State and parameter estimation from exact partial state observation in stochastic reaction networks

Muruhan Rathinam\(^1\) and Mingkai Yu\(^2\)

\(^1\)Department of Mathematics and Statistics, University of Maryland Baltimore County
\(^2\)Department of Mathematics and Statistics, University of Maryland Baltimore County

October 12, 2020

Abstract

We consider chemical reaction networks modeled by a discrete state and continuous in time Markov process for the vector copy number of the species and provide a novel particle filter method for state and parameter estimation based on exact observation of some of the species in continuous time. The conditional probability distribution of the unobserved states is shown to satisfy a system of differential equations with jumps. We provide a method of simulating a process that is a proxy for the vector copy number of the unobserved species along with a weight. The resulting weighted Monte Carlo simulation is then used to compute the conditional probability distribution of the unobserved species. Our method is exact in a certain sense and we also show how our algorithm can be adapted for a Bayesian estimation of parameters and for the estimation of a past state value based on observations up to a future time.

1 Introduction

Chemical reaction networks occurring at the intracellular level often have some or all molecular species present in low copy numbers. As such, these are best modeled by a discrete state Markov process in continuous time, where the state \(Z(t) \in \mathbb{Z}^+_n\) of the process is the vector copy number of the molecular species at time \(t\). This model and the corresponding stochastic simulation algorithm introduced by Gillespie [17, 18] assumed a well-stirred system where spatial variations in molecular concentrations were considered negligible. Later on when the spatial variations were considered important, this model still proved useful in that its framework allowed for a spatial compartmental model where transport of molecules across compartments could be treated mathematically as reactions [6].

In this paper, we consider the problem of estimating the state \(Z(t)\) from exact and continuous observations of some of the states in time. In other words, suppose that the reaction network consists of \(n\) molecular species where the vector copy number of \(n_2\) of the molecular species may be observed exactly as a function of time over the interval \([0, t]\). Based on this observation, we are interested in estimating the vector copy number of the remaining \(n_1 = n - n_2\) species at time \(t\). If we denote the vector copy number of the observable species by \(Y(t) \in \mathbb{Z}^+_{n_2}\) and that of the rest of the (unobservable) species by \(X(t) \in \mathbb{Z}^+_{n_1}\), then our goal is to compute the conditional probability distribution

\[
\pi(t, x) = P\{X(t) = x \mid Y(s) = y(s), 0 \leq s \leq t\}
\]

for \(x \in \mathbb{Z}^+_{n_1}\).

Stochastic filtering methods provide a framework for addressing this type of problem, which is not restricted to reaction networks. The stochastic filtering methods usually involve generating recursive updates in time so that the additional computation required in going from the knowledge of conditional probability at a time instant \(t\) to a future time instant \(t + h\) only involves the observations made during \([t, t + h]\) and the conditional probability computed by time \(t\). The widely known instance of stochastic filtering is the Kalman filter [20] which was concerned with a Gaussian process with a linear evolution in discrete time. Later on, the Kalman-Bucy filter [25] was introduced in the continuous time setting of linear stochastic differential equations (SDEs). In
these linear and Gaussian settings the conditional probability distribution is also Gaussian, and hence its time evolution can be reduced to studying the time evolution of the mean and the covariance.

In general, one may not expect the conditional probability to be Gaussian and hence the methods are more complex. In the general nonlinear setting for SDEs there are several methods and one may find [4] as a general reference. In the setting of Markov processes in discrete time, one may find recursive filtering equations which describe the time evolution of the conditional probability distribution [13].

For discrete and finite state Markov processes in continuous time, a rigorous derivation of the evolution equation for the conditional probability distribution \( \pi(t, x) \) was provided in [10]. These evolution equations take the form of differential equations with jumps. These equations, known as the filtering equations, are analogous to Kolmogorov’s forward equations for the time evolution of the (unconditional) probability distribution \( P[Z(t) = z] \) of a continuous time and discrete state Markov process. In the case of reaction networks, Kolmogorov’s forward equations are known as the Chemical Master Equations (CME).

It is well known that when the number of states is infinite or very large, the solution of the CME is not practical and Monte Carlo simulation is the more efficient method of choice. The well known Gillespie algorithm [17, 20] and its variants [16] [1] are the methods of choice for exact realizations of sample paths. Likewise, when the state space for \( X(t) \) is very large or infinite, direct numerical solution of the filtering equations is not practical.

To our best knowledge, an “exact” Monte Carlo algorithm for the computation of \( \pi(t, x) \) based on an exact partial state trajectory \( y(t) \) observed in continuous time, is not available in the literature. We provide such an “exact” Monte Carlo algorithm (in the context of reaction networks) in this paper. What is meant by exact needs to be explained. Our algorithm simulates \( N_s \) copies of identically distributed pairs \((V, w)\) of processes where \( V \) has \( Z_\infty^+ \) as its state space and \( w \) is a weight such that \( \pi(t, x) \) is estimated by

\[
\hat{\pi}(t, x) = \frac{\sum_{i=1}^{N_s} 1(V^{(i)}(t)) \, w^{(i)}(t)}{\sum_{i=1}^{N_s} w^{(i)}(t)}. 
\]

The term exact refers to the fact that the numerator and hence the denominator are unbiased estimators of \( \rho(t, x) \) and \( \sum_x \rho(t, x) \) respectively, where \( \rho(t, x) \) satisfies the unnormalized filtering equations (8) and (9) that will be defined in Section 2 and are related to \( \pi(t, x) \) by

\[
\pi(t, x) = \frac{\rho(t, x)}{\sum_x \rho(t, x)}. 
\]

Due to the division, \( \hat{\pi}(t, x) \) is not an unbiased estimator of \( \pi(t, x) \). However, by virtue of law of large numbers, as \( N_s \to \infty \) we expect \( \hat{\pi}(t, x) \) to converge to \( \pi(t, x) \). In fact, all Monte Carlo methods of filtering in most other contexts (discrete time Markov chains, nonlinear SDEs etc.) also obtain the conditional probabilities by estimating an unnormalized version and then normalizing.

We describe some previous results for state and parameter estimation in the context of reaction networks. All these methods are for observations based on discrete time snapshots while our work addresses the case of continuous time observation. [2] considers Bayesian inference of parameters based on exact partial state observations in discrete time snapshots and proposes a Markov Chain Monte Carlo (MCMC) method. [21] [22] use the Chemical Langevin Equation (CLE) as the underlying model and propose MCMC methods for state and parameter estimation based on partial state observations corrupted by additive Gaussian noise in discrete time snapshots. [3] considers the case where the reaction network is approximated by the linear noise approximation. The observations are assumed to be linear combination of the states corrupted by an additive Gaussian noise with an unknown variance and they propose an extended Kalman-Bucy filter. They also consider pure delays in the system. The recent work in [12] considers the case where the reaction network may be approximated by an ODE model with jumps. A function of the state corrupted by an additive Gaussian noise is observed in discrete time snapshots. Rigorous results are derived to show that by solving the filtering problem for the reduced model based on observations of the original model, one can also obtain a good approximation of the estimation of the original model.

In this paper, we provide a weighted Monte Carlo method, that is a particle filter, for the computation of the conditional distribution of the unobserved species based on exact observation of some species in continuous time. In reality, perfect observations are not possible. However, it is often reasonable to assume that the observation noise is discrete in nature and hence the noise could be incorporated into the model via the introduction of additional species and reactions. For instance, during fluorescence intensity observations, photons produced
constitute a species and their production by a fluorescently tagged molecule constitutes a reaction channel. The assumption of continuous in time observation is less realistic. Nevertheless, if the sampling rate is high compared to reaction rates, then we expect this to provide a good approximation. Moreover, having an algorithm for the idealised baseline case of continuous in time observations is expected to provide valuable insights into how to analyse and develop techniques for the case of discrete time observations of a continuous time process.

The rest of the paper is organized as follows. In Section 2 we briefly review the basics of stochastic reaction networks and provide the filtering equations. We also introduce the useful unnormalized filtering equations. In Section 3 we provide a Monte Carlo filtering algorithm which generates a pair of processes \((V, w)\) where \(V\) has the same state space as \(X\) such that
\[
\pi(t, x) = \frac{\mathbb{E}[1_{\{x\}}(V(t)) \, w(t)]}{\mathbb{E}[w(t)]}.
\]
We also show how our algorithm can be easily adapted to do parameter estimation based on exact partial state observation in a Bayesian framework and also how to estimate the conditional probability of a past event based on the observations made until a later time. That is, we show how to compute
\[
P\{X(t) = x \mid Y(s) = y(s), 0 \leq s \leq T\},
\]
where \(x \in \mathbb{Z}^n_+\) and \(0 < t < T\), via an easy modification of our algorithm. In Section 4 we illustrate our algorithms via numerical examples and Section 5 makes some concluding remarks and discusses future work.

2 Stochastic reaction networks and filtering equations

We consider a stochastic reaction network with \(n\) molecular species and \(m\) reaction channels. The process \(Z(t) \in \mathbb{Z}^n_+\) stands for the copy number vector of the species at time \(t\). The dynamics of the reaction network are characterized by propensity functions \(a_j(z, c)\) where \(z \in \mathbb{Z}^n_+\) is the state and \(c \in \mathbb{R}^p\) is a vector of parameters, and also by the stoichiometric vectors \(\nu_j \in \mathbb{Z}^n\) for \(j = 1, \ldots, m\). The occurrence of a reaction event \(j\) at time \(t\) leads to a state change \(Z(t) = Z(t^-) + \nu_j\) and given \(Z(t) = z\), the probability that a reaction event \(j\) occurs during \((t, t + h]\) is \(a_j(z, c)h + o(h)\) as \(h \to 0+\). We shall adopt the convention that the process \(Z(t)\) is right continuous and assume \(Z(t)\) to be non-explosive, meaning that there are only finitely many reaction events during any finite time interval. For brevity, we suppress the dependence on parameters except when we are concerned with parameter estimation. It is well known\cite{17, 20, 18, 20} that the time evolution of the (unconditional) probability mass function
\[
p(t, z) = P\{Z(t) = z\}
\]
is given by the Kolmogorov’s forward equations also known as the chemical master equations (CME):
\[
p'(t, z) = \sum_{j=1}^{m} a_j(z - \nu_j) p(t, z - \nu_j) - \sum_{j=1}^{m} a_j(z) p(t, z).
\]

We consider the situation where we make exact (noiseless) observations of the copy number of the last \(n_2\) species in continuous time. We write \(Z(t) = (X(t), Y(t))\) where \(X(t) \in \mathbb{Z}^n_+\) is the unobserved component of the state and \(Y(t) \in \mathbb{Z}^n_+\) is the observed component of the state. Suppose we make a particular observation \(Y(s) = y(s)\) for \(0 \leq s \leq t\) for \(t \geq 0\). We are interested in computing the conditional probability
\[
\pi(t, x) = P\{X(t) = x \mid Y(s) = y(s), 0 \leq s \leq t\} \quad \forall x \in \mathbb{Z}^n_+.
\]

Let us denote by \(\nu'_1\) the first \(n_1\) components and by \(\nu''_j\) the last \(n_2\) components of \(\nu_j\) for \(j = 1, \ldots, m\) and define the subset \(\mathcal{O} \subset \{1, \ldots, m\}\) consisting of observable reaction channels, that is those that alter \(Y(t)\). Thus \(j \in \mathcal{O}\) if and only if \(\nu''_j \neq 0\). We denote by \(\mathcal{U}\), the complement of \(\mathcal{O}\). Thus \(\mathcal{U}\) consists of the unobservable reaction channels.

We shall denote by \(t_k, k = 1, 2, \ldots\) the successive jump times of \(y(t)\) and let \(t_0 = 0\). Let’s define \(a^{\mathcal{O}}(x, y)\) and \(a^{\mathcal{U}}(x, y)\) by
\[
a^{\mathcal{O}}(x, y) = \sum_{j \in \mathcal{O}} a_j(x, y), \quad a^{\mathcal{U}}(x, y) = \sum_{j \in \mathcal{U}} a_j(x, y),
\]
which are respectively the total propensity of the observable and the unobservable reactions. For each \( k \in \mathbb{N} \) define \( \mathcal{O}_k \) by

\[
\mathcal{O}_k = \{ j \in \mathcal{O} | \nu''_j = y(t_k) - y(t_{k-1}) \}.
\]

Thus \( \mathcal{O}_k \) is the subset of observable reactions that are consistent with the observed jump \( y(t_k) - y(t_{k-1}) = y(t_k) - y(t_{k-1}) \) at time \( t_k \).

The time evolution of \( \pi(t,x) \) follows a system of differential equations in between jump times \( t_k \) and at the jump times \( \pi(t, x) \) undergoes jumps. A rigorous derivation of the evolution equations for \( \pi(t, x) \) (for finite state Markov processes) is given in [10] and we provide a more intuitive derivation in the Appendix. We summarize these equations which we call the filtering equations here.

For \( k = 0, 1, \ldots \), and for \( t_k \leq t < t_{k+1} \), \( \pi \) satisfies the following system of differential equations:

\[
\pi'(t, x) = \sum_{j \in \mathcal{U}} \pi(t, x - \nu'_j) a_j(x - \nu'_j, y(t_k)) - \sum_{j \in \mathcal{U}} \pi(t, x) a_j(x, y(t_k)) - \pi(t, x) \left( a^\mathcal{O}(x, y(t_k)) - \sum_{\tilde{x}} a^\mathcal{O}(\tilde{x}, y(t_k)) \pi(t, \tilde{x}) \right) \quad \forall x \in \mathbb{Z}^n_+.
\]

(4)

and for \( k = 1, 2, \ldots \), and at times \( t_k \), \( \pi(t, x) \) jumps according to:

\[
\pi(t_k, x) = \sum_{j \in \mathcal{O}} a_j(x - \nu'_j, y(t_{k-1})) \pi(t_{k-1}, x - \nu'_j) \sum_{\tilde{x}} a^\mathcal{O}(\tilde{x}, y(t_{k-1})) \pi(t_{k-1}, \tilde{x}) \quad \forall x \in \mathbb{Z}^n_+.
\]

(5)

We note that \( \pi(t, x) \) is right continuous since by our convention the process \( Z(t) \) is right continuous and as a consequence the observed trajectory \( y(t) \) is right continuous as well.

Since we are interested in the case where the state space of the system of ODEs (4) is either an infinite or a very large finite subset of \( \mathbb{Z}^n_+ \), a direct solution of this equation is not practical. Instead our goal is a Monte Carlo simulation. Our Monte Carlo algorithm involves generating a trajectory \( V(t) \) with same state space \( \mathbb{Z}^n_+ \) as \( X(t) \), along with a weight trajectory \( w(t) \), such that at any time \( t \geq 0 \) and for \( x \in \mathbb{Z}^n_+ \)

\[
\pi(t, x) = \frac{\mathbb{E}[w(t) 1_{\{x\}}(V(t))]}{\mathbb{E}[w(t)]}.
\]

(6)

We note that, given a set \( A \), \( 1_A \) is the indicator function of the set. In practice, from a sample of \( N_s \) identically distributed trajectories \( V^{(i)} \) along with weights \( w^{(i)} \) for \( i = 1, \ldots, N_s \), \( \pi(t, x) \) is estimated by

\[
\hat{\pi}(t, x) = \frac{\sum_{i=1}^{N_s} 1_{\{x\}}(V^{(i)}(t)) w^{(i)}(t)}{\sum_{i=1}^{N_s} w^{(i)}(t)}.
\]

(7)

Since (4) is not linear, \( \pi \) cannot be interpreted as the probability mass function of some Markov process. However, we can define a related quantity \( \rho(t, x) \) which evolves according to a linear equation and can be related to a Markov process. We define \( \rho(t, x) \) to be a nonnegative function that satisfies the unnormalized filtering equations defined as below.

For \( k = 0, 1, \ldots \) on the interval \( t_k \leq t < t_{k+1} \), \( \rho(x, t) \) satisfies

\[
\rho'(t, x) = \sum_{j \in \mathcal{U}} \rho(t, x - \nu'_j) a_j(x - \nu'_j, y(t_k)) - \sum_{j \in \mathcal{U}} \rho(t, x) a_j(x, y(t_k)) - \rho(t, x) \left( a^\mathcal{O}(x, y(t_k)) - \sum_{\tilde{x}} a^\mathcal{O}(\tilde{x}, y(t_k)) \rho(t, \tilde{x}) \right) \quad \forall x \in \mathbb{Z}^n_+.
\]

(8)

For \( k = 1, 2, \ldots \) at jump times \( t_k \), \( \rho(t, x) \) jumps according to

\[
\rho(t_k, x) = \frac{1}{|\mathcal{O}_k|} \sum_{j \in \mathcal{O}_k} a_j(x - \nu'_j, y(t_{k-1})) \rho(t_{k-1}, x - \nu'_j) \quad x \in \mathbb{Z}^n_+.
\]

(9)

We note that \( |\mathcal{O}_k| \) is the number of reaction channels in the set \( \mathcal{O}_k \) and this is non-zero. Let \( \bar{\rho}(t) = \sum_{\tilde{x}} \rho(t, \tilde{x}) \). If we set \( \pi(t, x) = \rho(t, x)/\bar{\rho}(t) \), then (assuming that the derivative can be swapped with the sum) it is straightforward to verify that \( \pi(t, x) \) satisfies the filtering equations (4) and (5). In the next section we shall see that an appropriate stochastic interpretation of \( \rho(t, x) \) leads to the stochastic simulation algorithm to generate \( V \) and \( w \) such that

\[
\mathbb{E} \left[ 1_{\{x\}}(V(t)) w(t) \right] = \rho(t, x).
\]

(10)
3 Monte Carlo Algorithms

3.1 Weighted Monte Carlo and Resampling

In Section 3.2 we shall define a pair of processes \((V, w)\) where with state space \(Z^m_\infty \times [0, \infty)\) such that (10) holds where \(\rho\) satisfies the unnormalized filtering equations \(\ref{eq:8}\) and \(\ref{eq:9}\). The estimation of \(\pi(t, x)\) is accomplished by simulating \(N_s\) identically distributed copies of the pair of processes \((V, w)\) and estimating \(\pi(t, x)\) via \(\ref{eq:7}\).

We note that this type of procedure differs from the most common form of simulating a reaction network via \(N_s\) i.i.d. trajectories, say \(Z^{(i)}\) and then estimating \(E[\varphi(Z(t))]\), the expectation of a function \(\varphi\) of the state via the sample average

\[
\frac{1}{N_s} \sum_{i=1}^{N_s} \varphi(Z^{(i)}(t)).
\]

In this most familiar form of Monte Carlo simulation, all trajectories are weighted equally. In our situation, the Monte Carlo algorithm involves a weighted average. The weights carry information about the relative importance of trajectories of \(V\).

Corresponding to the sample \((V^{(i)}, w^{(i)})\) where \(i = 1, \ldots, N_s\), we may associate the (random) empirical measure (mass) \(M(t)\) at time \(t\) given by

\[
M(t) = \frac{1}{N_s} \sum_{i=1}^{N_s} w^{(i)}(t) \delta_{V^{(i)}(t)},
\]

where \(\delta_x\) stands for the Dirac mass or unit point mass concentrated at \(x \in Z^m_\infty\). If we choose \((V, w)\) to satisfy (10), then for each \(x \in Z^m_\infty\) the expected value of the empirical measure of the singleton \(\{x\}\) is \(\rho(t, x)\):

\[
E[M(t)(\{x\})] = \rho(t, x).
\]

The fact that the weights may grow (or decay) unevenly as time progresses leads to two different issues. The first issue is that the simulation may run into numerical problems where some weights become very large while others become very small, exceeding the finite precision of the computer. Especially, very large weights will become \(\text{Inf}\) in the finite precision representation of the computer, rendering computations impractical. The second issue is more fundamental and is unrelated to the finite precision nature of computations. The accuracy of the Monte Carlo estimate of \(\pi(t, x)\) depends on the variance of \(w(t)\) as well as the variance of \(1_{\{x\}}(V(t)) w(t)\) and also the covariance between these two terms. If the algorithm allows for a large variance in \(w(t)\), naturally this would have negative implications for the accuracy of the estimate.

In order to avoid these issues, one may \textit{resample} the empirical measure at certain time points to produce a new empirical measure that is still the sum of \(N_s\) Dirac masses but with equal weights. We shall employ one such algorithm described in \[4\]. This is given in Algorithm 4. We observe that if an empirical measure \(M(t)\) at time \(t\) is resampled to produce another empirical measure \(\tilde{M}(t)\), then we want the resampling procedure to satisfy the basic condition that

\[
E[\tilde{M}(t)(\{x\})] = E[M(t)(\{x\})], \quad \forall x \in Z^m_\infty.
\]

In order to maintain above requirement, the resampling procedure itself introduces extra randomness which in turn can result in loss of accuracy. Thus the frequency with which resampling is performed will be an important topic of investigation. We also note that the resampling procedure introduces dependence among the sample trajectories. Thus, while \((V^{(i)}, w^{(i)})\) for \(i = 1, \ldots, N_s\) are identically distributed, the collection is not necessarily independent.

Suppose that at time \(t_0 > 0\), we have simulated pairs \((V^{(i)}(t_0), w^{(i)}(t_0))\) for \(i = 1, \ldots, N_s\), and we resample to produce a new set of pairs \((\tilde{V}^{(i)}(t_0), \tilde{w}^{(i)}(t_0))\) for \(i = 1, \ldots, N_s\). Then we shall have that for all \(i\) and \(j\),

\[
E[1_{\{x\}}(\tilde{V}^{(j)}(t_0)) \tilde{w}^{(j)}(t_0)] = E[1_{\{x\}}(V^{(i)}(t_0)) w^{(i)}(t_0)],
\]

and \(\tilde{w}^{(j)}(t_0) = 1\) for \(j = 1, \ldots, N_s\). The consequent evolution of \((\tilde{V}^{(j)}(t), \tilde{w}^{(j)}(t))\) for \(j = 1, \ldots, N_s\) (which we shall rename \((V^{(j)}, w^{(j)})\)), will proceed in a conditionally independent fashion until the next resampling.

While above description is precise, it is most convenient to think of the algorithm as propagating \(N_s\) particles so that at time \(t\), the \(i\)th particle is situated at location \(V^{(i)}(t) \in Z^m_\infty\) and as has weight \(w^{(i)}(t) \geq 0\). When
resampling occurs at say, time $t_0$. Algorithm 1 considers the weight of each particle $i$ and assigns a random number $o^{(i)} \in \mathbb{Z}_+$ of offsprings to the particle. The number $o^{(i)}$ is chosen such that $\mathbb{E}[o^{(i)} | w^{(i)}(t_0)] = w^{(i)}(t_0)$ and subject to the condition that the total number of offsprings of all particles is $N_*$. Each particle $i$ is considered “dead” after it gives “birth” to $o^{(i)}$ offsprings each of which start at the same location $V^{(i)}(t_0)$ as its parent and with weight $w^{(i)}(t_0) = 1$. Subsequently all the offsprings evolve independently.

### 3.2 Processes $V$ and $w$

In this section we define a pair of processes $(V, w)$ where $V(t) \in \mathbb{Z}_{n+1}^n$ and $w(t) \geq 0$ as follows. We initialize $V(0)$ with the same distribution as that of the initial state $X(0) = x_0$ and set $w(0) = 1$. In between jump times, that is, for $t_k \leq t < t_{k+1}$ where $k = 0, 1, \ldots$, the process $V$ is evolved according to the underlying reaction network with all observable reactions removed. Thus only the copy number of the first $n_1$ species can change during $(t_k, t_{k+1})$. Moreover, for $t_k \leq t < t_{k+1}$ where $k = 0, 1, \ldots$, the weight process $w(t)$ is evolved according to

$$w(t) = w(t_k) \exp \left\{ - \int_{t_k}^t a^O(V(s), y(t_k)) \, ds \right\}.$$  

Equivalently, we note that $w(t)$ satisfies the ODE

$$w'(t) = w(t) a^O(V(t), y(t_k))$$

for $t_k \leq t < t_{k+1}$ (at $t = t_k$ the right hand derivative is considered). At jump times $t = t_k$ for $k = 1, 2, \ldots$ the process $w(t)$ jumps as follows. For $j \in \mathcal{O}_k$, set

$$w(t_k) = w(t_k-) a_j(V(t_k-), y(t_k-))$$

with probability $\frac{1}{|\mathcal{O}_k|}$.

Moreover, at jump times $t = t_k$ for $k = 1, 2, \ldots$ the process $V(t)$ jumps as follows. For $j \in \mathcal{O}_k$, set

$$V(t_k) = V(t_k-) + \nu_j'$$

with probability $\frac{1}{|\mathcal{O}_k|}$.

We note that if $a_j(V(t_k-), y(t_k-)) = 0$ for the chosen $j \in \mathcal{O}_k$, it may be possible that $V(t_k)$ is assigned an infeasible state, that is a state that has negative components. However, at the same time $w(t_k)$ would be zero and hence the value of $V(t_k)$ would not matter.

Our goal is to show that

$$\mathbb{E}[1_{\{x\}}(V(t)) \, w(t)] = \rho(t, x)$$

for all $t$ and $x$, where $\rho$ is defined by (3) and (5). The following two lemmas basically establish that.

**Lemma 1** Let $V$ and $w$ be defined as above, and let $\rho$ defined by

$$\rho(t, x) = \mathbb{E}[1_{\{x\}}(V(t)) \, w(t)].$$

*If $V$ is non-explosive then $\rho(t, x)$ satisfies (5) on $t_k \leq t < t_{k+1}$.***

**Proof** As a function of $t$, $1_{\{x\}}(V(t))$ is of bounded variation and piecewise constant on every interval bounded interval of time. Moreover, $w(t)$ is absolutely continuous in $t$. Thus we may write for $t_k \leq t < t_{k+1}$

$$1_{\{x\}}(V(t)) \, w(t) = 1_{\{x\}}(V(t_k)) \, w(t_k) + \int_{t_k}^t 1_{\{x\}}(V(s)) \, w'(s) ds$$

$$+ \sum_{t_k < s \leq t} w(s) \left( 1_{\{x\}}(V(s)) - 1_{\{x\}}(V(s-)) \right)$$

$$= 1_{\{x\}}(V(t_k)) \, w(t_k) - \int_{t_k}^t 1_{\{x\}}(V(s)) \, w(s) \, a^O(V(s), y(t_k)) \, ds$$

$$+ \sum_{j \in \mathcal{O}_k} \sum_{t_k < s \leq t} w(s) \left( 1_{\{x\}}(V(s-)) + \nu_j' - 1_{\{x\}}(V(s-)) \right) (R_j(s) - R_j(s-)).$$
where \( R_j \) is the process that counts the number of firings of reaction channel \( j \) during \((0,t]\). Since the stochastic intensity \( \mathbb{E}[R_j(t)] \) of \( R_j \) is given by \( a_j(V(t-)) \), taking expectations we obtain

\[
\rho(t,x) = \rho(t_k, x) + \sum_{j \in \mathcal{U}} \mathbb{E} \left[ \int_0^t (1 \{x\}(V(s) + \nu'_j) - 1 \{x\}(V(s))) \, w(s) \, a_j(V(s), y(t_k)) \, ds \right] \\
- \mathbb{E} \left[ \int_0^t 1 \{x\}(V(s)) \, w(s) \, a^O(V(s), y(t_k)) \, ds \right].
\]

Using Fubini and differentiating with respect to \( t \), We obtain that

\[
\rho'(t,x) = \sum_{j \in \mathcal{U}} \mathbb{E}[1 \{x\}(V(t) + \nu'_j) - 1 \{x\}(V(t))) \, a_j(V(t), y(t_k)) \, w(t)] \\
- \mathbb{E}[1 \{x\}(V(t)) \, w(t) \, a^O(V(t), y(t_k))]) \\
= \sum_{j \in \mathcal{U}} \mathbb{E}[1(x - \nu'_j)(V(t)) \, a_j(x - \nu'_j, y(t_k)) \, w(t)] - \sum_{j \in \mathcal{U}} \mathbb{E}[1 \{x\}(V(t))) \, a_j(x, \nu'_j) \, w(t)] \\
- \mathbb{E}[1 \{x\}(V(t)) \, w(t) \, a^O(x, y(t_k))]) \\
= \sum_{j \in \mathcal{U}} \rho(t, x - \nu'_j) \, a_j(x - \nu'_j, y(t_k)) - \sum_{j \in \mathcal{U}} \rho(t, x) \, a_j(x, y(t_k)) - \rho(t, x) \, a^O(t, x),
\]

which agrees with \( \mathbb{E} \).

Lemma 2 Let \( k \in \{1, 2, \ldots \} \) and suppose

\[
\mathbb{E}[1 \{x\}(V(t_k-)) \, w(t_k-)] = \rho(t_k-, x)
\]

for all \( x \). Then

\[
\mathbb{E}[1 \{x\}(V(t_k)) \, w(t_k)] = \rho(t_k, x)
\]

Proof It follows that the conditional expectation

\[
\mathbb{E}[w(t_k) \, 1 \{x\}(V(t_k)) \mid w(t_k-), \nu(t_k-)] = \frac{1}{|O_k|} \sum_{j \in O_k} w(t_k-) \, a_j(x - \nu'_j, y(t_k-1)) \, 1(x - \nu'_j)(V(t_k-)).
\]

Taking expectation, we obtain

\[
\mathbb{E}[w(t_k) \, 1 \{x\}(V(t_k))] = \frac{1}{|O_k|} \sum_{j \in O_k} \rho(t_k-, x - \nu'_j) \, a_j(x - \nu'_j, y(t_k-1)).
\]

The result follows from \( \mathbb{E} \).

From the above lemmas and mathematical induction, it follows that

\[
\mathbb{E}[1 \{x\}(V(t)) \, w(t)] = \rho(t, x)
\]

for all \( t \geq 0 \) and \( x \).

Algorithm 1 describes the overall filtering algorithm whereas Algorithm 2 describes the continuous evolution and Algorithm 3 describes the jumps at observed jump times \( t_k \). Algorithm 4 describes the resampling via offsprings. We observe that in Algorithm 1 resampling is only performed at the observed jump times \( t_k \) and not necessarily always. In Section 5.5 we numerically explore different resampling strategies. One extreme option is always to resample at \( t_k \), the other extreme is never to resample. The third option is to resample at \( t_k \) if either the number of zero weights among the \( N_x \) particles exceeds a predetermined number or if the ratio of the largest weight to the smallest non-zero weight among the particles exceeds a predetermined value. If resampling is not performed at \( t_k \), then the weights are rescaled so that the average weight is 1. This latter step prevents all weights from growing to be too large or too small. We also note that it is possible to consider resampling at time points which may not coincide with the observed jump times \( t_k \). However, since the only point in time where some weights may become zero is at the jump times \( t_k \), it makes sense to consider the jump times as the points in time to determine if a resampling step is required.
Algorithm 1 Overall scheme

1. **Input:** Jump times of \( Y(t_1, \ldots, t_{N_y}) \) and observed \( Y \) at jump times \( (y_1, \ldots, y_{N_y}) \). Initial distribution \( \mu_0 \) for \( X(0) = x_0 \), parameter value \( c \), final time \( T \), filter sample size \( N_s \).
2. Generate \( N_s \) i.i.d sample from \( \mu_0 \) and assign to \( V^{(1)}, \ldots, V^{(N_s)} \). Set \( w^{(i)} = 1 \) for \( i = 1, \ldots, N_s \).
3. \( k = 1, t = 0, y = y_0 \)
4. for \( k = 1 \) to \( N_k \) do
5. for \( i = 1 \) to \( N_s \) do
6. \((V^{(i)}, w^{(i)}) = \text{Continuous-evolution}(V^{(i)}, w^{(i)}, t, t_k, y, c)\)
7. \((V^{(i)}, w^{(i)}) = \text{Jump}(V^{(i)}, w^{(i)}, y, y_k, c)\)
8. end for
9. if resampling then \((V, w) = \text{Offsprings}(V, w)\)
10. else rescale \( w \) so that \( \sum_{i=1}^{N_s} w^{(i)} = N_s \)
11. end if
12. Set \( t = t_k, y = y_k \)
13. end for
14. for \( i = 1 \) to \( N_s \) do
15. \((V^{(i)}, w^{(i)}) = \text{Continuous-evolution}(V^{(i)}, w^{(i)}, t_{N_k}, T, y_{N_k}, c)\)
16. end for

3.3 Treating parameters as random variables

In general, the propensity functions \( a_j(x, y) \) depend on a vector \( c \) of parameters, thus \( a_j = a_j(x, y, c) \) where \( c \in \mathbb{R}^p \). Here we consider a Bayesian framework for inferring the parameters \( c \) from the observation \( Y(s) = y(s) \) for \( 0 \leq s \leq t \). This involves treating \( c \) as a random variable, or rather a stochastic process \( C(t) \) which remains constant in time \( t \geq 0 \). Thus, we may “absorb” \( C(t) \) into \( X(t) \), thus expanding the unobserved components of the state by \( p \) extra dimensions. The Bayesian framework involves starting with a prior distribution \( \mu \) on the parameter space \( \mathbb{R}^p \). Then we compute the posterior distribution probability mass/density function \( \pi(t, x, c) \) which is characterized by

\[ P\{X(t) = x, C(t) \in A | Y(s) = y(s), 0 \leq s \leq t\} = \int_{c \in A} \pi(t, x, c) \, dc. \]

Since the parameters are distributed continuously, the filtering equations (14) and (15) need to be modified as follows. For \( t_k \leq t < t_{k+1} \), \( \pi \) satisfies the following system of differential equations:

\[
\begin{align*}
\pi'(t, x, c) &= \sum_{j \in U} \pi(t, x - v_j, c) a_j(x - v_j, y(t_k), c) - \sum_{j \in U} \pi(t, x, c) a_j(x, y(t_k), c) \\
&\quad - \pi(t, x, c) \left( a^O(x, y(t_k), c) - \int_{\bar{c}} \sum_x a^O(\bar{x}, y(t_k), \bar{c}) \pi(t, \bar{x}, \bar{c}) d\bar{c} \right),
\end{align*}
\]

which holds for all \( x \in \mathbb{Z}^n_+ \) and \( c \in \mathbb{R}^p \). At times \( t_k \), \( \pi(t, x, c) \) jumps according to:

\[
\pi(t_k, x, c) = \sum_{l \in \mathcal{O}_x} \frac{a_l(x - v'_l, y(t_{k-1}), c) \pi(t_{k-1}, x - v'_l, c)}{\sum_{l \in \mathcal{O}_x} \sum_{l \in \mathcal{O}_k} a_l(\bar{x}, y(t_{k-1}), \bar{c}) \pi(t_{k-1}, \bar{x}, \bar{c}) d\bar{c}},
\]

which holds for all \( x \in \mathbb{Z}^n_+ \) and \( c \in \mathbb{R}^p \).

Algorithm 5 describes the Monte Carlo algorithm to generate a weighted sample from the posterior distribution for the parameter. Having computed \( C^{(i)} \) and \( w^{(i)} \) for \( i = 1, \ldots, N_s \), we may estimate the posterior mean \( \bar{C} \) and standard deviation \( \sigma_C \) by

\[
\bar{C} = \frac{\sum_{i=1}^{N_s} w^{(i)} C^{(i)}}{\sum_{i=1}^{N_s} w^{(i)}}, \tag{20}
\]

and

\[
\sigma_C^2 = \frac{\sum_{i=1}^{N_s} w^{(i)} (C^{(i)} - \bar{C})^2}{\sum_{i=1}^{N_s} w^{(i)}}. \tag{21}
\]

8
Algorithm 2 Continuous evolution

1. function CONTINUOUS-evolution(\(V_0, w_0, t_0, t_f, y\))
2. Assume unobservable reactions are numbered 1, 2, \(\ldots\), \(m_u\)
3. Assume observable reactions are numbered \(m_u + 1, \ldots, m\)
4. Set \(V = V_0\), \(w = w_0\), \(t = t_0\)
5. while \(t < t_f\) do
6. for \(j = 1\) to \(m_u\) do
7. Set \(\lambda_j = a_j(V, y)\)
8. end for
9. Set \(\lambda_0 = \sum_{j=m_u+1}^{m} \lambda_j\)
10. Set \(\lambda^{\nu} = \sum_{j=1}^{m_u} \lambda_j\)
11. Generate \(\tau \sim \text{Exponential}(\lambda^{\nu})\)
12. if \(t + \tau < t_f\) then
13. Generate \(u \sim \text{Uniform}[0,1]\)
14. Find \(j \in \{1, \ldots, m_u\}\) such that \(\sum_{l=1}^{j-1} \lambda_l < u \lambda^{\nu} \leq \sum_{l=1}^{j} \lambda_l\)
15. Set \(V = V + \nu_j\)
16. end if
17. Set \(t_n = \min\{t + \tau, t_f\}\)
18. Set \(w = w \times \exp\{- (t_n - t) \lambda^{\nu} \}\)
19. Set \(t = t_n\)
20. end while
21. return \((V, w)\)
22. end function

3.4 Estimating past state

Besides the real-time update of the conditional probability mass function \(\pi(t, x) = P\{X(t) = x \mid Y(s) = y(s), 0 \leq s \leq t\}\), we would like to look back and consider estimation of the state at time \(t\) with the observation made until a later time \(T\). A special case is when the initial state of the system is not known precisely, but rather a prior distribution is the best of our knowledge.

To tackle the problem of finding

\[
P\{X(t_0) = x \mid Y(s) = y(s), 0 \leq s \leq T\}
\]

with \(0 \leq t_0 \leq T\), we expand the state of the underlying process as \((X, \tilde{X}, Y)\) where \(\tilde{X}\) is simply a copy of \(X\) until time \(t_0\) and there after \(\tilde{X}(t)\) remains fixed at the value \(\tilde{X}(t_0)\).

The process \((X, \tilde{X}, Y)\) with state space \(\mathbb{Z}_+^{n_1} \times \mathbb{Z}_+^{n_1} \times \mathbb{Z}_+^{n_2}\) will be a piecewise time homogeneous Markov process in the sense that on the interval \([0, t_0]\) it will evolve according to a certain reaction network and during \([t_0, T]\) it will evolve according to a different reaction network. During \([0, t_0]\) whenever \(X\) jumps by \(\nu_j\), \(\tilde{X}\) also jumps by \(\tilde{\nu}_j = \nu_j\). However, during \([t_0, T]\) the stoichiometric vectors \(\nu_j\) corresponding to \(\tilde{X}\) are all zero indicating no change in state. Based on this, the original filtering equations are still valid for this extended process. We are simply estimating

\[
P\{\tilde{X}(T) = x \mid Y(s) = y(s), 0 \leq s \leq T\}
\]

which equals

\[
P\{X(t_0) = x \mid Y(s) = y(s), 0 \leq s \leq T\}
\]

since \(\tilde{X}(T) = X(t_0)\). Algorithm 6 describes this.

4 Estimation error

In this section, we discuss two notions of error in state or parameter estimation. The first is a measure of the error in the estimated conditional distribution and the true conditional distribution while the second is the \(L^2\) error in estimating the state or a parameter.
where $Y$ is the state or a parameter, one is frequently interested in obtaining a point estimate. The ideal point estimate of the distribution we introduce the mean total variation error (MTVE):

$$\text{MTVE} = \mathbb{E} \left( \sum_{x \in \mathbb{Z}_+^1} |\hat{\pi}(t, x, Y_{[0,t]}) - \pi(t, x, Y_{[0,t]})| \right). \tag{22}$$

We also introduce the conditional mean total variation error (CMTVE):

$$\text{CMTVE} = \mathbb{E} \left( \sum_{x \in \mathbb{Z}_+^1} |\hat{\pi}(t, x, Y_{[0,t]}) - \pi(t, x, Y_{[0,t]})| \bigg| \mathcal{Y}_t \right), \tag{23}$$

where $\mathcal{Y}_t$ is the $\sigma$-algebra generated by the observed process $Y$ up to time $t$. The reader unfamiliar with $\sigma$-algebras may regard this as condition on the trajectory of $Y$ up to time $t$. Thus, CMTVE is a function of the observed trajectory. If we are estimating conditional density of a parameter as opposed to state, the summations in the above definitions need to be replaced by an integrals.

On the other hand, instead of attempting to describe the entire conditional distribution for the (unobserved) state or a parameter, one is frequently interested in obtaining a point estimate. The ideal point estimate of the unobserved state $X(t)$ is the conditional expectation $\mathbb{E}[X(t) \mid \mathcal{Y}_t]$. In practice one estimates the latter via the estimator $\mathcal{E}(t)$ defined by

$$\mathcal{E}(t) = \frac{\sum_{i=1}^{N_t} V(i)(t) w(i)(t)}{\sum_{i=1}^{N_t} w(i)(t)}. \tag{24}$$

Thus the estimation error is given by $e(t) = \mathcal{E}(t) - X(t)$. The quantities of interest are, the conditional bias $\mathbb{E}[e(t) \mid \mathcal{Y}_t]$, the bias $\mathbb{E}[e(t)]$, the conditional $L^2$ error $\mathbb{E}[e^2(t) \mid \mathcal{Y}_t]^{1/2}$ and the $L^2$ error $\mathbb{E}[e^2(t)]^{1/2}$.

It is instructive to split $e(t)$ as

$$e(t) = (\mathcal{E}(t) - \mathbb{E}[X(t) \mid \mathcal{Y}_t]) + (\mathbb{E}[X(t) \mid \mathcal{Y}_t] - X(t)).$$

We make the important observation that $\mathcal{E}(t)$ and $X(t)$ are independent when conditioned on $\mathcal{Y}_t$. Hence

$$\mathbb{E}\left\{ \left( \mathcal{E}(t) - \mathbb{E}[X(t) \mid \mathcal{Y}_t] \right) \left( \mathbb{E}[X(t) \mid \mathcal{Y}_t] - X(t) \right) \bigg| \mathcal{Y}_t \right\} = \mathbb{E}\left\{ \left( \mathcal{E}(t) - \mathbb{E}[X(t) \mid \mathcal{Y}_t] \right) \bigg| \mathcal{Y}_t \right\} \mathbb{E}\left\{ \left( \mathbb{E}[X(t) \mid \mathcal{Y}_t] - X(t) \right) \bigg| \mathcal{Y}_t \right\} = 0.$$
Hence we may expand
\[
E[e^2(t) \mid \mathcal{Y}_i] = E \left[ (E(t) - E[X(t) \mid \mathcal{Y}_i])^2 \mid \mathcal{Y}_i \right] + E \left[ (E[X(t) \mid \mathcal{Y}_i] - X(t))^2 \mid \mathcal{Y}_i \right] \\
= E \left[ (E(t) - E[X(t) \mid \mathcal{Y}_i])^2 \mid \mathcal{Y}_i \right] + \text{Var}[X(t) \mid \mathcal{Y}_i] \tag{25}
\]
We observe that the second term depends only on the filtering problem and not on the filtering algorithm, while the first term depends on the filtering algorithm. We expect as the filter sample size \( N_s \) approaches infinity the first term to approach zero in some sense. Taking expectation on (25) we obtain that
\[
E[e^2(t)] = E \left[ (E(t) - E[X(t) \mid \mathcal{Y}_i])^2 \right] + E \left[ \text{Var}[X(t) \mid \mathcal{Y}_i] \right], \tag{26}
\]
where as observed earlier, the second term depends only on the filtering problem and provides a lower bound on the \( L^2 \) error.

We observe that for any given observation trajectory, the conditional error \( E[e^2(t) \mid \mathcal{Y}_i] \) may be estimated as follows assuming that we know the true state \( X(t) \) corresponding the observation. We simulate \( N_r \) independent runs of the overall filter (each time with \( N_s \) samples) to obtain estimates \( \mathcal{E}_1(t), \ldots, \mathcal{E}_{N_r}(t) \). Then estimate \( E[e^2(t) \mid \mathcal{Y}_i] \) as
\[
E[e^2(t) \mid \mathcal{Y}_i] \approx \frac{1}{N_r} \sum_{j=1}^{N_r} \mathcal{E}_j(t) - X(t))^2.
\]
On the other hand \( E[e^2(t)] \) may be estimated as follows. We simulate the original system via the Gillespie algorithm \( N_r \) independent times to obtain \((X^{(1)}, Y^{(1)}), \ldots, (X^{(N_r)}, Y^{(N_r)})\). For each such simulation \( j \), we run
where $\lambda$ the overall filter algorithm to record $E$ Algorithm 5 Overall scheme for Bayesian inference

1. **Input:** Jump times of $Y$ ($t_1, \ldots, t_{N_k}$) and observed $Y$ at jump times ($y_1, \ldots, y_{N_k}$). Initial distribution $\mu_0$ for $X(0) = x_0$, prior distribution $\mu$ for parameter $c$, final time $T$, filter sample size $N_s$.
2. Generate a sample $x_0$ from $\mu_0$ and a sample $c$ from $\mu$. Set $V(i) = x_0$, $w(i) = 1$, $C(i) = c$ for $i = 1$ to $N_s$.
3. Set $k = 1$, $t = 0$, $y = y_0$.
4. for $k = 1$ to $N_k$ do
5. for $i = 1$ to $N_s$ do
6. $(V(i), w(i)) = \text{Continuous-evolution}(V(i), w(i), t, t_k, y, C(i))$
7. $(V(i), w(i)) = \text{Jump}(V(i), w(i), y, C(i))$
8. end for
9. Set $U(i) = (V(i), C(i))$ for $i = 1$ to $N_s$.
10. if resampling then $(U, w) = \text{Offsprings}(U, w)$
11. else rescale $w$ so that $\sum_{i=1}^{N_s} w(i) = N_s$
12. end if
13. Set $(V(i), C(i)) = U(i)$ for $i = 1$ to $N_s$.
14. Set $t = t_k$, $y = y_k$.
15. end for
16. for $i = 1$ to $N_s$ do
17. $(V(i), w(i)) = \text{Continuous-evolution}(V(i), w(i), t_{N_k}, T, y_{N_k}, C(i))$
18. end for

the overall filter algorithm to record $E_j(t)$. Then estimate $E[e^2(t)]$ as

$$E[e^2(t)] \approx \frac{1}{N_s} \sum_{j=1}^{N_s} |E_j(t) - X^{(j)}(t)|^2.$$ 

5 Numerical Examples

In this section we illustrate the filtering algorithms via examples. In all examples, observations were made by simulation of the underlying reaction network using the Gillespie algorithm [17] to obtain one or more independent samples of $(X, Y)$, the unobserved and observed trajectories. Then the filtering algorithms were applied to the observed trajectories.

5.1 A linear propensity example

We consider the simple reaction network

$$S \xrightarrow{c_1} S + A,$$
$$\emptyset \xrightarrow{c_2} S,$$
$$S \xrightarrow{c_3} \emptyset,$$  (27)

consisting of two species and three reaction channels and assume mass action form of propensities. Thus the propensities are given by $a_1(z) = c_1 z_2$, $a_2(z) = c_2$, $a_3(z) = c_3 z_2$.

First, consider the case where the copy number of species $S$ is observed exactly while the species $A$ is unobserved. Thus, $Z(t) = (X(t), Y(t)) = (#A(t), #S(t))$. In this situation, since the second two reaction channels alone determine the copy number of $S$, the conditional probability density function $\pi(t, x)$ could be computed exactly as

$$\pi(t, x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

where $\lambda = \int_0^t c_1 y(t) dt$. With initial state $#A = 0$ and $#S = 5$, parameter values $c = (1, 5, 1)$, and filter sample size $N_s = 10,000$, we ran Algorithm 1 and Figure 1 shows the comparison between the conditional distribution computed by the filter with the exact conditional distribution. In order to estimate the CMTVE defined by (23), with same initial state, parameter values, final time $T$, filter sample size $N_s = 10,000$, and the same observation
formed a Bayesian estimation of parameter $c$ distribution (prior distribution on $\theta$ to compute $\bar{\theta}$). Not having slope 1 is due to this as well as due to finite sample size of $N$. We randomly generated a sample of $N$ independent realizations of the system $X(t)$, and applied the filtering algorithm with filter sample size $N_f$ to get an estimate $\hat{E}(t)$ for each of the realizations. Figure 2 shows a scatter plot of the values of $\hat{E}(T)$ against $X(T)$. We note that $\hat{E}(T)$ is a biased estimator whose bias is expected to approach zero as $N_f$ tends to infinity. The fact that the regression line does not have slope 1 is due to this as well as due to finite sample size of $N_f$.

Next, with the same system, we considered observing species $A$ rather than species $S$. So $Z(t) = (X(t), Y(t)) = (\#S(t), \#A(t))$. We kept the initial state and parameter values of $c_1$ and $c_3$ to be the same as before and performed a Bayesian estimation of parameter $c_2$. Thus we considered $c_2$ as a random variable $C_2$ with a uniform prior distribution on $[4, 6]$. We randomly generated a sample of $N_r = 1000$ values of $C_2$ following uniform distribution $[4, 6]$ and generated one observation trajectory $Y(t)$ for each value of $C_2$. Then we applied Algorithm 5 to compute $\hat{E}(t)$, the filter estimate of $\hat{E}(C_2 | Y(t))$ (see (20)). The result is shown in Figure 2.

---

**Algorithm 6 Past state estimation**

1. **Input:** Jump times of $Y(t_1, \ldots, t_N)$ and observed $Y$ at jump times $(y_1, \ldots, y_N)$. Initial distribution $\mu_0$ for $X(0) = x_0$, parameter value $c$, intermediate time $t_0$, final time $T$, filter sample size $N_f$.
2. Set $k = 1$, $t = 0$, $y = y_0$
3. for $k = 1$ to $k_1$ do
4. for $i = 1$ to $N_f$ do
5. $(V^{(i)}, w^{(i)}) = \text{Continuous-evolution}(V^{(i)}, w^{(i)}, t, t_k, y, c)$
6. $(V^{(i)}, w^{(i)}) = \text{Jump}(V^{(i)}, w^{(i)}, y, y_k, c)$
7. end for
8. if resampling then $(V, w) = \text{Offsprings}(V, w)$
9. else rescale $w$ so that $\sum_{i=1}^{N_f} w^{(i)} = N_f$
10. end if
11. Set $t = t_k$, $y = y_k$
12. end for
13. $(V^{(i)}, w^{(i)}) = \text{Continuous-evolution}(V^{(i)}, w^{(i)}, t, t_0, y_N, c)$ for $i = 1, \ldots, N_f$
14. Set $V^{(i)} = V^{(i)}$ for $i = 1, \ldots, N_f$
15. for $k = k_1 + 1$ to $N_f$ do
16. for $i = 1$ to $N_f$ do
17. $(V^{(i)}, w^{(i)}) = \text{Continuous-evolution}(V^{(i)}, w^{(i)}, t, t_k, y, c)$
18. $(V^{(i)}, w^{(i)}) = \text{Jump}(V^{(i)}, w^{(i)}, y, y_k, c)$
19. end for
20. Set $U^{(i)} = (V^{(i)}, \hat{V}^{(i)})$ for $i = 1, \ldots, N_f$
21. if resampling then $(U, w) = \text{Offsprings}(U, w)$
22. else rescale $w$ so that $\sum_{i=1}^{N_f} w^{(i)} = N_f$
23. end if
24. Set $(V^{(i)}, \hat{V}^{(i)}) = U^{(i)}$ for $i = 1, \ldots, N_f$
25. Set $t = t_k$, $y = y_k$
26. end for
27. $(V^{(i)}, w^{(i)}) = \text{Continuous-evolution}(V^{(i)}, w^{(i)}, t, y_N, c)$ for $i = 1, \ldots, N_f$

---

1. From this we estimated CMTVE for the algorithm to be 0.0475 with a 95% confidence interval of [0.0461, 0.0488].
5.2 Genetic Circuit Example

We consider a genetic transcription example \[31\] where a protein $A$ encoded by gene $D_A$ could bind with its own gene promoter to enhance or suppress its transcription:

\[
\begin{align*}
D_A + A & \xrightarrow{c_1} D'_A, \\
D'_A & \xrightarrow{c_2} D_A + A, \\
D_A & \xrightarrow{c_3} D_A + A, \\
D'_A & \xrightarrow{c_4} D'_A + A, \\
A & \xrightarrow{c_5} \emptyset.
\end{align*}
\]  

(28)

Let $Z(t) = (\#D_A(t), \#D'_A(t), \#A(t))$ and suppose the propensity function has the mass action form $a_1(z) = c_1z_1z_3, a_2(z) = c_2z_2, a_3(z) = c_3z_1, a_4(z) = c_4z_2, a_5(z) = c_5z_3$. Suppose we only observe the molecule counts of protein $A$ and would like to estimate the copy number of the naked form of gene promoter $D_A$ and the bounded form of the gene promoter $D'_A$. We ran all numerical experiments with initial condition $Z(0) = (3, 0, 15)$ and nominal parameter values $c_1 = 0.3, c_2 = 3, c_3 = 0.5, c_4 = 0.2, c_5 = 0.06$ and filter sample size $N_s = 10,000$. The estimation of the copy number of $D_A$ right after each jump of $Z_3(t) = \#A(t)$ for a particular observed trajectory of $A$ is shown in Figure 4.

We estimated one parameter at a time while fixing the other parameters at their nominal values mentioned before. We chose a uniform distribution as the prior and the posterior conditional distributions corresponding to the same single observed trajectory at several snapshots in time are shown in Figure 5.
Figure 2: Scatter plot of the point estimates $\mathcal{E}(T)$ of the copy number of species $A$ (based on the observation of species $S$) against the actual copy numbers $X(T)$, in the linear propensity example with $T = 20$. The dashed yellow line is the regression line and solid red line is the line with slope 1. The bias is $-0.2810$ with a 95% confidence interval of $[-1.20427, 0.642256]$ the estimated $L^2$ error is $10.3159$ with a 95% confidence interval of $[9.58143, 11.0014]$.

Figure 3: On the left is a scatter plot of $\hat{C}_2(T)$, the estimated value of $C_2$ (based on the observation of species $A$) against the actual value of $C_2$ in the linear propensity example. Here $T = 40$. Filter sample size was $N_s = 1000$ and trial size $N_r = 500$. On the right is the result of running the same experiment for different values of $T$. The $L^2$ error $\mathbb{E}[(\hat{C}_2(T) - C_2)^2]^{1/2}$ as well as the bias $\mathbb{E}(\hat{C}_2(T) - C_2)$ are plotted against $T$ suggesting that longer observations lead to better estimations.
Figure 4: Estimation of copy number of $D_A$ in the genetic circuit example. Estimates are shown at jump times of the observed species $A$ along with their 95% confidence intervals. Filter sample size used was $N_s = 10,000$.

Figure 5: The posterior conditional probability density function of parameters $C_1$ (left) and $C_2$ (right) based on one observed trajectory on interval $[0, T]$ in the genetic circuit example for $T = 0, 20$ and $100$. The filter sample size was $N_s = 10,000$. $T = 0$ is simply the uniform priors on $C_1$ and $C_2$. The vertical lines show the true parameter values.
5.3 Genetic toggle switch

Consider the system of genetic toggle switch\[15\]

\[\emptyset \xrightarrow{a_1} S_1, \]

\[S_1 \xrightarrow{a_2} \emptyset, \]

\[\emptyset \xrightarrow{a_3} S_2, \]

\[S_2 \xrightarrow{a_4} \emptyset. \]

(29)

Let \(Z(t) = (X(t), Y(t)) = (\#S_1(t), \#S_2(t))\) and suppose the propensity functions have the form \(a_1(z) = \frac{\alpha_1}{1 + y}, a_2(z) = x, a_3(z) = \frac{\alpha_2}{1 + z^2}, a_4(z) = y.\) Suppose we only observe species \(S_2\) and we estimate the molecule counts of species \(S_1.\) We ran all numerical experiments with the initial condition \(Z(0) = (0, 0)\) and the nominal parameter values \(\alpha_1 = 50, \alpha_2 = 16, \beta = 2.5, \gamma = 1.\) The estimation of the molecule counts of species \(S_1\) right after each jump of \(\#S_2(t)\) for a particular observed trajectory is shown in Figure 6 where a filter sample size of \(N_s = 10,000\) was used.

As in the genetic circuit example, we estimated one parameter at a time while fixing the other parameter at its nominal value mentioned before. We chose a uniform distribution as the prior and the posterior conditional distributions computed based on the same single observed trajectory at several snapshots in time are shown in Figure 7. Since the parameter \(\alpha_2\) affects the species \(S_2\) in a more immediate way, its inference is better. On the other hand, \(\alpha_1\) affects \(S_2\) in a more indirect way, we had to observe a trajectory up to \(T = 1000\) to obtain a posterior distribution that was sufficiently narrow.
5.4 SEIR model

Consider the SEIR model for the spread of infectious diseases

\[ S + I \xrightarrow{\beta} E + I, \]
\[ E \xrightarrow{\kappa} I, \]
\[ I \xrightarrow{\gamma} R. \]  

(30)

Let \( Z(t) = (\#S(t), \#E(t), \#R(t), \#I(t)) \) where \( S, E, I \) and \( R \) stand for susceptible, exposed, infected and recovered respectively. We assume that we could observe the infectious population \( I \) exactly. The propensity functions \( a_j(\cdot) \) have the form \( a_1(z) = \beta z_1 z_4 / N \), \( a_2(z) = \kappa z_2 \), \( a_3(z) = \gamma z_4 \), where \( N = z_1 + z_2 + z_3 + z_4 \) is the total population. We ran all numerical experiments with initial condition \( Z(0) = (s_0, e_0, r_0, i_0) = (500, 20, 0, 5) \) and parameter values \( \beta = 0.05, \kappa = 0.2, \gamma = 0.05 \).

The estimation of the susceptible and exposed population after each change of \( Z_4(t) = \#I(t) \) for a particular observed trajectory of \( I \) is shown in Figure 8 where a filter sample size of \( N_s = 10,000 \) was used. Figure 9 shows scatter plots corresponding to state estimation of susceptible and exposed number of individuals.

Keeping other parameters at their nominal values, we explored Bayesian inference of parameter \( \kappa \) (the reciprocal of the incubation period) based on a uniform prior. The scatter plot of the parameter estimation of \( \kappa \) based on \( N_r = 1000 \) trials each with a filter sample size \( N_s = 1000 \) is shown in Figure 10.

Next we explore estimation of the state at a past time \( t_0 \) based on observation up to a future time \( T \). That is \( 0 \leq t_0 \leq T \). The case of \( t_0 = 0 \) corresponds to the situation where the initial state itself is not known, but we only have a prior distribution for it. We assumed that the exposed population at initial time \( t = 0 \) followed a binomial distribution, with parameters \( N = 520 \) and \( p = 0.04 \). Since \( \#I(0) = 5 \) and the total population is 525, this left us with 520 individuals. We assumed a probability \( p = 0.04 \) of being exposed, which lead us to choose the binomial distribution. Algorithm 6 was used to estimate the conditional probability mass function

\[ P\{X_2(t_0) = x | Y(s) = y(s), 0 \leq s \leq T\} \]

for various \( t_0 \) and \( T \). The results are shown in Figure 11.

5.5 Investigation of resampling

Here we investigate the strategies of resampling in our filter. In the numerical examples presented so far, we applied the offspring algorithm to resample right after each jump \( t_k \). Here we present the results of three
Figure 8: Estimation of the susceptible population and the exposed population in the SEIR example. Estimates are shown at jump times of the observed new infection or recovery along with the 95% confidence intervals. Filter sample size used was $N_s = 10,000$.

Figure 9: Estimate of the susceptible (left) and exposed (right) population in SEIR example at time $T = 40$. The filter sample size was $N_s = 1000$, and trial size $N_r = 500$. For the susceptible population, the bias is 0.0156 within a 95% confidence interval $[-0.160955, 0.192202]$ and $L_2$ error is 1.9723 within $[1.81684, 2.11637]$. For the exposed population, the bias is $-0.0156$ within a 95% confidence interval $[-0.192202, 0.160955]$ and $L_2$ error is 1.9723 within $[1.81684, 2.11637]$. 
Figure 10: On the left is a scatter plot of $\hat{\kappa}(T)$, the estimated value of $\kappa$ against the actual value of $\kappa$. Here $T = 40$. Filter sample size was $N_s = 1000$ and trial size $N_r = 500$. On the right is the result of running the same experiment for different values of $T$. The $L^2$ error $E[(\hat{\kappa}(T) - \kappa)^2]^{1/2}$ as well as the bias $E[\hat{\kappa}(T) - \kappa]$ are plotted against $T$.

Figure 11: Conditional distribution of the exposed population at the initial time $X_2(0)|Y_T$ for $T = 0, 20$ and 40 on the left and at an intermediate time $X_2(10)|Y_T$ for $T = 10, 20$ and 40 on the right. Filter sample size was $N_s = 10,000$. The prior of the initial distribution of the exposed population follows a binomial distribution $B(520, 0.04)$. 
we applied an adaptive resampling method where resampling is applied at normalized instead so that the average weight was 1 right after any jump time third alternative was not to resample at all. We note that, when ever resampling was not used, the weights were weights or if the ratio of the maximum weight to the minimum nonzero weight was greater than 1000. The 95%

Table 1: The bias and error of estimation of the states or parameters when different resampling schemes were applied.

| System              | Method      | $L^2$ error | Confidence interval          | bias     | Confidence interval          |
|---------------------|-------------|-------------|------------------------------|----------|------------------------------|
| linear propensity   | each jump   | 9.7470      | [9.0677, 10.3819]            | −0.3346  | [−1.2068, 0.5375]            |
| linear propensity   | adaptive (0%) | 10.0216    | [9.3373, 10.6621]            | −0.0699  | [−0.9671, 0.8273]            |
| linear propensity   | never       | 10.0783     | [9.4464, 10.6728]            | −0.4910  | [−1.3922, 0.4103]            |
| genetic circuit     | each jump   | 0.5638      | [0.5241, 0.6009]             | 0.0100   | [−0.0405, 0.0604]            |
| genetic circuit     | adaptive (75.85%) | 0.5456      | [0.5066, 0.5820]            | −0.0085  | [−0.0574, 0.0403]            |
| genetic circuit     | never       | NaN         | NaN                          | NaN      | NaN                          |
| genetic toggle      | each jump   | 5.5083      | [5.0655, 5.9181]             | 0.3701   | [−0.1291, 0.8622]            |
| genetic toggle      | adaptive (2.20%) | 5.5307      | [5.0344, 5.9860]             | 0.2406   | [−0.2541, 0.7353]            |
| genetic toggle      | never       | 5.4175      | [4.9015, 5.8886]             | −0.8134  | [−1.2930, −0.3339]           |
| SEIR                | each jump   | 2.0853      | [1.9323, 2.2279]             | −0.0176  | [−0.2043, 0.1691]            |
| SEIR                | adaptive (9.91%) | 2.3726      | [2.2102, 2.5245]            | 0.1238   | [−0.0884, 0.3359]            |
| SEIR                | never       | 2.0823      | [1.9572, 2.2003]             | −0.0029  | [−0.1893, 0.1835]            |
| linear propensity c2| each jump   | 0.4099      | [0.3850, 0.4334]             | 0.0233   | [−0.0134, 0.0599]            |
| linear propensity c2| adaptive (23.00%) | 0.4237      | [0.3977, 0.4481]            | −0.0295  | [−0.0674, 0.0083]            |
| linear propensity c2| never       | 0.5380      | [0.5022, 0.5716]             | 0.0628   | [0.0149, 0.1106]             |
| SEIR $\kappa$      | each jump   | 0.0377      | [0.0354, 0.0398]             | 0.0002   | [−0.003, 0.004]              |
| SEIR $\kappa$      | adaptive (20.61%) | 0.0351      | [0.0329, 0.0372]            | −0.003   | [−0.004, 0.0028]             |
| SEIR $\kappa$      | never       | 0.0371      | [0.0343, 0.0396]            | −0.0064  | [−0.0097, −0.003]            |

different resampling strategies. One was to resample at each jump time $t_k$ as we did earlier. As an alternative, we applied an adaptive resampling method where resampling is applied at $t_k$ if more than 10 particles had zero weights or if the ratio of the maximum weight to the minimum nonzero weight was greater than 1000. The third alternative was not to resample at all. We note that, whenever resampling was not used, the weights were normalized instead so that the average weight was 1 right after any jump time $t_k$.

Table 1 shows the $L^2$ error as well as the bias in estimating the conditional mean of the state or a parameter along with the 95% confidence intervals, for the various examples considered earlier. The results are for state estimation unless a parameter is mentioned in the first column. The entries NaN indicate numerical issues in MATLAB mainly due to the fact that all weights became either zero or infinity within numerical precision. These results show that the three approaches yield more or less the same accuracy (the estimated errors are within the confidence intervals of each other) except in the case of the genetic circuit example where serious numerical issues manifest without resampling. We suspect that in the other examples, if final time $T$ is increased, the need for resampling will become evident.

6 Concluding and future work

We presented a novel filtering algorithm to compute the conditional probability mass function

$$\pi(t, x) = P\{X(t) = x | Y(s) = y(s), 0 \leq s \leq t\},$$

in the context of reaction networks. Here $Y(t) = y(t)$ is the vector copy number of a subset of species that are assumed to be observed exactly in continuous time and $X(t)$ is the vector copy number of the remaining unobserved species. We also showed how this algorithm can be adapted for the purposes of Bayesian parameter estimation based on exact partial state observation. Furthermore, we also showed how the state $X(t_0)$ at time $t_0$ can be estimated based on observations up to a later time $T$ where $0 \leq t_0 \leq T$.

The filtering algorithm involves a weighted Monte Carlo method and a resampling strategy needs to be employed. We explored some possibilities for resampling at the observed jump times $t_k$. Our investigations in this regards were numerical and based on relatively simple examples, and hence not exhaustive. An investigation of adaptive resampling based on some theoretical analysis is the subject of future work.

While we presented an intuitive derivation of the filtering equations, our derivation is not mathematically rigorous. [10] provides a rigorous derivation in the context of finite state Markov processes which in the case of reaction networks correspond to systems where species conservation relations limit the species counts to be
bounded. A rigorous derivation is certainly possible for the case of unbounded species copy numbers provided certain integrability or moment bound conditions hold. In this context conditions in [29, 23, 11] will be relevant.

As is well known, many intracellular reaction networks may have some species in greater abundance and a discrete state model can be tedious to simulate one event at a time. Tau-leap methods [19] as well as model reduction approaches have been proposed for efficient simulation of such systems. See [20] and references therein for several methods as well as [5, 27, 24, 13, 29, 23, 30] for rigorous mathematical analysis of methods. These same considerations could be applied to the filtering method proposed here to develop reduced order models and tau-leap simulations.

Our assumption of exact (noisless) observation of some species may appear unrealistic. However, as mentioned in the introduction, most observation noise may be modeled via extra reactions and extra species such as photons. If the photon counts are very large, the same considerations of reduced order models or tau-leaping mentioned above apply. Less realistic is the assumption of continuous in time observations. In reality, observations are recorded in discrete time snapshots. If the frequency of the snapshots is very high, then the theory of continuous in time observations provides a good approximation. If the frequency is low, then this is not the case. Future work will involve the case where observations of some species are made at certain time snapshots as well as the limiting behavior as the time snapshots increase in frequency. Either way, the theory and the algorithm discussed in this paper provides a baseline for the exploration of observations at discrete time snapshots.

Acknowledgments: We thank Ankit Gupta for introducing us to the rich and subtle topic of stochastic filtering.

A Derivation of the evolution equations for $\pi(t, x)$

We note that [10] provides a rigorous derivation of the evolution equation for the conditional probability $\pi(t, z)$ when the state space is finite and the exact observation is of the form $y = h(z)$ where $h$ is a function of the state space. The derivation in [10] may not be easily accessible to applied scientists who may not be familiar with the language of stochastic analysis. Moreover, our filtering equations (while in agreement with [10]) are somewhat simpler in appearance since in our case $h$ corresponds to the projection onto the last $n_2$ components of the state and also due to the structure of the reaction network. The derivation shown here is more intuitive to follow (at the expense of some rigor) and results in equations consistent with [10]. Moreover, we shall not make the assumption that the state space is finite. We believe that the rigorous derivation in [10] can be extended to infinite state space under reasonable assumptions, but such an endeavor is beyond the scope of this paper.

We start with a discretization of the time interval $[0, \infty)$ by a mesh

$$\{\ell h | \ell = 0, 1, \ldots \}$$

of spacing $h$. We consider $(X(\ell h), Y(\ell h))$ as a discrete time Markov chain. If the observed trajectory of $Y(t)$ is given by $y(t)$, then the observations on the mesh points will be given by

$$\bar{y}_\ell = y(\ell h), \quad \ell = 0, 1, \ldots$$

We may use the filtering equations for a partially observed discrete time Markov chain derived in [13]. For the discrete time Markov chain $(X(\ell h), Y(\ell h))$ (for $\ell = 0, 1, \ldots$) where $Y$ is observed exactly, the conditional probability

$$\pi_\ell(x) = P(X(\ell h) = x | Y(jh) = \bar{y}_j, j = 0, \ldots, n),$$

is shown in [13] to satisfy

$$\pi_\ell(x') = \frac{\sum_y \kappa((x, \bar{y}_{\ell-1}), (x', \bar{y}_\ell))\pi_{\ell-1}(x)}{\sum_{x_1} \sum_{x_{\ell-1}} \kappa((x_{\ell-1}, \bar{y}_{\ell-1}), (x_\ell, \bar{y}_\ell))\pi_{\ell-1}(x_{\ell-1})},$$

(31)

where

$$\kappa((x, y), (x', y')) = P(X(\ell h) = x', Y(\ell h) = y' | X((\ell - 1)h) = x, Y((\ell - 1)h) = y).$$

22
From the infinitesimal characteristics of continuous time Markov chains, as \( h \) approaches 0+, 
\[
\kappa((x, y), (x', y')) = a_j(x, y)h + o(h) \quad (x', y') = (x, y) + \nu_j,
\]
\[
\kappa((x, y), (x, y)) = 1 - h \sum_{j=1}^{m} a_j(x, y) + o(h),
\]
\[
\kappa((x, y), (x', y')) = o(h) \quad \text{otherwise.}
\]

Let's consider two adjacent mesh points \( t \) and \( t + h \). There are two possibilities; there are no jumps of \( y \) on the interval \( (t, t + h) \) or there are jumps. Again from the infinitesimal characteristics of continuous time Markov chains, as \( h \) approaches 0+, there is either no jump or one jump during \( (t, t + h) \). We approximate \( \pi(t, x) \) by \( \pi_{t-1}(x) \) and \( \pi(t + h, x) \) by \( \pi_t(x) \). For the case when there is no jump during \( (t, t + h) \), if we suppose \( t_k \leq t < t + h < t_{k+1} \), then
\[
\bar{y}_t = \bar{y}_{t-1} = y(t_k).
\]

Using (31) and (32), we obtain \( \pi(t + h, x) \) as a ratio where the numerator is
\[
\sum_{j \in \mathcal{U}} \pi(t, x - \nu'_j)a_j(x - \nu'_j, y(t_k))h + \pi(t, x) - \pi(t, x) \sum_{j=1}^{m} a_j(x, y(t_k))h + o(h)
\]
and the denominator is
\[
\sum_{\tilde{x}} \left( 1 - \sum_{j \in \mathcal{O}} a_j(\tilde{x}, y(t_k))h \right) \pi(t, \tilde{x}) + o(h).
\]

From the above, we may obtain an expression for \( (\pi(t + h, x) - \pi(t, x))/h \) which upon taking limit as \( h \to 0+ \) yields [4]. The second case is when \( t < t_k < t + h \) and in this case
\[
\bar{y}_{t-1} = y(t_{k-1}) \neq y(t_k) = \bar{y}_t.
\]

Using (31) and (32), we obtain
\[
\pi(t + h, x) = \frac{\sum_{l \in \mathcal{O}} a_l(x - \nu'_l, y(t_{k-1})) \pi(t, x - \nu'_l) h + o(h)}{\sum_{\tilde{x}} \sum_{l \in \mathcal{O}} a_l(\tilde{x} - \nu'_l, y(t_{k-1})) \pi(t, \tilde{x}) h + o(h)}.
\]

Noting that \( \pi(t + h, x) \to \pi(t_k, x) \) and \( \pi(t, x) \to \pi(t_k-, x) \) as \( h \to 0+ \), we obtain [3].

We note that for a rigorous treatment one needs to use the language of measure theory, since unlike in the discrete time Markov chain case, in the continuous time case we are conditioning on a zero probability event of observing \( Y(s) = y(s) \) for \( 0 \leq s \leq t \). Moreover, a rigorous and mathematically “cleaner” treatment involves working with the integral representation of the differential equation with jumps. The derivation provided here, we hope, provides the “essence” of the idea.

References

[1] David F Anderson. A modified next reaction method for simulating chemical systems with time dependent propensities and delays. The Journal of chemical physics, 127(21):214107, 2007.

[2] David F Anderson, Arnab Ganguly, Thomas G Kurtz, et al. Error analysis of tau-leap simulation methods. The Annals of Applied Probability, 21(6):2226–2262, 2011.

[3] David F Anderson and Thomas G Kurtz. Continuous time markov chain models for chemical reaction networks. In Design and analysis of biomolecular circuits, pages 3–42. Springer, 2011.

[4] Alan Bain and Dan Crisan. Fundamentals of stochastic filtering, volume 60. Springer Science & Business Media, 2008.

[5] Karen Ball, Thomas G Kurtz, Lea Popovic, Greg Rempala, et al. Asymptotic analysis of multiscale approximations to reaction networks. The Annals of Applied Probability, 16(4):1925–1961, 2006.
[6] David Bernstein. Simulating mesoscopic reaction-diffusion systems using the gillespie algorithm. Physical Review E, 71(4):041103, 2005.

[7] Richard J Boys, Darren J Wilkinson, and Thomas BL Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. Statistics and Computing, 18(2):125–135, 2008.

[8] Pierre Brémaud. Point processes and queues: martingale dynamics. Springer-Verlag, 1981.

[9] Silvia Calderazzo, Marco Brancaccio, and Bärbel Finkenstädt. Filtering and inference for stochastic oscillators with distributed delays. Bioinformatics, 35(8):1380–1387, 2019.

[10] Fulvia Confortola and Marco Fuhrman. Filtering of continuous-time markov chains with noise-free observation and applications. Stochastics An International Journal of Probability and Stochastic Processes, 85(2):216–251, 2013.

[11] S. Engblom. On the stability of stochastic jump kinetics. App. Math., 5(6):3217–3239, 2014.

[12] Zhou Fang, Ankit Gupta, and Mustafa Khammash. Stochastic filters based on hybrid approximations of multiscale stochastic reaction networks. arXiv preprint arXiv:2008.11682, 2020.

[13] Bert Fristedt, Naresh Jain, N Krylov, and Nikolaì Vladimirovich Krylov. Filtering and Prediction: A Primer: A Primer, volume 10. American Mathematical Soc., 2007.

[14] Arnab Ganguly, Derya Altintan, and Heinz Koeppl. Jump-diffusion approximation of stochastic reaction dynamics: error bounds and algorithms. Multiscale Modeling & Simulation, 13(4):1390–1419, 2015.

[15] Timothy S Gardner, Charles R Cantor, and James J Collins. Construction of a genetic toggle switch in escherichia coli. Nature, 403(6767):339–342, 2000.

[16] Michael A Gibson and Jehoshua Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. The journal of physical chemistry A, 104(9):1876–1889, 2000.

[17] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem., 81:2340–2361, 1977.

[18] Daniel T Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. Journal of computational physics, 22(4):403–434, 1976.

[19] Daniel T Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. The Journal of chemical physics, 115(4):1716–1733, 2001.

[20] Daniel T Gillespie. Stochastic simulation of chemical kinetics. Annu. Rev. Phys. Chem., 58:35–55, 2007.

[21] Andrew Golightly and Darren J Wilkinson. Bayesian sequential inference for stochastic kinetic biochemical network models. Journal of Computational Biology, 13(3):838–851, 2006.

[22] Andrew Golightly and Darren J Wilkinson. Bayesian parameter inference for stochastic biochemical network models using particle markov chain monte carlo. Interface focus, 1(6):807–820, 2011.

[23] Ankit Gupta, Corentin Briat, and Mustafa Khammash. A scalable computational framework for establishing long-term behavior of stochastic reaction networks. PLoS Comput Biol, 10(6):e1003669, 2014.

[24] Benjamin Hepp, Ankit Gupta, and Mustafa Khammash. Adaptive hybrid simulations for multiscale stochastic reaction networks. The Journal of chemical physics, 142(3):034118, 2015.

[25] Rudolph E Kalman and Richard S Bucy. New results in linear filtering and prediction theory. 1961.

[26] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[27] Hye-Won Kang, Thomas G Kurtz, et al. Separation of time-scales and model reduction for stochastic reaction networks. The Annals of Applied Probability, 23(2):529–583, 2013.
[28] Tiejun Li. Analysis of explicit tau-leaping schemes for simulating chemically reacting systems. Multiscale Modeling & Simulation, 6(2):417–436, 2007.

[29] Muruhan Rathinam. Moment growth bounds on continuous time markov processes on non-negative integer lattices. Quart. Appl. Math., 53(2):347–364, 2015.

[30] Muruhan Rathinam. Convergence of moments of tau leaping schemes for unbounded markov processes on integer lattices. SIAM Journal on Numerical Analysis, 54(1):415–439, 2016.

[31] Muruhan Rathinam and Hana El Samad. Reversible-equivalent-monomolecular tau: A leaping method for “small number and stiff” stochastic chemical systems. Journal of Computational Physics, 224(2):897–923, 2007.