should be pointed out that in these cases of moderate $\overline{N}$, we will typically have $\gamma \le 0$ and so the analysis will hold.

The conclusion of this analysis, backed up by the example in Section 8, is that MLMC, with processes coupled via the representations (19) and (22), and the unbiased MLMC in particular, produce substantial gains in computational efficiency and could become standard algorithms in the sciences. Further techniques, however, will need to be developed in the case $\gamma > 0$, and this will be a focus for future work.

6 Delayed proofs of Theorems 1 and 2

We begin by focussing on the proof of Theorem 2, which is restated here for completeness.

Theorem 2. *Suppose (X^N, Z_ℓ^N) satisfy (20) and (21) with $X^N(0) = Z_\ell^N(0)$. Then, there exist positive constants C_1, C_2, that do not depend on N but do depend upon both T and γ, such that*

$$\sup_{t\le T} \mathbb{E}|X^N(t) - Z_\ell^N(t)|^2 \le C_1(T,\gamma)N^{-\rho}h_\ell + C_2(T,\gamma)h_\ell^2.$$

We start with the following lemma, which already points out why our analytical results are most useful in the case when $\gamma \le 0$. The case $\gamma \le 0$ is certainly common, though is by no means ubiquitous.

Lemma 3. Suppose (X^N, Z_ℓ^N) satisfy (20) and (21) with $X^N(0) = Z_\ell^N(0)$. Then, there exists positive constants c_1, c_2, independent of N, γ, and T, such that

$$\sup_{t\le T} \mathbb{E}|X^N(t) - Z_\ell^N(t)| \le \left(c_1 N^{2\gamma} T e^{c_2 N^\gamma T}\right) h_\ell.$$

Proof. Note that

$$\begin{aligned}
\mathbb{E}|X^N(t) - Z_\ell^N(t)| \\
&= \mathbb{E}\Big| \sum_k Y_{k,2}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(X^N(s)) - \lambda_k(X^N(s)) \wedge \lambda_k(Z_\ell^N \circ \eta_\ell(s))ds\right)\zeta_k^N \\
&\quad - \sum_k Y_{k,3}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N \circ \eta_\ell(s)) - \lambda_k(X^N(s)) \wedge \lambda_k(Z_\ell^N \circ \eta_\ell(s))ds\right)\zeta_k^N\Big| \\
&\le \sum_k |\zeta_k^N|\Big[\mathbb{E}Y_{k,2}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(X^N(s)) - \lambda_k(X^N(s)) \wedge \lambda_k(Z_\ell^N \circ \eta_\ell(s))ds\right) \\
&\qquad + \mathbb{E}Y_{k,3}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N \circ \eta_\ell(s)) - \lambda_k(X^N(s)) \wedge \lambda_k(Z_\ell^N \circ \eta_\ell(s))ds\right)\Big] \\
&= \sum_k |\zeta_k^N| N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \mathbb{E}|\lambda_k(X^N(s)) - \lambda_k(Z_\ell^N \circ \eta_\ell(s))|ds \\
&\le N^\gamma C\int_0^t \mathbb{E}|X^N(s) - Z_\ell^N \circ \eta_\ell(s)|ds,
\end{aligned}$$

where $C > 0$ is some constant and we used that the λ_k are Lipschitz (recall Remark 3). Adding and subtracting the obvious terms yields

$$\mathbb{E}|X^N(t) - Z_\ell^N(t)| \le N^\gamma C\int_0^t \mathbb{E}|Z_\ell^N(s) - Z_\ell^N \circ \eta_\ell(s)|ds + N^\gamma C\int_0^t \mathbb{E}|X^N(s) - Z_\ell^N(s)|ds. \tag{30}$$

The integrand of the first term on the right hand side of (30) satisfies

$$\mathbb{E}|Z_\ell^N(s) - Z_\ell^N \circ \eta_\ell(s)| \le \sum_k |\zeta_k^N| N^{\beta_k+\nu_k\cdot\alpha}\mathbb{E}\int_{\eta_\ell(s)}^s \lambda_k(Z_\ell^N(\eta_\ell(r))dr \le CN^\gamma h_\ell, \tag{31}$$

where $C > 0$ is a constant. Collecting the above yields,

$$\mathbb{E}|X^N(t) - Z_\ell^N(t)| \le C_1 N^{2\gamma} t h_\ell + C_2 N^\gamma \int_0^t \mathbb{E}|X^N(s) - Z_\ell^N(s)|ds,$$

for some positive constants C_1, C_2 that are independent of N, γ, and T. The result now follows from Gronwall's inequality. $\square$

We note that Lemma 3 is a worst case scenario due to the appearance of the term N^γ in the exponent. However, considering the network $S_1 \overset{N^\gamma}{\to} 2S_1$ (exponential growth), shows this to actually be a sharp estimate. The vast majority of networks will behave much better than this case, and it is an interesting future problem to classify those networks for which the methods being introduced in this paper behave in a manner that is vastly superior to the analytical bound captured in Lemma 3.

We are now in position to prove Theorem 2.

Proof. (of Theorem 2.) We have

$$X^N(t) - Z_\ell^N(t) = M^N(t) + \int_0^t F^N(X^N(s)) - F^N(Z_\ell^N \circ \eta_\ell(s))ds,$$

where

$$\begin{aligned} M^N(t) \overset{\text{def}}{=} \sum_k \Bigg[Y_{k,2}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(X^N(s)) - \lambda_k(X^N(s))\wedge\lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\right) \\ - N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(X^N(s)) - \lambda_k(X^N(s))\wedge\lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\Bigg]\zeta_k^N \\ - \sum_k \Bigg[Y_{k,3}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s)) - \lambda_k(X^N(s))\wedge\lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\right) \\ + N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s)) - \lambda_k(X^N(s))\wedge\lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\Bigg]\zeta_k^N, \end{aligned}$$

is a martingale, and

$$F^N(x) = \sum_k N^{\beta_k+\nu_k\cdot\alpha}\lambda_k(x)\zeta_k^N.$$

Note that based upon our assumptions, we have that

$$|F^N(x) - F^N(y)| \le CN^\gamma|x-y|, \tag{32}$$

where $C > 0$ is a constant that does not depend upon N or γ. The quadratic covariation matrix of M^N is

$$[M^N](t) = \sum_k \zeta_k^N(\zeta_k^N)^T(J_{k,2}^N(t) + J_{k,3}^N(t)),$$

where

$$\begin{aligned} J_{k,2}^N(t) &\overset{\text{def}}{=} Y_{k,2}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(X^N(s)) - \lambda_k(X^N(s))\wedge\lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\right) \\ J_{k,3}^N(t) &\overset{\text{def}}{=} Y_{k,3}\left(N^{\beta_k+\nu_k\cdot\alpha}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s)) - \lambda_k(X^N(s))\wedge\lambda_k(Z_\ell^N\circ\eta_\ell(s))ds\right). \end{aligned}$$

Thus,

$$\mathbb{E}[M^N](t) = \sum_k \zeta_k^N(\zeta_k^N)^T N^{\beta_k+\nu_k\cdot\alpha}\mathbb{E}\int_0^t \left|\lambda_k(X^N(s)) - \lambda_k(Z_\ell^N\circ\eta_\ell(s))\right| ds,$$

and, in particular,

$$E[M^N]_{ii}(t) = \sum_k (\zeta_{ik}^N)^2 N^{\beta_k+\nu_k\cdot\alpha}\mathbb{E}\int_0^t \left|\lambda_k(X^N(s)) - \lambda_k(Z_\ell^N\circ\eta_\ell(s))\right| ds. \tag{33}$$

We note that

$$\mathbb{E}[|X^N(t) - Z_\ell^N(t)|^2] \le 2\mathbb{E}[|M^N(t)|^2] + 2\mathbb{E}\left|\int_0^t F^N(X^N(s)) - F^N(Z_\ell^N\circ\eta_\ell(s))ds\right|^2, \tag{34}$$

and we may handle the two terms on the right hand side of the above equation separately.

First, by (33),

$$\begin{aligned} \mathbb{E}[|M^N(t)|^2] &\le \sum_i\sum_k (\zeta_{ik}^N)^2 N^{\beta_k+\nu_k\cdot\alpha}\mathbb{E}\int_0^t \left|\lambda_k(X^N(s)) - \lambda_k(Z_\ell^N\circ\eta_\ell(s))\right| ds \\ &= \sum_k |\zeta_k^N|^2 N^{\beta_k+\nu_k\cdot\alpha}\mathbb{E}\int_0^t \left|\lambda_k(X^N(s)) - \lambda_k(Z_\ell^N\circ\eta_\ell(s))\right| ds \qquad (35) \\ &\le 2CN^\gamma N^{-\rho}\mathbb{E}\int_0^t \left|X^N(s) - Z_\ell^N\circ\eta_\ell(s)\right| ds, \end{aligned}$$

where C is a constant independent of N, t, and γ. After adding and subtracting $Z_\ell^N(s)$, using (31), and applying Lemma 3, we conclude that

$$\mathbb{E}[|M^N(t)|^2] \leq (c_1 N^{3\gamma} T^2 e^{c_2 N^\gamma T}) N^{-\rho} h_\ell, \tag{36}$$

for some constants c_1, c_2 that do not depend upon T, γ, or N, and which will change during the course of the proof.

Turning to the second term on the right hand side of (34), making use of (32) we have for some $C > 0$ independent of T, γ, and N,

$$\begin{aligned}\mathbb{E}\bigg(\int_0^t |F^N(X^N(s)) - F^N(Z_\ell^N \circ \eta_\ell(s))| ds\bigg)^2 \\ \leq C\mathbb{E}\bigg(\int_0^t |Z_\ell^N \circ \eta_\ell(s) - Z_\ell^N(s)| ds\bigg)^2 + C\mathbb{E}\bigg(\int_0^t |X^N(s) - Z_\ell^N(s)| ds\bigg)^2.\end{aligned} \tag{37}$$

The first term on the right of (37) can be bounded via

$$\begin{aligned}\mathbb{E}\bigg(\int_0^T |Z_\ell^N \circ \eta_\ell(s) - Z_\ell^N(s)| ds\bigg)^2 &\leq T\mathbb{E}\int_0^T |Z_\ell^N \circ \eta_\ell(s) - Z_\ell^N(s)|^2 ds \\ &= T\sum_{i=1}^n \int_{t_i}^{t_i+h} \mathbb{E}|Z_\ell^N \circ \eta_\ell(s) - Z_\ell^N(s)|^2 ds.\end{aligned} \tag{38}$$

We have that

$$\begin{aligned}\mathbb{E}|Z_\ell^N \circ \eta_\ell(s) - Z_\ell^N(s)|^2 &\leq \sum_k |\zeta_k|^2 \bigg[N^{\beta_k + \nu_k \cdot \alpha}\mathbb{E}\int_{\eta_\ell(s)}^s \lambda_k(Z_\ell^N \circ \eta_\ell(r)) dr \\ &\qquad + N^{2(\beta_k + \nu_k\cdot\alpha)}\mathbb{E}\bigg(\int_{\eta_\ell(s)}^s \lambda_k(Z_\ell^N \circ \eta_\ell(r)) dr\bigg)^2\bigg] \\ &\leq C N^\gamma N^{-\rho} h_\ell + C N^{2\gamma} h_\ell^2,\end{aligned} \tag{39}$$

for some constant $C > 0$. Combining (38) and (39) shows

$$\mathbb{E}\bigg(\int_0^T |Z_\ell^N \circ \eta_\ell(s) - Z_\ell^N(s)| ds\bigg)^2 \leq T(C N^\gamma N^{-\rho} h_\ell + C N^{2\gamma} h_\ell^2). \tag{40}$$

Combining (40) with (37) then yields

$$\begin{aligned}\mathbb{E}\bigg(\int_0^t |F^N(X^N(s)) - F^N(Z_\ell^N \circ \eta_\ell(s))| ds\bigg)^2 &\leq c_1 T N^\gamma N^{-\rho} h_\ell + c_2 T N^{2\gamma} h_\ell^2 \\ &\quad + c_3 t \int_0^t \mathbb{E}|X^N(s) - Z_\ell^N(s)|^2 ds,\end{aligned} \tag{41}$$

for some constants c_1, c_2, c_3 that do not depend upon T, N, or γ. Equations (34), (36), and (41) yield

$$\begin{aligned}\mathbb{E}[|X^N(t) - Z_\ell^N(t)|^2] &\leq (c_1 N^{3\gamma} T^2 e^{c_2 N^\gamma T}) N^{-\rho} h_\ell + c_1 T N^\gamma N^{-\rho} h_\ell + c_2 T N^{2\gamma} h_\ell^2 \\ &\quad + c_3 t \int_0^t \mathbb{E}|X^N(s) - Z_\ell^N(s)|^2 ds.\end{aligned}$$

The result now follows from Gronwall's inequality. □

We turn our focus to the proof of Theorem 1, which is restated here for completeness.

Theorem 1. *Suppose* $(Z_\ell^N, Z_{\ell-1}^N)$ *satisfy* (17) *and* (18) *with* $Z_\ell^N(0) = Z_{\ell-1}^N(0)$. *Then, there exist positive constants* C_1, C_2, *that do not depend on* N *but do depend upon both* T *and* γ, *such that*

$$\sup_{t \le T} \mathbb{E}|Z_\ell^N(t) - Z_{\ell-1}^N(t)|^2 \le C_1(T,\gamma)N^{-\rho}h_\ell + C_2(T,\gamma)h_\ell^2.$$

Proof. (of Theorem 1.) A direct proof can be written along the lines of that for Theorem 1. A separate, cruder, proof would simply add and subtract $X^N(t)$ to $|Z_\ell^N(t) - Z_{\ell-1}^N(t)|^2$ and use Theorem 1 combined with the triangle inequality. □

7 Implementation issues

The analysis in Sections 5 and 6 gives an order of magnitude for the number of paths, n_ℓ, that must be chosen at each level so as to attain a desired accuracy. This was needed to prove that the computational complexity of a given problem can be greatly reduced with an appropriate choice of the n_ℓ. However, the analysis does not tell us what the n_ℓ should be with precision, nor does it tell us that these are the optimal n_ℓ. Letting V_ℓ denote the variance of $\widehat{Q}_\ell$ for a given n_ℓ, and CPU_ℓ be the CPU time needed to generate n_ℓ paths, we know that

$$CPU_\ell = \frac{K_\ell}{V_\ell} + O(1),$$

for some K_ℓ. Further, for a given tolerance, ϵ, we need

$$\mathsf{Var}(\widehat{Q}) = \sum_\ell V_\ell = (\epsilon/1.96)^2, \tag{42}$$

for, say, a 95% confidence interval (where the term 1.96 will be changed depending upon the size of the confidence interval desired). We may approximate each K_ℓ with a number of preliminary simulations (not used in the full implementation), and then minimize

$$\sum_\ell \frac{K_\ell}{V_{n_\ell}},$$

subject to the constraint (42). This will give us target variances, V_ℓ, for each level. We may then simulate each level until enough paths have been generated for the variance of the estimator at that level to be below the target V_ℓ.

In Section 8, we use 300 paths for solving this embedded optimization problem for the example considered.

8 An Example

For an instructive example, consider a simple model of gene transcription and translation:

$$G \overset{25}{\to} G + M, \quad M \overset{1000}{\to} M + P, \quad 2P \overset{0.001}{\to} D, \quad M \overset{0.1}{\to} \emptyset, \quad P \overset{1}{\to} \emptyset.$$

Here a single gene is being translated into mRNA, which is then being transcribed into proteins, and finally the proteins produce stable dimers. The final two reactions represent degradation of mRNA and proteins, respectively. Suppose we start with one gene and no other molecules, and want to estimate the expected number of dimers at time $T = 1$ to an accuracy of ± 1 with 95% confidence. Therefore, we want the variance of our estimator to be smaller than $(1/1.96)^2 = .2603$.

While $\epsilon = 1$ for the unscaled version of this problem, the simulation of just a few paths of the system will show that there will be somewhere in the magnitude of 3,500 dimers at time $T = 1$. Therefore, for the scaled system, we are asking for an accuracy of $\widetilde{\epsilon} = 1/3500 \approx 0.0002857$. Also, a few paths (100 is sufficient) shows that the order of magnitude of the variance of the normalized number of dimers is approximately 0.11. Thus, the approximate number of exact sample paths we will need to generate can be found by solving

$$\frac{1}{n}\mathsf{Var}(\text{normalized \# dimers}) = (\widetilde{\epsilon}/1.96)^2 \implies n = 5.18 \times 10^6.$$

Therefore, we will need approximately five million independent sample paths generated via an exact algorithm. Implementing the modified next reaction method [1] on our machine (using Matlab), each path takes approximately 0.03 CPU seconds to generate. Therefore, the approximate amount of time to solve this particular problem will be 155,000 CPU S, which is about forty three hours. The outcome of such a simulation is detailed in the table below where "# updates" refers to the total number, over all paths, of updates to the system performed, and is used as a quantification for the computational complexity of the different methods under consideration:

Method: Exact algorithm with crude Monte Carlo.

Approximation	# paths	CPU Time	Variance of estimator	# updates
3714.2 $\pm$1.0	4,740,000	1.49 $\times 10^5$ CPU S	0.25995	8.27 $\times 10^{10}$

Next, we solved the problem using tau-leaping with various step-sizes, combined with a crude Monte Carlo estimator. The results of those simulations are detailed in the table below, in which the bias of the approximate algorithm has become apparent:

Method: tau-leaping with crude Monte Carlo.

Step-size	Approximation	# paths	CPU Time	Variance of estimator	# updates
$h = 3^{-7}$	3,712.5$\pm$1.0	4,720,000	46,000 CPU S	0.26029	4.99$\times 10^{10}$
$h = 3^{-6}$	3,708.4 $\pm$1.0	4,720,000	19,764 CPU S	0.26014	1.66$\times 10^{10}$
$h = 3^{-5}$	3,694.5 $\pm$1.0	4,690,000	7,081 CPU S	0.26019	5.52$\times 10^{9}$
$h = 3^{-4}$	3,654.8 $\pm$1.0	4,635,000	2,476 CPU S	0.26029	1.80$\times 10^{9}$

Note that implementing tau-leaping with a step-size of $h = 3^{-8}$ would take nearly as long as simply implementing an exact algorithm.

Next, we implemented the biased version of MLMC with various step-sizes. The results of those simulations are detailed in the table below, where the approximations and CPU times should be compared with the tables above. The CPU times stated include the time needed to solve the embedded optimization problem discussed in Section 7, which took approximately 20, 10, and 4 CPU seconds, for $L = 7$, $L = 6$, and $L = 5$, respectively. We used 300 simulations at each level to solve the optimization problem.

Method: biased MLMC with $M = 3$, $\ell_0 = 2$, and L ranging from 7 to 5.

Finest step-size	Approximation	CPU Time	Variance of estimator	# updates
$h_L = 3^{-7}$	3,711.4 $\pm$1.0	1,669.5 CPU S	0.25910	5.75 $\times 10^8$
$h_L = 3^{-6}$	3,707.1 $\pm$1.0	1,479.0 CPU S	0.26012	5.22 $\times 10^8$
$h_L = 3^{-5}$	3,693.7 $\pm$1.0	1,171.2 CPU S	0.25996	4.35 $\times 10^8$

Note that the gain in computational complexity, as quantified by the # updates, over straight tau-leaping with a finest level of $h_L = 3^{-7}$ is 87 fold, with straight tau-leaping taking 27.5 times longer. Also note that the bias of the approximation method is still apparent.

Finally, we implemented the unbiased version of MLMC with various step-sizes. The results of those simulations are detailed in the table below. The CPU times stated include the time needed to solve the embedded optimization problem discussed in Section 7, each of which took approximately 20 CPU S. We used 300 simulations at each level to solve the optimization problem.

Method: unbiased MLMC with $\ell_0 = 2$, and M and L detailed below.

Step-size parameters	Approximation	CPU Time	Variance of estimator	# updates
$M = 3$, $L = 6$	3,713.5 ± 1.0	1,699.9 CPU S	0.25961	5.87×10^8
$M = 3$, $L = 5$	3,713.9 ± 1.0	1,656.1 CPU S	0.25647	5.78×10^8
$M = 3$, $L = 4$	3,713.3 ± 1.0	2,257.6 CPU S	0.26010	7.39×10^8
$M = 4$, $L = 4$	3,714.4 ± 1.0	1,726.9 CPU S	0.25869	8.00×10^8
$M = 4$, $L = 5$	3,713.1 ± 1.0	1,568.4 CPU S	0.25988	7.35×10^8

We see that the unbiased MLMC estimator behaves as the analysis predicts. Further, the exact algorithm with crude Monte Carlo, by far the most commonly used method in the literature, demanded 143 times more updates and 90 times more CPU time than our unbiased MLMC estimator, with the precise speedups depending upon the choice of M and L. Note that $M, L,$ and ℓ_0 can be chosen via an optimization problem similar to that discussed in Section 7 as a pre-computation that will typically cost relatively little in terms of CPU time.

We feel it is instructive to give more details to at least one choice of M and L for the unbiased MLMC estimator. For the case with $M = 3$, $L = 5$, $\ell_0 = 2$, we give below the relevant data for the different levels. Below, by $(X, Z_{3^{-5}})$ we mean the level in which the exact process is coupled to the approximate process with $h = 3^{-5}$, and by $(Z_{3^{-\ell}}, Z_{3^{-\ell+1}})$ we mean the level with $Z_{3^{-\ell}}$ coupled to $Z_{3^{-\ell+1}}$.

Level	# paths	CPU Time	Variance of estimator	# updates
$(X, Z_{3^{-5}})$	900	58.8 CPU S	0.060300	1.56×10^7
$(Z_{3^{-5}}, Z_{3^{-4}})$	19,700	109.9 CPU S	0.029179	3.30×10^7
$(Z_{3^{-4}}, Z_{3^{-3}})$	99,600	193.7 CPU S	0.026804	5.70×10^7
$(Z_{3^{-3}}, Z_{3^{-2}})$	445,800	334.7 CPU S	0.036469	8.85×10^7
Tau-leap with $h = 3^{-2}$	9,920,000	935.9 CPU S	0.103718	3.84×10^8
Totals		1,633.1 CPU S	0.256470	5.78×10^8

The total time with the optimization problem was 1,656.1 CPU S. Note that most of the CPU time was taken up at the coarsest level. Also, while the exact algorithm with crude Monte Carlo demanded the generation of almost five million exact sample paths, we needed only 900 such paths. Of course, we needed nearly ten million paths at the coarsest level, but these paths are very cheap to generate. Finally, we note that the optimization problem divided up the total desired variance into non-uniform sizes with the more computationally intensive levels being allowed to have a higher variance than the computationally non-intensive levels.

References

[1] David F. Anderson, *A modified next reaction method for simulating chemical systems with time dependent propensities and delays*, J. Chem. Phys. **127** (2007), no. 21, 214107.

[2] ______, *Incorporating postleap checks in tau-leaping*, J. Chem. Phys. **128** (2008), no. 5, 054103.

[3] David F. Anderson, Arnab Ganguly, and Thomas G. Kurtz, *Error analysis of tau-leap simulation methods*, to appear in the Annals of Applied Probability. Available on arxiv.org, 2011.

[4] David F. Anderson and Masanori Koyama, *Weak error analysis of tau-leaping methods for multiscale stochastic chemical kinetic systems*, Submitted to Annals of Applied Probability. Available on arxiv.org.

[5] David F. Anderson and Thomas G. Kurtz, *Continuous time Markov chain models for chemical reaction networks*, Design and Analysis of Biomolecular Circuits: Engineering Approaches to Systems and Synthetic Biology (H. Koeppl et al., ed.), Springer, 2011, pp. 3–42.

[6] Adam Arkin, John Ross, and Harley H. McAdams, *Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected Escherichia coli cells*, Genetics **149** (1998), 1633–1648.

[7] Karen Ball, Thomas G. Kurtz, Lea Popovic, and Greg Rempala, *Asymptotic analysis of multiscale approximations to reaction networks*, Ann. Appl. Prob. **16** (2006), no. 4, 1925–1961.

[8] A. Barth, C. Schwab, and N. Zollinger, *Multi-level Monte Carlo finite element method for elliptic pdes with stochastic coefficients*, to appear in Numerische Mathematik, 2011.

[9] A. Becskei, B. B. Kaufmann, and A. Van Oudenaarden, *Contributions of low molecule number and chromosomal positioning to stochastic gene expression*, Nature Genetics **37** (2005), no. 9, 937–944.

[10] Yang Cao, Daniel T. Gillespie, and Linda R. Petzold, *Avoiding negative populations in explicit Poisson tau-leaping*, J. Chem. Phys. **123** (2005), 054104.

[11] ______, *Efficient step size selection for the tau-leaping simulation method*, J. Chem. Phys. **124** (2006), 044109.

[12] Abhijit Chatterjee and Dionisios G. Vlachos, *Binomial distribution based τ-leap accelerated stochastic simulation*, J. Chem. Phys. **122** (2005), 024112.

[13] K.A. Cliffe, M.B. Giles, R. Scheichl, and A.L. Teckentrup, *Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients*, to appear in Computing and Visualization in Science, 2011.

[14] Michael B. Elowitz, Arnold J. Levin, Eric D. Siggia, and Peter S. Swain, *Stochastic gene expression in a single cell*, Science **297** (2002), no. 5584, 1183–1186.

[15] Stewart N. Ethier and Thomas G. Kurtz, *Markov processes: Characterization and convergence*, John Wiley & Sons, New York, 1986.

[16] M.A. Gibson and J. Bruck, *Efficient exact stochastic simulation of chemical systems with many species and many channels*, J. Phys. Chem. A **105** (2000), 1876–1889.

[17] M.B. Giles, *Improved multilevel Monte Carlo convergence using the Milstein scheme*, Monte Carlo and Quasi-Monte Carlo Methods 2006 (A. Keller, S. Heinrich, and H. Niederreiter, eds.), Springer-Verlag, 2007, pp. 343–358.

[18] M.B. Giles, *Multilevel Monte Carlo path simulation*, Operations Research **56** (2008), 607–617.

[19] M.B. Giles and B.J. Waterhouse, *Multilevel quasi-Monte Carlo path simulation*, Advanced Financial Modelling, Radon Series on Computational and Applied Mathematics (2009), 165–181.

[20] Mike Giles, Desmond J. Higham, and Xuerong Mao, *Analysing multi-level Monte Carlo for options with non-globally Lipschitz payoff*, Finance ad Stochastics **13** (2009), 403–413.

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

[22] ______, *Exact stochastic simulation of coupled chemical reactions*, J. Phys. Chem. **81** (1977), no. 25, 2340–2361.

[23] ______, *Approximate accelerated simulation of chemically reaction systems*, J. Chem. Phys. **115** (2001), no. 4, 1716–1733.

[24] D. T. Gillespie and Linda R. Petzold, *Improved leap-size selection for accelerated stochastic simulation*, J. Chem. Phys. **119** (2003), no. 16, 8229–8234.

[25] Peter W. Glynn, *A generalized semi-Markov process formalism for discrete event systems*, Proc. IEEE **77** (1989), no. 1, 14–23.

[26] Stefan Heinrich, *Multilevel Monte Carlo methods*, Springer, Lect. Notes Comput. Sci. **2179** (2001), 58–67.

[27] Fred J. Hickernell, Thomas Müller-Gronbach, Ben Niu, and Klaus Ritter, *Multi-level Monte Carlo algorithms for infinite-dimensional integration on RN*, Journal of Complexity **26** (2010), 229–254.

[28] D. J. Higham, X. Mao, M. Roj, Q. Song Q, and G Yin, *Mean exit times and the multi-level Monte Carlo method*, Tech. Report 5, Unversity of Strathclyde, Department of Mathematics and Statistics, 2011.

[29] Desmond J. Higham, *Stochastic ordinary differential equations in applied and computational mathematics*, IMA J. Applied Math. **76** (2011), 449–474.

[30] Marting Hutzenthaler, Arnulf Jenstzen, and Peter E. Kloeden, *Divergence of the multilevel Monte Carlo method*, available on arxiv.org, 2011.

[31] Hye-Won Kang and Thomas G. Kurtz, *Separation of time-scales and model reduction for stochastic reaction networks*, submitted, 2011.

[32] Thomas G. Kurtz, *The relationship between stochastic and deterministic models for chemical reactions*, J. Chem. Phys. **57** (1972), no. 7, 2976–2978.

[33] ______, *Strong approximation theorems for density dependent Markov chains*, Stoch. Proc. Appl. **6** (1977/78), 223–240.

[34] ______, *Approximation of population processes*, CBMS-NSF Reg. Conf. Series in Appl. Math.: 36, SIAM, 1981.

[35] Tiejun Li, *Analysis of explicit tau-leaping schemes for simulating chemically reacting systems*, SIAM Multiscale Model. Simul. **6** (2007), no. 2, 417–436.

[36] Hédia Maamar, Arjun Raj, and David Dubnau, *Noise in gene expression determines cell fate in* bacillus subtilis, Science **317** (2007), no. 5837, 526–529.

[37] Donald A. McQuarrie, *Stochastic approach to chemical kinetics*, J. Appl. Prob. **4** (1967), 413–478.

[38] J. Paullson, *Summing up the noise in gene networks*, Nature **427** (2004), 415–418.

[39] J. M. Pedraza and A. van Oudenaarden, *Noise propagation in gene networks*, Science **307** (2005), 1886–1888.

[40] J. M. Raser and E. K. O'Shea, *Control of stochasticity in eukaryotic gene expression*, Science **304** (2004), 1811–1814.

[41] Muruhan Rathinam, Linda R. Petzold, Yang Cao, , and Daniel T. Gillespie, *Consistency and stability of tau-leaping schemes for chemical reaction systems*, SIAM Multiscale Model. Simul. **3** (2005), 867–895.

[42] Eric Renshaw, *Stochastic population processes*, Oxford University Press, Oxford, 2011.

[43] Brian D. Ripley, *Stochastic simulation*, John Wiley & Sons, Inc., New York, NY, USA, 1987.

[44] T. Tian and K. Burrage, *Binomial leap methods for simulating stochastic chemical kinetics*, J. Chem. Phys. **121** (2004), 10356.

[45] D. J. Wilkinson, *Stochastic modelling for systems biology*, Chapman and Hall/CRC Press, 2006.