MULTISCALE MODEL. SIMUL.
Vol. 12, No. 2, pp. 581–615

HYBRID CHERNOFF TAU-LEAP*

ALVARO MORAES†, RAUL TEMPONE‡, AND PEDRO VILANOVA‡

Abstract. Markovian pure jump processes model a wide range of phenomena, including chemical reactions at the molecular level, dynamics of wireless communication networks, and the spread of epidemic diseases in small populations. There exist algorithms such as Gillespie's stochastic simulation algorithm (SSA) and Anderson's modified next reaction method (MNRM) that simulate a single path with the exact distribution of the process, but this can be time consuming when many reactions take place during a short time interval. Gillespie's approximated tau-leap method, on the other hand, can be used to reduce computational time, but it may lead to nonphysical values due to a positive one-step exit probability, and it also introduces a time discretization error. Here, we present a novel hybrid algorithm for simulating individual paths which adaptively switches between the SSA and the tau-leap method. The switching strategy is based on a comparison of the expected interarrival time of the SSA and an adaptive time step derived from a Chernoff-type bound for the one-step exit probability. Because this bound is nonasymptotic, we do not need to make any distributional approximation for the tau-leap increments. This hybrid method allows us (i) to control the global exit probability of any simulated path and (ii) to obtain accurate and computable estimates of the expected value of any smooth observable of the process with minimal computational work. We present numerical examples that illustrate the performance of the proposed method.

Key words. tau-leap, error estimates, error control, exit probability, weak approximation, hybrid algorithms

AMS subject classifications. 60J75, 60J27, 65G20, 92C40

DOI. 10.1137/130925657

1. Introduction.

In this work, we present a hybrid algorithm to accurately compute

$$\mathrm{E}\left[g(X(T))\right], \tag{1.1}$$

the expected value of some given smooth function, $g : \mathbb{R}^d \to \mathbb{R}$, where X is a nonhomogeneous Poisson process taking values in $\mathbb{Z}_+^d$, and T is a given final time. Here, $\mathbb{Z}_+$ denotes the set of nonnegative integers, and the ith component, $X_i(t)$, describes, for example, the number of particles of species i present in a chemical system at time t. In that type of system, different species undergo reactions at random times by changing the number of particles of at least one of the species. The probability that a reaction will happen in a small time interval is modeled by a propensity function that depends on the current state of the system.

Pathwise realizations of such pure jump processes (see [10]) can be simulated exactly using the stochastic simulation algorithm (SSA) introduced by Gillespie in [13]. Independently, an equivalent kinetic Monte Carlo algorithm was developed in the physics community in the 1960s (see [25] for references).

Although these algorithms generate exact realizations of the Markov process, X, they are computationally feasible only for relatively low propensities. For example,

*Received by the editors June 19, 2013; accepted for publication (in revised form) February 12, 2014; published electronically April 24, 2014.
http://www.siam.org/journals/mms/12-2/92565.html
†Computer, Electrical and Mathematical Sciences and Engineering, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia (alvaro.moraesgutierrez@kaust.edu.sa).
‡Mathematical and Computer Sciences and Engineering Division, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia (raul.tempone@kaust.edu.sa, pedro.guerra@kaust.edu.sa).

in the SSA, at each time step, the process is simulated exactly by sampling the next reaction to occur and the waiting time for this reaction to happen (see section 1.2). Then, the total computational work of the SSA roughly becomes proportional to the expected value of the total propensity integrated over an SSA path (see Remark 3.3). For that reason, Gillespie proposed in [14] the tau-leap method to approximate the SSA by evolving the process with fixed time steps, keeping the propensity fixed within each time step. In fact, the tau-leap method can be seen as a forward Euler method for a stochastic differential equation driven by Poisson random measures (see [19]). In the limit, as the time steps go to zero, the tau-leap solution converges to the SSA solution (see [23]).

A drawback of the tau-leap method is that the simulated process may take negative values, which is an undesirable consequence of the approximation and is not a feature of the original process. For this purpose, a Chernoff-type bound for the time step size is developed here. It controls the probability of taking negative values by adjusting the time steps. Nevertheless, there are two main scenarios in which we could obtain extremely small time steps by using the Chernoff bound: either in the case of very stringent probabilities of taking negative values, or because the current state of the tau-leap approximate process is relatively close to the boundary. On the contrary, by using an exact step, the probability of taking negative values is obviously zero, and, when the process is relatively close to the boundary, the expected time step size of the exact method is usually larger than that obtained by the Chernoff bound. Therefore, to avoid extremely small time steps, we propose switching between the SSA and the Chernoff tau-leap method adaptively, creating a hybrid SSA–Chernoff tau-leap method. The selection of the simulation method depends on the current state of the approximate process through the total propensity, which is a measure of the activity of the system around the current state. Therefore, the hybrid algorithm reveals the existence of two scales (low/high) of activity that determine whether to choose an exact or approximate simulation method. Moreover, our hybrid Chernoff tau-leap method gives accurate estimates of the global error of the approximation and also its corresponding computational work.

In [6], a hybrid SSA tau-leap algorithm is proposed. In that work, the proposed switching rule depends on two free parameters, and it is based on the so-called *leap condition*, which can be interpreted as a local time discretization error control. While they are focused on avoiding negative population values, the global error control and its computational work are not treated. Methods to prevent negative values for the tau-leap method can roughly be divided into three classes: preleap checks, postleap checks, and modifications of the Poisson distributed increments. A preleap check calculates the largest possible time step fulfilling some leap criterion, often based on controlling the relative change in the propensity function before taking the step (see [7, 6, 15]). This is primarily aimed at reducing the local time discretization error, but it also reduces the probability of taking negative values. The approach presented here includes a preleap check that strictly bounds the exit probability, and it is better suited for estimating the tails of the Poisson distribution than a standard Gaussian approximation is. In [3], an alternative postleap check was introduced to guarantee a nonnegative population in each step. If a step leading to a negative population has been taken, the postleap procedure retakes a shorter step, conditioned on already sampled data from the failed step, to avoid sampling bias. However, this procedure may be expensive since, when computing the new step, binomial-distributed Poisson bridges need to be simulated. A third way to prevent negative populations is to replace the Poisson-distributed increments in the tau-leap method with bounded

increments from the binomial or multinomial distributions (see [8, 22, 24]). This technique introduces another approximation error but also imposes a restriction on the maximum step size in order to preserve the expected value of the tau-leap increment.

In this work, we derive a novel preleap check that is based on the general Chernoff bound used in large deviation theory [9]. More specifically, let $\bar{x}$ be the state of the approximate process at time t, and let $\delta \in (0,1)$ be given. We compute a time step, $\tau = \tau(\delta, \bar{x})$, such that the probability that the approximate process reaches a nonphysical negative value in the interval $[t, t+\tau)$ is less than δ. Also, by bounding the one-step exit probability by δ, we are able to control the probability that a whole hybrid path exits the $\mathbb{Z}_+^d$ lattice. Simply put, this is a global exit probability.

The global error arising from the hybrid method can be decomposed into three components: the global exit error, the time discretization error, and the statistical error. This global error should be less than a prescribed tolerance, TOL, with probability larger than a certain confidence level. The global exit error is a quantity that is derived from the global exit probability and therefore can be controlled by δ. The analysis and control of this component together are among the main contributions of this work. The discretization error inherent in the tau-leap method is controlled through a time mesh of size h (see [16]). Finally, the statistical error is controlled by the number of hybrid paths, M, by making use of the central limit theorem [21]. The parameters δ, h, and M are functions of TOL since they are obtained by approximately minimizing the computational work of the hybrid method under the constraint that the global error must be less than TOL. Here, the computational work is measured as the amount of time needed for computing an estimate of $\mathrm{E}\left[g(X(T))\right]$ within TOL with a given level of confidence. This is known in the literature as CPU runtime.

The methodology presented here also allows the determination of when an exact method is preferred over the hybrid method. Similar hybrid methods have been proposed for the regular tau-leap method (see [6]), but without the rigorous global error estimation and control that are presented here.

1.1. The pure jump process. To describe the pure jump process, $X : [0,T] \times \Omega \to \mathbb{Z}_+^d$, occurring in (1.1), we consider a system of d species interacting through J different reaction channels. For the sake of brevity, we write $X(t,\omega) \equiv X(t)$. Let $X_i(t)$ be the number of particles of species i in the system at time t. We want to study the evolution of the state vector,

$$X(t) = (X_1(t), \ldots, X_d(t)) \in \mathbb{Z}_+^d,$$

modeled as a continuous-time, discrete-space Markov chain starting at some state, $X(0) \in \mathbb{Z}_+^d$. Each reaction can be described by the vector $\nu_j \in \mathbb{Z}^d$ such that, for a state vector $x \in \mathbb{Z}_+^d$, a single firing of reaction j leads to the change

$$x \to x + \nu_j.$$

The probability that reaction j will occur during the small interval $(t, t+\mathrm{d}t)$ is then assumed to be

$$\mathrm{P}\left(\text{reaction } j \text{ fires during } (t, t+\mathrm{d}t) \;\middle|\; X(t) = x\right) = a_j(x)\mathrm{d}t + o\left(\mathrm{d}t\right), \tag{1.2}$$

with a given nonnegative polynomial propensity function, $a_j : \mathbb{R}^d \to \mathbb{R}$. We set $a_j(x) = 0$ for those x such that $x + \nu_j \notin \mathbb{Z}_+^d$.

A process, X, that satisfies the Markov property together with (1.2) is a continuous-time, discrete-space Markov chain that can be characterized by the nonhomogeneous

Poisson process,

$$X(t) = X(0) + \sum_{j=1}^{J} \nu_j Y_j \left(\int_0^t a_j(X(s))\, \mathrm{d}s \right), \tag{1.3}$$

where $Y_j : \mathbb{R}_+ \times \Omega \to \mathbb{Z}_+$ are independent unit-rate Poisson processes [10]. In this work, we do not assume that the species can only be transformed into other species or be consumed like in [16]. In our numerical examples, we allow the set of possible states of the system to be infinite, but we explicitly avoid cases in which one or more species grows exponentially fast or blows up in the time interval $[0, T]$.

Remark 1.1. In chemical kinetics, the above setting can be used to describe well-stirred systems of chemical species, interacting through different chemical reactions, characterized by stoichiometric vectors, ν_j, and polynomial propensities, a_j, derived from the mass-action principle (see [18]). Such systems are assumed to be confined to a constant volume and to be in thermal, but not necessarily chemical, equilibrium at some constant temperature. Other popular applications can be found in population biology, epidemiology, and communication networks (see, e.g., [5, 11]).

Example 1.2 (simple decay model). Consider the reaction $X \xrightarrow{c} \emptyset$, where one particle is consumed. In this case, the state vector $X(t)$ is in $\mathbb{Z}_+$, where X denotes the number of particles in the system. The vector for this reaction is $\nu = -1$. The propensity functions in this case could be, for example, $a(X) = c\,X$, where $c > 0$.

The classical approach to chemical kinetics deals with state vectors of nonnegative real numbers representing the concentration of species at time t, usually measured in moles per liter. In this setting, the concentrations are assumed to vary continuously in time, according to the mass action principle, which says that each reaction in the system affects the rate of change of the species. More precisely, the effect on the instantaneous rate of change is proportional to the product of the concentrations of the reacting species. For the simple decay example, we have the reaction rate ODE (or mean field): $\dot{x}(t) = -cx(t)$ for $t \in \mathbb{R}_+$ and $x(0) = x_0 \in \mathbb{R}_+$. In general, let ν be the stoichiometric matrix with columns ν_j, and let $a(x)$ be the column vector of propensities. Then, we have

$$\begin{cases} \dot{x}(t) &= \nu a(x(t)),\ t \in \mathbb{R}_+, \\ x(0) &= x_0 \in \mathbb{R}_+. \end{cases} \tag{1.4}$$

1.2. Gillespie's SSA method. The SSA method simulates exact paths of X using (1.3). It requires the sampling of two random variables per time step: one to find the time of the next reaction and another to determine which is the reaction that is firing at that time.

In [13], Gillespie presented the original SSA or the direct method.

Given a state $X(t)$, the direct method is carried out by drawing two uniform random numbers, $U_1, U_2 \sim \mathcal{U}(0,1)$, which give the time to, and index of, the next reaction, i.e.,

$$j = \min \left\{ k \in \{1, \ldots, J\} : \sum_{i=1}^{k} \frac{a_i(X(t))}{a_0(X(t))} > U_1 \right\},\ \tau_{\min} = \frac{1}{a_0(X(t))} \ln\left(\frac{1}{U_2}\right),$$

where $a_0(x) := \sum_{j=1}^{J} a_j(x)$. The new state is $X(t + \tau_{\min}) = X(t) + \nu_j$, and by repeating the above procedure until final time T, a complete path of the process, X, can be simulated.

The drawback of this algorithm appears clearly as the sum of the intensities of all reactions, $a_0(x)$, becomes large: since all the jump times have to be included in the time discretization, the corresponding computational work may become unaffordable. Indeed, we have that the mean value of the jump times on the interval $(t, t+\tau)$ is approximately $a_0(X(t))\tau + o(\tau)$.

1.3. The tau-leap approximation. In the following, we denote by $\bar{X} : [0,T] \times \Omega \to \mathbb{Z}^d$ the tau-leap approximation of X. To avoid the computational drawback of the exact methods, i.e., when many reactions occur during a short time interval, the tau-leap method was proposed in [14]: given a population, $\bar{X}(t)$, and a time step, $\tau > 0$, the population at time $t+\tau$ is generated by

$$\bar{X}(t+\tau) = \bar{X}(t) + \sum_{j=1}^{J} \nu_j Y_j \left(a_j(\bar{X}(t))\tau\right), \tag{1.5}$$

where $\{Y_j(\lambda_j)\}_{j=1}^{J}$ are independent Poisson distributed random variables with parameter λ_j, used to model the number of times that the reaction j fires during the $(t, t+\tau)$ interval. This is nothing else than a forward Euler discretization of the stochastic differential equation of the pure jump process (1.3), realized by the Poisson random measure with state dependent intensity (see [19]).

In the limit, when $\tau \to 0$, the tau-leap method gives the same solution as the exact methods, using the property that, for a constant propensity, the firing probability in one reaction channel is independent of the other reaction channels. The total number of firings in each channel is then a Poisson distributed stochastic variable depending only on the initial population, $\bar{X}(t)$. The error thus comes from the variation of $a(X(s))$ for $s \in (t, t+\tau)$.

1.4. Outline of this work. The outline of this work is as follows. In section 2, we derive and give an implementation of the Chernoff-type bound that guarantees that the *one-step* exit probability in the tau-leap method is less than a predefined quantity. We also show that the Gaussian preleap selection step is not accurate and should not be used as a reliable bound. In section 3, we motivate and give implementation details of the one-step switching decision rule, which will be the key ingredient for generating hybrid paths. We show how to choose between the SSA and the tau-leap method on the basis of the current state of the approximated process. Next, we show how to generate hybrid paths and obtain an estimate of the path exit error based on the probability that one hybrid path exits the $\mathbb{Z}_+^d$ lattice. This estimation of the global exit probability depends on the expected number of tau-leap steps taken by the hybrid algorithm. It is easy to prove that this number is finite. Hybrid paths can also be used for estimating the expected number of steps that the SSA needs in order to reach the final time. In section 4, we decompose the total error into three components, the discretization error, the statistical error, and the global exit error, which were studied in the previous section. To control these errors, we give an algorithm capable of estimating the error components. We also compute the necessary ingredients for obtaining the desired estimate, i.e., a time mesh, a bound for the one-step exit probability, and the total number of Monte Carlo hybrid paths to be simulated. These ingredients are computed by optimizing the expected work of the hybrid method constrained to the error requirements. In section 5, we present some numerical experiments using well-known examples taken from the literature. Finally, in section 6, we provide conclusions and suggest directions for future work.

2. The Chernoff bound: One-step exit probabilities. In this section, we derive a Chernoff-type bound that helps us guarantee that the one-step exit probability in the tau-leap method is less than a predefined quantity, $\delta > 0$. This is crucial to controlling the computational global error, $\mathcal{E}$, which is defined below in section 4. To motivate the main ideas, the bound is first derived for the single reaction case and then generalized to several reactions. At the end of this section, we present an algorithm that efficiently computes the step size.

2.1. The single reaction case. Let $Q \equiv Q(\lambda)$ be a Poisson random variable with parameter $\lambda > 0$. Given a nonnegative integer, n, consider the following two upper bounds for $\mathrm{P}\,(Q \geq n)$: the Klar bound [17] and a Chernoff-type bound [9], which we derive below. The Klar bound is given by

$$\mathrm{P}\,(Q \geq n) \leq \left(1 - \frac{\lambda}{n+1}\right)^{-1} \exp(-\lambda)\frac{\lambda^n}{n!} \tag{2.1}$$

and is valid for $\lambda < n+1$, while the Chernoff bound is given by

$$\mathrm{P}\,(Q \geq n) \leq \exp\Big(n(1 - \log(n/\lambda) - \lambda)\Big) \tag{2.2}$$

and is valid for $\lambda < n$; otherwise, it is trivial.

In order to prove the Chernoff bound (2.2), we first note that the Markov inequality gives, for every $s > 0$,

$$\mathrm{P}\,(Q \geq n) = \mathrm{P}\left(e^{sQ} \geq e^{sn}\right) \leq \frac{\mathrm{E}\left[e^{sQ}\right]}{e^{sn}},$$

and thus

$$\mathrm{P}\,(Q \geq n) \leq \exp\left(\inf_{s>0}\{-sn + \lambda(e^s - 1)\}\right).$$

When $\lambda \in (0, n)$, the infimum,

$$\inf_{s>0}\{-sn + \lambda(e^s - 1)\},$$

is achieved at $s^* = \log(n/\lambda)$, and its value is $n(1 - \log(n/\lambda) - \lambda)$. From this simple calculation, we obtain the Chernoff bound (2.2).

Given a positive integer, n, representing the state of the system at a certain time, and $\delta \in (0, 1)$, we would like to obtain the largest value for λ such that $\mathrm{P}\,(Q(\lambda) \geq n) \leq \delta$. From the Chernoff bound, we have

$$n(1 - \log(n/\lambda) - \lambda) \leq \log(\delta)$$

or, equivalently,

$$\log(\lambda) - \frac{\lambda}{n} \leq \log(n) + \frac{\log(\delta)}{n} - 1. \tag{2.3}$$

If in the Klar bound (2.1) we neglect the factor

$$\left(1 - \frac{\lambda}{n+1}\right)^{-1},$$

which lies between 1 and $n+1$ when $\lambda \in (0, n)$, then we obtain

$$\exp(\lambda)\frac{\lambda^n}{n!} \leq \delta.$$

Taking logarithms on both sides, we arrive exactly at the Chernoff bound (2.3). We can see in Figure 1 that the Klar bound (2.1) is sharp, except when λ gets close to the singularity at $n+1$. The Chernoff bound (2.2) is not as sharp as Klar's bound but, as we will see in the next subsection, it has a generalization to the more practical many-reaction case. We observe that the Gaussian approximation in Figure 1 performs poorly for small values of λ and is not a bound in general.

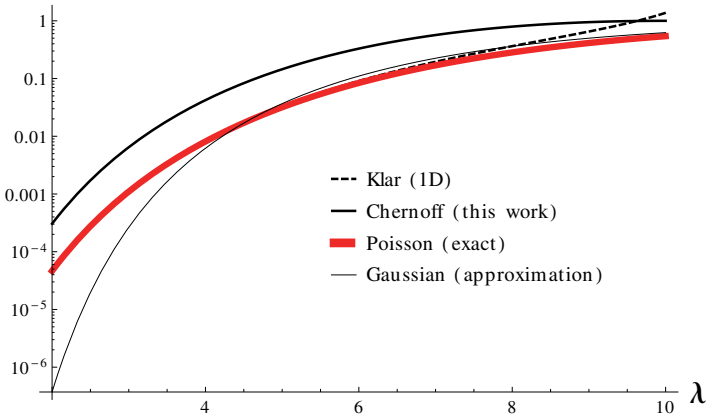

FIG. 1. *Let $n = 10$ and $\lambda \in (2, 10)$. Here, we show the semilogarithmic plot of $\mathrm{P}(Q(\lambda) \geq n)$, the Chernoff bound $\exp((n(1 - \log(n/\lambda) - \lambda)))$, the Klar bound, and the Gaussian approximation.*

2.2. The many-reaction case. To the best of our knowledge, there is no simple expression for the cumulative distribution function of a linear combination of independent Poisson random variables. For that reason, we propose a Chernoff-type bound for estimating the maximum size of the tau-leap step when many reactions are involved.

Consider the following preleap check problem: find the largest possible τ such that, with high probability, the next step of the tau-leap method will take a value in the $\mathbb{Z}_+^d$ lattice of nonnegative integers, i.e.,

$$\mathrm{P}\left(\bar{X}(t) + \sum_{j=1}^{J} \nu_j Y_j\left(a_j(\bar{X}(t))\tau\right) \in \mathbb{Z}_+^d \,\middle|\, \bar{X}(t)\right) \geq 1 - \delta \tag{2.4}$$

for some small $\delta > 0$. Observe that this value of τ depends on $\bar{X}(t)$.

Condition (2.4) can be achieved by solving d auxiliary problems, one for each x-coordinate, $i = 1, 2, \ldots, d$. Find the largest possible $\tau_i \geq 0$, such that

$$\mathrm{P}\left(\bar{X}_i(t) + \sum_{j=1}^{J} \nu_{ji} Y_j\left(a_j(\bar{X}(t))\tau_i\right) < 0 \,\middle|\, \bar{X}(t)\right) \leq \delta_i, \tag{2.5}$$

where $\delta_i = \delta/d$ and ν_{ji} is the ith coordinate of the jth reaction channel, ν_j. Inequality (2.4) is then fulfilled if we let $\tau := \min\{\tau_i : i = 1, 2, \ldots, d\}$.

In the following sections, we show how to find the largest time steps, τ_i.

2.2.1. Defining the function $\tau_i(s)$. Consider the random variable $Q_i(t,\tau_i)$ representing the opposite of the increment of the process, $\bar{X}_i(t)$:

$$Q_i(t,\tau_i) := \sum_{j=1}^{J} (-\nu_{ji}) Y_j\left(a_j(\bar{X}(t))\tau_i\right).$$

Observe that $Q_i(t,\tau_i)$ is a linear combination of J independent Poisson random variables whose intensities are multiples of τ_i.

For all $s > 0$, using the Markov inequality, we obtain an upper bound for the probabilities we want to control:

$$\begin{aligned} \mathrm{P}\left(Q_i(t,\tau_i) > \bar{X}_i(t) \,\middle|\, \bar{X}(t)\right) &= \mathrm{P}\left(\exp\left(sQ_i(t,\tau_i)\right) > \exp\left(s\bar{X}_i(t)\right) \,\middle|\, \bar{X}(t)\right) \\ &\leq \frac{\mathrm{E}\left[\exp\left(sQ_i(t,\tau_i)\right)\right]}{\exp\left(s\bar{X}_i(t)\right)}. \end{aligned} \tag{2.6}$$

Observe that the independent Poisson random variables, $Y_j\left(a_j(\bar{X}(t))\tau_i\right)$, have moment-generating functions,

$$M_j(s) = \exp\left(a_j(\bar{X}(t))\tau_i(e^s - 1)\right),$$

and, therefore,

$$\begin{aligned} \mathrm{E}\left[\exp\left(sQ_i(t,\tau_i)\right)\right] &= \prod_{j=1}^{J} M_j(-s\nu_{ji}) \\ &= \exp\left(\tau_i \sum_{j=1}^{J} a_j(\bar{X}(t))(e^{-s\nu_{ji}} - 1)\right). \end{aligned} \tag{2.7}$$

By combining (2.6) and (2.7), we obtain the Chernoff bound for the multireaction case, namely,

$$\mathrm{P}\left(Q_i(t,\tau_i) > \bar{X}_i(t) \,\middle|\, \bar{X}(t)\right) \leq \inf_{s>0} \exp\left(-s\bar{X}_i(t) + \tau_i \sum_{j=1}^{J} a_j(\bar{X}(t))(e^{-s\nu_{ji}} - 1)\right). \tag{2.8}$$

To avoid the computational problem of finding exactly the above infimum and to guarantee that

$$\mathrm{P}\left(Q_i(t,\tau_i) > \bar{X}_i(t) \,\middle|\, \bar{X}(t)\right) \leq \delta_i,$$

we proceed as follows. First, according to (2.5) and (2.8),

$$-s\bar{X}_i(t) + \tau_i \sum_{j=1}^{J} a_j(\bar{X}(t))(e^{-s\nu_{ji}} - 1) = \log(\delta_i).$$

Using this fact, we can express τ_i as a function of s:

$$\tau_i(s) = \frac{\log(\delta_i) + s\bar{X}_i(t)}{-a_0\left(\bar{X}(t)\right) + \sum_{j=1}^{J} a_j(\bar{X}(t))e^{-s\,\nu_{ji}}}, \tag{2.9}$$

where

$$a_0(\bar{X}(t)) := \sum_{j=1}^{J} a_j(\bar{X}(t)).$$

2.2.2. Study of $\tau_i(s)$. In this section, we study how much we can increase τ_i while satisfying condition (2.5). Obviously, it is satisfied for $\tau_i = 0^+$. By a continuity argument, we want to obtain τ_i^* defined as the maximum τ_i such that every point of the interval $[0, \tau_i]$ satisfies (2.5). Note that τ_i^* could be $+\infty$.

We discuss how, depending on certain relations among the pairs $\{(a_j(\bar{X}(t)), \nu_{ji})\}_{j=1}^J$, we can conclude that τ_i^* is either a real number or $+\infty$. First, if $\nu_{ji} > 0$ for all j, then τ_i^* must be $+\infty$, since no reaction is pointing to zero. From now on, we assume that, given the coordinate i, there is at least one reaction pointing to zero, i.e.,

$$\exists j \text{ such that } \nu_{ji} < 0. \tag{2.10}$$

The denominator of (2.9) is the function

$$D_i(s) := -a_0\left(\bar{X}(t)\right) + \sum_{j=1}^{J} a_j(\bar{X}(t)) e^{-s\,\nu_{ji}}, \tag{2.11}$$

which is convex since it is a positive linear combination of the convex functions $e^{-s\nu_{ji}}$ plus the constant term $-a_0(\bar{X}(t))$. We also notice that $D_i(0) = 0$ and $D_i(+\infty) = +\infty$ when (2.10) holds.

On the other hand, the numerator of (2.9),

$$R_i(s) := \log(\delta_i) + s\bar{X}_i(t),$$

is a straight line crossing the vertical axis at $\log(\delta_i) < 0$, and we can assume that its slope, $\bar{X}_i(t)$, is positive. Otherwise, the $\bar{X}(t)$ process is at the boundary of $\mathbb{Z}_+^d$, and therefore no reaction is pointing outside the lattice, $\mathbb{Z}_+^d$. We therefore set $\tau_i^* = +\infty$.

Let us define s_i as the root of the numerator $R_i(s)$, i.e.,

$$s_i := -\log(\delta_i)/\bar{X}_i(t). \tag{2.12}$$

By direct substitution of (2.12) into (2.11), we obtain

$$D_i(s_i) = -a_0(\bar{X}(t)) + \sum_{j=1}^{J} a_j(\bar{X}(t)) \delta_i^{\nu_{ji}/\bar{X}_i(t)} \tag{2.13}$$

and

$$D_i'(s_i) = -\sum_{j=1}^{J} a_j(\bar{X}(t)) \nu_{ji} \delta_i^{\nu_{ji}/\bar{X}_i(t)}. \tag{2.14}$$

In order to determine whether $\tau_i^* < \infty$ or $\tau_i^* = \infty$, we have to analyze all possible cases regarding the pair $(R_i(s), D_i(s))$.

Indeed, note that

$$D_i'(0) = -\sum_{j=1}^{J} a_j(\bar{X}(t)) \nu_{ji},$$

and if $D_i'(0) \geq 0$, which could be interpreted as a drift pointing to the boundary, then $D_i(s)$ is monotonically increasing in $[0, +\infty)$. This situation is illustrated in Figure 2: in the left panel, we see the pair $(R_i(s), D_i(s))$; in the right panel, we see the quotient

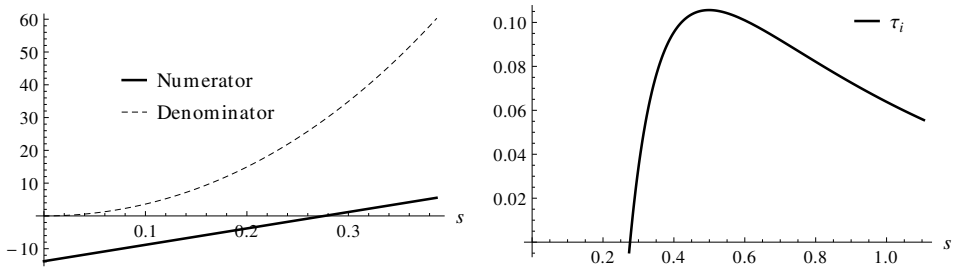

FIG. 2. *Left: Numerator $R_i(s)$ and denominator $D_i(s)$. Right: Quotient $\tau_i(s) = R_i(s)/D_i(s)$. Both plots are for the case $D_i'(0) \geq 0$.*

$\tau_i(s) = R_i(s)/D_i(s)$. The function τ_i achieves its maximum, τ_i^*, at a unique point, s_i^*.

If $D_i'(0) < 0$, which can be interpreted as a drift pointing to $+\infty$, the value of τ_i^* depends on $\bar{X}_i(t)$, i.e., on the size of the slope of $R_i(s)$. Observe that $D_i(s)$ is then negative in an interval $(0, d_i)$, with $D_i(d_i) = 0$, and in general there is not a closed form for d_i. Also, since $D_i(s)$ and $R_i(s)$ may have opposite signs for some $s \leq \max(s_i, d_i)$, this allows for artificially negative values of τ_i, which should not be taken into account.

The value of τ_i^* is finite or $+\infty$ according to the sign of $D_i(s_i)$. These three cases are shown in the left panel of Figure 3. When $\bar{X}_i(t)$ is large enough, i.e., when $D_i(s_i) < 0$, we can see in the right panel of Figure 3 that $\tau_i* = +\infty$. This is true because the limit of $\tau_i(s)$, as $s \to d_i^+$, is $+\infty$. Therefore, if $\bar{X}_i(t)$ is far from the boundary and the drift is pointing to $+\infty$, we can take τ_i to be as large as we wish.

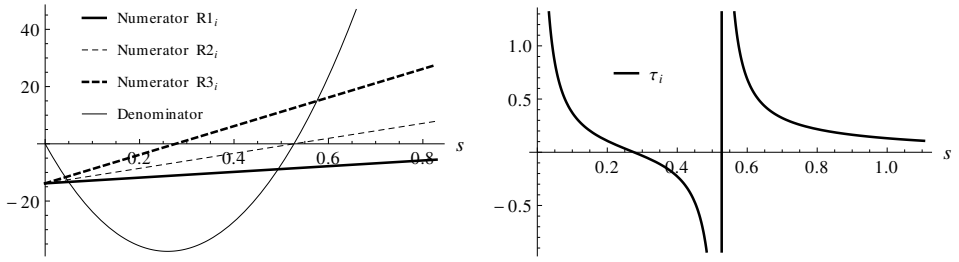

FIG. 3. *In this case $\sum_{j=1}^{J} a_j(\bar{X}(t))\nu_{ji} > 0$. Left: Relative positions of $(R_i(s), D_i(s))$, depending on the sign of $D_i(s_i)$. Right: $\tau_i(s) = R_i(s)/D_i(s)$ in the case $D_i(s_i) < 0$.*

The two other cases are as follows: if $D_i(s_i) > 0$, then $\bar{X}_i(t)$ is, in a certain sense, close to the boundary, and even if the drift is pointing to $+\infty$, there exists an upper bound for τ_i. This is illustrated in the left part of Figure 4, where τ_i^* is the maximum to the right of s_i. Finally, if $D_i(s_i) = 0$, then τ_i^* can be obtained as the limit of $\tau_i(s)$ as $s \to d_i^+$. By l'Hôpital's rule, we have that $\tau_i^* = \bar{X}_i(t)/D_i'(s_i)$.

We can summarize the previous discussion as follows: If $\nu_{ji} \geq 0$ for all j, then $\tau_i^* = +\infty$; otherwise, we have the following three cases:

1. $D_i(s_i) > 0$. In this case, $\tau_i(s_i) = 0$ and $D_i(s)$ is positive and increasing for all $s \geq s_i$. Therefore, $\tau_i(s)$ is equal to the ratio of two positive increasing functions. The numerator, $R_i(s)$, is a linear function, and the denominator, $D_i(s)$, grows exponentially fast. Then, there exist an upper bound, τ_i^*, and a unique number, s_i^*, which satisfies $\tau_i(s_i^*) = \tau_i^*$. We develop an algorithm for

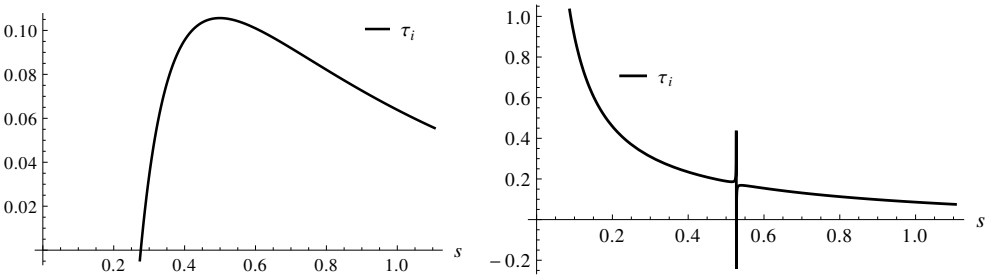

FIG. 4. *The other two cases for $\tau_i(s)$ when $\sum_{j=1}^{J} a_j(\bar{X}(t))\nu_{ji} > 0$. Left: $D_i(s_i) > 0$. Right: $D_i(s_i) = 0$.*

approximating s_i^*, using the relation $\tau_i'(s_i^*) = 0$.
2. If $D_i(s_i) < 0$, then $\tau_i^* = +\infty$.
3. If $D_i(s_i) = 0$, then $\tau_i^* = \bar{X}_i(t)/D_i'(s_i)$.

2.2.3. Approximating s_i^*. In this section, we present a simple and fast algorithm for approximating s_i^*, which was defined in case 1 above. We proceed in two steps. In the first step, we find an initial guess, $s_{i,0}^*$, and in the second step, we improve this guess and obtain $s_{i,1}^*$. Therefore, $\tau_i^*{=}\tau_i(s_i^*)$ will be approximated by $\tau_i(s_{i,1}^*)$.

From (2.10), the equation $\tau_i'(s) = 0$ is equivalent to

$$-a_0(\bar{X}(t))+\sum_{j=1}^{J} a_j(\bar{X}(t))\exp(-s\nu_{ji}) = (s-s_i)\sum_{j=1}^{J} a_j(\bar{X}(t))(-\nu_{ji})\exp(-s\nu_{ji}). \tag{2.15}$$

Let us define

$$\hat{s} := s - s_i \quad \text{and} \quad b_{ji}(\bar{X}(t)) := a_j(\bar{X}(t))\delta_i^{\nu_{ji}/\bar{X}_i(t)} > 0.$$

As a consequence,

$$\exp(-s\nu_{ji}) = \delta_i^{\nu_{ji}/\bar{X}_i(t)}\exp(-\hat{s}\nu_{ji}),$$

and therefore (2.15) can be written as

$$\sum_{j=1}^{J} b_{ji}(\bar{X}(t))\exp(-\hat{s}\nu_{ji}) = a_0(\bar{X}(t)) + \hat{s}\sum_{j=1}^{J} b_{ji}(\bar{X}(t))(-\nu_{ji})\exp(-\hat{s}\nu_{ji}). \tag{2.16}$$

Once we introduce the auxiliary functions Ψ_{ji},

$$\Psi_{ji}(y) := \exp(-\nu_{ji}y)(1+\nu_{ji}y),$$

(2.16) becomes equivalent to finding s_i^* such that $G(s_i^*) = 0$, where

$$G(y) = -a_0(\bar{X}(t)) + \sum_{j=1}^{J} b_{ji}(\bar{X}(t))\Psi_{ji}(y).$$

The left graph in Figure 5 shows the shape of Ψ_{ji} depending on the sign of ν_{ji}. We deduce that G is a decreasing function such that $G(0) = D(s_i)$ and $G(+\infty) = -\infty$.

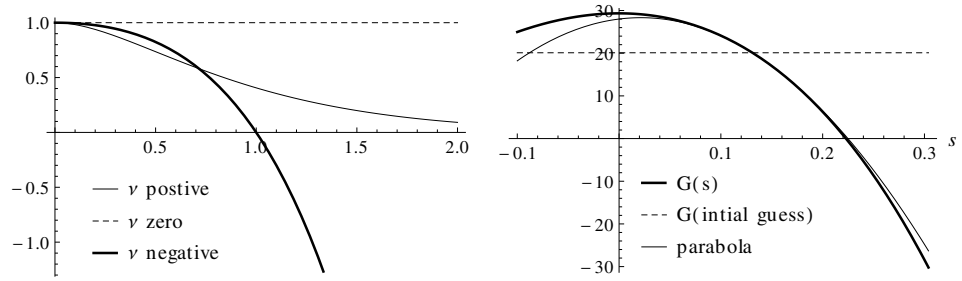

FIG. 5. *Left: Function $\Psi(s)$ for different values of ν. Right: Function G and its approximating parabola.*

By neglecting the exponential term in Ψ_{ji}, we can obtain an initial guess for s_i^*, i.e.,

$$s_{i,0}^* = \frac{-a_0(\bar{X}(t)) + \sum_{j=1}^J b_{ji}(\bar{X}(t))}{\sum_{j=1}^J b_{ji}(\bar{X}(t))(-\nu_{ji})} = \frac{D_i(s_i)}{D_i'(s_i)}.$$

As we observed in case 1, the values for $D_i(s_i)$ and $D_i'(s_i)$ are positive, and our initial guess, $s_{i,0}^*$, is a positive number.

In the right graph in Figure 5, we can see that the parabola obtained as the second-order approximation of G at $s_{i,0}^*$ is a good approximation of G close to its root, s_i^*. Therefore, we obtain $s_{i,1}^*$ as the largest root of the approximating parabola. By evaluating $\tau_i(s_{i,1}^*)$, we obtain a sharp lower bound of $\sup_{s>0} \tau_i(s)$.

An expression for $s_{i,1}^*$ in terms of G and its derivatives up to the second order evaluated at $s_{i,0}^*$ is given by

$$s_{i,1}^* = s_{i,0}^* + \left(-G'(s_{i,0}^*) + \sqrt{G'(s_{i,0}^*)^2 - 2G''(s_{i,0}^*)G(s_{i,0}^*)}\right)/G''(s_{i,0}^*). \tag{2.17}$$

An efficient implementation for computing $\tau_i(s_{i,1}^*) \approx \tau_i^*$ can be found in Algorithm 1 (see the definition of τ_i^* in case 1 at the end of section 2.2.2).

2.3. Computational work of the preleap methods: Chernoff bound versus Gaussian approximation. In this section, we first summarize an alternative preleap method, introduced in [16], which uses a Gaussian-type approximation. We then compare the algorithm that computes the Chernoff step size with the one that computes the Gaussian-type step size, τ_{gau}.

Given $\delta > 0$, we want to find the largest τ_{gau} such that

$$\mathrm{P}\left(\bar{X}_i(t+\tau_{gau}) < 0 \;\middle|\; \bar{X}(t)\right) \le \delta, \quad i{=}1,\dots,d. \tag{2.18}$$

Using (1.5), we get

$$\begin{aligned}\mathrm{P}\left(\bar{X}_i(t) - Q_i(t,\tau_{gau}) < 0 \,\middle|\, \bar{X}(t)\right) &= \mathrm{P}\left(Q_i(t,\tau_{gau}) > \bar{X}_i(t) \,\middle|\, \bar{X}(t)\right)\\ &= 1 - \mathrm{P}\left(Q_i(t) \le \bar{X}_i(t) \,\middle|\, \bar{X}(t)\right) \le \delta\,,\end{aligned}$$

where $Q_i(t,\tau_{gau}) := -\sum_{j=1}^J \nu_{ji} Y_j(a_j(\bar{X}(t))\tau_{gau})$.

Now, we approximate $Q_i(t,\tau_{gau})$ by

$$\hat{Q}_i(t,\tau_{gau}) := \mathrm{E}\left[Q_i(t,\tau_{gau})\right] + \sqrt{\mathrm{Var}\left[Q_i(t,\tau_{gau})\right]}N\,,$$

Algorithm 1. Computes the Chernoff tau-leap step size. Inputs: The current state of the approximate process, $\bar{X}$, the propensity functions evaluated at $\bar{X}$, $(a_j(\bar{X}))_{j=1}^J$, and the stoichiometric matrix, ν_{ji}. Output: τ. Notes: For a fixed coordinate, i, such that (2.10) is fulfilled (otherwise $\tau_i = +\infty$), this algorithm determines whether or not τ_i^* is finite. When τ_i^* is finite, this algorithm computes an approximation for $\tau_i(s_{i,1}^*)$ based on (2.17).

Require: $a_0 \leftarrow \sum_{j=1}^J a_j > 0$
for $i = 1$ **to** d **do**
  if $\exists j : \nu_{ji} < 0$ **and** $\bar{X}_i(t) > 0$, **then**
    $x \leftarrow \bar{X}_i(t)$
    $b_j \leftarrow a_j \, \delta_i^{\nu_{ji}/x}$
    $\hat{b} \leftarrow \sum_{j=1}^J b_j$
    if $\hat{b} - a_0 < 0$, **then**
      $\tau_i \leftarrow +\infty$
    else
      if $\hat{b} - a_0 > 0$, **then**
        $s \leftarrow (a_0 - \hat{b}) / \sum_{j=1}^J b_j \nu_{ji}$
        $\xi_j \leftarrow b_j \exp(-s\nu_{ji})$
        $c_p \leftarrow \sum_{j=1}^J \xi_j \nu_{ji}^p, \quad p = 0, 1, 2, 3,$
        $\alpha \leftarrow \frac{1}{2}(c_3 s - c_2)$
        $\beta \leftarrow -c_2 s$
        $\gamma \leftarrow -a_0 + c_0 + c_1 s$
        $s \leftarrow s - (\beta + \sqrt{\beta^2 - 4\alpha\gamma})/2\alpha$
        $\tau_i \leftarrow sx / (-a_0 + \sum_{j=1}^J b_j \exp(-s\nu_{ji}))$
      else
        $\tau_i \leftarrow -x / \sum_{j=1}^J b_j \nu_{ji}$
      end if
    end if
  else
    $\tau_i \leftarrow +\infty$
  end if
end for
return $\min\{\tau_1, \ldots, \tau_d\}$

where $N \sim \mathcal{N}(0,1)$ is a standard normal random variable. We get

$$\mathrm{P}\left(\hat{Q}_i(t) \le \bar{X}_i(t) \,\middle|\, \bar{X}(t)\right) = \Phi\left(\frac{\bar{X}_i(t) + \sum_{j=1}^J \nu_{ji} a_j(\bar{X}(t))\tau_{gau}}{\sqrt{\sum_{j=1}^J \nu_{ji}^2 a_j(\bar{X}(t))\tau_{gau}}}\right),$$

where Φ is the cumulative density function for the standard normal distribution. Finally, let z_δ satisfy $\Phi(z_\delta) = 1 - \delta$. Then, the τ_{gau} that approximately solves (2.18) is obtained from

$$\bar{X}_i(t) + \sum_{j=1}^J \nu_{ji} a_j(\bar{X}(t))\tau_{gau} = z_\delta \sqrt{\sum_{j=1}^J \nu_{ji}^2 a_j(\bar{X}(t))\tau_{gau}}.$$

Algorithm 2 efficiently computes the step size, τ_{gau}, using the Gaussian approxi-

mation.

Algorithm 2. Computes the tau-leap step size using a Gaussian approximation. Inputs: The current state of the approximate process, $\bar{X}$, the propensity functions evaluated at $\bar{X}$, $(a_j(\bar{X}))_{j=1}^J$, and the stoichiometric matrix, ν_{ji}. Output: τ. Notes: For a fixed coordinate, i, this algorithm determines whether or not τ_i^* is finite. When τ_i^* is finite, this algorithm computes its value.

Require: $\sum_{j=1}^J a_j > 0$
for i=1 **to** d **do**
  $x \leftarrow \bar{X}_i(t)$
  $c_p \leftarrow \sum_{j=1}^J a_j \nu_{ji}^p, \quad p = 1, 2,$
  $\rho \leftarrow z_{\delta_i}^2$
  $\alpha \leftarrow \rho^2 c_2^2 - 4\rho c_1 c_2 x$
  if $c_2 = 0$ **or** ($c_1 > 0$ **and** $\alpha < 0$), **then**
    $\tau_i \leftarrow +\infty$
  else
    if $c_2 \neq 0$ **and** ($c_1 < 0$ **or** ($c_1 > 0$ **and** $\alpha \geq 0$)), **then**
      $\beta \leftarrow \rho c_2 - 2c_1 x$
      $\tau_i \leftarrow (\beta - \sqrt{\alpha})/2c_1^2$
    else
      $\tau_i \leftarrow x^2/\rho c_2$
    end if
  end if
end for
return $\min\{\tau_1, \ldots, \tau_d\}$

To quantify the relative efficiency of Algorithm 1 versus Algorithm 2, we use the following nominal operation count convention (based on McMahon [20]): add-mul, subtraction, and division 1 flop; square root 4 flops; and exp function 8 flops. We do not count the flow control work, and we assume $d = 1$ because it is easily extended to $d > 1$. Moreover, we are not taking into account the memory access cost, which usually is dominant. The total flop count for Algorithm 1 is $33 + 26J$, and for Algorithm 2 it is $19 + 2J$. The ratio tends to 13 when $J \to \infty$. However, the actual runtime in the MATLAB implementation is, in all the examples we tested, more optimistic than that predicted using the flop count. Empirically, we observed that the dominant computational work of the hybrid algorithm at each step is due to the simulation of a Poisson random variable (see [1] for details). The additional work of computing the Chernoff step size is, in fact, almost negligible.

In Figure 6, we show the comparison between the Chernoff bound and the Gaussian approximation for the simple decay model, with initial condition $X_0 = 100$ (see section 5). The Chernoff bound appears to be conservative, and it holds for any δ, which is not the case for the Gaussian approximation, whereas their computational work is of the same order. We can see that in the Gaussian case, the approximation does not attain the required one-step exit probability, with a confidence level of 95%, for most δ.

3. The one-step switching rule and hybrid trajectories. In this section, we first present a one-step switching rule that, given the current state of the approximate process, $\bar{X}(t)$, adaptively determines whether to use an exact or an approximated

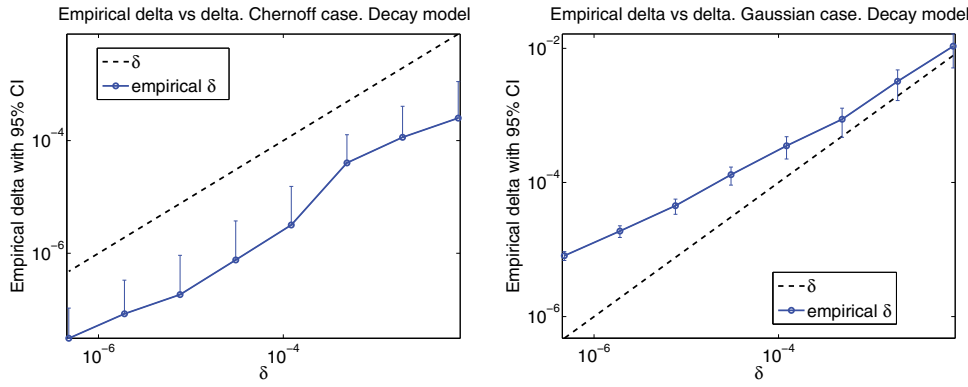

FIG. 6. *The Chernoff bound versus the Gaussian approximation in the simple decay model example, with initial condition* X_0=100 *(see section* 5*). Left: The empirical one-step exit probability bound with asymptotic confidence intervals (*95%*) versus a reference line with a unit slope (solid line) for the Chernoff tau-leap. Missing confidence intervals mean that the values are zero or negative. Right: The Gaussian approximation case. We can observe that the Chernoff bound holds for any* δ*, with a confidence level of* 95%*, which is not the case for the Gaussian approximation.*

method for the next step. Then, we present an algorithm for simulating a whole hybrid path. This algorithm consists of a certain number of exact and approximate steps. Next, we estimate the probability that one hybrid path exits the lattice, $\mathbb{Z}_+^d$, which is an event that depends on the expected number of tau-leap steps, as we will see. Finally, we show how to estimate, based only on hybrid paths, the expected number of steps of a pure SSA path.

3.1. The one-step switching rule algorithm. Here, we provide a justification for the one-step switching rule algorithm, as described in Algorithm 3.

Let $x = \bar{X}(t)$ be the current state of the approximate process, $\bar{X}$. Therefore, the expected time step of the SSA is given by $1/a_0(x)$. Let τ_{Ch}=$\tau_{Ch}(x, \delta)$ be the Chernoff tau-leap step, obtained using Algorithm 1. To move one step forward using the SSA method, we should compute at least $a_0(x)$ and sample two uniform random variables. On the other hand, to move one step forward using the Chernoff tau-leap method, we not only have to compute τ_{Ch} (discussed at the end of section 2), but we also have to generate J Poisson random variables, where J is the number of reaction channels. It is critical to observe that the computational work of generating J Poisson random variables is much larger than the computational work of generating only two uniform random variables. This computational work could be measured, for example, as the average execution time for the operations involved in it.

We now describe K_1 and K_2. In order to avoid the overhead caused by unnecessary computations of τ_{Ch}, we first estimate the computational work of moving forward from the current time, t, to the next grid point, T_0, by using the SSA method. If this work is less than the work of computing τ_{Ch}, we take an exact step. This motivates us to define K_1 as the ratio between the work of computing τ_{Ch} and the work of computing $a_0(x)$ plus sampling two uniform random variables. Otherwise, we compute τ_{Ch} and decide whether to take an SSA step or a tau-leap one, according to the comparison between τ_{Ch} and $K_2/a_0(x)$. Here $K_2 = K_2(x, \delta)$ is defined as the work of taking a Chernoff tau-leap step given the current state of the process, divided by the work of taking an SSA step plus the work of computing τ_{Ch}. As we mentioned, associated with each type of step, there is computational work. In the first case, when

$K_1/a_0(x) > T_0 - t$, the work is C_1 and includes the computation of $1/a_0(x)$ and the generation of two uniform random variates. In the same way, when $K_1/a_0(x) > T_0 - t$ and $K_2/a_0(x) > \tau_{Ch}$, the work is C_2 and involves the work contained in C_1 and of computing $\tau_{Ch}(x, \delta)$, which is denoted by C_3. On the other hand, when a Chernoff tau-leap step is taken, we have not only the constant work, C_3, but also *variable* work, which is the work of generating the Poisson random variates. The latter work is a function of the propensities of all the reaction channels, namely, $a(x)\tau_{Ch}(x, \delta)$. We model the computational work of generating *one* Poisson random variate according to [1], and this work is denoted by $C_P(\cdot)$. In the Gamma simulation method developed by Ahrens and Dieter in [1], which is used by MATLAB, the work grows like $b_1 + b_2 \ln \lambda$, where $\lambda > 15$ is the rate of the Poisson random variable. For $\lambda \leq 15$, the growth is linear. In practice, it is possible to estimate b_1 and b_2 using a Monte Carlo method with a least squares fit, as shown in Figure 7.

Summarizing, $K_1 := \frac{C_3}{C_1}$, and $K_2(\bar{X}(t), \delta) := \frac{C_3 + \sum_{j=1}^{J} C_P(a_j(\bar{X}(t))\tau_{Ch}(\bar{X}(t), \delta))}{C_1 + C_3}$.
Observe that $K_2(x, \delta) \to \frac{C_3 + Jb_1}{C_1 + C_3} =: \tilde{C} > 0$ as $\delta \to 0$.

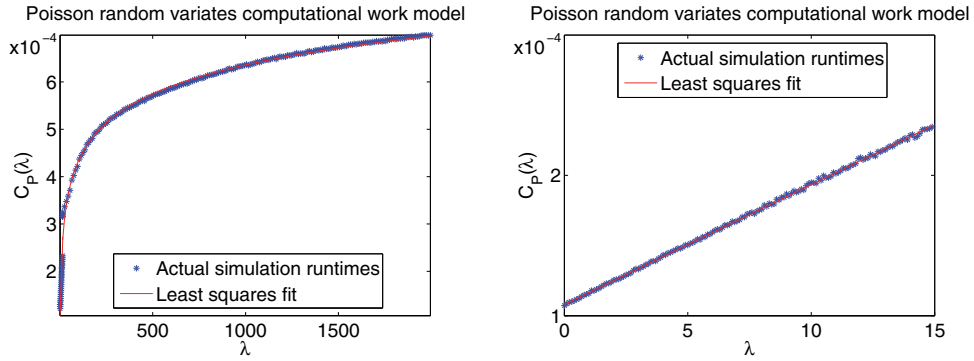

FIG. 7. *Left: The computational work (runtime) model for generating a Poisson random variate using the Gamma method by Ahrens and Dieter* [1]. *Right: Linear growth detail for* $\lambda \in [0, 15]$.

Here, we estimate the coefficients (offline precomputed, machine-dependent quantities) C_1, C_2, C_3, b_1, and b_2 by computing average execution times of the corresponding machine code block (in this case MATLAB code).

We now briefly describe Algorithm 3. The first decision is made through the comparison between the expected SSA step size and the remaining time until the next grid point, T_0. To interpret this rule, we first assume that $T_0 - t$ tends to zero. Then, the selected method tends to be SSA. This decision rule favors SSA over tau-leap and trivially guarantees the Chernoff bound. In the case of problems where the SSA method is more convenient, the advantage is obvious: it is not necessary to superfluously compute the tau-leap step size. On the other hand, this choice has "reasonable" computational work in terms of choosing SSA over tau-leap, since there is little time left until T_0. Now assume that K_1/a_0 tends to infinity; that is, a_0 tends to zero. Then, the reasonable choice is SSA, because the Chernoff tau-leap step size tends to zero in this case. It should be noted that this first decision rule has no extra computational work, because a_0 must be computed anyway. If $K_1/a_0 < T_0 - t$ holds, then the tau-leap size is computed and the second decision is made (line 3). In this case, first assume that τ_{Ch} tends to zero. Then, the selected method tends to be SSA, which is a natural choice. If, on the contrary, τ_{Ch} tends to infinity, the chosen method

Algorithm 3. The one-step switching rule. Inputs: The current time, t, the current state of the approximate process, $\bar{X}(t)$, the propensity functions, $(a_j(\bar{X}(t)))_{j=1}^{J}$, and the next grid point, T_0. Outputs: Method and τ. Notes: Based on $\mathrm{E}\left[\tau_{SSA}(\bar{X}(t))\right] = 1/a_0(\bar{X}(t))$ and $\tau_{Ch}(\bar{X}(t), \delta)$, this algorithm adaptively selects which method to use: SSA or tau-leap. We denote by τ_{SSA} (τ_{Ch}) the step size when the decision is to use the SSA (tau-leap) method.

Require: $a_0 \leftarrow \sum_{j=1}^{J} a_j > 0$
1: **if** $K_1/a_0 < T_0 - t$, **then**
2: $\quad \tau_{Ch} \leftarrow$ Algorithm 1
3: $\quad$ **if** $\tau_{Ch} < K_2(\bar{X}(t), \delta)/a_0$, **then**
4: $\quad\quad$ **return** (SSA, τ_{SSA})
5: $\quad$ **else**
6: $\quad\quad$ **return** (TL, τ_{Ch})
7: $\quad$ **end if**
8: **else**
9: $\quad$ **return** (SSA, τ_{SSA})
10: **end if**

tends to be the tau-leap, which again is a natural choice. Now, assume that K_2/a_0 tends to infinity. Then, a reasonable choice is SSA, because the step size is large and the bound is guaranteed. If K_2/a_0 tends to zero, the reasonable choice is tau-leap.

A summary of the one-step switching rule decisions is given in Table 1.

TABLE 1
One-step switching rule summary. Decision 1 *is made at line* 1 *of Algorithm* 3*, whereas decision* 2 *is made at line* 3.

	If \ tends to	∞	0
Decision 1	$T_0 - t$	go to Decision 2	SSA
	K_1/a_0	SSA	TL
Decision 2	τ_{Ch}	TL	SSA
	K_2/a_0	SSA	TL

Remark 3.1. In Figure 8, we illustrate the result of the one-step switching rule in the gene transcription and translation model (see section 5). As δ (the parameter that controls the one-step exit probability) decreases, the SSA region, in the state space of the problem, increases. We observe that, for $\delta = 10^{-2}$, almost all the state space is a Chernoff tau-leap region. For smaller δ, we observe that, if the number of proteins (y-axis) is zero, and the number of mRNAs (x-axis) is large enough, the states belong to the tau-leap region, because the propensity of the reactions pointing outside the lattice is weaker than the propensity of the reactions pointing inside the lattice. When the number of proteins increases, there is a narrow region in which the propensity of the reactions pointing out dominates, and, consequently, the switching rule chooses for the SSA method. After that, the Chernoff tau-leap is preferred. The situation is almost symmetric in the $x = y$ axis.

Remark 3.2. According to Algorithm 3, the selection of the simulation method depends on the current state, x, of the approximate process, $\bar{X}$, through the total propensity, $a_0(x)$, which is a measure of the activity of the system around the state, x. High activity around x leads to the Chernoff tau-leap method. Therefore, Algorithm

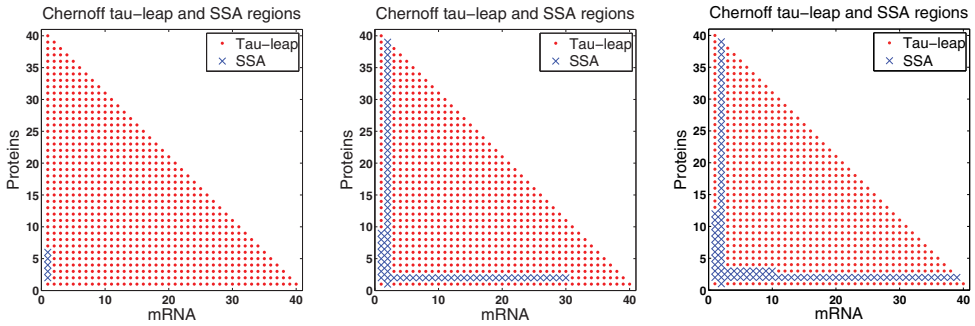

FIG. 8. *Regions of the one-step switching rule in the gene transcription and translation model (see section 5). The blue and red dots show the Chernoff tau-leap and the SSA regions, respectively. From left to right,* $\delta = 10^{-2}, 10^{-4}, 10^{-6}$*, respectively.*

3 reveals the existence of two scales (low/high) of activity that determine whether to choose an exact or approximate simulation method. Observe that the scale of activity depends not only on x but also on the one-step exit probability bound, δ, through the Chernoff step size, τ_{Ch}, and the time grid.

3.2. The hybrid algorithm. In this subsection, we present a novel algorithm that adaptively switches between the approximate (Chernoff tau-leap) and the exact (SSA) method to generate a whole hybrid path. Algorithm 4 presents this idea.

On the one hand, a path generated by an exact method never exits the lattice, $\mathbb{Z}_+^d$, although the computational work could be unaffordable due to many small interarrival times typically occurring when the process is "far" from the boundary. On the other hand, a tau-leap path, which may be cheaper than an exact one, could leave the lattice at any step. It depends on the size of the next time step and the current state of the approximate process, $\bar{X}(t)$. This one-step exit probability could be large, especially when the approximate process is "close" to the boundary. In section 2 we show how to control this one-step exit probability adaptively, by adjusting the tau-leap step size. As we previously mentioned, a hybrid path consists of a certain number of exact and approximate steps. A hybrid path could therefore leave the lattice. In section 3.3 we show how to estimate and control the probability of this event.

Given a problem, Algorithm 4 returns the last system state, $\bar{X}(t_K)$, and its respective time, t_K, such that the process belongs to the lattice. At each time, t_k, Algorithm 3 chooses the method to use (exact or approximate) for taking the $(k+1)$th step and its size.

3.3. The global exit probability bound. Once we introduce the hybrid approximate process, $\bar{X}$, one issue is to estimate the probability that one hybrid path exits the lattice, $\mathbb{Z}_+^d$. Let $\bar{\Omega}$ be the sample space for the set of all hybrid paths generated by Algorithm 4. The event $A = \{\bar{\omega} \in \bar{\Omega} : t_K = T\}$ means that the whole hybrid path, $(\bar{X}(t_k, \bar{\omega}))_{k=0}^K$, belongs to the lattice, $\mathbb{Z}_+^d$. Among these paths, the number of successful leaps using the tau-leap method is $N_{TL}(\bar{\omega}) \equiv N_{TL}$. Then,

$$\bar{\Omega} = A^c \cup A = \{\bar{\omega} \in \bar{\Omega} : t_K < T\} \cup \{\bar{\omega} \in \bar{\Omega} : t_K = T\}.$$

Algorithm 4. The hybrid tau-leap algorithm. Inputs: The initial state, $X(0)$, the propensity functions, $(a_j)_{j=1}^J$, the stoichiometric vectors, $\nu = (\nu_j)_{j=1}^J$, and the final time, T. Outputs: A sequence of states, $(\bar{X}(t_k))_{k=0}^K \subset \mathbb{Z}_+^d$, such that $t_K \leq T$. If $t_K < T$, then the path exited the $\mathbb{Z}_+^d$ lattice before the final time T. It also returns the number of times, N_{TL}, the tau-leap method was successfully applied (i.e., from $\bar{X}(t_k) \in \mathbb{Z}_+^d$, apply the tau-leap method and obtain $\bar{X}(t_{k+1}) \in \mathbb{Z}_+^d$), the number of SSA steps such that $K_1/a_0(\bar{X}(t)) > t_k - t$ is true, N_{SSA,K_1}, and the number of SSA steps such that $K_1/a_0(\bar{X}(t)) > t_k - t$ is false and $K_2(\bar{X}(t))/a_0(\bar{X}(t)) > \tau_{Ch}$ is true, N_{SSA,K_2} (see Algorithm 3). Notes: Given the current state, $next_{SSA}$ computes the next state using the SSA method. Here, t_i denotes the current time at the ith step.

$i \leftarrow 0$, $t_i \leftarrow 0$, $\bar{X}(t_i) \leftarrow X(0)$, $\bar{Z} \leftarrow X(0)$
while $t_i < T$ **do**
  $T_0 \leftarrow$ next grid point greater than t_i
  $(m, \tau) \leftarrow$ Algorithm 3 with $(t_i, \bar{Z}, (a_j(\bar{Z}))_{j=1}^J, T_0)$
  if $m = SSA$, **then**
    $N_{SSA} \leftarrow N_{SSA} + 1$
    if $t_i + \tau < T$, **then**
      $\bar{Z} \leftarrow next_{SSA}(\bar{Z})$
    end if
    $t_{i+1} \leftarrow \min\{T, t_i + \tau\}$
  else
    $\tau \leftarrow \min\{\tau, T - t_i\}$
    $\bar{Z} \leftarrow \bar{Z} + \mathcal{P}(a(\bar{Z})\tau)\nu$
    if $\bar{Z} \in \mathbb{Z}_+^d$, **then**
      $N_{TL} \leftarrow N_{TL} + 1$
      $t_{i+1} \leftarrow t_i + \tau$
    else
      return $((\bar{X}(t_k))_{k=0}^i, N_{TL}, N_{SSA})$
    end if
  end if
  $i \leftarrow i + 1$
  $\bar{X}(t_i) \leftarrow \bar{Z}$
end while
return $((\bar{X}(t_k))_{k=0}^i, N_{TL}, N_{SSA})$

Let $t_k = \sum_k \tau_k$, where each τ_k is obtained using either SSA or the tau-leap method:

$$\{\bar{\omega} \in A\} \Leftrightarrow \{\exists k \in \mathbb{N} : t_k = T\} \Leftrightarrow \bigcup_{n=0}^{+\infty} \left(\{\exists k \in \mathbb{N} : t_k = T\} \cap \{N_{TL} = n\}\right).$$

Then, we can write

$$\begin{aligned} \mathrm{P}(A) &= \sum_{n=0}^{+\infty} \mathrm{P}\left(\{\exists k \in \mathbb{N} : t_k = T, N_{TL} = n\}\right) \\ &= \sum_{n=0}^{+\infty} \mathrm{P}\left(\{\exists k \in \mathbb{N} : t_k = T \,\big|\, N_{TL} = n\}\right) \mathrm{P}\left(N_{TL} = n\right). \end{aligned}$$

At each step, the probability of exiting the lattice will be less than δ when the step size is computed using the Chernoff method (Algorithm 1), and it will be equal to zero if the SSA is adopted. At this stage, it should be pointed out that if we use the Gaussian approximation (Algorithm 2), it will not be possible to guarantee an upper bound for the probability of event A. Observe that

$$\begin{aligned}&\mathrm{P}\left(\{\exists k \in \mathbb{N} : t_k = T\}\right) = \mathrm{P}\left(\bar{X}(t_0)\in\mathbb{Z}_+^d, \bar{X}(t_1)\in\mathbb{Z}_+^d, \ldots, \bar{X}(t_k)\in\mathbb{Z}_+^d\right)\\ &= \mathrm{P}\left(\bar{X}(t_1)\in\mathbb{Z}_+^d \,\middle|\, \bar{X}(t_0)\right)\mathrm{P}\left(\bar{X}(t_2)\in\mathbb{Z}_+^d \,\middle|\, \bar{X}(t_1)\right)\ldots\mathrm{P}\left(\bar{X}(t_k)\in\mathbb{Z}_+^d \,\middle|\, \bar{X}(t_{k-1})\right),\end{aligned}$$

where the notation $\mathrm{P}\left(Y \in \mathbb{Z}_+^d \,\middle|\, X\right)$ assumes that $X \in \mathbb{Z}_+^d$. Now, by construction,

$$\mathrm{P}\left(\{\exists k \in \mathbb{N} : t_k = T \,\middle|\, N_{TL} = n\}\right) \geq (1-\delta)^n$$

because

$$\begin{aligned}\mathrm{P}\left(\bar{X}(t_j) \in \mathbb{Z}_+^d \,\middle|\, \bar{X}(t_{j-1})\right) &\geq 1-\delta \text{ if we use the Chernoff algorithm}\\ &= 1 \text{ if we use the SSA.}\end{aligned}$$

That is, if the path reached time T, and $N_{TL} = n$, then the Chernoff algorithm was successfully applied n times. By definition,

$$\mathrm{P}(A) = \sum_{n=0}^{+\infty} \mathrm{P}\left(\{\exists k \in \mathbb{N} : t_k = T \,\middle|\, N_{TL} = n\}\right)\mathrm{P}\left(N_{TL} = n\right) \geq \mathrm{E}\left[(1-\delta)^{N_{TL}}\right].$$

Moreover, for small values of δ, using a second-order Taylor approximation for the function $(1-\delta)^{N_{TL}}$ and taking expectations, we obtain the following:

$$\mathrm{E}\left[(1-\delta)^{N_{TL}}\right] = 1 - \delta\mathrm{E}\left[N_{TL}\right] + \frac{\delta^2}{2}(\mathrm{E}\left[N_{TL}^2\right] - \mathrm{E}\left[N_{TL}\right]) + o(\delta^2).$$

Finally, we arrive at the desired path exit probability,

$$\mathrm{P}\left(A^c\right) \leq \delta\mathrm{E}\left[N_{TL}\right] - \frac{\delta^2}{2}(\mathrm{E}\left[N_{TL}^2\right] - \mathrm{E}\left[N_{TL}\right]) + o(\delta^2).$$

In practice, we use $\delta\mathrm{E}\left[N_{TL}\right]$ as an upper bound of $\mathrm{P}\left(A^c\right)$ since δ is very small and $\mathrm{Var}\left[N_{TL}\right]$ is moderate. In Appendix A, we prove that $\mathrm{E}\left[N_{TL}\right]$ is bounded for polynomial propensity functions and tends to zero when $\delta \to 0$.

Remark 3.3 (hybrid estimation of $\mathrm{E}\left[N_{SSA^*}\right]$: The expected number of steps of a pure SSA path). In the SSA algorithm, the expected time spent in the state $X(s)$, namely, $\Delta t|X(s)$, is an exponential random variable with intensity $a_0(X(s))$. Therefore, the quantity

$$\int_0^T a_0(X(s))ds = \int_0^T \frac{1}{\mathrm{E}\left[\Delta t|X(s)\right]}ds$$

is an approximation of the number of steps of an exact path, $(X(s))_{0\leq s\leq T}$. By sampling M hybrid paths, we have that the sample mean, $\mathcal{A}(\int_0^T a_0(\bar{X}(s))ds; M)$, defined by $\mathcal{A}(Y;M) := \frac{1}{M}\sum_{i=1}^M Y(\omega_i)$, is an estimator of $\mathrm{E}\left[N_{SSA^*}\right]$.

This allows us, for example, to approximate $\mathrm{Work}_{SSA}(TOL)$, i.e., the computational work that the SSA method requires to estimate $\mathrm{E}\left[g(X(T))\right]$ for a given tolerance (TOL). This remark is used below in Algorithm 5.

4. Error decomposition, estimation, and control. In this section, we define the computational global error, $\mathcal{E}$, and show how it can be naturally decomposed into three components: the discretization error, $\mathcal{E}_I$, and the exit error, $\mathcal{E}_E$, both coming from the tau-leap part of the hybrid method, and the Monte Carlo statistical error, $\mathcal{E}_S$. Next, we show how to model and control the global error, $\mathcal{E}$, giving upper bounds for each one of the three components. Finally, given a prescribed tolerance, TOL, we present a procedure for obtaining the parameters needed for estimating $\mathrm{E}\left[g(X(T))\right]$ by sampling hybrid paths. These parameters are the time mesh, $(t_k)_{k=0}^{K}(TOL)$, the one-step exit probability bound, $\delta(TOL)$, and the number of Monte Carlo samples, $M_{Hyb}(TOL)$.

4.1. Global error decomposition. As we already mentioned, the main goal of this work is to estimate accurately and efficiently the expected value $\mathrm{E}\left[g(X(T))\right]$, where $X : [0,T] \to \mathbb{Z}_+^d$ is a Markov pure jump process and $g : \mathbb{R}^d \to \mathbb{R}$ is a smooth observable of the process at final time T. We propose the following estimator:

$$\frac{1}{M}\sum_{m=1}^{M}\left(g(\bar{X}(T))\mathbf{1}_{\{A\}}\right)(\bar{\omega}_m), \tag{4.1}$$

where $\bar{X} : [0,T] \to \mathbb{Z}^d$ is the hybrid approximate process introduced in section 3.2, and $\bar{\omega} \in \bar{\Omega}$. The set $A \subset \bar{\Omega}$ was defined in section 3.3. We recall that $\mathbf{1}_{\{A\}}(\bar{\omega}_m) = 1$ if and only if the m-hybrid path did not exit $\mathbb{Z}_+^d$.

We define the computational global error, $\mathcal{E}$, as

$$\mathcal{E} := \mathrm{E}\left[g(X(T))\right] - \frac{1}{M}\sum_{m=1}^{M}\left(g(\bar{X}(T))\mathbf{1}_{\{A\}}\right)(\bar{\omega}_m). \tag{4.2}$$

We can split $\mathcal{E}$ into three parts:

$$\mathrm{E}\left[g(X(T))\right] - \frac{1}{M}\sum_{m=1}^{M}\left(g(\bar{X}(T))\mathbf{1}_{\{A\}}\right)(\bar{\omega}_m) = \underbrace{\mathrm{E}\left[\left(g(X(T)) - g(\bar{X}(T))\right)\mathbf{1}_{\{A\}}\right]}_{=:\mathcal{E}_I}$$

$$+ \underbrace{\mathrm{E}\left[g(X(T))\mathbf{1}_{\{A^c\}}\right]}_{=:\mathcal{E}_E} + \underbrace{\frac{1}{M}\sum_{m=1}^{M}\left(\mathrm{E}\left[g(\bar{X}(T))\mathbf{1}_{\{A\}}\right] - g(\bar{X}(T))\mathbf{1}_{\{A\}}\right)(\bar{\omega}_m)}_{=:\mathcal{E}_S}.$$

Here, $\mathcal{E}_I$ and $\mathcal{E}_S$ are the discretization and Monte Carlo statistical errors, respectively, and they are associated with the hybrid paths, $\bar{X}$ on A. $\mathcal{E}_E$ is the global exit error. We observe that the error term, $\mathcal{E}_E$, is defined as the expected value of $g(X(T))\mathbf{1}_{\{A^c\}}$, which is a random variable defined on $\Omega \times \bar{\Omega}$. More specifically, we set $\mathcal{E}_E$ such that

$$|\mathcal{E}_E| = \min_{P \in \mathcal{P}} |\mathcal{E}_E(P)|,$$

where $\mathcal{P}$ is the set of all probability measures on $\Omega \times \bar{\Omega}$. By choosing $P \in \mathcal{P}$ as the product probability measure, we have that $g(X(T))$ and $\mathbf{1}_{\{A^c\}}$ are independent random variables. As a consequence,

$$|\mathcal{E}_E| \leq |\mathrm{E}\left[g(X(T))\right]|\,\mathrm{P}\left(A^c\right).$$

An approximate upper bound, $\mathcal{B}$, for $|\mathrm{E}\left[g(X(T))\right]|$ could be obtained, for instance, as the 95% quantile of a bootstrap sample for $|\mathcal{A}(g(X(T));\cdot)|$. As we showed in section 3.3, $\mathrm{P}\left(A^c\right)$ can be approximated by $\delta\,\mathrm{E}\left[N_{TL}\right]$. Therefore, $\mathcal{B}\,\delta\,\mathcal{A}(N_{TL};\cdot)$ is an approximated upper bound for $|\mathcal{E}_E|$, where $\mathcal{A}(N_{TL};\cdot)$ is the estimator of $\mathrm{E}\left[N_{TL}\right]$.

The discretization error, $\mathcal{E}_I = \mathrm{E}\left[\left(g(X(T)) - g(\bar{X}(T))\right)\mathbf{1}_{\{A\}}\right]$, is actually the weak error associated with the hybrid paths in A. An efficient procedure for accurately estimating this quantity in the context of the tau-leap method is described in [16]. This procedure computes $\mathcal{E}_I(\bar{\omega})$ for every simulated hybrid path, $(\bar{X}(t_k,\bar{\omega}))_{k=0}^K$, as a weighted sum of local errors at the mesh times, $(t_k)_{k=0}^K$. The sequence of weights, $(\varphi_k(\bar{\omega}))_{k=1}^K$, considered in [16], is defined as the duals motivated by approximate variations of $g(\bar{X}(T))$ with respect to the initial data. According to this method, $\mathcal{E}_I$ is approximated by $\mathcal{A}(\mathcal{E}_I(\bar{\omega});\cdot)$. We adapt this method in Algorithm 7 for estimating the weak error in the hybrid context. A brief description follows. For each hybrid path, we compute backward the sequence of dual weights:

$$\begin{aligned}\varphi_K &= \nabla g(\bar{X}_K),\\ \varphi_k &= \left(Id + \tau_k \mathbb{J}_a^T(\bar{X}_k)\,\nu^T\right)\varphi_{k+1}, \quad k = K-1, K-2, \ldots, 1,\end{aligned}$$

where ∇ is the gradient operator and $\mathbb{J}_a(\bar{X}_k) = [\partial_i a_j(\bar{X}_k)]_{j,i}$ is the Jacobian matrix of the propensity function, a_j, for $j = 1,\ldots,J$ and $i = 1,\ldots,d$. Then, we have

$$\mathcal{E}_I(\bar{\omega}) = \sum_{k=1}^{K}\left(\frac{\tau_k}{2}\varphi_k \mathbf{1}_{\{TL\}}(k)\sum_{j=1}^{J}\nu_j^T\,\Delta a_{j,k}\right)(\bar{\omega}).$$

Here, $\bar{X}_k \equiv \bar{X}(t_k)$, $\tau_k = t_{k+1} - t_k$, $\Delta a_{j,k} = a_j(\bar{X}_{k+1}) - a_j(\bar{X}_k)$, $\mathbf{1}_{\{TL\}}(k) = 1$ if and only if, at time t_k, the tau-leap method was used and Id is the $d \times d$ identity matrix.

We model the Monte Carlo statistical error, $\mathcal{E}_S$, as a Gaussian random variable that has zero mean and variance $\mathrm{Var}\left[g(X(T))\right]/M$, which could be controlled by obtaining a rough estimate of $\mathrm{Var}\left[g(X(T))\right]$. The sample variance is denoted as $\mathcal{S}^2(Y;M) := \mathcal{A}(Y^2;M) - \mathcal{A}(Y;M)^2$. Therefore, $C_A\sqrt{\mathcal{S}^2(g(X(T));\cdot)/M}$ is used as an estimation of $\mathcal{E}_S$, where $C_A \geq 2$ is a desired confidence level.

4.2. Error estimation and control. Given a tolerance, TOL, we would like to have a procedure that determines whether we should use the SSA method or the hybrid one. This decision should be based on the expected computational work of both methods, and the procedure should provide, in any case, the necessary elements for computing the estimator. When the SSA method is chosen, the procedure should provide the number of simulations, $M_{SSA}(TOL)$. When the hybrid method is chosen, the procedure should provide not only the number of simulations, $M_{Hyb}(TOL)$, but also the time mesh, $(t_k)_{k=0}^K(TOL)$, and the one-step exit probability bound, $\delta(TOL)$. Let us describe such a decision procedure. The building block of a hybrid path is Algorithm 3, which adaptively determines whether to use an SSA or a tau-leap step. According to this algorithm, given the current state of the approximate process, x, there are two ways of taking an SSA step, depending on the logical conditions $K_1/a_0(x) > T_0 - t$ and $K_2(x,\delta)/a_0(x) > \tau_{Ch}$. The first way is when $K_1/a_0(x) > T_0 - t$ is true. In this case, we take an SSA step and avoid the computation of $\tau_{Ch}(x)$. The second is when $K_1/a_0(x) > T_0 - t$ is false and $K_2(x,\delta)/a_0(x) > \tau_{Ch}$ is true; but in this case, we have to compute $\tau_{Ch}(x)$. We consider one particular hybrid path, and we let $N_{SSA,K_1}(h,\delta)$ be the number of SSA steps such that $K_1/a_0(x) > T_0 - t$ is true. In the

same way, let $N_{SSA,K_2}(h,\delta)$ be the number of SSA steps such that $K_1/a_0(x) > T_0 - t$ is false and $K_2(x,\delta)/a_0(x) > \tau_{Ch}$ is true. Finally, let $N_{TL}(h,\delta)$ be the total number of tau-leap steps. We define $\Psi(h,\delta)$ as the expected work of a hybrid path, i.e.,

$$\Psi(h,\delta) = C_1 \mathrm{E}\left[N_{SSA,K_1}(h,\delta)\right] + C_2 \mathrm{E}\left[N_{SSA,K_2}(h,\delta)\right] + C_3 \mathrm{E}\left[N_{TL}(h,\delta)\right] + \sum_{j=1}^{J} \mathrm{E}\left[\int_{[0,T]} C_p(a_j(\bar{X}(s))\tau_{Ch}(\bar{X}(s),\delta))\mathbf{1}_{\{TL\}}(\bar{X}(s))ds\right].$$

Therefore, the expected computational work of the hybrid method is $M\Psi(h,\delta)$, where M is the total number of hybrid paths.

Given $TOL > 0$, we consider the problem

$$\begin{cases} \min_{M,h,\delta} M\Psi(h,\delta) \\ \text{s.t.} \\ \mathcal{E}_I + \mathcal{E}_E + \mathcal{E}_S \leq TOL. \end{cases} \tag{4.3}$$

In Algorithm 5, we propose an iterative method for obtaining an approximate solution to this problem.

A brief description of the ideas involved in this algorithm follows. Consider that a relative tolerance, $TOL > 0$, is given. By using Algorithm 6, we simulate a number of hybrid paths in a coarse mesh of size h_0, with sufficiently small δ (say, $\delta = 10^{-6}$), to obtain accurate estimates of $\mathrm{Var}\left[g(\bar{X}(T))\right]$ and $\mathcal{E}_I$. The total runtime of this procedure is recorded in the variable r.

Now, we estimate $\Psi(h_0,\delta)$ and, in particular, the error bound, $\mathcal{B}\,\delta\,\mathcal{A}(N_{TL};M_s)$, for the exit error, $\mathcal{E}_E$. It is desired that this error be of order $\mathcal{O}\left(TOL^2\right)$. We thus divide δ by a factor (e.g., 10) and re-estimate $\mathcal{B}\,\delta\,\mathcal{A}(N_{TL};M_s)$ until this condition is fulfilled. Then, we compute the discretization error, $\mathcal{E}_I$, and $\mathcal{S}^2(g(X(T);M_s)$.

For fixed $\delta > 0$ and $\epsilon > 0$, let us consider an auxiliary problem:

$$\begin{cases} \min_{M,h} M\Psi(h,\delta) \\ \text{s.t.} \\ \mathcal{E}_I(h,\delta) + C_A\sqrt{\mathcal{S}^2(g(X(T));M_s)/M} = \epsilon, \end{cases} \tag{4.4}$$

where $C_A \geq 2$.

Instead of solving (4.4), we proceed as follows. First, we fix $h = h_0$ and derive M_{aux} and ϵ_0 as functions of h_0 and δ:

$$M_{aux}(h_0,\delta) = \left(\frac{\partial_h \Psi(h_0,\delta)}{\Psi(h_0,\delta)} \cdot \frac{C_A\sqrt{\mathcal{S}^2(g(X(T));M_s)}}{2\partial_h \mathcal{E}_I(h_0,\delta)}\right)^2, \tag{4.5}$$

$$\text{and } \epsilon_0(h_0,\delta) = \mathcal{E}_I(h_0,\delta) + C_A\frac{\sqrt{\mathcal{S}^2(g(X(T));\cdot)}}{\sqrt{M_{aux}(h_0,\delta)}}.$$

If $\epsilon_0 < TOL - TOL^2$, we take the current values of h_0 and δ as solutions of our optimization problem (4.3). Otherwise, we divide the time mesh by a factor (e.g., 4, which is near the optimal value of the multilevel tau-leap) and proceed iteratively. Each time we refine the mesh or δ, we set the budget for the computational work, r_0, as $2{\cdot}r$, which is the current total computational work of the calibration algorithm (see the details in Algorithm 6). In this way, we can guarantee that the current computational work of the calibration is less than or equal to two times the computational work at the last refinement.

Algorithm 5. Calibration and error estimation. Inputs: The initial state, $X(0)$, the final time, T, the propensity functions, $(a_j)_{j=1}^J$, the stoichiometric vectors, $(\nu_j)_{j=1}^J$, the smooth observable, g, and $TOL > 0$. Outputs: (SSA, M_{SSA}) or (Hyb, M_{Hyb}, δ, $(t_k)_{k=0}^K$). Notes: The values C_A and C_1 are defined in section 3.1. For the sake of simplicity, we omit the arguments of the algorithms when there is no risk of confusion.

Set initial mesh $\{t_k\}_{k=0}^K$ (h_0 its diameter)
$\delta \leftarrow O(TOL^3)$
$r_0 \leftarrow \infty$
$(\hat{\Psi}, r, \mathcal{S}^2(g(\bar{X}(T)); \cdot), \mathcal{A}(\{g(\bar{X}(T)), N_{SSA^*}, \mathcal{E}_I, N_{TL}\}; \cdot)) \leftarrow$ Algorithm 6
$M_{SSA} \leftarrow C_A^2 \mathcal{S}^2(g(\bar{X}(T)); \cdot)/(TOL - TOL^2)^2$
$a \leftarrow -1$
$b \leftarrow \log(\hat{\Psi}) - a \log(h_0)$
fin $\leftarrow$ **false**
while not fin **and** $\hat{\Psi} < C_1 \mathcal{A}(N_{SSA^*}; \cdot)$ **do**
  while $|\mathcal{A}(g(\bar{X}(T)); \cdot)| \delta \mathcal{A}(N_{TL}; \cdot) > TOL^2$ **do**
    Refine δ
    $r_0 \leftarrow 2\, r$
    $(\hat{\Psi}, r, \mathcal{S}^2(g(\bar{X}(T)); \cdot), \mathcal{A}(\{g(\bar{X}(T)), N_{SSA^*}, \mathcal{E}_I, N_{TL}\}; \cdot)) \leftarrow$ Algorithm 6
    $M_{SSA} \leftarrow C_A^2 \mathcal{S}^2(g(\bar{X}(T)); \cdot)/(TOL - TOL^2)^2$
  end while
  Compute $\partial_h \mathcal{E}_I$ and $\partial_h \tilde{\Psi}(h; a, b)$
  Compute $M_{aux}(h_0; \delta)$ and ϵ; see (4.5)
  if $\epsilon < TOL - TOL^2$, **then**
    fin $\leftarrow$ **true**
    Compute M_{Hyb} and ϵ; see (4.6)
    if $M_{Hyb}\, \hat{\Psi} < M_{SSA}\, C_1 \mathcal{A}(N_{SSA^*}; \cdot)$, **then**
      return (Hyb, M_{Hyb}, δ, $\{t_k\}_{k=0}^K$)
    else
      return (SSA, M_{SSA})
    end if
  else
    Refine the mesh $\{t_k\}_{k=0}^K$, and set h_0
    $r_0 \leftarrow 2\, r$
    $(\hat{\Psi}, r, \mathcal{S}^2(g(\bar{X}(T)); \cdot), \mathcal{A}(\{g(\bar{X}(T)), N_{SSA^*}, \mathcal{E}_I, N_{TL}\}; \cdot)) \leftarrow$ Algorithm 6
    $M_{SSA} \leftarrow C_A^2 \mathcal{S}^2(g(\bar{X}(T)); \cdot)/(TOL - TOL^2)^2$
    Update a and b using a linear regression
  end if
end while

Once the previous process is finished, we can take advantage of the slack $TOL - TOL^2 - \mathcal{E}_I(h_0, \delta)$ for reducing the value of M_{aux} and obtain

$$M_{Hyb}(h_0, \delta) = \left(\frac{C_A \sqrt{\mathcal{S}^2(g(X(T)); M_s)}}{TOL - TOL^2 - \mathcal{E}_I(h_0, \delta)} \right)^2. \tag{4.6}$$

The estimation of $\frac{\partial_h \Psi(h_0, \delta)}{\Psi(h_0, \delta)}$ and $\partial_h \mathcal{E}_I(h_0, \delta)$ in (4.5) deserves some remarks.

First, note that $\frac{\partial_h \Psi(h_0, \delta)}{\Psi(h_0, \delta)} = \partial_h \log(\Psi(h_0, \delta))$. In a pure tau-leap regime, the number of steps is approximately inversely proportional to the size of the mesh. We

Algorithm 6. Auxiliary function for Algorithm 5. Inputs: Same as Algorithm 4, and constant r_0, used to control the total computational work of the algorithm (budget). Outputs: The estimated runtime of the hybrid path, $\hat{\Psi}$, the total accumulated runtime, r, an estimate of $\mathrm{Var}\,[g(X(T))]$, $\mathcal{S}^2(g(\bar{X}(T));\cdot)$, an estimate of $\mathrm{E}\,[g(X(T))]$, $\mathcal{A}(g(\bar{X}(T));\cdot)$, an estimate of $\mathrm{E}\,[\mathcal{E}_I]$, $\mathcal{A}(\mathcal{E}_I;\cdot)$, and an estimate of the expected number of steps needed by the SSA and tau-leap methods, $\mathcal{A}(N_{SSA^*};\cdot)$ and $\mathcal{A}(N_{TL};\cdot)$. Here, $\mathbf{1}_{\{TL\}}(k) = 1$ if and only if the decision at time t_k was tau-leap. Notes: The values C_1, C_2, and C_3 are defined in section 3.1. Set appropriate values for M_0 and CV_0. For the sake of simplicity, we omit the arguments of the algorithms when there is no risk of confusion.

$M \leftarrow M_0$, $cv \leftarrow \infty$, $r \leftarrow 0$, $M_f \leftarrow 0$
while $cv > CV_0$ **and** $r \leq r_0$ **do**
  for $m \leftarrow 1$ to M **do**
    $((\bar{X}(t_k))_{k=0}^K, N_{TL}, N_{SSA,K_1}, N_{SSA,K_2}) \leftarrow$ Algorithm 4
    if the path does not exit $\mathbb{Z}_+^d$, **then**
      $M_f \leftarrow M_f + 1$
      Compute $g(\bar{X}(T;\bar{\omega}_m))$
      $\mathcal{E}_I \leftarrow$ Algorithm 7
      Use Remark 3.3 for estimating $N_{SSA^*}(\bar{\omega}_m)$
      $C_{Poi}(\bar{\omega}_m) \leftarrow \sum_{j=1}^J \sum_{k=0}^K C_P(a_j(\bar{X}(t_k))(t_{k+1}-t_k))\mathbf{1}_{\{TL\}}(k)$
    end if
  end for
  Estimate the coefficients of variation cv_g and $cv_{\mathcal{E}_I}$ of the estimators of $\mathrm{Var}\,[g(X(T))]$ and $\mathrm{E}\,[\mathcal{E}_I]$, respectively.
  $cv \leftarrow \max\{cv_g, cv_{\mathcal{E}_I}\}$
  $\hat{\Psi} \leftarrow C_1\mathcal{A}(N_{SSA,K_1};M_f) + C_2\mathcal{A}(N_{SSA,K_2};M_f) + C_3\mathcal{A}(N_{TL};M_f) + \mathcal{A}(C_{Poi};M_f)$
  $r \leftarrow r + M_f\ \hat{\Psi}$
  $M \leftarrow 2M$
end while
return $(\hat{\Psi}, r, \mathcal{S}^2(g(\bar{X}(T));M_f), \mathcal{A}(\{g(\bar{X}(T)), \mathcal{E}_I, N_{SSA^*}, N_{TL}\};M_f))$

Algorithm 7. Computes the discretization error, $\mathcal{E}_I \equiv \mathcal{E}_I(\bar{\omega}_m)$. Inputs: $(\bar{X}(t_k))_{k=0}^K$. Here, $\mathbf{1}_{\{TL\}}(k) = 1$ if and only if the decision at time t_k was tau-leap, and Id is the $d \times d$ identity matrix Output: $\mathcal{E}_I(\bar{\omega}_m)$.

$\mathcal{E}_I \leftarrow 0$
Compute $\varphi_K \leftarrow \nabla g(\bar{X}(t_K))$
for $k \leftarrow K-1$ to 1 **do**
  $\Delta t_k \leftarrow t_{k+1} - t_k$
  Compute $\mathbb{J}_a(\bar{X}(t_k)) = [\partial_i a_j(\bar{X}(t_k))]_{j,i}$
  $\varphi_k \leftarrow \left(Id + \Delta t_k\, \mathbb{J}_a^T(\bar{X}(t_k))\,\nu^T\right)\varphi_{k+1}$
  $\Delta a_k \leftarrow a(\bar{X}(t_{k+1})) - a(\bar{X}(t_k))$
  $\mathcal{E}_I \leftarrow \mathcal{E}_I + \frac{\Delta t_k}{2}(\Delta a_k\, \mathbf{1}_{\{TL\}}(k)\,\nu^T)\,\varphi_k$
end for
return $\mathcal{E}_I$

therefore have $\mathrm{E}\,[N_{TL}(h)] = \mathcal{O}\left(h^{-1}\right)$. In a hybrid regime, we model $\mathrm{E}\,[\Psi(h,\delta)] = \mathcal{O}\left(h^a\right)$. Therefore, for large values of h, a plausible model for $\log(\Psi(h,\delta))$ is $a\log(h) +$

b. We denote it with $\tilde{\Psi}(h; a, b)$. See Algorithm 5 for details. An initial guess for a is -1.

For the estimation of $\partial_h \mathcal{E}_I(h_0, \delta)$, we simply take numerical derivatives when consecutive meshes are available as follows:

$$\partial_h \mathcal{E}_I(h_k, \delta) \approx -2/\mathcal{E}_I(h_{k-1}, \delta_{k-1}) \left(\mathcal{E}_I(h_k, \delta_k) - \mathcal{E}_I(h_{k-1}, \delta_{k-1})\right).$$

As an initial value, we can consider $\mathcal{E}_I(h_0, \delta)/h_0$.

When h is close to zero, $\Psi(h, \delta)$ is the expected work of an exact path, $C_1 \mathrm{E}\left[N_{SSA^*}\right]$ (see Remark 3.3). Therefore, if in any iteration $\Psi(h, \delta)$ is greater than $C_1 \mathcal{A}(N_{SSA^*}; \cdot)$, then we decide to use the SSA method.

5. Numerical examples. In this section, we present two examples to illustrate the performance of our proposed method.

5.1. A simple decay model. The classical radioactive decay model provides a simple and important example for the application of the hybrid method. This model has only one species and one reaction,

$$X \xrightarrow{c} \emptyset.$$

Its stoichiometric matrix, $\nu \in \mathbb{R}$, and the propensity function, $a : \mathbb{Z}_+ \to \mathbb{R}$, are given by

$$\nu = -1 \quad \text{and} \quad a(X) = c\, X.$$

Here, we choose $c = 1$ and $g(x) = x$. In this particularly simple example, we have the exact solution, namely, $\mathrm{E}\left[g(X(T))|X(t) = X_0\right] = X_0 \exp(-c(T-t))$.

In Figure 9, we show the behavior of the time step size of the Chernoff tau-leap method, τ_{Ch}, as a function of the one-step exit probability bound, δ. We compare τ_{Ch} with the expected value of the SSA step size, τ_{SSA}, in a log-log scale, for $x_0 \in \{5, 10, 15, 20\}$. We can see that τ_{Ch} goes to zero as δ goes to zero. For small values of δ, we have that $\tau_{SSA} = 1/a_0(x_0) = 1/x_0$ is larger than τ_{Ch}, and, therefore, the SSA method is chosen by the hybrid algorithm (Algorithm 3). The expected SSA step size, which is independent of δ, is shown with horizontal dotted lines starting from the right, until the intersection with the τ_{Ch} curve. For example, if $x_0 = 10$, τ_{Ch} is larger than the expected τ_{SSA} whenever $\delta > 0.0259$. These dotted lines show two regimes: as we mentioned, below the dotted line, we can say that the process is close to the boundary, but, when τ_{Ch} is larger than the expected τ_{SSA}, we can say that the process, X, is far from the boundary. In this regime, the Chernoff tau-leap method will be chosen by the hybrid algorithm.

Summarizing, in Figure 9, we can observe when the SSA method is preferred over the Chernoff tau-leap, either because we have very stringent probabilities of taking negative values yielding a small value for δ, or because the current state of the process (x_0) is relatively close to the boundary.

In Figure 10, we show τ_{Ch} as a function of x_0, using a log scale on the x-axis, for different values of δ. It is interesting to observe that the maximum value of τ_{Ch} is 1, even when the final time is $T = 2$. This is influenced by the propensity function and the value of c. For smaller values of c, the maximum increases. This figure shows that when x_0 is small, the values of τ_{Ch} decrease rapidly and become much smaller than τ_{SSA}. As we mentioned, to be close to or far from the boundary is a relative notion, and it must be seen according to the probability of exiting the lattice. For instance,

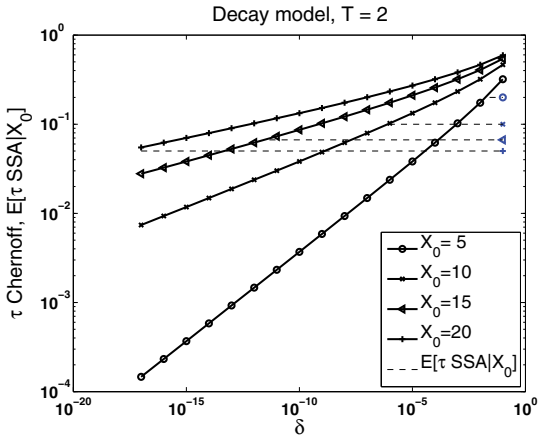

FIG. 9. *Chernoff step size, τ_{Ch}, as a function of δ, for $x_0 \in \{5, 10, 15, 20\}$, compared to* $\mathrm{E}[\tau_{SSA}|X_0]$. *For x_0 fixed, we can observe two regimes delimited by the dotted lines. Above the dotted line, the Chernoff tau-leap method is preferred, and below the line, the preferred method is the SSA.*

when $x_0 = 10$, we have that τ_{SSA} is approximately equal to τ_{Ch} for $\delta = 10^{-5}$, which is greater than the values of δ typically needed to achieve small tolerances. In the figure, we can see that, when x_0 tends to 1 (its minimum value), the expected τ_{SSA} tends to 1, and it is greater than τ_{Ch}. This shows that, as we are getting closer to the boundary by decreasing x_0, the τ_{Ch} becomes too small. On the other hand, when x_0 increases, the Chernoff tau-leap step size becomes larger, and, therefore, the tau-leap method is preferred.

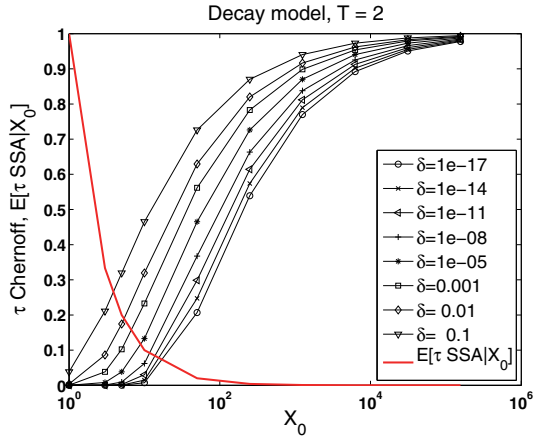

FIG. 10. *Chernoff step size, τ_{Ch}, as a function of x_0, for different values of δ. We observe two regimes: As x_0 decreases, the SSA method is preferred; as x_0 increases, the Chernoff tau-leap is preferred.*

Consider the initial condition $X_0 = 100$ and final time $T = 2$. We can observe that the process starts at a regime where the expected SSA step size is smaller than the Chernoff tau-leap, but after a certain time, it is the opposite. In Figure 11, we show 20 SSA paths and 20 hybrid paths.

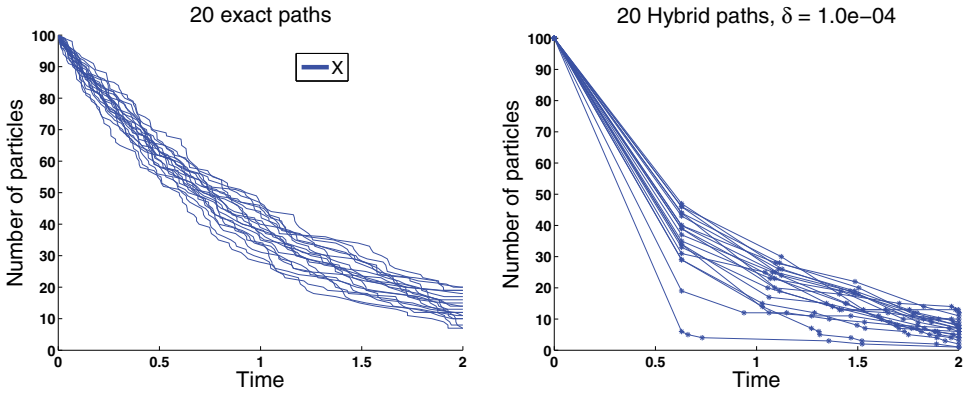

FIG. 11. *Left:* 20 *SSA paths for the simple decay model with* $X_0 = 100$ *and* $T = 2$. *Right:* 20 *hybrid trajectories, with linear interpolation between sample points (time steps). We can observe that, near the x-axis, the hybrid algorithm takes more SSA steps and fewer tau-leap steps.*

Now, we consider the initial condition, $X_0 = 10^5$, and the final time, $T = 0.5$. In this case, the process starts far from the boundary. First, we observe in Figure 12 that the SSA paths are very close to each other; that is, the variance of $g(X(T))$ is small. We analyze an ensemble of five independent realizations of the calibration algorithm (Algorithm 5), using different relative tolerances. In Figure 13, we show, in the left panel, the total predicted work (runtime) given by the calibration algorithm for both methods, the hybrid and the SSA, versus the estimated error bound, and its corresponding confidence intervals at the 95% level. The method chooses for the hybrid algorithm for the first three tolerances (largest) and the SSA for the two smaller ones. For the fourth tolerance, the method chooses the hybrid in 80% of the runs and SSA for the rest (see Table 2). Note that as TOL decreases, the hybrid path converges to the exact one because δ goes to 0 (see Appendix A). In the right panel, we show, for different tolerances, the actual work (runtime) of both methods, using a 12 core Intel GLNXA64 architecture and MATLAB version R2012b. The actual runtimes are in accordance with our predictions.

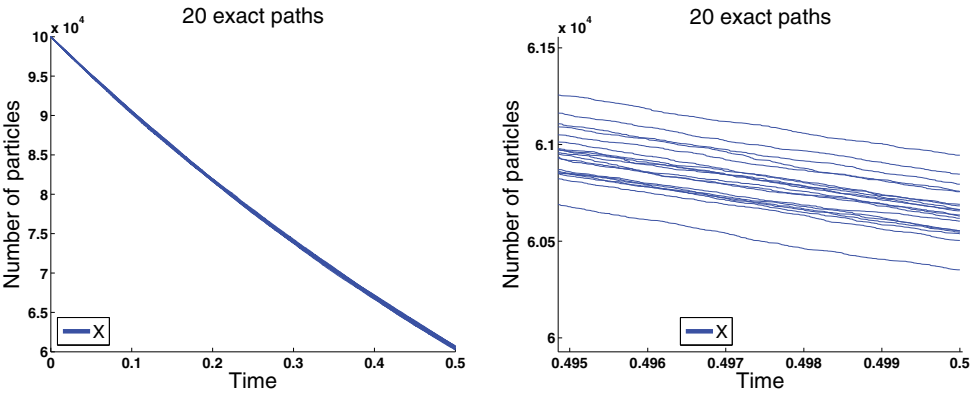

FIG. 12. *Left:* 20 *SSA paths for the simple decay model with* $X_0 = 10^5$ *and* $T = 0.5$. *Right: Details.*

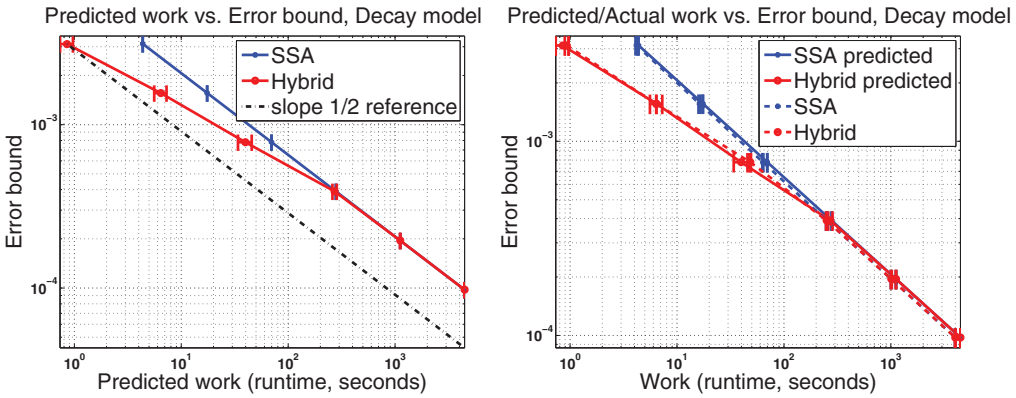

FIG. 13. *Left: Predicted work (runtime) versus the estimated error bound for* $X_0 = 10^5$ *and* $T = 0.5$. *The hybrid method is preferred over the SSA for the first three tolerances (larger ones). For the last two tolerances, the SSA is preferred. Therefore, in that case, the total predicted runtime is the same for the hybrid and SSA methods. Right: Predicted and actual work (runtime) versus the estimated error bound.*

TABLE 2

Details for an ensemble of five independent runs of Algorithm 5 *for the simple decay model with* $X_0 = 10^5$ *and* $T = 0.5$. *For example, the third row of the table tells us that we should run* $M = 64$ *hybrid paths, with a time mesh of size* $h = 4.9 \cdot 10^{-4}$ *and a one-step exit probability bound of* $\delta = 4.77 \cdot 10^{-10}$. *The work of the hybrid method is, on average,* 57% *of the work of the SSA (third column in the second part of the table). Here* $\hat{W}_{Hyb} := M_{Hyb}\,\hat{\Psi}$ *and* $\hat{W}_{SSA} := M_{SSA}\,C_1\,\mathcal{A}(N_{SSA^*};\cdot)$. *The fourth row shows, in the second and third columns, that in four runs of Algorithm* 5 *the SSA method is chosen, and in one run the hybrid method is chosen. In that case, we should simulate* $M_{SSA} = 180$ *SSA paths or* $M = 260$ *hybrid paths. Confidence intervals at* 95% *level are also provided.*

TOL	Method		$\delta(TOL)$	$h(TOL)$	$M(TOL)$
	SSA	HYB			
3.13e-03	0.00	1.00	3.05e-08	2.0e-03	5.0
1.56e-03	0.00	1.00	3.81e-09	9.8e-04	1.6e+01
7.81e-04	0.00	1.00	4.77e-10	4.9e-04	6.4e+01
3.91e-04	0.80	0.20	5.96e-11	2.4e-04	2.6e+02
1.95e-04	1.00	0.00	-	-	-
9.77e-05	1.00	0.00	-	-	-

TOL	M_{SSA}	$\frac{\hat{W}_{Hyb}}{\hat{W}_{SSA}}$	$\mathcal{A}(N_{TL};\cdot)$	$\mathcal{A}(N_{SSA^*};\cdot)$
3.13e-03	3.0	0.20 $\pm$0.03	2.6e+02	3.9e+04
1.56e-03	1.2e+01	0.37 $\pm$0.05	5.1e+02	3.9e+04
7.81e-04	4.6e+01	0.57 $\pm$0.09	1.0e+03	3.9e+04
3.91e-04	1.8e+02	0.97 $\pm$0.05	2.0e+03	3.9e+04
1.95e-04	7.2e+02	1.00	-	3.9e+04
9.77e-05	2.9e+03	1.00	-	3.9e+04

In the simple decay model, where an explicit expression for $\mathrm{E}\left[g(X(T))\right]$ is available, we can accurately compute the ratio between the estimated weak error and $\mathcal{E}_I$, which we call the efficiency index of the discretization error. We compute this quantity when the preferred method is the hybrid one. Recall that

$$\mathcal{E}_I = \mathrm{E}\left[\left(g(X(T)) - g(\bar{X}(T))\right)\mathbf{1}_{\{A\}}\right].$$

In order to compute that quantity, for each run of the calibration algorithm (Algorithm

5), we use a large sample in order to control the statistical error. The sample size is such that the statistical error in the estimation of $\mathcal{E}_I$ is ten times smaller than the prescribed tolerance. In Figure 14 we show the efficiency index of the discretization error, with confidence intervals at 95%. In the same figure we also show TOL versus the actual computational error. It can be seen that the prescribed tolerance is achieved with the required confidence of 95%, since C_A=1.96.

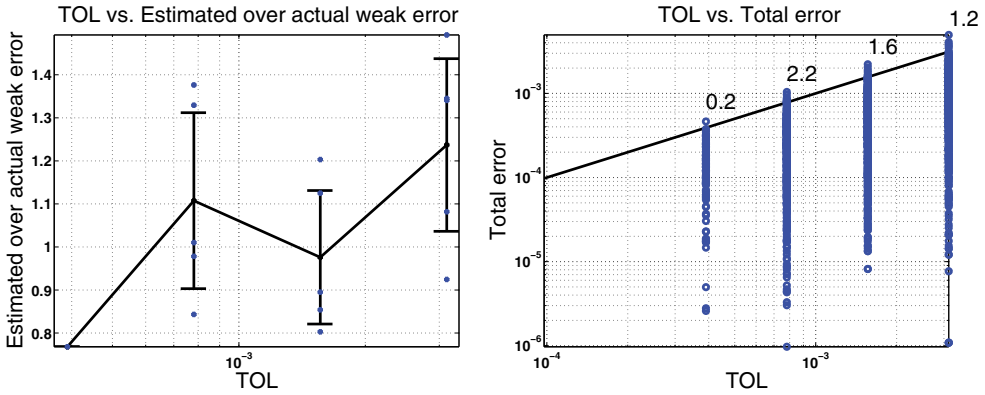

FIG. 14. *Left: Efficiency index for $\mathcal{E}_I$ and 95% confidence intervals. Right: TOL versus the actual computational error. The numbers above the straight line show the percentage of runs that had errors larger than the required tolerance. We observe that in all cases the computational error follows the imposed tolerance closely with the expected confidence of* 95%.

5.2. Gene transcription and translation [4]. This model has five reactions,

$$\begin{aligned} \emptyset &\xrightarrow{c_1} R, \quad R \xrightarrow{c_2} R + P, \\ 2P &\xrightarrow{c_3} D, \quad R \xrightarrow{c_4} \emptyset, \\ P &\xrightarrow{c_5} \emptyset, \end{aligned}$$

described by the stoichiometric matrix and the propensity function

$$\nu = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix} \quad \text{and} \quad a(X) = \begin{pmatrix} c_1 \\ c_2 R \\ c_3 P(P-1) \\ c_4 R \\ c_5 P \end{pmatrix},$$

respectively, where $X(t) = (R(t), P(t), D(t))$ and $c_1 = 25$, $c_2 = 10^3$, $c_3 = 0.001$, $c_4 = 0.1$, and $c_5 = 1$. In the simulations, the initial condition is $(0, 0, 0)$, and the final time is $T = 1$. The observable is given by $g(X) = X_3 = D$.

We can see that the abundance of the mRNA species, represented by R, is close to zero for $t \in [0, T]$. Therefore, we can interpret that the process is close to the boundary. However, according to Table 3, the calibration algorithm always chooses the hybrid method only in the first two tolerances. This happens because small tolerances induce small one-step exit probabilities, and, as a consequence, the Chernoff tau-leap steps are smaller than the expected SSA steps. This suggests that the reduced abundance of one of the species is not enough to ensure that the SSA method should be used. The tolerance also plays a role in this choice.

TABLE 3
Details for an ensemble of five independent runs of Algorithm 5 for the gene transcription and translation model. Details on how to read the table are provided in Table 2.

TOL	Method		$\delta(TOL)$	$h(TOL)$	$M(TOL)$
	SSA	HYB			
1.00e-01	0.00	1.00	8.0e-05 ±2e-05	2e-02 ±2e-03	66 ±3
5.00e-02	0.00	1.00	1.0e-05 ±2e-06	7e-03 ±7e-04	230 ±8
2.50e-02	0.40	0.60	1.1e-06 ±5e-07	3e-03 ±7e-04	840 ±70
1.25e-02	0.80	0.20	1.9e-07	2.0e-03	3e+03
6.25e-03	1.00	0.00	-	-	-
3.13e-03	1.00	0.00	-	-	-

TOL	M_{SSA}	$\frac{\hat{W}_{Hyb}}{\hat{W}_{SSA}}$	$\mathcal{A}(N_{TL};\cdot)$	$\mathcal{A}(N_{SSA^*};\cdot)$
1.00e-01	3.5e+01	0.39 ±0.04	7e+01 ±1e+01	1.8e+04
5.00e-02	1.4e+02	0.54 ±0.10	1.4e+02 ±2e+01	1.8e+04
2.50e-02	5.5e+02	0.88 ±0.10	3.2e+02 ±9e+01	1.7e+04
1.25e-02	2.2e+03	0.99 ±0.02	4.9e+02	1.8e+04
6.25e-03	8.8e+03	1.00	-	1.8e+04
3.13e-03	3.5e+04	1.00	-	1.8e+04

In Figure 15, we show an ensemble of five independent realizations of the calibration algorithm and the comparisons of its corresponding predicted and actual work. We can appreciate the robustness of the calibration procedure. We can also observe that the hybrid method converges to the SSA when the tolerance goes to zero.

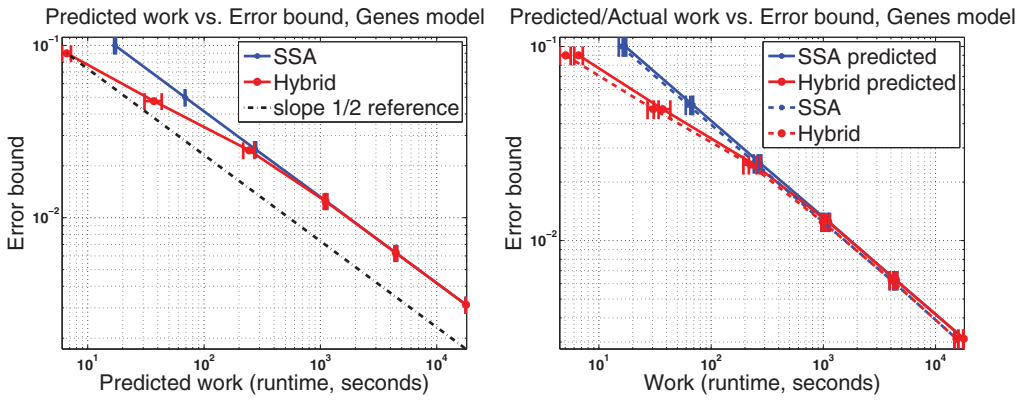

FIG. 15. *Left: Predicted work (runtime) versus the estimated error bound for the gene transcription and translation model. The hybrid method is preferred over the SSA for the first two tolerances (larger ones). For the last four tolerances, the SSA is preferred. Therefore, in the latter case, the total predicted runtime is the same for the hybrid and SSA methods. Right: Predicted and actual work (runtime) versus the estimated error bound.*

In Figure 16 we show the efficiency index of the discretization error, with confidence intervals at 95%. In the same figure we also show TOL versus the actual computational error. It can be seen that the prescribed tolerance is achieved with the required confidence of 95%, since C_A=1.96.

6. Conclusions. In this work, we addressed the problem of accurately estimating the expected value of an observable of a Markov pure jump process at a given

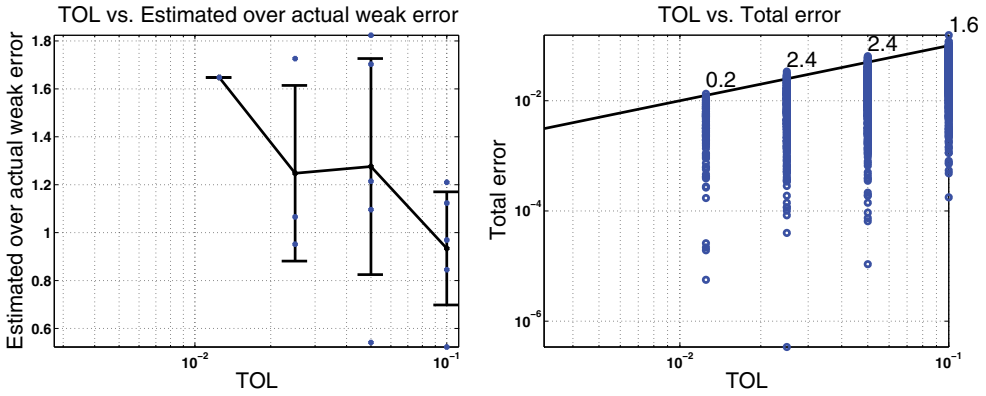

FIG. 16. *Left: Efficiency index for $\mathcal{E}_I$ and 95% confidence intervals. Right: TOL versus the actual computational error. The numbers above the straight line show the percentage of runs that had errors larger than the required tolerance. We observe that in all cases the computational error follows the imposed tolerance closely with the expected confidence of 95%.*

final time within a certain prescribed tolerance with high probability. Examples of settings where such estimation is necessary are message delivery times and connectivity in wireless communication networks and the number of infected agents in epidemic modeling of small populations. Although there are methods that simulate paths with the exact distribution of the process (e.g., Gillespie's SSA method), the computational work of generating the number of paths required to control the statistical error in a Monte Carlo setting turns out to be prohibitive for some real applications. On the other hand, Gillespie's approximate tau-leap method could produce, in certain cases, less expensive paths at the price of additionally introducing a time discretization error and an exit error.

In this work, we proposed a hybrid algorithm that, at each step, adaptively chooses to adopt the SSA method when the work of the tau-leap step becomes high. As a consequence, the expected work of a hybrid path remains bounded by the expected work of an SSA path and potentially can be much smaller.

The global exit error is related to the fact that, at any time, a tau-leap path can attain a nonphysical value. Preleap checks are common techniques for dealing with this problem by controlling the time step size. Here, we presented a novel nonasymptotic Chernoff-type hard bound to control large deviations of linear combinations of independent Poisson random variables. This bound allows us not only to obtain a preleap check for the tau-leap method, which neither changes the distribution of the increments nor requires any type of assumption regarding the reactions that can occur, but also to estimate and control the global exit error. To the best of our knowledge, there has been no previous attempt in the literature to estimate and control this type of error at the path level.

Another important contribution of this work is a calibration algorithm that can determine if it is suitable to use the hybrid algorithm for a given problem and also that can provide the associated simulation parameters. In the hybrid case, the calibration algorithm provides the one-step exit probability bound, the time mesh, and the number of hybrid paths that are needed for computing the mentioned expected value with low computational work.

It is worth mentioning that, by simulating hybrid paths, we obtained accurate

estimates of the average number of steps required by the SSA method to reach the final time. This is especially relevant in problems where the process visits regions of the state space where the total propensity is very high.

The numerical results that we obtain from different models show that the hybrid method proposed here is suitable for addressing problems in which one or more species has few individuals while the total propensity is high. In these types of problems (e.g., the gene transcription and translation model), the reaction-rate ODEs do not provide accurate approximation of the average behavior of the process, and the cost of the exact methods is also high. Moreover, we observed that generating Poisson random variables makes the computational work of a tau-leap step much higher than the work of an SSA step. This last argument, together with the advantages already discussed in terms of the time discretization error and the global exit error, adds more evidence in favor of avoiding the tau-leap whenever possible.

Our next step is to extend this hybrid algorithm to the multilevel Monte Carlo setting [12, 2]. We aim to obtain substantial computational work gains with respect to the traditional exact methods (SSA and modified next reaction method (MNRM)) and the single-level hybrid Chernoff tau-leap and to show that the computational complexity of this multilevel extension is of order $\mathcal{O}\left(TOL^{-2}\right)$.

Appendix A. An upper bound for the expected number of tau-leap steps of a hybrid trajectory, ($\mathbf{E}\left[\boldsymbol{N_{TL}(h,\delta)}\right]$). Let $\mathrm{E}\left[N_{TL}(h,\delta)\right]$ be the expected number of tau-leap steps of a hybrid path with a mesh of size h and a one-step exit probability bound, δ. Let $\{T_i\}$ be the sequence of grid points, t the current time, and $\bar{X}(t)$ the current state of the hybrid process, $\bar{X}$. Let $\tau_{Ch}(\bar{X}(t),\delta)$ be the Chernoff tau-leap step size computed using Algorithm 1, and, finally, let K_1 and $K_2 = K_2(\bar{X}(t),\tau_{Ch})$ be those introduced in section 3.1.

According to Algorithm 3, the logical conditions for choosing a tau-leap step are given by

$$\frac{K_1}{a_0(\bar{X}(t))} < T_i - t \quad \text{and} \quad \frac{K_2}{a_0(\bar{X}(t))} < \tau_{Ch}(\bar{X}(t),\delta).$$

The effective step size in this case is given by $\min\{\tau_{Ch}(\bar{X}(t),\delta), T_i - t\}$. Observe that $\{\frac{K_2}{a_0(\bar{X}(t))} < \tau_{Ch}(\bar{X}(t),\delta)\} \to \emptyset$ as $\delta \to 0$, because $K_2 \to \tilde{C}$ and $\tau_{Ch} \to 0$ (see section 3.1). By the definition of N_{TL}, we have that

$$\begin{aligned}
&\mathrm{E}\left[N_{TL}(h,\delta)\right] \\
&= \mathrm{E}\left[\sum_i \int_{T_{i-1}}^{T_i} \frac{\mathbf{1}\left\{\frac{K_1}{a_0(\bar{X}(t))} < T_i - t, \frac{K_2}{a_0(\bar{X}(t))} < \tau_{Ch}(\bar{X}(t),\delta)\right\}}{\min\{\tau_{Ch}(\bar{X}(t),\delta), T_i - t\}}\, dt\right] \\
&\le \mathrm{E}\left[\sum_i \int_{T_{i-1}}^{T_i} \frac{\mathbf{1}\left\{\frac{K_1}{a_0(\bar{X}(t))} < T_i - t, \frac{K_2}{a_0(\bar{X}(t))} < \tau_{Ch}(\bar{X}(t),\delta)\right\}}{\min\left\{\frac{K_1}{a_0(\bar{X}(t))}, \frac{K_2}{a_0(\bar{X}(t))}\right\}}\, dt\right] \\
&\le \mathrm{E}\left[\sum_i \int_{T_{i-1}}^{T_i} \frac{a_0(\bar{X}(t))\mathbf{1}\left\{\frac{K_2}{a_0(\bar{X}(t))} < \tau_{Ch}(\bar{X}(t),\delta)\right\}}{\min\{K_1, K_2\}}\, dt\right] \to 0 \quad \text{as } \delta \to 0.
\end{aligned}$$

It is also true that $\mathrm{E}\left[N_{TL}\right]$ has a polynomial bound since a_0 is polynomial and

$$\mathrm{E}\left[\sum_i \int_{T_{i-1}}^{T_i} \frac{a_0(\bar{X}(t))\mathbf{1}\left\{\frac{K_2}{a_0(\bar{X}(t))}<\tau_{Ch}(\bar{X}(t),\delta)\right\}}{\min\{K_1,K_2\}}\,dt\right] \leq \frac{\int_0^T \mathrm{E}\left[a_0(\bar{X}(t))\right]\,dt}{\min\{K_1,K_2\}}.$$

Finally, for the problems where $\max_{x\in\mathbb{Z}_+^d} a_0(x) < \infty$, we get the rough upper bound

$$\frac{\int_0^T \mathrm{E}\left[a_0(\bar{X}(t))\right]\,dt}{\min\{K_1,K_2\}} \leq \frac{T}{\min\{K_1,K_2\}} \max_{x\in\mathbb{Z}_+^d} a_0(x).$$

Observe that $\mathbb{Z}_+^d$ can be substituted by $\mathbb{Z}_+^d(x_0,T)\subset\mathbb{Z}_+^d$ defined by the subset of states that can be reached by a path starting from x_0 and evolving up to time T. Therefore, we have an upper bound for $\mathrm{E}\left[N_{TL}\right]$ that does not depend on δ. When the lattice is finite as in the exponential decay (Example 1.2), this bound is $c\,T\,x_0/K_2$.

Acknowledgments. The authors are members of the KAUST SRI Center for Uncertainty Quantification in the division of Computer, Electrical and Mathematical Sciences and Engineering at King Abdullah University of Science and Technology (KAUST). The authors would like to thank Jesper Karlsson for many interesting discussions in the early stages of this work.

REFERENCES

[1] J. Ahrens and U. Dieter, *Computer methods for sampling from gamma, beta, Poisson and bionomial distributions*, Computing, 12 (1974), pp. 223–246.
[2] D. F. Anderson and D. J. Higham, *Multilevel Monte Carlo for continuous time Markov chains, with applications in biochemical kinetics*, Multiscale Model. Simul., 10 (2012), pp. 146–179.
[3] D. F. Anderson, *Incorporating postleap checks in tau-leaping*, J. Chem. Phys., 128 (2008), 054103.
[4] D. F. Anderson and M. Koyama, *Weak error analysis of numerical methods for stochastic models of population processes*, Multiscale Model. Simul., 10 (2012), pp. 1493–1524.
[5] F. Brauer and C. Castillo-Chavez, *Mathematical Models in Population Biology and Epidemiology*, 2nd ed., Texts in Appl. Math. 40, Springer, New York, 2012.
[6] Y. Cao, D. Gillespie, and L. Petzold, *Avoiding negative populations in explicit Poisson tau-leaping*, J. Chem. Phys., 123 (2005), 054104.
[7] Y. Cao, D. T. Gillespie, and L. R. Petzold, *Efficient step size selection for the tau-leaping simulation method*, J. Chem. Phys., 124 (2006), 044109.
[8] A. Chatterjee, D. G. Vlachos, and M. A. Katsoulakis, *Binomial distribution based tau-leap accelerated stochastic simulation*, J. Chem. Phys., 122 (2005), 024112.
[9] H. Chernoff, *A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations*, Ann. Math. Statist., 23 (1952), pp. 493–507.
[10] S. N. Ethier and T. G. Kurtz, *Markov Processes: Characterization and Convergence*, 2nd ed., Wiley Series in Probability and Statistics, Wiley-Interscience, New York, 2005.
[11] F. Gebali, *Analysis of Computer and Communication Networks*, Springer, New York, 2008.
[12] M. Giles, *Multi-level Monte Carlo path simulation*, Oper. Res., 53 (2008), pp. 607–617.
[13] D. T. Gillespie, *A general method for numerically simulating the stochastic time evolution of coupled chemical reactions*, J. Comput. Phys., 22 (1976), pp. 403–434.
[14] D. T. Gillespie, *Approximate accelerated stochastic simulation of chemically reacting systems*, J. Chem. Phys., 115 (2001), pp. 1716–1733.
[15] D. T. Gillespie and L. R. Petzold, *Improved leap-size selection for accelerated stochastic simulation*, J. Chem. Phys., 119 (2003), pp. 8229–8234.
[16] J. Karlsson and R. Tempone, *Towards automatic global error control: Computable weak error expansion for the tau-leap method*, Monte Carlo Methods Appl., 17 (2011), pp. 233–278.

[17] B. Klar, *Bounds on tail probabilities of discrete distributions*, Probab. Engrg. Inform. Sci., 14 (2000), pp. 161–171.
[18] T. G. Kurtz, *The relationship between stochastic and deterministic models for chemical reactions*, J. Chem. Phys., 57 (1972), pp. 2976–2978.
[19] T. Li, *Analysis of explicit tau-leaping schemes for simulating chemically reacting systems*, Multiscale Model. Simul., 6 (2007), pp. 417–436.
[20] F. McMahon, *The Livermore Fortran Kernels: A Computer Test of the Numerical Performance Range*, Lawrence Livermore National Laboratory, Livermore, CA, 1986.
[21] V. Petrov, *Sums of Independent Random Variables*, Springer, New York, 1976.
[22] M. F. Pettigrew and H. Resat, *Multinomial tau-leaping method for stochastic kinetic simulations*, J. Chem. Phys., 126 (2007), 084101.
[23] M. Rathinam, L. R. Petzold, Y. Cao, and D. T. Gillespie, *Consistency and stability of tau-leaping schemes for chemical reaction systems*, Multiscale Model. Simul., 4 (2005), pp. 867–895.
[24] T. Tian and K. Burrage, *Binomial leap methods for simulating stochastic chemical kinetics*, J. Chem. Phys., 121 (2004), pp. 10356–10364.
[25] A. Voter, *Introduction to the kinetic Monte Carlo method*, in Radiation Effects in Solids, Springer, New York, 2007, pp. 1–23.