Multi-level Monte Carlo for continuous time Markov chains, with applications in biochemical kinetics

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

November 23, 2011

Abstract

We show how to extend a recently proposed multi-level Monte Carlo approach to the continuous time Markov chain setting, thereby greatly 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, exploiting a coupling of the requisite processes that is easy to simulate while providing a small variance for the estimator. Further, and in a stark departure from other implementations of multi-level Monte Carlo, we show how 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 dramatically improve, in a quantifiable manner, the basic computational complexity of current approaches that have many names and variants across the scientific literature, including the Bortz-Kalos-Lebowitz algorithm, discrete event simulation, dynamic Monte Carlo, kinetic Monte Carlo, the n-fold way, the next reaction method, the residence-time algorithm, the stochastic simulation algorithm, Gillespie's algorithm, and tau-leaping. The new algorithm applies generically, but we also give an example where the coupling idea alone, even without a multi-level discretization, can be used to improve efficiency by exploiting system structure. Stochastically modeled chemical reaction networks provide a very important application for this work. Hence, we use this context for our notation, terminology, natural scalings, and computational examples.

Keywords: continuous time Markov chain, reaction network, computational complexity, Gillespie, next reaction method, random time change, tau-leaping, variance.

1 Introduction

This paper concerns the efficient computation of expectations for continuous time Markov chains. Specifically, we extend the multi-level Monte Carlo approach of Giles [18], with related earlier work by Heinrich [24], to this setting. We study the wide class of systems that can be written using the random time change representation of Kurtz [15, Chapter 6] [32] in the form

$$X(t) = X(0) + \sum_{k=1}^{R} Y_k \left(\int_0^t \lambda_k(X(s))ds \right) \zeta_k, \tag{1}$$

where the Y_k are independent unit-rate Poisson processes, $\zeta_k \in \mathbb{R}^d$, and the functions λ_k are the associated intensity, or propensity, functions. While such models are used in nearly all branches of the sciences,

[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

especially in the studies of queues and populations, their use has recently exploded in the biosciences, and we use this application area for the setting of our work. We will formally introduce these models in Section 2, however we begin by demonstrating how two different models, one from chemistry and one from queuing, can be represented via (1).

First, consider a linear reversible chemical network

$$S_1 \underset{\kappa_2}{\overset{\kappa_1}{\rightleftarrows}} S_2,$$

in which molecules of type S_1 convert to molecules of type S_2 at rate $\kappa_1 X_1$, where X_1 is the number of S_1 molecules, and molecules of type S_2 convert to S_1 at rate $\kappa_2 X_2$. Here we are assuming the system satisfies mass action kinetics, see Section 2. The usual stochastic model, written in the framework of (1), is then

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

Next, consider an $M/M/k$ queue in which arrivals are happening at a constant rate $\lambda > 0$, and there are k servers, with each serving at a rate $\mu > 0$. Letting $X(t)$ denote the number of customers in the queue at time t,

$$X(t) = X(0) + Y_1(\lambda t) - Y_2\left(\mu \int_0^t (X(s) \wedge k)\, ds\right),$$

where we define $a \wedge b \overset{\text{def}}{=} \min\{a, b\}$.

There are multiple algorithms available to compute exact sample paths of continuous time Markov chains, and, though they are all only slight variants of each other, they go by different names depending upon the branch of science within which they are being applied. These include the Bortz-Kalos-Lebowitz algorithm, discrete event simulation, dynamic Monte Carlo, kinetic Monte Carlo, the n-fold way, the residence-time algorithm, the stochastic simulation algorithm, the next reaction method, and Gillespie's algorithm, where the final two are the most commonly referred to algorithms in the biosciences. As the computational cost of exact algorithms scales linearly with the number of jump events (i.e. reactions), 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 [21] in the chemical kinetic setting, have been developed with the explicit aim of greatly lowering the computational complexity of each path simulation while controlling the bias [3, 5, 6, 27, 34, 35].

A common task in the study of stochastic models, and the main focus of this paper, is to approximate $\mathbb{E}f(X(T))$, where f is a scalar-valued function of the state of the system which gives a measurement of interest. For example, the function f could be:

1. $f(X(T)) = X_i(T)$, yielding estimates for mean values, or
2. $f(X(T)) = X_i(T)X_j(t)$, which can be used with estimates for the mean values to provide estimates of variances (when $i = j$) and covariances (when $i \neq j$), or
3. $f(X(T)) = 1_{\{X(T)\in B\}}$, the indicator function giving 1 if the state is in some specified set. Such functions could also be used to construct histograms, for example, since $\mathbb{E}f(X(T)) = P\{X(T) \in B\}$.

Suppose we use an exact simulation algorithm to approximate $\mathbb{E}f(X(T))$ to $O(\epsilon)$ accuracy 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 many settings, 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 a weakly order one method). We first make the trivial observation that the estimator

$$\mu_n = \frac{1}{n}\sum_{j=1}^{n} f(Z_{h,[j]}(T)), \tag{2}$$

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

$$\mathbb{E}f(X(T)) - \mu_n = \big[\mathbb{E}f(X(T)) - \mathbb{E}f(Z_h(T))\big] + \big[\mathbb{E}f(Z_h(T)) - \mu_n\big], \tag{3}$$

we see that choosing $h = O(\epsilon)$, so that the first term on the right is $O(\epsilon)$, and $n = O(\epsilon^{-2})$, 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 [5, 6, 34] 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 multi-level Monte Carlo approach 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 $O(\epsilon^{-2}\log(\epsilon)^2)$ [18]. 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, 25, 28, 30], and the same ideas have been applied to partial differential equations [9, 13]. A key motivation for multi-level Monte Carlo is that optimizing the overall expected value computation is a different, and typically more relevant, goal than optimizing along each path. Computing an expectation using only an exact algorithm (or an algorithm with a very fine time-step) can require a large number of paths and an *extremely* large number of random variables and state updates. In general, the total number of paths cannot be reduced. The computational benefits of multi-level Monte Carlo arise because the number of random variables and state updates needed to approximate the expectation can be drastically reduced by averaging over a very carefully chosen combination of coordinated realizations, many of which are much cheaper to compute than an exact realization.

In this paper we extend the multi-level approach to the continuous time Markov chain setting, and especially the stochastic chemical kinetic setting. The extension involves a non-trivial coupling of the requisite processes that is easy to simulate while providing a very small variance for the estimator. In fact, showing the practical importance of the coupling (found in this paper in both equations (18) and (22)), which was first used in [33] and later in [5] as an analytical tool and subsequently in [1] towards the problem of computing parameter sensitivities, 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 representation (1) 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 approximating expected values of continuous time Markov chains, and especially stochastically modeled chemical reaction networks.

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. We provide such an example in Section 9. We also emphasize that our complexity analysis does not involve asymptotic limits. In particular, we do not consider infinitely large system size, where stochastic effects vanish, or infinitesimally small discretization time-step, where the benefits of an approximate method evaporate.

The outline of the remainder of the paper is as follows. In Section 2, we consider stochastically modeled chemical reaction networks, which is our main application area, discussing how such models can be represented via (1). In Section 3, we introduce an equivalent model to (1) that 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 it plays no role in the actual simulations. In Section 4, we briefly review Euler's method, often called tau-leaping in the chemical kinetic setting. In Section 5, we review the original multi-level Monte Carlo method. In Section 6, we extend multi-level Monte Carlo to the continuous time Markov chain setting in two different ways. In the first, exact algorithms are not used and we are led to an efficient method with an unquantified 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 7, we provide the delayed proofs of the main analytical results of Section 6. In Section 8, we briefly discuss some implementation issues. In Section 9, we provide computational examples demonstrating our main results. Finally, in Section 10 we provide some brief conclusions.

2 The basic stochastic model for chemical reaction networks

In this section we discuss how the basic stochastic model for chemical reaction networks can be represented via (1) for suitable choices of λ_k and ζ_k. A chemical reaction network consists of the interaction of multiple *species*, $\{S_1, \ldots, S_d\}$, through different possible *reactions*. If we denote by $\zeta_k \in \mathbb{R}^d$ the change to the state of the system after each occurrence of the kth reaction, then we have

$$X(t) = X(0) + \sum_k R_k(t)\zeta_k,$$

where $X_i(t)$ gives the number of molecules of S_i at time t, and $R_k(t)$ is the number of times the kth reaction has taken place up until time t. To model R_k, each reaction channel is assumed to have an associated intensity, or propensity, function, $\lambda_k : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$, and for the standard Markov chain model, the number of times that the kth reaction occurs by time t can then be represented by the counting process

$$R_k(t) = Y_k\bigg(\int_0^t \lambda_k(X(s))ds\bigg),$$

where the Y_k are independent unit-rate Poisson processes; see, for example, [32], [15, Chapter 6], or the recent survey [7]. The state of the system then satisfies (1). This 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.

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

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

where ν_{ki} denotes the number of molecules of S_i required for one instance of the reaction. Note that $\lambda_k(x) = 0$ 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 (1) *is*

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

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

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

3 Scaled models

To quantify the relative computational complexity of different methods, it is important that the natural scalings of a model be taken into account. However, we stress that such a change to the representation of the

model does not change the simulation—we simulate the unscaled model but analyze the methods on an appropriately scaled version.

Letting N be some natural parameter of the system, which is usually taken to be the abundance of the most abundant component, we scale the model by setting $X_i^N = N^{-\alpha_i} X_i$, where $\alpha_i \geq 0$ is chosen so that $X_i^N = O(1)$. The general form of such a scaled model is

$$X^N(t) = X^N(0) + \sum_k Y_k \left(N^\gamma \int_0^t N^{c_k} \lambda_k(X^N(s)) ds \right) \zeta_k^N, \tag{5}$$

where γ and c_k are scalars, $|\zeta_k^N| = O(N^{-c_k})$, and both X^N and $\lambda_k(X^N)$ are of order one. Note that we are explicitly allowing for $|\zeta_k^N|$ to be smaller than N^{-c_k}, a point made explicit in and around equation (7). We note that we should write λ_k^N, as the resulting intensity function may depend upon N, though we drop the superscript N for notational convenience. It is now natural to take

$$\overline{N} = N^\gamma \sum_k N^{c_k}$$

as the order of magnitude for the number of computations required to generate a single path using an exact algorithm. We will demonstrate how to arrive at such a scaled model for chemical systems below, however we first discuss the parameter γ.

The parameter γ should be interpreted as being related to the natural time-scale of the model. That is, if $\gamma > 0$ then the shortest timescale in the problem is much smaller than 1, while if $\gamma < 0$ it is much larger. The analysis in this paper is most applicable in the case that $\gamma \leq 0$, for otherwise the error bounds grow quite rapidly. However, and as will be demonstrated in the examples section, the methods developed can still behave very well even when $\gamma > 0$, pointing out that the present analysis does not fully capture the behavior of the methods.

We will show how to derive a model of the form (5) in the case of chemical reaction networks with mass action kinetics. Let $N \gg 1$, where N is the abundance of the most abundant species, or some other large parameter. Suppose we have 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

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

For each species, define the *normalized abundance* by $X_i^N(t) \stackrel{\text{def}}{=} N^{-\alpha_i} X_i(t)$, where $\alpha_i \geq 0$ should be selected so that $X_i^N = O(1)$. Here X_i^N may be the species number ($\alpha_i = 0$), 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 [29], and we recall that ν_k is the source vector of the kth reaction. Our model has therefore become

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

where $\zeta_{ki}^N \stackrel{\text{def}}{=} \zeta_{ki} / N^{\alpha_i}$ (so ζ_k^N is the scaled reaction vector). Define $\gamma \in \mathbb{R}$ via

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

Then, for each k define

$$c_k \stackrel{\text{def}}{=} \beta_k + \nu_k \cdot \alpha - \gamma. \tag{6}$$

With these definitions, our chemical model becomes (5).

Returning to the general setting of (5), for each k we define

$$\rho_k \stackrel{\text{def}}{=} \min\{\alpha_i \,:\, \zeta_{ki}^N \neq 0\}, \tag{7}$$

so that $|\zeta_k^N| \approx N^{-\rho_k}$, and define $\rho \stackrel{\text{def}}{=} \min\{\rho_k\}$. We have 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. Also, 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. Finally, it is worth explicitly noting that the *classical scaling* holds if and only if $c_k \equiv \rho_k \equiv 1$ and $\gamma = 0$ [5, 31].

Remark 1. We emphasise that the models (1) and (5) 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 7 it will be shown how the representation (5) 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.

Example 2
To 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 $\rho_1 = \rho_2 = 1$. 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 $X_1^N + X_2^N \equiv 2$.

Example 3
We provide a deterministic example to further explain the use of the scalings. Consider the ordinary differential equation

$$\dot{x}(t) = \lambda N - \mu x(t),$$

where $\lambda, \mu = O(1)$, $N \gg 1$, and $x_0 = O(N)$. Of course, the solution to this system is

$$x(t) = \frac{\lambda N}{\mu} - \left(\frac{\lambda N}{\mu} - x_0\right)e^{-\mu t}.$$

However, defining $x^N = N^{-1}x$, we see that x^N satisfies

$$\dot{x}^N(t) = \lambda - \mu x^N(t),$$

with $x_0^N = O(1)$. Solving yields

$$x^N(t) = \frac{\lambda}{\mu} - \left(\frac{\lambda}{\mu} - x_0^N\right) e^{-\mu t}.$$

Note, then, that solving for either x or x^N automatically yields the other after scaling. Also note the important property that in the ODE governing x, the driving force, λN, was an extremely large value. However, the forcing function of x^N, which is simply λ, was $O(1)$.

Example 3 points out an important feature: the functions λ_k of (5), together with their derivatives, are much better behaved, in terms of their magnitude, than the intensity functions of the original model (1). Therefore, after possibly redefining the kinetics by multiplication with a cutoff function, see, for example, [5, 6], it is reasonable to assume that each λ_k is, in fact, a globally Lipschitz function of X^N. We formalize this assumption here.

Running assumption: Throughout, we assume that the functions λ_k of (5) are globally Lipschitz.

4 A review of Euler's method in the current setting

We briefly review Euler's method, termed tau-leaping in the chemical kinetic literature [21], as applied to the models (1), and equivalently (5). 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. 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) to denote a sample from the Poisson distribution with parameter x, with all such samples being independent of each other and of all other sources of randomness used.

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

(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 $\Lambda_k = \text{Poisson}(\lambda_k(Z_h(t_n))h)$.

(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. Some concern adaptive step-size selection along a path [11, 22]. Others focus on ensuring non-negative population values [3, 10, 12, 37]. The latter issue is easily dealt with in our context; for example, it is sufficient to return a value to zero if it ever goes negative in the course of a simulation. This is discussed further in subsection 6.2.

Analogously to (1), 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:

$$Z_h(t) = Z_h(0) + \sum_k Y_k \left(\int_0^t \lambda_k(Z_h \circ \eta(s))ds\right) \zeta_k, \tag{8}$$

where the Y_k are as before, and $\eta(s) \stackrel{\text{def}}{=} \left\lfloor \frac{s}{h} \right\rfloor h$. 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 $i \in \{1, \ldots, d\}$ we let $Z^N_{h,i} \stackrel{\text{def}}{=} N^{-\alpha_i} Z_{h,i}$, so the scaled version of (8) is

$$Z^N_h(t) = Z^N_h(0) + \sum_k Y_k \left(N^\gamma \int_0^t N^{c_k} \lambda_k(Z^N_h \circ \eta(s))ds \right) \zeta^N_k, \tag{9}$$

where all other notation is as before. We again stress that the models (8) and (9) are equivalent, with (8) usually giving the counts of each component and (9) providing the normalized abundances.

Remark 2. 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.

5 A review of multi-level Monte Carlo

Given a stochastic process, $X(\cdot)$, let $f : \mathbb{R}^d \to \mathbb{R}$ be a function of the state of the system which gives a measurement of interest. Our task is to approximate $\mathbb{E}f(X(T))$ efficiently. As discussed in Section 1, using the "crude Monte Carlo" estimator (2) 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 $\ell \in \{0, 1, \ldots, L\}$, 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 (3)) is of the desired order of magnitude. We then have

$$\mathbb{E}f(Z_L(T)) = \mathbb{E}[f(Z_0(T))] + \sum_{\ell=1}^{L} \mathbb{E}[f(Z_\ell(T)) - f(Z_{\ell-1}(T))], \tag{10}$$

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

$$\widehat{Q}_0 \stackrel{\text{def}}{=} \frac{1}{n_0} \sum_{i=1}^{n_0} f(Z_{0,[i]}(T)), \quad \text{and} \quad \widehat{Q}_\ell \stackrel{\text{def}}{=} \frac{1}{n_\ell} \sum_{i=1}^{n_\ell} (f(Z_{\ell,[i]}(T)) - f(Z_{\ell-1,[i]}(T))), \tag{11}$$

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, 26] for details on how to do this in the diffusive setting). We then let

$$\widehat{Q} \stackrel{\text{def}}{=} \sum_{\ell=0}^{L} \widehat{Q}_\ell, \tag{12}$$

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 and f is Lipschitz, we may set

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

which yields $\mathsf{Var}(\widehat{Q}) = O(\epsilon^2)$, but with a total computational complexity of $O(\epsilon^{-2}(\log \epsilon)^2)$. We make the following observations.

1. The gains in computational efficiency come about for two reasons. First, a coordinated sequence of simulations are being done, with nested step-sizes, and the simulations with larger step-size are much cheaper than those with very fine step sizes. Second, while we do still require the generation of paths with fine step-sizes, the variance of $f(Z_\ell) - f(Z_{\ell-1})$ will be small, thereby requiring significantly fewer of these expensive paths in the estimation of $\widehat{Q}_\ell$ of (11).

2. For the analysis in [18], 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, even though we are only solving the *weak* approximation problem.

3. The estimator (12) is a biased estimator of $\mathbb{E}f(X(T))$, and the number of levels L was chosen to ensure that the bias is within the desired tolerance.

6 Multi-level Monte Carlo for continuous time Markov chains

We now consider the problem of estimating $\mathbb{E}f(X^N(T))$, where X^N satisfies the general system (5). We again stress that as X^N of (5) is equivalent to the process X of (1), 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 $\overline{N} = N^\gamma \sum_k N^{c_k}$ 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 total computational complexity of $O(\overline{N}\epsilon^{-2})$.

We will now extend the core ideas of multi-level Monte Carlo as described in Section 5 to the continuous time Markov chain setting with Euler tau-leaping, given in (9), as our approximation method. We again fix an integer $M > 0$, and for $\ell \in \{\ell_0, \ldots, L\}$, 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 (9) generated with a step-size of h_ℓ. By [6], for suitable f

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

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

$$\mathbb{E}f(Z_L^N(T)) = \mathbb{E}[f(Z_{\ell_0}^N(T))] + \sum_{\ell=\ell_0+1}^{L} \mathbb{E}[f(Z_\ell^N(T)) - f(Z_{\ell-1}^N(T))]. \tag{13}$$

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 by $\widehat{Q}_\ell$. That is

$$\widehat{Q}_0 \stackrel{\text{def}}{=} \frac{1}{n_0}\sum_{i=1}^{n_0} f(Z_{\ell_0,[i]}^N(T)), \quad \text{and} \quad \widehat{Q}_\ell \stackrel{\text{def}}{=} \frac{1}{n_\ell}\sum_{i=1}^{n_\ell} (f(Z_{\ell,[i]}^N(T)) - f(Z_{\ell-1,[i]}^N(T))), \tag{14}$$

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

$$\widehat{Q} \stackrel{\text{def}}{=} \widehat{Q}_0 + \sum_{\ell=\ell_0+1}^{L} \widehat{Q}_\ell, \tag{15}$$

be the unbiased estimator for $\mathbb{E}[f(Z^N_L(T))]$. 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 now come into focus. First, we must be able to simulate the paths Z^N_ℓ and $Z^N_{\ell-1}$ simultaneously in a manner that is efficient and produces small variances between the paths. Second, we must be able to quantify this variance in order to control the variance of the associated $\widehat{Q}_\ell$ terms of (14). Both requirements demand a good coupling of the processes Z^N_ℓ and $Z^N_{\ell-1}$.

We motivate our choice of coupling by first treating two simpler tasks. First, consider the problem of trying to understand the difference between $Z_1(t)$ and $Z_2(t)$, where Z_1, Z_2 are Poisson processes with rates 13.1 and 13, respectively. A simple approach is to let Y_1 and Y_2 be independent, unit-rate Poisson processes, set

$$Z_1(t) = Y_1(13.1t) \quad \text{and} \quad Z_2(t) = Y_2(13t),$$

and consider $Z_1(t) - Z_2(t)$. Using this representation, these processes are independent and, hence, not coupled. Further, the variance of their difference is the sum of their variances, and so

$$\mathsf{Var}(Z_1(t) - Z_2(t)) = \mathsf{Var}(Z_1(t)) + \mathsf{Var}(Z_2(t)) = 26.1t.$$

Another choice is to let Y_1 and Y_2 be independent unit-rate Poisson processes, and set

$$Z_1(t) = Y_1(13t) + Y_2(0.1t) \quad \text{and} \quad Z_2(t) = Y_1(13t),$$

where we have used the additivity property of Poisson processes. The important point to note is that both Z_1 and Z_2 are using the process $Y_1(13t)$ to generate simultaneous jumps. The process Z_1 then uses the auxiliary process $Y_2(0.1t)$ to jump the extra times that Z_2 does not. The processes Z_1 and Z_2 will jump together the vast majority of times, and hence are tightly coupled; by construction $\mathsf{Var}(Z_1(t) - Z_2(t)) = \mathsf{Var}(Y_2(0.1t)) = 0.1t$. More generally, if Z_1 and Z_2 are instead inhomogeneous Poisson processes with intensities $f(t)$ and $g(t)$, respectively, then we could let Y_1, Y_2, and Y_3 be independent, unit-rate Poisson processes and define

$$\begin{aligned}
Z_1(t) &= Y_1\left(\int_0^t f(s) \wedge g(s) ds\right) + Y_2\left(\int_0^t f(s) - (f(s) \wedge g(s))\, ds\right), \\
Z_2(t) &= Y_1\left(\int_0^t f(s) \wedge g(s) ds\right) + Y_3\left(\int_0^t g(s) - (f(s) \wedge g(s))\, ds\right),
\end{aligned}$$

where we are using that, for example,

$$Y_1\left(\int_0^t f(s) \wedge g(s) ds\right) + Y_2\left(\int_0^t f(s) - (f(s) \wedge g(s))\, ds\right) \stackrel{\mathcal{D}}{=} Y\left(\int_0^t f(s) ds\right),$$

where Y is a unit rate Poisson process and we recall that $a \wedge b \stackrel{\text{def}}{=} \min\{a, b\}$.

We now return to the main problem of coupling the processes Z^N_ℓ and $Z^N_{\ell-1}$, each satisfying (9) with their respective step-sizes. We couple the processes Z^N_ℓ and $Z^N_{\ell-1}$ in the following manner, which is similar

to a coupling originally used in [33], and later in [5], as an analytical tool, and subsequently in [1] towards the problem of computing parameter sensitivities:

$$\begin{aligned} Z_\ell^N(t) &= Z_\ell^N(0) + \sum_k Y_{k,1}\left(N^\gamma N^{c_k}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s))\wedge\lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s))ds\right)\zeta_k^N \\ &\quad + \sum_k Y_{k,2}\left(N^\gamma N^{c_k}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s)) - \lambda_k(Z_\ell^N\circ\eta_\ell(s))\wedge\lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s))ds\right)\zeta_k^N, \end{aligned} \tag{16}$$

$$\begin{aligned} Z_{\ell-1}^N(t) &= Z_{\ell-1}^N(0) + \sum_k Y_{k,1}\left(N^\gamma N^{c_k}\int_0^t \lambda_k(Z_\ell^N\circ\eta_\ell(s))\wedge\lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s))ds\right)\zeta_k^N \\ &\quad + \sum_k Y_{k,3}\left(N^\gamma N^{c_k}\int_0^t \lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s)) - \lambda_k(Z_\ell^N\circ\eta_\ell(s))\wedge\lambda_k(Z_{\ell-1}^N\circ\eta_{\ell-1}(s))ds\right)\zeta_k^N, \end{aligned} \tag{17}$$

where the $Y_{k,i},\ i\in\{1,2,3\}$, are independent, unit-rate Poisson processes, and for each ℓ, we define $\eta_\ell(s)\stackrel{\text{def}}{=}\lfloor s/h_\ell\rfloor h_\ell$. Note that we essentially used the coupling of the simpler examples above (pertaining to Z_1 and Z_2) for each of the reaction channels.

The paths of the coupled processes can easily be computed simultaneously and the distributions of the marginal processes are the same as the usual scaled Euler approximate paths (9) with similar step-sizes. More precisely, the system (16)–(17) is the scaled version of, and is hence equivalent to, the system

$$\begin{aligned} Z_\ell(t) =& Z_\ell(0) + \sum_k Y_{k,1}\left(\int_0^t \lambda_k(Z_\ell\circ\eta_\ell(s))\wedge\lambda_k(Z_{\ell-1}\circ\eta_{\ell-1}(s))ds\right)\zeta_k \\ &+ \sum_k Y_{k,2}\left(\int_0^t \lambda_k(Z_\ell\circ\eta_\ell(s)) - \lambda_k(Z_\ell\circ\eta_\ell(s))\wedge\lambda_k(Z_{\ell-1}\circ\eta_{\ell-1}(s))ds\right)\zeta_k, \\ Z_{\ell-1}(t) =& Z_{\ell-1}(0) + \sum_k Y_{k,1}\left(\int_0^t \lambda_k(Z_\ell\circ\eta_\ell(s))\wedge\lambda_k(Z_{\ell-1}\circ\eta_{\ell-1}(s))ds\right)\zeta_k \\ &+ \sum_k Y_{k,3}\left(\int_0^t \lambda_k(Z_{\ell-1}\circ\eta_{\ell-1}(s)) - \lambda_k(Z_\ell\circ\eta_\ell(s))\wedge\lambda_k(Z_{\ell-1}\circ\eta_{\ell-1}(s))ds\right)\zeta_k, \end{aligned} \tag{18}$$

where now the marginal processes are distributionally equivalent to the approximate processes (8) with similar step-sizes, and all notation is as before. The natural algorithm to simulate the representation (18) (and hence (16)–(17)) to a time $T>0$ is the following.

Algorithm 2 (Simulation of the representation (18))**.** Fix an integer $M\geq 2$. 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 $t_n\geq T$:

(i) For $j=0,\ldots,M-1$,

(a) Set

- $A_{k,1}=\lambda_k(Z_\ell(t_n+j\times h_\ell))\wedge\lambda_k(Z_{\ell-1}(t_n))$.
- $A_{k,2}=\lambda_k(Z_\ell(t_n+j\times h_\ell))-A_{k,1}$.
- $A_{k,3}=\lambda_k(Z_{\ell-1}(t_n))-A_{k,1}$.

(b) For each k, let

- $\Lambda_{k,1}=\text{Poisson}(A_{k,1}h_\ell)$.

- $\Lambda_{k,2} = \text{Poisson}(A_{k,2}h_\ell)$.
- $\Lambda_{k,3} = \text{Poisson}(A_{k,3}h_\ell)$.

(c) Set

- $Z_\ell(t_n + (j+1) \times h_\ell) = Z_\ell(t_n + j \times h_\ell) + \sum_k (\Lambda_{k,1} + \Lambda_{k,2})\zeta_k$.
- $Z_{\ell-1}(t_n + (j+1) \times h_\ell) = Z_{\ell-1}(t_n + j \times h_\ell) + \sum_k (\Lambda_{k,1} + \Lambda_{k,3})\zeta_k$.

(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 (18), the scaled version of the process generated via Algorithm 2 satisfies (16)–(17). 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 $j = 0, \ldots, M-1$. Third, at most one of A_2, A_3 will be non-zero during each step, with both being zero whenever $\lambda_k(Z_\ell(t_n)) = \lambda_k(Z_{\ell-1}(t_n))$. 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 (18) and/or (16)–(17), is no harder to simulate, from an implementation standpoint, than 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 $\Lambda_{k,1}$,[1] which is the same amount of work as would be needed for the generation of a *single* path of Euler tau-leaping.

In Section 7 we will prove the following theorem, which is one of our main analytical results.

Theorem 1. *Suppose* $(Z_\ell^N, Z_{\ell-1}^N)$ *satisfy* (16) *and* (17) *with* $Z_\ell^N(0) = Z_{\ell-1}^N(0)$. *Then, there exist functions* C_1, C_2, *that do not depend on* h_ℓ, *such that*

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

In particular, for $\gamma \le 0$ *the values* $C_1(N^\gamma T)$ *and* $C_2(N^\gamma T)$ *may be bounded above uniformly in* N.

Remark 3. The specific forms of $C_1(N^\gamma T)$ and $C_2(N^\gamma T)$ for Theorem 1 and Theorem 2 below are given in Section 7. However, we note here that if $\gamma > 0$, the factors $C_1(N^\gamma T)$ and $C_2(N^\gamma T)$ could be huge, leading to upper bounds in Theorem 1 and Theorem 2 of no practical use. So we henceforth assume that $\gamma \le 0$ and thus regard $C_1(N^\gamma T)$ and $C_2(N^\gamma T)$ as constants independent of N. We note that the classical chemical kinetics scaling, with $\gamma = 0$, satisfies this assumption. However, good performance is observed in Section 9 with $\gamma > 0$, suggesting that further analysis may extend the range of validity for this method.

Note that Theorem 1 together with f Lipschitz gives us the estimate

$$\begin{aligned} \left|\mathsf{Var}\left(f(Z_\ell^N(t))\right) - \mathsf{Var}\left(f(Z_{\ell-1}^N(t))\right)\right| &\le \mathbb{E}\left|f(Z_\ell^N(t)) - f(Z_{\ell-1}^N(t))\right|^2 \\ &\le C\mathbb{E}|Z_\ell^N(t) - Z_{\ell-1}^N(t)|^2 \\ &\le C\left[C_1(N^\gamma T)N^{-\rho}(N^\gamma h_\ell) + C_2(N^\gamma T)(N^\gamma h_\ell)^2\right], \quad (19) \end{aligned}$$

which we will use to control the variance of $\widehat{Q}_\ell$ in (14).

[1] The cost of generating a Poisson random variable generally increases with the size of the mean

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. We define X^N and Z_ℓ^N via

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

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

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 (5) and (9). 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 (1) and (8).

The natural algorithm to simulate (22), and hence (20)–(21), is the next reaction method [2, 16], where the system is viewed as having dimension $2d$ 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 [2] 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 $U \sim \text{rand}(0,1)$, 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 $k \in \{1, \ldots, R\}$ and $i \in \{1, 2, 3\}$, set $P_{k,i} = \ln(1/r_{k,i})$, where $r_{k,i}$ is rand$(0,1)$, and $T_{k,i} = 0$.

(i) For each k, set

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

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

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

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

(iv) If $t + \Delta \geq T_{\text{tau}}$,

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

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

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

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

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

(v) Else,

(a) Update. For $\{k, i\} = \mu$, where μ is from (iii),

- If $i = 1$, set $X(t + \Delta) = X(t) + \zeta_k$ and $Z_\ell(t + \Delta) = Z_\ell(t) + \zeta_k$.
- If $i = 2$, set $X(t + \Delta) = X(t) + \zeta_k$.
- If $i = 3$, set $Z_\ell(t + \Delta) = Z_\ell(t) + \zeta_k$

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

(c) Set $P_\mu = P_\mu + \ln(1/r)$, where r is rand$(0, 1)$, and μ is from (iii).

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

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

The following theorem, which should be compared with Theorem 1, is proven in Section 7 and is our second main analytical result.

Theorem 2. *Suppose* (X^N, Z_ℓ^N) *satisfy* (20) *and* (21) *with* $X^N(0) = Z_\ell^N(0)$. *Then, there exist functions* C_1, C_2, *that do not depend on* h_ℓ, *such that*

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

Moreover, for $\gamma \leq 0$ *the values* $C_1(N^\gamma T)$ *and* $C_2(N^\gamma T)$ *may be bounded above uniformly in* N.

We are now in a position to develop MLMC in the stochastic chemical kinetic setting. Recall our assumption that $\gamma \leq 0$, so C_1 and C_2 in Threorems 1 and 2 are bounded. We return to the $\widehat{Q}_\ell$ terms in (14). Supposing that the test function f is uniformly Lipschitz in our domain of interest (note that this is automatic for any reasonable f in the case when mass is conserved), then for $\ell > \ell_0$, we know from (19) that

$$\mathsf{Var}(\widehat{Q}_\ell) \leq C\frac{1}{n_\ell}\left[C_1(N^\gamma T)N^{-\rho}(N^\gamma h_\ell) + C_2(N^\gamma T)(N^\gamma h_\ell)^2\right].$$

Note that if $N^{-\rho} \leq h_\ell$, the leading order of the error is the h_ℓ^2 term. As a heuristic argument for this behavior, note that if $N^{-\rho} \leq h_\ell$ and N is large while h_ℓ is small, 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 (15) satisfies

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

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

$$A(h) \stackrel{\text{def}}{=} N^{-\rho}(N^\gamma h) + (N^\gamma h)^2. \tag{23}$$

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

$$\begin{aligned}n_0 h_{\ell_0}^{-1} + \sum_{\ell=\ell_0+1}^{L} n_\ell h_\ell^{-1} &= \epsilon^{-2}h_{\ell_0}^{-1} + \sum_{\ell=\ell_0+1}^{L} \epsilon^{-2}(L-\ell_0)A(h_\ell)h_\ell^{-1} \\ &= \epsilon^{-2}\left(h_{\ell_0}^{-1} + (L-\ell_0)\sum_{\ell=\ell_0+1}^{L}(N^{-\rho}N^\gamma + h_\ell N^{2\gamma})\right) \\ &\leq \epsilon^{-2}\left(h_{\ell_0}^{-1} + \ln(\epsilon)^2 N^{-\rho}N^\gamma + \ln(\epsilon^{-1})\frac{1}{M-1}h_{\ell_0}N^{2\gamma}\right),\end{aligned} \tag{24}$$

where we used that

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

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 8, the n_ℓ can be chosen algorithmically, by optimizing for a given problem.

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

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

and note that

$$\widehat{Q} \stackrel{\text{def}}{=} \widehat{Q}_E + \sum_{\ell=\ell_0+1}^{L}\widehat{Q}_\ell + \widehat{Q}_0 \tag{26}$$

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

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

where $K_1(N^\gamma T)$ and $K_2(N^\gamma T)$ are independent of h_ℓ, and, under our assumption that $\gamma \le 0$, can be bounded uniformly in N. It follows that the choice

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

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

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

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

6.2 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 8 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 [3, 10, 12, 37] 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 so long as 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. Of course, this cost could be quite high as negativity of population numbers can lead to instability if they have not defined the intensity functions outside the positive orthant in a reasonable manner. However, in Section 8 we discuss how intelligent implementation of the method can greatly reduce this cost by ensuring that the approximate paths remain stable with high probability.

Third, inspecting (24) and (28) shows that the unbiased MLMC estimator (26) has an additional term of $O(\overline{N}A(h_L)\epsilon^{-2})$ in its computational complexity bound, as compared with the biased MLMC estimator (15). The authors feel that $\overline{N}A(h_L)$ would have to be quite substantial to warrant not using the unbiased version.

Fourth, note that we always have the following:

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

Thus, under our standing assumption $\gamma \leq 0$, 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, consider the case when the system satisfies the classical scaling, for which $\rho = 1$, $\gamma = 0$ and $c_k \equiv 1$. In this case, $\overline{N} = N$ 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, $A(h_L) = O(h_L^2)$ 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 an example found in Section 9, 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 $\gamma \leq 0$ and so the analysis will hold.

The conclusion of this analysis, backed up by the examples in Section 9, is that MLMC methods, with processes coupled via the representations (18) and (22), and the unbiased MLMC in particular, produce substantial gains in computational efficiency and could become standard algorithms in the sciences. Further attention, however, needs to be given to the case $\gamma > 0$, and this will be a focus for future work.

7 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 functions* C_1, C_2, *that do not depend on* h_ℓ, *such that*

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

Moreover, for $\gamma \leq 0$ *the values* $C_1(N^\gamma T)$ *and* $C_2(N^\gamma T)$ *may be bounded above uniformly in* N.

We start with the following lemma.

Lemma 3. 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, independent of N, γ, and T, such that for $t \geq 0$

$$\mathbb{E}|X^N(t) - Z_\ell^N(t)| \leq c_1 \left(e^{c_2 N^\gamma t} - 1\right)(N^\gamma h_\ell).$$

Proof. Note that

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

where $C > 0$ is some constant and we used that the λ_k are assumed to be Lipschitz. Adding and subtracting the obvious terms yields

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

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

$$\mathbb{E}|Z_\ell^N(s) - Z_\ell^N \circ \eta_\ell(s)| \leq \sum_k |\zeta_k^N| N^\gamma N^{c_k} \mathbb{E} \int_{\eta_\ell(s)}^s \lambda_k(Z_\ell^N(\eta_\ell(r))dr \leq \tilde{C} N^\gamma h_\ell, \tag{31}$$

where $\tilde{C} > 0$ is a constant, and we recall that λ is $O(1)$ in our region of interest. Collecting the above yields

$$\mathbb{E}|X^N(t) - Z_\ell^N(t)| \leq \widehat{C}_1 N^{2\gamma} t h_\ell + \widehat{C}_2 N^\gamma \int_0^t \mathbb{E}|X^N(s) - Z_\ell^N(s)|ds,$$

for some positive constants $\widehat{C}_1, \widehat{C}_2$ that are independent of N, γ, and T. The result now follows from Gronwall's inequality. □

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 be a sharp estimate. A future research direction will be classifying those networks for which this upper bound can be decreased substantially.

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

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

is a martingale, and

$$F^N(x) = \sum_k N^\gamma N^{c_k} \lambda_k(x) \zeta_k^N.$$

Note that based upon our assumptions, we have that

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

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

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

Thus,

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

and, in particular,

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

We note that

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

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

First, by (33) and the Burkholder-Davis-Gundy inequality,

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

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 for $t \le T$

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

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,

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

The expected value in the first term on the right hand side of (37) can be bounded via

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

We have that

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

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

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

Combining (40) with (37) then yields

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

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

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

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* (16) *and* (17) *with* $Z_\ell^N(0) = Z_{\ell-1}^N(0)$. *Then, there exist functions* C_1, C_2, *that do not depend on* h_ℓ, *such that*

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

In particular, for $\gamma \leq 0$ *the values* $C_1(N^\gamma T)$ *and* $C_2(N^\gamma T)$ *may be bounded above uniformly in* N.

Proof. (of Theorem 1.) A direct proof can be written along the lines of that for Theorem 2. 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 2 combined with the triangle inequality. □

8 Implementation issues

The analysis in Sections 6 and 7 specified an order of magnitude for the number of paths, n_ℓ, to be used at each level so as to attain the desired accuracy. This was needed to prove that the computational complexity 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, which, of course, will depend on the function f, and the model itself.

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

$$CPU_\ell \approx \frac{K_\ell}{V_\ell},$$

for some K_ℓ as both CPU_ℓ and $1/V_\ell$ scale linearly with n_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 quantity 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

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

subject to the constraint (42). This will give us target variances, V_ℓ, for each level, and an estimate on the time needed until the computation is completed. 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. Note that this is similar to the strategy proposed in [18].

We make the important observation that with such an optimizing pre-computation, we can choose both L and ℓ_0, that is the finest and crudest levels, before attempting the full calculation. This reduces the probability of the approximate processes becoming negative, or non-physical in other ways, during the course of a simulation. This, in turn, helps keep the approximate processes stable, which leads to increased efficiency. Further, if we find during such a pre-computation that no choice of levels and number of paths will be significantly faster than using an exact algorithm combined with crude Monte Carlo, then we should simply revert to solely using an exact algorithm. We may conclude, therefore, that *the developed method will never, for any example, be appreciably slower than using an exact algorithm with crude Monte Carlo*. As will be demonstrated in the next section, however, the method will often times, even in cases not yet predicted by the analysis, be *significantly* faster.

Finally, we note that in each of the examples in Section 9, and each method tested, we use Matlab's built in Poisson random number generator. Further, we produce the necessary approximate paths in batches ranging from the 100s to 10s of thousands so as to reduce the number of separate calls to the Poisson random number generator.

9 Examples

We present three examples to demonstrate the performance of the proposed method.

Example. We begin by considering a model of gene transcription and translation also used in [6]:

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

Here, a single gene is being transcribed into mRNA, which is then being translated into proteins, and finally the proteins produce stable dimers. The final two reactions represent degradation of mRNA and proteins, respectively. We suppose the system starts with one gene and no other molecules, so $X(0) = (1, 0, 0)$ where X_1, X_2, X_3 give the molecular counts of the mRNA, proteins, and dimers, respectively. Finally, we suppose that we want to estimate the expected number of dimers at time $T = 1$ to an accuracy of ± 1 with 95% confidence. Thus, we want the variance of our estimator to be smaller than $(1/1.96)^2 \approx .2603$. We will also estimate the second moment of the number of dimers, which could be used in conjunction with the mean to estimate the variance. For comparison purposes, we will use each method discussed in this paper to approximate the mean, and will use an exact method combined with crude Monte Carlo and the unbiased MLMC method to approximate the second moment.

While $\epsilon = 1$ for the unscaled version of this problem, the simulation of just a few paths of the system shows that there will be approximately 23 mRNA molecules, 3,000 proteins, and 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

Mean	Variance	$\mathbb{E}X_3^2(1)$	# paths	CPU Time	# updates
3714.2 $\pm$ 1.0	$\approx$ 1,232,418	1.5035 $\times 10^7 \pm 8 \times 10^3$	4,740,000	1.49 $\times 10^5$ CPU S	8.27 $\times 10^{10}$

Table 1: Performance of an exact algorithm with crude Monte Carlo. The mean number of dimers at time 1 is reported with 95% a confidence interval. The approximated variance of the number of dimers is provided for completeness. An estimate of the second moment is also provided with a 95% confidence interval.

Step-size	Mean	# paths	CPU Time	# updates
$h = 3^{-7}$	3,712.3 $\pm$ 1.0	4,750,000	13,374.6 S	6.2×10^{10}
$h = 3^{-6}$	3,707.5 $\pm$ 1.0	4,750,000	6,207.9 S	2.1×10^{10}
$h = 3^{-5}$	3,693.4 $\pm$ 1.0	4,700,000	2,803.9 S	6.9×10^9
$h = 3^{-4}$	3,655.2 $\pm$ 1.0	4,650,000	1,219.0 S	2.6×10^9

Table 2: Performance of Euler tau-leaping with crude Monte Carlo for the computation of the first moment of X_3, the number of dimers. The bias of the method is apparent.

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.

We also note that with the rough orders of magnitude computed above for the different molecular counts at time $T = 1$, we have $N \approx$ 3,500, $\alpha_1 \approx .38$, and $\alpha_2 = \alpha_3 \approx 1$. Therefore, we have that $N^\gamma \approx 23,000/3,000 = 7.6 \implies \gamma \approx 0.2485$ for this problem (where we chose the "stiffest" reaction for this calculation, which is that of $M \to M + P$). However, we note that the parameter γ changes throughout the simulation and is quite a bit higher near $t \approx 0$.

Implementing the modified next reaction method, which produces exact sample paths [2], on our machine[2] (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 Table 1 where "# updates" refers to the total number, over all paths, of steps, and is used as a crude quantification for the computational complexity of the different methods under consideration.

Next, we solved the problem using Euler tau-leaping with various step-sizes, combined with a crude Monte Carlo estimator. The results of those simulations are detailed in Table 2. Note that the bias of the approximate algorithm has become apparent.[3] We then implemented the biased version of MLMC with various step-sizes. The results of those simulations are detailed in Table 3, where the approximations and CPU times should be compared with those of Euler tau-leaping. The CPU times stated include the time needed to solve the embedded optimization problem discussed in Section 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 56 fold, with straight tau-leaping taking 17.1 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 Table 4. As in the biased MLMC case, the CPU times stated include the time

[2]We used an Apple machine with 8GB Ram and an i7 chip.

[3]This data also appears in [6].

Step-size parameters	Mean	CPU Time	# updates
$M = 3$, $L = 7$	3,712.6 $\pm$ 1.0	781.8 S	1.1 $\times 10^9$
$M = 3$, $L = 6$	3,708.5 $\pm$ 1.0	623.9 S	7.9 $\times 10^8$
$M = 3$, $L = 5$	3,694.5 $\pm$ 1.0	546.9 S	6.6 $\times 10^8$

Table 3: Performance of biased MLMC with $M = 3$, $\ell_0 = 2$, and L ranging from 7 to 5. The reported times include the time needed for the pre-computations used to choose the number of paths per level as discussed in Section 8.

Step-size parameters	Mean	CPU Time	Var. of estimator	# updates
$M = 3$, $L = 6$	3,713.9 $\pm$ 1.0	1,063.3 S	0.2535	1.1 $\times 10^9$
$M = 3$, $L = 5$	3,714.7 $\pm$ 1.0	1,114.9 S	0.2565	9.4 $\times 10^8$
$M = 3$, $L = 4$	3,714.2 $\pm$ 1.0	1,656.6 S	0.2595	1.0 $\times 10^9$
$M = 4$, $L = 4$	3714.2 $\pm$ 1.0	1,334.8 S	0.2580	1.1 $\times 10^9$
$M = 4$, $L = 5$	3,713.8 $\pm$ 1.0	1,014.9 S	0.2561	1.1 $\times 10^9$

Table 4: Performance of unbiased MLMC with $\ell_0 = 2$, and M and L detailed above. The reported times include the time needed for the pre-computations used to choose the number of paths per level as discussed in Section 8.

needed to solve the embedded optimization problem discussed in Section 8. We see that the unbiased MLMC estimator behaves as the analysis predicts: there is no bias for any choice of M or L, and the required CPU time are analogous to Euler's method with a course time-step. Further, the exact algorithm with crude Monte Carlo, by far the most commonly used method in the literature, demanded approximately 80 times more updates and 140 times more CPU time than our unbiased MLMC estimator, with the precise speedups depending upon the choice of M and L.

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 $M = 3$, $L = 5$, and $\ell_0 = 2$, we provide in Table 5 the relevant data for the different levels. As already stated in Table 4, the total time with the optimization problem was 1,114.9 CPU S, more than the total CPU time reported in Table 5, which does not include the time needed to solve the optimization problem. Note that most of the CPU time was taken up at the coarsest level, as is common with MLMC methods and predicted by the analysis. Also, while the exact algorithm with crude Monte Carlo demanded the generation of almost five million exact sample paths, we needed only 3,900 such paths at our finest level. This difference is the main reason for the dramatic reduction in CPU time. Of course, we needed more than eight 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 generally being allowed to have a higher variance.

In Table 6 we provide the data pertaining to the estimate of the second moment using the unbiased version of MLMC with $M = 3$ and $L = 5$ that appears in the second row of Table 4. The 95% confidence interval for the second moment is 1.5031 $\times 10^7 \pm 9 \times 10^3$, which should be compared with the confidence interval generated using an exact method.

Example. We turn now to a simple example that allows us to study how the behavior of the developed

Level	# paths	Mean	Var. estimator	CPU Time	# updates
$(X, Z_{3^{-5}})$	3,900	20.1	0.0658	279.6 S	6.8 $\times 10^7$
$(Z_{3^{-5}}, Z_{3^{-4}})$	30,000	39.2	0.0217	49.0 S	8.8 $\times 10^7$
$(Z_{3^{-4}}, Z_{3^{-3}})$	150,000	117.6	0.0179	71.7 S	1.5 $\times 10^8$
$(Z_{3^{-3}}, Z_{3^{-2}})$	510,000	350.4	0.0319	112.3 S	1.7 $\times 10^8$
Euler, $h = 3^{-2}$	8,630,000	3,187.4	0.1192	518.4 S	4.7 $\times 10^8$
Totals	N.A.	3,714.7	0.2565	1031.0 S	9.5 $\times 10^8$

Table 5: Details of the different levels for the implementation of the unbiased MLMC method with $M = 3$, $L = 5$, and $\ell_0 = 2$. 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	Estimate	Var. of estimator	95% Confidence inteval
$(X, Z_{3^{-5}})$	157,000	5.8 $\times 10^6$	N.A.
$(Z_{3^{-5}}, Z_{3^{-4}})$	303,000	2.0 $\times 10^6$	N.A.
$(Z_{3^{-4}}, Z_{3^{-3}})$	894,000	2.2 $\times 10^6$	N.A.
$(Z_{3^{-3}}, Z_{3^{-2}})$	2.4888 $\times 10^6$	4.2 $\times 10^6$	N.A.
Euler, $h = 3^{-2}$	1.1188 $\times 10^7$	5.7 $\times 10^6$	N.A.
Totals	1.5031 $\times 10^7$	2.0 $\times 10^7$	1.5031 $\times 10^7 \pm 9 \times 10^3$

Table 6: Details of the different levels for the implementation of the unbiased MLMC method with $M = 3$, $L = 5$, and $\ell_0 = 2$ for the approximation of the second moment.

methods depends upon the parameter γ. Consider the family of models indexed by θ,

$$A \underset{\theta}{\overset{\theta}{\rightleftarrows}} B,$$

with,

$$X_A(0) = X_B(0) = \lfloor 1{,}000\,\theta^{-1} \rfloor,$$

where $\lfloor x \rfloor$ is the greatest integer less than or equal to x. The stochastic equation governing X_A, giving the number of A molecules, is

$$X_A(t) = X_A(0) + Y_1\left(\int_0^t \theta(2,000\theta^{-1} - X_A(s))ds\right) - Y_2\left(\int_0^t \theta X_A(s)ds\right),$$

with the Euler approximation given by

$$Z_A(t) = Z_A(0) + Y_1\left(\int_0^t \theta(2,000\theta^{-1} - Z_A \circ \eta(s))ds\right) - Y_2\left(\int_0^t \theta Z_A \circ \eta(s)ds\right).$$

Letting $N = X_A(0)$, we see that $\theta = N^\gamma$, implying

$$\gamma = \ln(\theta)/\ln(N).$$

Note, in particular, that $\gamma > 0 \iff \theta > 1$, with γ being a strictly increasing function of θ. Therefore, we may test the dependence of the behavior of the MLMC method on this model by varying the single

θ	Method	Estimate of $X_A(1)$	CPU time	# Paths	Speedup
.1	Crude MC	9,999.97 $\pm$ 0.20	1,476.9 S	164,800	N.A.
.1	MLMC: $M = 3$, $L = 4$, $\ell_0 = 0$	10,000.07 $\pm$ 0.19	6.5 S	N.A.	227.2
.5	Crude MC	1,999.90 $\pm$ 0.16	1,110.9 S	124,100	N.A.
.5	MLMC: $M = 3$, $L = 4$, $\ell_0 = 1$	2,000.10 $\pm$ 0.16	15 S	N.A.	74.1
1	Crude MC	999.96 $\pm$ 0.11	1,464.4 S	163,800	N.A.
1	MLMC: $M = 3$, $L = 5$, $\ell_0 = 2$	999.99 $\pm$ 0.11	28 S	N.A.	52.3
2	Crude MC	500.01 $\pm$ 0.11	739.9 S	82,700	N.A.
2	MLMC: $M = 6$, $L = 6$, $\ell_0 = 3$	499.96 $\pm$ 0.11	21	N.A.	35.2
10	Crude MC	99.983 $\pm$ 0.044	900.2 S	100,600	N.A.
10	MLMC: $M = 3$, $L = 6$, $\ell_0 = 5$	99.965 $\pm$ 0.044	65 S	N.A.	13.8
25	Crude MC	40.012 $\pm$ 0.028	898.0 S	100,500	N.A.
25	MLMC: $M = 3$, $L = 6$, $\ell_0 = 6$	39.996 $\pm$ 0.028	98 S	N.A.	9.2
50	Crude MC	20.008 $\pm$ 0.0139	1,789.0 S	200,200	N.A.
50	MLMC: $M = 3$, $L = 7$, $\ell_0 = 7$	20.005 $\pm$ 0.0138	360 S	N.A.	5.0
100	Crude MC	10.002 $\pm$ 0.0139	892.6 S	100,100	N.A.
100	MLMC: $M = 3$, $L = 7$, $\ell_0 = 7$	9.988 $\pm$ 0.0138	250 S	N.A.	3.6
200	Crude MC	4.9996 $\pm$ 0.0088	1,120.3 S	125,400	N.A.
200	MLMC: $M = 3$, $L = 7$, $\ell_0 = 7$	4.9958 $\pm$ 0.0087	486 S	N.A.	2.3
500	Crude MC	2.0029 $\pm$ 0.0044	1,781.6 S	199,400	N.A.
500	MLMC: $M = 3$, $L = 7$, $\ell_0 = 7$	1.9953 $\pm$ 0.0044	1,625.9 S	N.A.	1.1
1,000	Crude MC	1.0038 $\pm$ 0.0043	897.2	100,200	N.A.
1,000	MLMC: $M = 3$, $L = 7$, $\ell_0 = 7$	1.0015 $\pm$ 0.0044	1,412.3 S	N.A.	0.64

Table 7: Approximation of $X_A(1)$ with 95% confidence intervals. Note that the speedup factor decreases as θ increases, with MLMC becoming less efficient than an exact method when $\theta = 1,000$.

parameter θ. We will let θ range from 0.1 to $1,000$ and for each θ use both an exact method with crude Monte Carlo and the unbiased MLMC method developed here to estimate the mean number of A molecules at time 1. We choose different values for our tolerance parameter, ϵ, for different values of θ. The results of these computations can be found in Table 7.

The analysis of this paper predicts that as θ increases, MLMC should progressively lose its computational advantage over an exact algorithm, and this is borne out in the data provided in Table 7. Note, however, that the unbiased version of MLMC remains significantly more efficient than an exact algorithm until $\theta = 1,000$, in which case $X_A(0) = 1$. Having $\theta =$ 1,000 is arguably the *worst* case scenario for an approximate algorithm such as tau-leaping, and the coupling method performs slightly worse than using an exact method with crude Monte Carlo. As discussed in Section 8, we would normally in this case simply use the exact method alone. However, we report the MLMC data for the sake of comparison. Further, it is extremely encouraging that there were still large gains in efficiency even when $\theta \in \{25, 50, 100\}$, something not predicted by the current analysis. Interestingly, the speedup factor of MLMC over an exact method appears, for this example, to be a function of θ. Specifically, as demonstrated by the log-log plot in Figure 1, we observed the relation

$$\text{Speedup factor} \approx 54.6\,\theta^{-0.62}.$$

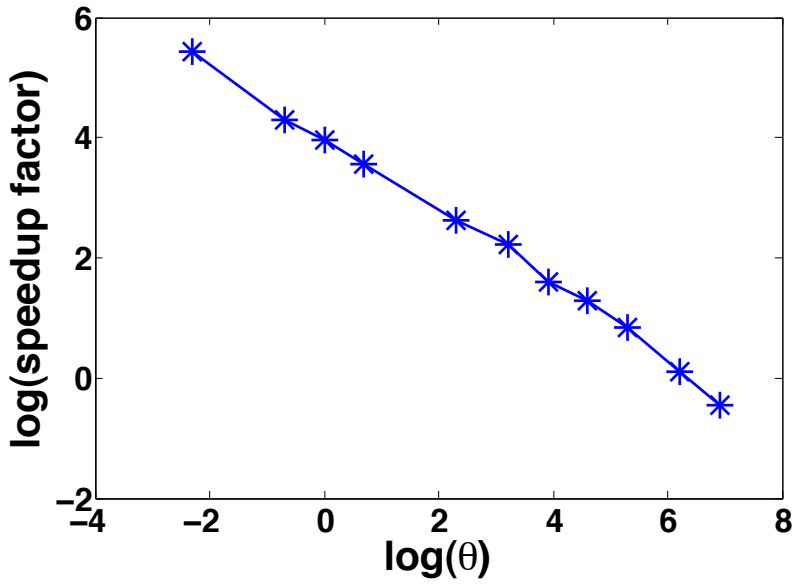

Figure 1: Log-log plot for the speedup factor of unbiased MLMC over an exact method. The best fit line, not shown, is $4 - 0.62x$.

Example. We finish with a model of viral kinetics first developed in [36] by Yin et al., and subsequently studied by Haseltine and Rawlings in [23], Ball et al., in [8], and E et al., in [14]. One reason the interest in this model has been so high from the biological, engineering, and mathematical communities is that it exemplifies a feature of many stochastic models arising in the biosciences: a separation of time scales. We will use this model to demonstrate that one of the main ideas of this paper, the coupling, is not restricted to the use of approximate methods defined using time-discretizations.

The model includes four time-varying "species": the viral genome (G), the viral structural protein (S), the viral template (T), and the secreted virus itself (V). We denote these as species 1, 2, 3, and 4, respectively, and let $X_i(t)$ denote the number of molecules of species i in the system at time t. The model involves six reactions,

$$
\begin{aligned}
R_1: &\quad T \xrightarrow{\kappa_1} T + G, && \kappa_1 = 1,\\
R_2: &\quad G \xrightarrow{\kappa_2} T, && \kappa_2 = 0.025,\\
R_3: &\quad T \xrightarrow{\kappa_3} T + S, && \kappa_3 = 1000,\\
R_4: &\quad T \xrightarrow{\kappa_4} \emptyset, && \kappa_4 = 0.25,\\
R_5: &\quad S \xrightarrow{\kappa_5} \emptyset, && \kappa_5 = 2,\\
R_6: &\quad G + S \xrightarrow{\kappa_6} V, && \kappa_6 = 7.5 \times 10^{-6},
\end{aligned}
$$

where the units of time are in days. The stochastic equations for this model are

$$\begin{aligned}
X_1(t) &= X_1(0) + Y_1\left(\int_0^t X_3(s)ds\right) - Y_2\left(0.025\int_0^t X_1(s)ds\right) \\
&\quad - Y_6\left(7.5\times 10^{-6}\int_0^t X_1(s)X_2(s)ds\right) \\
X_2(t) &= X_2(0) + Y_3\left(1000\int_0^t X_3(s)ds\right) - Y_5\left(2\int_0^t X_2(s)ds\right) \\
&\quad - Y_6\left(7.5\times 10^{-6}\int_0^t X_1(s)X_2(s)ds\right) \\
X_3(t) &= X_3(0) + Y_2\left(0.025\int_0^t X_1(s)ds\right) - Y_4\left(0.25\int_0^t X_3(s)ds\right) \\
X_4(t) &= X_4(0) + Y_6\left(7.5\times 10^{-6}\int_0^t X_1(s)X_2(s)ds\right).
\end{aligned}$$

Following [14], we assume an initial condition of $X(0) = (0, 0, 10, 0)^t$. We see that whenever the number of viral templates is positive, that is whenever $X_3 > 0$, the rates of the third and fifth reactions will be substantially larger than the others. At the times when $X_3 > 0$ and $X_2 = O(1)$, we have that $\gamma \gg 1$, with γ remaining large until $X_2 = O(1000)$. However, even when $X_2 = O(1000)$, the natural time-scale of the problem is $O(1/1000)$, whereas the time-scale in which we would like to answer questions is $O(1)$.

Instead of implementing our MLMC method directly, we take an alternative approach that makes use of the idea of the coupling, though not the multi-level aspect of the paper. That is, we will build an approximate process Z that will be used as a control variate for X. Towards that end, note that when the number of templates is positive, reactions 3 and 5 are much faster than the others. Ignoring the other reactions, we see that when $X_3 > 0$, the "system" governing the dynamical behavior of S is

$$\emptyset \underset{2}{\overset{1000X_3(t)}{\rightleftarrows}} S,$$

which has an equilibrium distribution that is Poisson with a parameter of $500X_3(t)$, see [4]. Believing that we may use this mean value of $X_2(s)$ in the integrated intensity of reaction 6, that is

$$\int_0^t X_1(s)X_2(s)ds \approx \int_0^t X_1(s)(500X_3(s))ds, \tag{43}$$

we hope a good approximate model for G, T, and V, which we denote by $Z = (Z_1, Z_3, Z_4)$ so as to remain consistent with the enumeration of X, is

$$\begin{aligned}
Z_1(t) &= X_1(0) + Y_1\left(\int_0^t Z_3(s)ds\right) - Y_2\left(0.025\int_0^t Z_1(s)ds\right) \\
&\quad - Y_6\left(3.75\times 10^{-3}\int_0^t Z_1(s)Z_3(s)ds\right) \\
Z_3(t) &= X_3(0) + Y_2\left(0.025\int_0^t Z_1(s)ds\right) - Y_4\left(0.25\int_0^t Z_3(s)ds\right) \\
Z_4(t) &= X_4(0) + Y_6\left(3.75\times 10^{-3}\int_0^t Z_1(s)Z_3(s)ds\right).
\end{aligned} \tag{44}$$

Note that while Z is an approximate model of X, it is still a valid continuous time Markov chain satisfying the natural non-negativity constraints. In particular, there is no time-discretization parameter in Z, which

is where many technical problems related to tau-leaping (stability concerns, negativity of molecular counts, etc.) arise.

We will now couple the two processes in a manner similar to (22) and build our estimator. Let ζ_k denote the reaction vector for the kth reaction. Let $\lambda_6(X) = 7.5 \times 10^{-6} X_1 X_2$, and $\Lambda_6(Z) = 3.75 \times 10^{-3} Z_1 Z_3$. For arbitrary f, we can estimate $\mathbb{E}f(X(T))$ via

$$\mathbb{E}f(X(T)) = \mathbb{E}(f(X(T)) - f(Z(T))) + \mathbb{E}f(Z(T)), \tag{45}$$

where $\mathbb{E}f(Z(T))$ is estimated by crude Monte Carlo using the representation (44), which is relatively cheap to simulate, and we estimate $\mathbb{E}(f(X(T)) - f(Z(T)))$ using independent realizations from the coupled processes (X, Z) below

$$\begin{aligned}
X(t) &= X(0) + Y_{1,1}\left(\int_0^t \min\{X_3(s), Z_3(s)\}ds\right)\zeta_1 + Y_{1,2}\left(\int_0^t X_3(s) - \min\{X_3(s), Z_3(s)\}ds\right)\zeta_1 \\
&\quad + Y_{2,1}\left(0.025\int_0^t \min\{X_1(s), Z_1(s)\}ds\right)\zeta_2 + Y_{2,2}\left(0.025\int_0^t X_1(s) - \min\{X_1(s), Z_1(s)\}ds\right)\zeta_2 \\
&\quad + Y_3\left(1000\int_0^t X_3(s)ds\right)\zeta_3 \\
&\quad + Y_{4,1}\left(0.25\int_0^t \min\{X_3(s), Z_3(s)\}(s)ds\right)\zeta_4 + Y_{4,2}\left(0.25\int_0^t X_3(s) - \min\{X_3(s), Z_3(s)\}(s)ds\right)\zeta_4 \\
&\quad + Y_5\left(2\int_0^t X_2(s)ds\right)\zeta_5 \\
&\quad + Y_{6,1}\left(\int_0^t \min\{\lambda_6(X(s)), \Lambda_6(Z(s))\}ds\right)\zeta_6 - Y_{6,2}\left(\int_0^t \lambda_6(X(s)) - \min\{\lambda_6(X(s)), \Lambda_6(Z(s))\}ds\right)\zeta_6 \\
Z(t) &= Y_{1,1}\left(\int_0^t \min\{X_3(s), Z_3(s)\}ds\right)\zeta_1 + Y_{1,3}\left(\int_0^t Z_3(s) - \min\{X_3(s), Z_3(s)\}ds\right)\zeta_1 \\
&\quad + Y_{2,1}\left(0.025\int_0^t \min\{X_1(s), Z_1(s)\}ds\right)\zeta_2 + Y_{2,3}\left(0.025\int_0^t Z_1(s) - \min\{X_1(s), Z_1(s)\}ds\right)\zeta_2 \\
&\quad + Y_{4,1}\left(0.25\int_0^t \min\{X_3(s), Z_3(s)\}(s)ds\right)\zeta_4 + Y_{4,3}\left(0.25\int_0^t Z_3(s) - \min\{X_3(s), Z_3(s)\}(s)ds\right)\zeta_4 \\
&\quad + Y_{6,1}\left(\int_0^t \min\{\lambda_6(X(s)), \Lambda_6(Z(s))\}ds\right)\zeta_6 - Y_{6,3}\left(\int_0^t \Lambda_6(Z(s)) - \min\{\lambda_6(X(s)), \Lambda_6(Z(s))\}ds\right)\zeta_6,
\end{aligned} \tag{46}$$

where the $Y_{k,i}$'s are independent, unit-rate Poisson processes. Note that we have coupled the process through the reaction channels 1, 2, 4, and 6, in the usual way, though not through 3 or 5, which are not incorporated in the model for Z. Simulation of the coupled processes, which is itself just a continuous time Markov chain in $Z^6_{\geq 0}$, may proceed by any exact algorithm. Here we used the next reaction method [2, 16].

Supposing we want to estimate $\mathbb{E}X_4(20)$, giving the mean number of virus molecules at time 20, we calculate this value using both a naive application of the next reaction method with crude Monte Carlo, and the control variate approach of (45) with the coupling (46). The details of the two computations are found in Table 8. We see that the crude Monte Carlo implementation required 60 times more updates and 22 times more CPU seconds than the control variate/coupling approach, again demonstrating the usefulness of the core ideas of this paper.

10 Conclusions

This work focused on the Monte Carlo approach to estimating expected values of continuous time Markov chains. In this context there is a trade off between the accuracy and the cost of each Monte Carlo sample.

Method	Approximation	# paths	CPU Time	# updates
Crude Monte Carlo	13.85 ± 0.07	75,000	24,800 CPU S	1.45×10^{10}
Coupling	13.91 ± 0.07	N.A.	1,118.5 CPU S	2.41×10^{8}

Table 8: Details of the approximated expected virus level at time 20 using crude Monte Carlo with an exact algorithm and a control variate approach using the coupling (46).

Exact samples are available, but these are typically very expensive, especially in our target application of biochemical kinetics. Approximate samples can be computed by tau-leaping, with the bias governed by a discretization parameter, h. A realistic analysis of the cost of tau-leaping must acknowledge the importance of system scaling. In particular, for a fixed system, in the limit $h \to 0$ tau-leaping becomes infinitely more expensive than exact sampling, since it needlessly refines the waiting times between reactions. In this work, we studied tau-leaping in a general setting that incorporates system scaling without taking asymptotic limits. Motivated by the work of Giles [18] on diffusion processes, we then introduced a new multi-level version of the algorithm that combines coordinated pairs of tau-leaping paths at different h resolutions. The two main conceptual advances in this work were (a) pointing out the practical benefits of a coupling process that had previously been introduced solely as a theoretical tool, and (b) exploiting the availability of an exact sampling algorithm to give an unbiased estimator. Our theoretical analysis of the computational complexity showed that the new algorithm dramatically outperforms the existing state of the art in a wide range of scaling regimes, including the classical scaling arising in chemical kinetics. The new algorithm is straightforward to summarize and implement, and numerical results confirmed that the predicted benefits can be seen in practice.

There are several avenues for future work in this area, including,

- using Quasi-Monte Carlo sampling to improve practical performance,
- customizing the method in the context of multi-scale or hybrid models, where it is possible to exploit special structure in the form of fast/slow reactions or species, or where the discrete space Markov chain is coupled to diffusion or ODE models,
- extending the theoretical analysis to the $\gamma > 0$ regime, in order to explain why we continued to observe excellent results in practice,
- using the coupling idea without discretization to obtain a control variate method that exploits specific problem structure, as illustrated in the third example of section 9.

Acknowledgement
The authors are grateful to the Banff International Research Station for Mathematical Innovation and Discovery for supporting their attendance at the workshop on *Multi-scale Stochastic Modeling of Cell Dynamics*, January 2010, where this collaboration began. We also thank Mike Giles for very useful feedback on an earlier version of this manuscript.

References

[1] David F. Anderson, *An efficient finite difference method for parameter sensitivities of continuous time Markov chains*, Submitted. Available on arxiv.org at `http://arxiv.org/abs/1109.2890`.

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

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

[4] David F. Anderson, Gheorghe Craciun, and Thomas G. Kurtz, *Product-form stationary distributions for deficiency zero chemical reaction networks*, Bulletin of Mathematical Biology **72** (2010), no. 8, 1947–1970.

[5] 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.

[6] David F. Anderson and Masanori Koyama, *Weak error analysis of numerical methods for stochastic models of population processes*, Submitted. Available on arxiv.org at `http://arxiv:1102.2922`.

[7] 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.

[8] 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.

[9] Andrea Barth, Christoph Schwab, and Nathaniel Zollinger, *Multi-level monte carlo finite element method for elliptic PDEs with stochastic coefficients*, Numerische Mathematik **119** (2011), 123–161.

[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. Cliffe, M. Giles, R. Scheichl, and A. Teckentrup, *Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients*, Computing and Visualization in Science **14** (2011), 3–15.

[14] Weinan E, Di Liu, and Eric Vanden-Eijnden, *Nested stochastic simulation algorithms for chemical kinetic systems with multiple time scales*, J. Comp. Phys. **221** (2007), no. 1, 158–180.

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

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

[23] Eric L. Haseltine and James B. Rawlings, *Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics*, J. Chem. Phys. **117** (2002), no. 15, 6959–6969.

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

[25] 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.

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

[27] Yucheng Hu, Tiejun Li, and Bin Min, *The weak convergence analysis of tau-leaping methods: Revisited*, Comm. Math. Sci. **9** (2011), 965–996.

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

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

[30] Peter Kloeden, Andreas Neuenkirch, and Raffaella Pavani, *Multilevel Monte Carlo for stochastic differential equations with additive fractional noise*, Annals of Operations Research **189** (2011), 255–276.

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

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

[33] _____, *Representation and approximation of counting processes*, Advances in filtering and optimal stochastic control, vol. 42, Springer, Berlin, 1982.

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

[35] 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.

[36] R. Srivastava, L. You, J. Summers, and J. Yin, *Stochastic vs. deterministic modeling of intracellular viral kinetics*, J. Theor. Biol **218** (2002), 309–321.

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