Filtered finite state projection method for the analysis and estimation of stochastic biochemical reaction networks

Elena D'Ambrosio[1], Zhou Fang[1], Ankit Gupta[1], Mustafa Khammash[1*]

1 Department of Biosystems Science and Engineering, ETH Zurich, Basel, Switzerland

*mustafa.khammash@bsse.ethz.ch

Abstract

Recent advances in fluorescence technologies and microscopy techniques have significantly improved scientists' ability to investigate biological processes at the single-cell level. However, fluorescent reporters can only track the temporal dynamics of a limited number of critical components in a cell (e.g., fluorescent proteins), leaving other pivotal dynamic components (such as gene-state) hidden. Moreover, the nature of the interactions among intracellular biomolecular species is inevitably stochastic in the low copy number regime, which adds more difficulties to the investigation of these hidden species dynamics. Therefore developing mathematical and computational tools for inferring the behaviour of stochastic reaction networks from time-course data is urgently needed.

Here we develop a finite-dimensional filter for estimating the conditional distribution of the hidden (unobserved) species given continuous-time and noise-free observations of some species (e.g. a fluorescent reporter). It was proposed that in this setting, the conditional distribution evolves in time according to a large or potentially infinite-dimensional system of coupled ordinary differential equations with jumps, known as the filtering equation. We first formally verify the validity of this filtering equation under the non-explosivity condition and then develop a Finite-State Projection method, which provides an approximate solution by truncating the infinite-dimensional system. Additionally, we give computable error bounds for the algorithm. Finally, we present several numerical examples to illustrate our method and compare its performance with an existing particle filtering method for estimating the conditional distribution.

1 Introduction

The genetics revolution of the 20th century [1] undoubtedly impacted several fundamental aspects of biological sciences. The extraordinary discovery of the Green Fluorescent Protein, dated to 1962, opened the doors for the rapid development of fluorescent technologies [2] and microscopic techniques [3]. Consequently, the scientists' ability to characterise living cells' dynamic behaviours has significantly increased. Nonetheless, even state of art fluorescent reporters, only a limited number of critical components in a cell, such as fluorescent proteins and mRNAs, can be tracked [4]. Therefore, other dynamic states of pivotal interest, such as gene (on/off) state, transcription factors and promoters' activity, remain obscure. Moreover, the interactions among intracellular biomolecular species are inevitably stochastic in the low copy number regime [5, 6], which adds more difficulties to investigating these hidden species. Therefore, developing mathematical and computational tools for inferring the behaviour of stochastic reaction networks from time-course data is urgently needed.

One of the most common ways to model stochastic biochemical reaction processes is to employ a continuous-time discrete-state Markov Chain which keeps track of the molecular species copy number. The transition probability of this Markov Chain satisfies a very large or possibly infinite dimensional linear ordinary differential system of coupled equations, also known as the Chemical Master Equation (CME), that describe the chain's possible state transitions over time. One dominant numerical strategy for solving this extensive system is the finite state projection (FSP) method [7], which truncates the Markov Chain's state space and gives rise to a finite-dimensional subsystem for the transition probability while providing an exact error certificate of the approximated solution.

In addition to this relatively direct approach, the CME can also be solved by simulation based approaches, like the Gillespie algorithm [8, 9] and its variants [10, 11], which exactly simulate the Markov chain model and approximate the probability distribution with the empirical distribution. However, these simulation-based approaches become computationally infeasible when the system size increases or several reactions are firing in a time unit. Therefore, other approximations of the CME have become available. Some of these methods, such as the Chemical Langevin Equation and the Linear Noise Approximation (see [12] for a detailed reference), focus on approximating the underlying stochastic process of the CME with a continuous state process. Instead, other methods look at the CME's statistics and are provided with moment closure techniques to close the moment equations [13–15].

Although several approaches are available to approximate the CME's solution, few computational methods exist to obtain real-time estimations of latent cellular states. In this direction, stochastic filtering theory provides a means for obtaining real-time estimations of latent states from noisy or noise-free time-course measurement data. Specifically, this means estimating the conditional probability distribution π_t of the latent state, modelled by a stochastic process X_t, given the available information until time t, typically represented by a sigma-algebra generated by an observation process, usually denoted by $\mathcal{Y}_t$. Usually, such conditional probability π_t satisfies a stochastic differential equation, known as the filtering equation, which is generally non-linear and infinite-dimensional [16]. These features hinder the real-time estimations of latent cellular states based on time-course measurement data.

One of the first pivotal advances in stochastic filtering has been the linear filter introduced by Kalman in the discrete-time setting [17]. The extension to the continuous-time case was conducted later by Bucy [18]. For linear dynamical system corrupted with Gaussian noise, the linear filter provides an analytical expression for the conditional distribution, π_t, which is Gaussian and therefore determined by its mean and covariance matrix. This Kalman-Bucy filter has been widely used in practical applications, including space missions like the Apollo project [19]. For the non-linear setting, the filtering equation (referred to as the Kushner-Stratonovich equation [20, 21] in some cases) is generally infinite-dimensional [22]; an analytical expression of the solution is not always available. Facing this challenge, a surge of interest has occurred in particle filter methods since the seminal paper by Gordon, Salmond, and Smith [23]. The key feature of such methods is that finitely many random samples (or particles) are exploited to represent the posterior probability distribution. These methods look promising and particularly suited to address the infinite-dimensional feature of filtering problems.

In the stochastic reaction networks framework, state and parameter estimations have been mainly carried out with the Kalman filter and particle filtering (also known as the sequential Monte-Carlo method). In [24], and [25], Sun et al. and Chuang et al. applied the Extended Kalman filter and the Kalman filter to non-linear state-space models of gene regulatory networks. Moreover, in [26], Calderazzo et al. used an Extended

Kalman-Bucy filter to obtain parameters and state estimations in discrete-time of a stochastic negative feedback model. In [27], Liu et al. employ (i) extended Kalman filter (EKF), (ii) unscented Kalman filter (UKF) and (iii) the particle filter to estimate unobserved states of cellular responses in *Escherichia Coli* subjected to a sudden temperature increase. Moreover, in [28], and [29], Boys et al. and Golightly et al. applied sequential Monte-Carlo methods to diffusion approximations models of gene regulatory networks to infer kinetic parameters. Along these lines, Fang et al. [30–32] used a particle filter algorithm on a reduced model of a stochastic reaction network, derived through time-scale separation, rigorously showing that the filter of the reduced model converges to the one of the original model. For a noise-free observation process, in [33–35], Zechner and his coworkers solved the filtering equation through a moment-closure approach. In the same direction, Rathinam et al. [36] proposed a particle filtering algorithm to solve numerous estimations problems in biologically relevant examples.

Despite the popularity of the above methods, they show multiple drawbacks in several scenarios. Kalman filter requires linearity of the dynamical system, which is not very practical in real applications. Furthermore, the Kalman filter is not very accurate at modelling low-copy number regimes for discrete-state processes. As shown in the example in Fig. 1, the Kalman Filter fails at capturing the hidden process behaviour; it even provides negative-valued estimates for naturally positive species copy numbers. The particle filters usually show high accuracy in estimating summary statistics of the conditional distribution. However, they tend to perform poorly in approximating specific probability, especially in the case of rare events [37], as can be seen in Fig. 2. These facts motivate us to propose a new method that can more accurately solve the filtering problem for biochemical reaction systems.

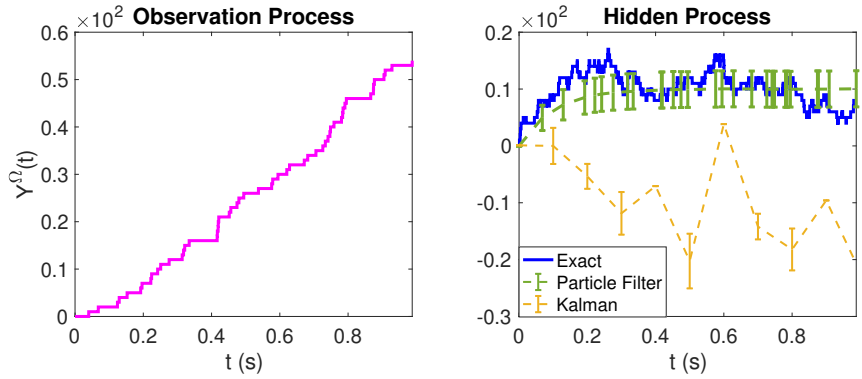

Figure 1. Performance of the Kalman filter (derivation of the filter can be found in S.3 section of the supplementary material), and particle filter in the simple transcription-translation model. In this example we studied the following simple network: $\text{mRNA} \xrightarrow{c_1} \text{mRNA} + \text{Protein}, \quad \varnothing \xrightarrow{c_2} \text{mRNA}, \quad \text{mRNA} \xrightarrow{c_3} \varnothing.$ We exactly observed in continuous time the Protein copy number and estimated the mRNA copy number. Kalman filter and particle filter were used for this purpose. On the left side, we are showing the trajectory of the Protein copy number, generated with a Gillespie simulation algorithm; whereas, on the right side, we plotting the hidden process trajectory, together with the conditional expectations and standard deviations estimated by the filters. Although particle filter shows a good performance, the Kalman filter fails at capturing the hidden process dynamics.

In this paper, we concern ourselves with the situation of noise-free observations, which is fairly reasonable given the significant advance of the modern technologies. In this setting, we first rigorously verified the validity of the continuous-time filtering

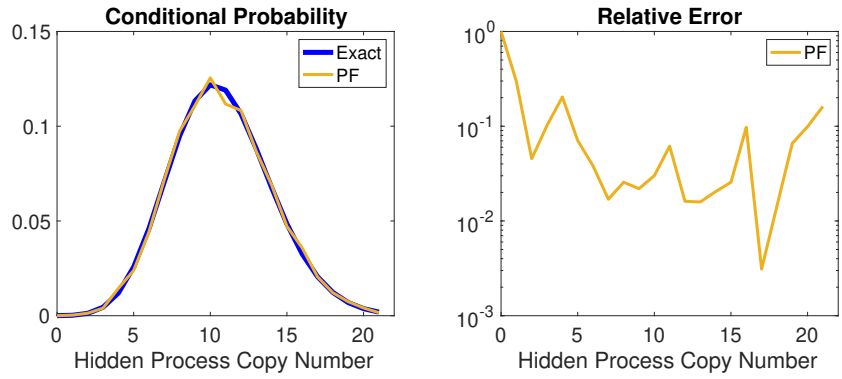

Figure 2. Comparison of the exact and estimated conditional distribution in the simple transcription-translation model (observing the mRNA dynamics): $\mathbf{mRNA} \xrightarrow{c_1} \mathbf{mRNA} + \mathbf{Protein}$, $\varnothing \xrightarrow{c_2} \mathbf{mRNA}$, $\mathbf{mRNA} \xrightarrow{c_3} \varnothing$ The left-sided plot displays a comparison between the exact conditional distribution and the one estimated by the particle filter at t=3s. We can notice that generally there is a good match between the two distributions, even though the peak is not accurately captured by the particle filter. On the right-side, it is shown the relative distance between the exact values of the conditional distribution and the ones estimated by the particle filter. The error is quite high, especially near 0.

equation, previously derived in [33, 34, 36]. The filtering equation has an unnormalized version, which resembles the CME very closely except for a term driven by the observation process. Inspired by this, we propose the Filtered Finite State Projection (FFSP) approach for the filtering problem by solving the unnormalized filtering equation on the truncated state space of the hidden species. Since the FFSP method solves the filtering equation in a direct fashion, it turns out to be more accurate than other existing methods. Based on different normalization steps, we provide two slightly different FFSP algorithms together with their error bounds. Finally, several biologically relevant numerical examples are presented to illustrate our approach.

The paper is organised as follows: in Section 2.1 we introduce the necessary notation and the stochastic reaction network framework; in Section 2.2, we introduce the conditions for which the filtering equation in the noise-free case holds; in Section 2.3 we outline the Filtered Finite State Projection (FFSP) method together with its accuracy analysis; in Section 2.4, we propose an alternative FFSP algorithm with a tighter error bound and finally, in Section 3 we show several numerical examples in which we compare the performance of both the algorithms, compute the error bounds and compare our algorithm with the famous Kalman filter and with a particle filter algorithm available in literature [36].

Notations

In this paper, we term $\mathbb{0}_n$ as a n-dimensional zero vector and $\mathbb{1}(\cdot)$ as the indicator function, which equals one if the argument holds and zero otherwise. $\mathbf{1}$ is a 1-column-vector of a proper size.

2 Materials and methods

2.1 Stochastic modelling of intracellular chemical reacting processes and their associated filtering problems

We consider an intracellular chemical reacting system that has n species $(S_1, \dots, S_n)$ and M reactions:

$$\sum_1^n \xi_{ik} S_i \xrightarrow{c_k} \sum_1^n \xi'_{ik} S_i, \quad k = 1, \dots M,$$

where ξ_{ik} and ξ'_{ik} are the numbers of S_i molecules consumed and produced in the kth reaction. We indicate with $a_1, a_2, \dots a_M$ the propensity functions representing the rates of these M reactions. Moreover, we term $\nu_k \triangleq \xi'_k - \xi_k$, as the stoichiometry vector associated with the kth reaction channel and define the stoichiometry matrix as follows:

$$\mathbb{S} = \begin{bmatrix} | & & | \\ \nu_1 & \dots & \nu_M \\ | & & | \end{bmatrix}$$

In the low copy number regime, the nature of the interactions among intracellular biomolecular species is inevitably stochastic, and its dynamics is usually modelled by a stochastic dynamic equation, known as the Random Time Change (RTC) Representation [38]:

$$\mathbf{Z}(t) = \mathbf{Z}(0) + \sum_{j=1}^{M} \nu_j R_j \left(\int_0^t a_j(\mathbf{Z}(s)) ds \right). \tag{1}$$

Here, $\mathbf{Z}(t)$ is a continuous-time discrete state Markov Chain keeping track of the molecules copy numbers, and $R_1, \dots, R_M$ are independent unit rate Poisson processes, which count the firing events of every reaction. For rigorousness, we only consider the processes $\mathbf{Z}(t)$ that satisfy the following non-explosivity condition:

$$\sup_{t \in [0,T]} \sum_{j=1}^{M} \mathbb{E}\left[a_j^2(\mathbf{Z}(t))\right] < \infty \qquad \forall T > 0 \tag{2}$$

which suggests that the system state will almost surely not grow to infinity in finite time. Also, to avoid negative molecular copies, we assume that $a_j(z) = 0$ for every $j \in \{1, \dots, M\}$ and $z + \nu_j \notin \mathbb{Z}^n_{\geq 0}$.

An alternative dynamical model to (1) is given by the Kolmogorov's forward equation also known as the Chemical Master Equation (CME):

$$\frac{d}{dt} p(t,z) = \sum_{j=1}^{M} a_j(z - \nu_j) p(t, z - \nu_j) - \sum_{j=1}^{M} a_j(z) p(t,z) \qquad \forall z \in \mathbb{Z}^n, \text{ and } \forall t \geq 0, \tag{3}$$

where $p(t,z) = P\{\mathbf{Z}(t) = z\}$ is the (unconditional) probability mass function associated with the Markov Chain, indicating the probability of finding the chemical system in state $z \in \mathbb{Z}^n$ at time $t > 0$. Under the condition (2), the two dynamical representations (1) and (3) of the system are equivalent.

With modern time-lapse fluorescent microscopes, scientists can now measure single-cell time-evolving behaviours at high resolutions [2, 3]. However, due to the limited choices of distinguishable fluorescent reporters, this technology cannot keep track of all the chemical species in a cell [4]. This obstacle gives rise to a stochastic filtering problem for the considered system, i.e., to infer the hidden state based on the partially observed cell states. An accurate solution to this problem can provide deep insights into intracellular dynamics and enables the design of better controllers.

2.2 The Filtering Problem for the Noise-Free Observation Process

To describe this filtering problem, we further decompose the system into two sub-networks, $\mathbf{Z}(t) = (\mathbf{X}(t), \mathbf{Y}(t))$, where $\mathbf{X}(t) \in \mathcal{X} \subseteq \mathbb{Z}_{\geq 0}^{n_1}$ keeps track of the copy numbers of the n_1 hidden (unobserved) species, and $\mathbf{Y}(t) \in \mathbb{Z}_{\geq 0}^{n_2}$ (with $n_2 = n - n_1$) keeps track of the copy numbers of the observed species. For simplicity, we rearrange the species orders so that the first n_1 species are the hidden species, and the last n_2 species are the observed ones. In this paper, we assume that we can time-continuously and exactly (noise-freely) observe the trajectory of the observed species $\mathbf{Y}(t)$. Our goal is to compute the conditional distribution $\pi(t,x) \triangleq P\{\mathbf{X}(t) = x | \mathbf{Y}(s) = y(s), 0 \leq s \leq t\}$[1], which is also viewed as the optimal estimate of the hidden process given the observations because for any $x \in \mathbb{Z}_{\geq 0}^{n_1}$ and $t \geq 0$

$$\pi(t,x) = \underset{\substack{K(t,x) \text{ is adapted to the} \\ \text{filtration generated by } \mathbf{Y}(t)}}{\arg\min} \mathbb{E}\left[\Big(K(t,x) - \mathbb{1}(\mathbf{X}(t) = x)\Big)^2\right]$$

where adapted means that the process under consideration is measurable with respect to the filtration generated by $\mathbf{Y}(t)$ for every $t \geq 0$.

To present the filtering equation characterising the time evolution of $\pi(t,x)$, we need first to introduce several notations related to the system decomposition. For each reaction stoichiometric vector ν_j $(j = 1, \ldots, M)$, we denote its first n_1 components by ν_j' and the last n_2 components as ν_j''. Following the notations in [36], we term $\mathcal{O} \triangleq \{j | \nu_j'' \neq \mathbb{0}_{n_2}\}$ as the set of observable reactions that can alter the dynamics of $\mathbf{Y}(t)$, and we term $\mathcal{U} \triangleq \mathcal{O}^c$ as the unobservable reactions that cannot alter $\mathbf{Y}(t)$. We define $a^{\mathcal{O}}(x,y) \triangleq \sum_{j \in \mathcal{O}} a_j(x,y)$, which is the total propensity of the observed reactions. Also, we term $\{t_k\}_{k \in \mathbb{Z}_{>0}}$ as the random jump times of the observed process $\mathbf{Y}(t)$ and denote $\mathcal{O}_k \triangleq \{j \in \mathcal{O} | \nu_j'' = y(t_k) - y(t_k^-)\}$ (for every $k \in \mathbb{Z}_{>0}$), which represents the reaction channels compatible with the jump of the observation process at time t_k. Finally, we define $t_0 = 0$.

The filtering equation for the conditional distribution $\pi(t,x)$ has been formally derived in the literature for finite state spaces' systems by [39]. In the infinite-dimensional setting, instead, it has been informally obtained by [33, 34, 36], and it is expressed by

$$\begin{aligned}\frac{d}{dt}\pi(t,x) = &\sum_{j \in \mathcal{U}} \pi(t, x - \nu_j') a_j(x - \nu_j', \mathbf{Y}(t_k)) - \sum_{j \in \mathcal{U}} \pi(t,x) a_j(x, \mathbf{Y}(t_k)) \\ &- \pi(t,x)\left(a^{\mathcal{O}}(x, \mathbf{Y}(t_k)) - \sum_{\tilde{x}} a^{\mathcal{O}}(\tilde{x}, \mathbf{Y}(t_k)) \pi(t, \tilde{x})\right)\end{aligned} \tag{4}$$

for all $k \in \mathbb{Z}_{\geq 0}$, $t \in (t_k, t_{k+1})$, and $x \in \mathbb{Z}_{\geq 0}^{n_1}$, and

$$\pi(t_k, x) = \frac{\sum_{j \in \mathcal{O}_k} a_j(x - \nu_j', \mathbf{Y}(t_{k-1})) \pi(t_k^-, x - \nu_j')}{\sum_{\tilde{x}} \sum_{j \in \mathcal{O}_k} a_j(\tilde{x}, \mathbf{Y}(t_{k-1})) \pi(t_k^-, \tilde{x})} \quad \forall k \in \mathbb{Z}_{>0} \text{ and } \forall x \in \mathbb{Z}_{\geq 0}^{n_1}. \tag{5}$$

In the supplementary material (section **S.1.3**), we rigorously verify this filtering equation under the non-explosivity condition (2), and we state the result as the following theorem.

[1] The well-posedness of $\pi(t,x)$ is guaranteed by [16, Theorem 2.24], and we give a more detailed discussion about it in the supplementary material.

Theorem 1. *Under the non-explosivity assumption* (2)*, the conditional distribution* $\pi(t,x)$ *is the unique (up to indistinguishability) non-negative solution of the filtering equation* (4) *and* (5) *that starts from* $\pi(0,\cdot)$ *and satisfies*

$$\int_0^t \sum_{x\in\mathbb{Z}_+^{n_1}} \sum_{j=1}^{M} a_j(x,\mathbf{Y}(s))\pi(s,x)ds < \infty \qquad \forall t \geq 0 \text{ almost surely.}$$

The filtering equation (4) is nonlinear due to the last term (being quadratic in $\pi(t,x)$), which brings many difficulties to solving this equation. To tackle this issue, the literature [36] provided a Monte-Carlo method based on a linear equation characterising the un-normalised conditional probability $\rho(t,x)$, from which you can obtain the filtering distribution via normalisation. Specifically, this un-normalised distribution is given by[2] $\rho(t,x) \triangleq \pi(t,x)\exp\left(-\int_{\max\{t_k|t_k\leq t\}}^t \left[\sum_{\tilde{x}\in\mathbb{Z}_{\geq 0}^{n_1}} a^{\mathcal{O}}(\tilde{x},\mathbf{Y}(s))\pi(s,\tilde{x})\right] ds\right)$ and therefore satisfies

$$\frac{d}{dt}\rho(t,x) = \sum_{j\in\mathcal{U}} \rho(t,x-\nu_j')a_j(x-\nu_j',\mathbf{Y}(t_k)) - \sum_{j=1}^{M}\rho(t,x)a_j(x,\mathbf{Y}(t_k)) \tag{6}$$
$$\forall k\in\mathbb{Z}_{\geq 0},\ \forall t\in(t_k,t_{k+1}),\ \text{and } x\in\mathbb{Z}_{\geq 0}^{n_1},$$

and

$$\rho(t_{k+1},x) = \frac{\sum_{j\in\mathcal{O}_{k+1}} a_j(x-\nu_j',\mathbf{Y}(t_k))\rho(t_{k+1}^-,x-\nu_j')}{\sum_{\tilde{x}}\sum_{j\in\mathcal{O}_k} a_j(\tilde{x},\mathbf{Y}(t_{k-1}))\rho(t_k^-,\tilde{x})} \quad \forall k\in\mathbb{Z}_{\geq 0} \text{ and } \forall x\in\mathbb{Z}_{\geq 0}^{n_1}. \tag{7}$$

Then, the literature [36] related the linear equation (6) with a new chemical reacting system and a random variable associated with that; afterward, a particle filter (a type of Monte-Carlo method) was proposed to compute the filtering equation. Different from this Monte-Carlo method, the literature [33, 34] proposed moment closure for the filtering equation (4) and (5) to compute the conditional moments of $\pi(t,x)$. More recently, an autonomous algorithm for this moment closure was proposed in [35].

2.3 Filtered finite state projection method

Here, we introduce a new method for the aforementioned filtering problem using the finite state projection (FSP) approach proposed in [7]. The idea is that the filtering equation (4) has a close resemblance to the CME (3), and, therefore, the FSP (an effective approach for solving CMEs) should be applicable to this filtering problem after some extensions. In this paper, we name this new method for the filtering problem as filtered finite state projection (FFSP) method.

Similar to the FSP which solves the CME directly by truncating the system's state space, the FFSP also straightforwardly solves the linear differential equation (6) in the time intervals (t_k,t_{k+1}) (for $k\in\mathbb{Z}_{>0}$) by truncating the hidden species' state space and updates the filter according to (7) at the jump times $\{t_k\}_{k\in\mathbb{Z}_{>0}}$. After approximating the un-normalised distribution $\rho(t,x)$, the FFSP obtain estimates of $\pi(t,x)$ by normalisation. To be more specific, when solving (6), the FFSP decomposes the hidden-state space $Z_{\geq 0}^{n_1}$ into a finite but large state space $\mathcal{X}_J = \{x_1,\ldots,x_J\} \subseteq Z_{\geq 0}^{n_1}$ and an infinity space $\mathcal{X}_{J'} \triangleq Z_{\geq 0}^{n_1} \setminus \mathcal{X}_J = \{x_{J+1},x_{J+2},\ldots\}$; then the FFSP only solves (4) on the finite space $\mathcal{X}_J$. More precisely, by denoting $\rho(t,\mathcal{X}_J) = (\rho(t,x_1),\ldots,\rho(t,x_J))^\top$ and

[2] This relation between normalised and un-normalised conditional distributions were not explicitly given in [36].

$\rho(t, \mathcal{X}_{J'}) = (\rho(t, x_{J+1}), \rho(t, x_{J+2}), \dots,)^\top$, the linear equation (4) can be equivalently written as

$$\begin{bmatrix} \dot{\rho}(t, \mathcal{X}_J) \\ \dot{\rho}(t, \mathcal{X}_{J'}) \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbb{A}_J(\mathbf{Y}(t_k)) & \mathbb{A}_{JJ'}(\mathbf{Y}(t_k)) \\ \mathbb{A}_{J'J}(\mathbf{Y}(t_k)) & \mathbb{A}_{J'}(\mathbf{Y}(t_k)) \end{bmatrix}}_{\mathbb{A}(\mathbf{Y}(t_k))} \begin{bmatrix} \rho(t, \mathcal{X}_J) \\ \rho(t, \mathcal{X}_{J'}) \end{bmatrix} \quad \forall k \in \mathbb{Z}_{\geq 0} \text{ and } \forall t \in (t_k, t_{k+1}). \tag{8}$$

where $\mathbb{A}(\mathbf{Y}(t_k))$ is an infinite dimensional matrix with each element defined by

$$\mathbb{A}_{ij}(\mathbf{Y}(t_k)) = \begin{cases} -\sum_{b=1}^{M} a_b(x_i, \mathbf{Y}(t_k)) & \text{if } x_j = x_i \\ \sum_{b=1}^{M} a_b(x_j, \mathbf{Y}(t_k)) \mathbb{1}\left(x_j + \nu'_b = x_i\right) & \text{if } x_j \neq x_i \end{cases}, \tag{9}$$

and $\mathbb{A}_J(\mathbf{Y}(t_k))$, $\mathbb{A}_{JJ'}(\mathbf{Y}(t_k))$, $\mathbb{A}_{JJ'}(\mathbf{Y}(t_k))$, and $\mathbb{A}_{J'}(\mathbf{Y}(t_k))$ are sub-matrices of $\mathbb{A}(\mathbf{Y}(t_k))$ with proper dimensions. Instead of solving the infinite dimensional ODE (8), the FFSP only implements a finite dimensional ODE

$$\dot{\rho}_{\text{FFSP}}(t, \mathcal{X}_J) = \mathbb{A}_J(\mathbf{Y}(t_k)) \rho_{\text{FFSP}}(t, \mathcal{X}_J) \qquad \forall k \in \mathbb{Z}_{\geq 0} \text{ and } \forall t \in (t_k, t_{k+1})$$

to approximate (8). Here, $\rho_{\text{FFSP}}(t, \mathcal{X}_J)$ is the FFSP's estimates to the the un-normalised filter $\rho(t, \mathcal{X}_J)$, and the solution of this finite dimensional ODE is

$$\rho_{\text{FFSP}}(t, \mathcal{X}_J) = \exp\left\{\mathbb{A}_J(\mathbf{Y}(t_k))(t - t_k)\right\} \rho_{\text{FFSP}}(t_k, \mathcal{X}_J) \quad \forall k \in \mathbb{Z}_{\geq 0} \text{ and } \forall t \in (t_k, t_{k+1}). \tag{10}$$

The detailed algorithm for the FFSP method is presented in Algorithm 1.

Algorithm 1 Filtered finite state projection (FFSP)

Require: The observation $\mathbf{Y}(t_k)$ and the initial distribution of the hidden states $\pi(0, x)$.

1: Initialization: $\pi_{\text{FFSP}}(0, x) = \pi(0, x)$, $\forall x \in \mathcal{X}_J$, and $\pi_{\text{FFSP}}(0, x) = 0$.

2: **for** $k = 0, 1, 2, \dots,$ **do**

3: Approximate the un-normalised filter before the next jump event:

$$\rho_{\text{FFSP}}(t, \mathcal{X}_J) = \exp\left\{\mathbb{A}_J(\mathbf{Y}(t_k))(t - t_k)\right\} \pi_{\text{FFSP}}\left(t_k, \mathcal{X}_J\right) \qquad t \in (t_k, t_{k+1})$$

4: Normalisation: $\pi_{\text{FFSP}}(t, x) = \frac{\rho_{\text{FFSP}}(t,x)}{\sum_{x \in \mathcal{X}_J} \rho_{\text{FFSP}}(t,x)}$ for all $x \in \mathcal{X}_J$ and $t \in (t_k, t_{k+1})$.

5: Approximate the filter at the next jump event:

$$\pi_{\text{FFSP}}(t_{k+1}, x) = \frac{\sum_{j \in \mathcal{O}_{k+1}} a_j\left(x - \nu'_j, \mathbf{Y}(t_k)\right) \rho_{\text{FFSP}}\left(t_{k+1}^-, x - \nu'_j\right)}{\sum_{x \in \mathcal{X}_J} \sum_{j \in \mathcal{O}_{k+1}} a_j\left(x, \mathbf{Y}(t_k)\right) \rho_{\text{FFSP}}\left(t_{k+1}^-, x\right)} \qquad x \in \mathcal{X}_J.$$

6: **end for**

In general, both the original FSP approach and our FFSP method solve their associated problems in a straightforward manner, and they therefore can reach good accuracy at a potential cost of high computational complexity in large dimensional systems. However, they still have some major differences in their structures. For instance, compared with the original FSP [7] which only solves one single finite-dimensional ODE, our FFSP algorithm needs to compute the ODEs (10) recursively for each time interval (t_k, t_{k+1}) due to the recursive structure of the filtering equation. Moreover, Algorithm 1 does not have the dominance property between the FFSP solution $\pi_{\text{FFSP}}(t, x)$ and the exact filter $\pi(t, x)$ because of the normalisation step

(in the sense that the exact solution is greater than the approximate one component-wise), whereas the original FSP algorithm does have this dominance property.

Also, our approach has several advantages over other existing computational methods for solving the filtering equation (4) and (5). Compared with the moment-closure approach in [33, 34], our FFSP method provides more detailed information about the conditional distribution, e.g., the specific conditional probability of being in any subset of the state-space, in addition to the conditional moments. Moreover, as a Monte-Carlo method, the particle filtering approach can provide good estimates for summary statistics (e.g., conditional mean and variance) but less accurate estimates for specific conditional probabilities (especially for the rare events). In contrast, our method (as a direct approach) can consistently provide very accurate estimates of the conditional probability at each state. More detailed comparisons between the particle filtering and the FFSP are presented in Section 3.

Once the FFSP method is established, we are interested in the estimation error of this algorithm, which characterises the reliability of this method. In the following, we perform error analysis for the FFSP. For technical reasons, we only consider the case where the propensity function $a^{\mathcal{O}}(\cdot, y)$ is upper bounded for each fixed $y \in \mathbb{Z}_{\geq 0}^{n_2}$, i.e.,

$$\text{there exists a function } \bar{a}^{\mathcal{O}}(y) \text{ such that } \sup_{x \in \mathbb{Z}_{\geq 0}^{n_1}} a^{\mathcal{O}}(x, y) \leq \bar{a}^{\mathcal{O}}(y). \tag{11}$$

Moreover, we also assume that the network with only unobserved reactions

$$\tilde{\mathbf{X}}_y(t) = \tilde{\mathbf{X}}_y(0) + \sum_{j \in \mathcal{U}} \nu_j' R_j \left(\int_0^t a_j(\tilde{\mathbf{X}}_y(s), y) ds \right)$$

is non-explosive in finite time for any $y \in \mathbb{Z}_{\geq 0}^{n_2}$ as long as its initial moments are finite. In other words,

$$\begin{aligned} &\textstyle\sum_{j \in \mathcal{U}} \mathbb{E}\left[\int_0^t a_j\left(\tilde{\mathbf{X}}_y(s), y\right) ds\right] < \infty \forall t \geq 0 \text{ and } \forall y \in \mathbb{Z}_{\geq 0}^{n_2}, \\ &\text{as long as } \mathbb{E}\left[\left\|\tilde{\mathbf{X}}_y(0)\right\|_1^q\right] < +\infty \text{ for all } q \geq 0. \end{aligned} \tag{12}$$

Both of these assumptions are reasonable and easily checkable in practice. Usually, under a microscope, scientists observe fluorescent reporters, which are catalytically produced and spontaneously degraded. In this setting, all the observable reactions satisfy (11) because of the saturation of catalytic production rates (due to the limited resources) and the independence of the degradation rate with other species. Also, some easily checkable conditions for the assumption (12) have already been provided in [40], and those conditions are shown to cover a large class of practical biochemical reacting systems.

An upper bound for the estimation error of Algorithm 1 is given in the following theorem.

Theorem 2. *Let us denote filters* $\pi_{FFSP}(t, \mathcal{X}) \triangleq (\pi_{FFSP}(t, x_1), \pi_{FFSP}(t, x_2), \dots)^\top$ *and* $\pi(t, \mathcal{X}) \triangleq (\pi(t, x_1), \pi(t, x_2), \dots)^\top$. *Then, under the conditions* (2), (11), *and* (12), *Algorithm 1 almost surely has an estimation error*

$$\begin{aligned} &\|\pi_{FFSP}(t_k, \mathcal{X}) - \pi(t_k, \mathcal{X})\|_1 = \epsilon(t_k) && \forall k \in \mathbb{Z}_{\geq 0} \\ &\|\pi_{FFSP}(t, \mathcal{X}) - \pi(t, \mathcal{X})\|_1 = \min\left\{ \frac{2\left[\epsilon(t_k) + \tilde{\epsilon}(t)\right]}{\|\rho_{FFSP}(t, \mathcal{X})\|_1}, 2 \right\} && \forall k \in \mathbb{Z}_{\geq 0} \text{ and } \forall t \in (t_k, t_{k+1}), \end{aligned}$$

where

$$\tilde{\epsilon}(t) = \|\rho_{FFSP}(t_k, \mathcal{X})\|_1 + \int_0^t \mathbf{1}^\top \mathbb{A}\left(\mathbf{Y}(t_k)\right) \rho_{FFSP}(s, \mathcal{X})\, ds - \|\rho_{FFSP}(t, \mathcal{X})\|_1$$

for all $k \in \mathbb{Z}_{\geq 0}$ *and* $t \in (t_k, t_{k+1})$*, and*

$$\epsilon(t_k) = \begin{cases} \sum_{x \notin \mathcal{X}_J} \pi(t_0, x) & t = 0 \\ \min\left\{ \frac{2\,\bar{a}^{\mathcal{O}_k}\left[\epsilon(t_{k-1}) + \tilde{\epsilon}(t_k^-)\right]}{\|\tilde{\rho}_{FFSP}(t_k, \mathcal{X})\|_1}, 2 \right\} & k \in \mathbb{Z}_{>0} \end{cases}$$

with $\bar{a}^{\mathcal{O}_k} \triangleq \sup_{x \in \mathbb{Z}_{\geq 0}^{n_1}} \left\{ \sum_{j \in \mathcal{O}_k} a_j(x, \mathbf{Y}(t_k)) \right\}$.

The proof of this theorem is given in the supporting material (section **S.2.2**. The major challenge is to analyze the discrepancy between the un-normalised filter $\rho(t, x)$ and the FFSP estimate $\rho_{\text{FFSP}}(t, x)$. Since the total mass of $\rho(t, x)$ is not conserved, we cannot directly use the techniques in [7]. Instead, we propose an extended FSP theorem (shown as follows) by exploiting the structure of $\mathbb{A}(\mathbf{Y}(t_k))$, especially the Metzler matrix property and the non-positivity of its column sums. This extended FSP theorem plays a fundamental role in the proof of Theorem 2; also, due to its general form, we believe this extended FSP theorem can be used in many other contexts as well, in which, for instance, the initial distribution may be unknown or the total mass of the distribution is not conserved, as in [41]. More specifically, in [41], a non-linear partial differential equation is derived to describe the probability of a protein concentration in a population of diving cells. Given the intractability of this non-linear equation, a linear version is proposed, whose solution has a non-constant total mass. Therefore, the following theorem would be incredibly suited in such a context to better characterise the evolution of the said total mass.

Theorem 3 (Extended FSP theorem for un-normalised probability distributions)**.** *We consider a set of un-normalised probability distributions* $\{\bar{p}(t, \cdot)\}_{t \geq 0}$ *defined on a discrete state space* $\mathcal{X} = \{x_1, x_2, \ldots\}$ *and evolving according to*

$$\dot{\bar{p}}(t, \mathcal{X}) = \mathbb{A}(y)\, \bar{p}(t, \mathcal{X}) \qquad \forall t \geq 0$$

where $\bar{p}(t, \mathcal{X}) \triangleq (\bar{p}(t, x_1), \bar{p}(t, x_2), \ldots)^\top$*, the variable* y *is a vector in* $\mathbb{Z}_{\geq 0}^{n_2}$*, and the matrix* $\mathbb{A}(y)$ *is defined in* (9)*. We also consider an FSP system of this infinite dimensional ODE, denoted by* $\{p_{FSP}(t, \cdot)\}_{t \geq 0}$*, which is defined on the same state space but evolves according to*

$$\dot{p}_{FSP}(t, \mathcal{X}) = \left[\begin{array}{c|c} \mathbb{A}_J(y) & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] p_{FSP}(t, \mathcal{X}) \qquad \forall t \geq 0$$

where $p_{FSP}(t, \mathcal{X}) \triangleq (p_{FSP}(t, x_1), p_{FSP}(t, x_2), \ldots)^\top$*, the matrix* $\mathbb{A}_J(y)$ *is the first* $J \times J$ *sub-matrix of* $\mathbb{A}(y)$*, and* $p_{FSP}(0, x) = 0$ *for all* $x \notin \mathcal{X}_J$*.*

Then, under conditions (12) *and* $\int_0^t \sum_i |\mathbb{A}_{ii}(y)|\, \bar{p}(s, x_i) ds < +\infty$*, the difference in the* L_1 *norm between these two sets of measures can be evaluated by*

$$\|p_{FSP}(t, \mathcal{X}) - \bar{p}(t, \mathcal{X})\|_1 < \epsilon(t) + \|p_{FSP}(0, \mathcal{X}) - \bar{p}(0, \mathcal{X})\|_1 \qquad \forall t \geq 0$$

where $\epsilon(t) = \sum_{x \in \mathcal{X}} p_{FSP}(0, x) + \int_0^t \mathbf{1}^\top \mathbb{A}(y)\, p_{FSP}(s, \mathcal{X}) ds - \|p_{FSP}(t, \mathcal{X})\|_1$*. (Here,* $\epsilon(t)$ *is computable because the function* $p_{FSP}(t, \cdot)$ *has finite support.)*

Proof. The proof is given in the section **S.2.1** of the supplementary material. □

The error bound provided in Theorem 2 is computable because the FFSP solution $\pi(t, \mathcal{X})$ only has finitely many non-zero elements at each time point. This computable error bound provides us with a measure to evaluate the performance of the FFSP algorithm and tells users how trustable the algorithm is. Also, as the size of $\mathcal{X}_J$ grows to the full (infinite) state-space $\mathcal{X}$, both $\epsilon(t_0)$ and and $\tilde{\epsilon}(t)$ converge to zero, which in turn suggests the convergence of $\epsilon(t_k)$ $(k = 1, 2, \dots)$ to zero as well. This observation indicates that we can indefinitely improve the accuracy of the algorithm by expanding the size of the truncated state space.

Despite this positive attribute, this error bound also has a major drawback that the error $\epsilon(t_k)$ grows exponentially fast along the observable jump events due to the multiplication of $\epsilon(t_k)$ by $\bar{a}^{\mathcal{O}_k}$ in the error updating. Because of this exponential growth rate, the error bound can quickly reach a very large number in some cases (see Section 3). Though a large error bar does not necessarily mean Algorithm 1 is inaccurate (see Section 3 for many examples), it provides practitioners less confidence about this computational algorithm, which is still not preferred. In what follows, we mitigate this problem by proposing another FFSP algorithm with a tighter error bound at a cost of narrower applicability. We have to highlight that both the algorithms produce the same result up to a constant of normalisation. In this regard, the first algorithm exploits the most natural strategy to address the filtering problem in the noise-free case. Moreover, it is also versatile enough to be applicable to most chemical reacting systems. However, it leads to a rather conservative error bound for the approximate solution. In contrast, the second algorithm exploits a different normalisation strategy and leads to an exact error certificate for the approximate solution. Nonetheless, a drawback of the second algorithm lies in the fact that the stochastic reaction network, to which it is applied, needs to satisfy conditions (11) and (12).

2.4 Another FFSP algorithm with tighter error bounds

In this section, we propose another FFSP algorithm with tighter error bounds. Note that in [7], the error of the FSP algorithm can be exactly computed due to the dominance property between the FSP solution and the exact one (i.e., FSP solution is always less than or equal to the exact solution component-wise). However, in Algorithm 1, we lose this property immediately after the normalization step, when the total mass of the FFSP solution becomes 1, and therefore it cannot always be dominated by another probability distribution unless they are strictly equal (which rarely happens). This complicates the error analysis for Algorithm 1 and leads to the somehow conservative error bound in Theorem 2.

From these observations, we provide a new FFSP algorithm by modifying the normalization step so that the exact solution always dominates the FFSP solution; by doing so, we can explicitly express the error of this new FFSP algorithm. More specifically, when approximating the filter at the next observable jump time, the new algorithm divides $\rho_{\text{FFSP}}(t_{k+1}, x)$ $(k = 1, 2, \dots,)$ by a quantity that upper bounds the denominator in (5) rather than $\sum_{x \in \mathcal{X}_J} \sum_{j \in \mathcal{O}_k} a_j\left(x, \mathbf{Y}(t_k)\right) \rho_{\text{FFSP}}\left(t_{k+1}^-, x\right)$. We present the new algorithm in Algorithm 2 and its estimation error in Theorem 4.

Theorem 4. *Under conditions* (2) *and* (11)*, Algorithm 2 has the properties that*

- $\pi_{FFSP}(t, x) \leq \pi(t, x)$ *for all* $t \geq 0$ *and* $x \in Z_{\geq 0}^{n_1}$*, and, therefore,*
- *the estimation error is given by* $\|\pi_{FFSP}(t, \mathcal{X}) - \pi(t, \mathcal{X})\|_1 = 1 - \|\pi_{FFSP}(t, \mathcal{X}_J)\|_1$ *for all* $t \geq 0$*, where* $\pi_{FFSP}(t, \mathcal{X}) \triangleq (\pi_{FFSP}(t, x_1), \pi_{FFSP}(t, x_2), \dots)^\top$ *and* $\pi(t, \mathcal{X}) \triangleq (\pi(t, x_1), \pi(t, x_2), \dots)^\top$.

Algorithm 2 Another FFSP algorithm with a tighter error bound

Require: The observation $\mathbf{Y}(t_k)$ and the initial distribution of the hidden states $\pi(0,x)$.

1: Initialization: $\pi_{\text{FFSP}}(0,x) = \pi(0,x)$, $\forall x \in \mathcal{X}_J$.

2: **for** $k = 0, 1, 2, \ldots,$ **do**

3: Approximate the un-normalised filter before the next jump event:

$$\rho_{\text{FFSP}}(t, \mathcal{X}_J) = \exp\left\{\mathbb{A}_J(\mathbf{Y}(t_k))(t - t_k)\right\} \pi_{\text{FFSP}}\left(t_k, \mathcal{X}_J\right) \qquad t \in (t_k, t_{k+1})$$

4: Estimate an upper bound for the normalisation factor before the next jump:

$$c_{\text{FFSP}}(t) = 1 + \int_{t_i}^{t} \mathbf{1}^T \mathbb{A}(\mathbf{Y}(t_k))\, p_{\text{FFSP}}\left(s, \mathcal{X}\right) ds \qquad t \in (t_k, t_{k+1})$$

5: Estimate an upper bound for the normalisation factor at the next jump:

$$\begin{aligned} c_{\text{FFSP}}(t_{k+1}) = &\sum_{x \in \mathcal{X}_J} \sum_{j \in \mathcal{O}_{k+1}} a_j\left(x, \mathbf{Y}(t_k)\right) \rho_{\text{FFSP}}(t_{k+1}^-, x) \\ &+ \bar{a}^{\mathcal{O}_{k+1}} \left[c_{\text{FFSP}}(t_{k+1}^-) - \left\|\rho_{\text{FFSP}}(t_{k+1}^-, \mathcal{X})\right\|_1\right] \end{aligned}$$

with $\bar{a}^{\mathcal{O}_{k+1}} \triangleq \sup_{x \in \mathbb{Z}_{\geq 0}^{n_1}} \left\{\sum_{j \in \mathcal{O}_{k+1}} a_j(x, \mathbf{Y}(t_k))\right\}$.

6: Normalisation: for every $x \in X_J$, we compute

$$\begin{aligned} \pi_{\text{FFSP}}(t, x) =& \frac{\rho_{\text{FFSP}}(t, x)}{c_{\text{FFSP}}(t)} \qquad t \in (t_k, t_{k+1}) \\ \pi_{\text{FFSP}}(t_{k+1}, x) =& \frac{\sum_{j \in \mathcal{O}_{k+1}} a_j\left(x - \nu_j', y(t_k)\right) \rho_{\text{FFSP}}\left(t_{k+1}^-, x - \nu_j'\right)}{c_{\text{FFSP}}(t_{k+1})}. \end{aligned}$$

7: **end for**

Proof. The proof is given in section **S.2.1**. □

The second part of Theorem 4 tells that we can exactly estimate the error of Algorithm 2 rather than obtaining a conservative error bound. This benefit mainly comes from the new normalization step, where we divide the un-normalised FFSP solution by a quantity larger than the actual normalization factors in (4) and (5). In addition to this theoretical result, our numerical examples in Section 3 also show that this second FFSP algorithm can provide a much tighter error bound than the first algorithm. All these results suggest that at this stage, the second FFSP algorithm (Algorithm 2) can provide estimates of lower error bounds than the first FFSP algorithm (Algorithm 1).

However, we need to mention that the tighter error bound of Algorithm 2 does not necessarily mean that this second FFSP algorithm is always more accurate than the first one. Mainly, this tighter error bound is obtained by a more elegant analysis technique which does not work for the first algorithm, but we cannot preclude the possibility that the first algorithm is actually more accurate than the second one. Our numerical examples in Section 3 also illustrate that the actual error of the first FFSP algorithm can be extremely tiny even though its error bound is very large. Moreover, by requiring the dominance property for the second FFSP algorithm, the solution of Algorithm 2 has a leaky property (similar to the original FSP in [7]) that its total mass constantly flows out of the truncated state space. Consequently, the second FFSP can lose most of its mass at some time point and stop being functioning since then. This leakiness is also observed in some numerical examples in Section 3, and in these cases, the first FFSP can perform much more accurately than the second one.

Furthermore, we have to point out that, even though we recover the dominance property for the approximate solution by exploiting a different normalisation strategy, the output of Algorithm 2 is not necessarily a probability distribution, as the total mass may be less than 1. In fact, Algorithm 1 and Algorithm 2 are equivalent up to a constant of normalisation, as their unnormalised filters follow the same linear equation and the algorithms only differs in the normalization step. The key difference is, that, the former employs a more natural strategy for solving the filtering problem, which gives rise to a probability distribution as an output. The latter, instead, utilises a different constant of normalisation to recover the dominance property, which therefore outputs an un-normalised probability distribution.

Also, the second FFSP has a narrower working range compared to the first FFSP algorithm. In the normalization step, the second FFSP algorithm requires the system to satisfy the condition (11), whereas the first FFSP algorithm does not. This suggests that the first FFSP can be applied to a broader class of chemical reacting systems. Nevertheless, both algorithms need this condition to generate error bounds.

Comparisons of these two FFSP algorithms are summarized in Table 1.

	First FFSP algorithm (Algorithm 1)	Second FFSP algorithm (Algorithm 2)
Error bound	A loose error bound	An exact error bound 👍
Actual accuracy	Unknown	Computable
Applicability	All chemical reacting systems 👍	Chemical reacting systems satisfying (11)

Table 1. Comparisons of the two FFSP algorithm in terms of the error bound, actual accuracy, and applicability.

3 Numerical Results

Now, we illustrate our filters using several biologically relevant examples. The experiments were performed on a laptop with a 2.3 GHz Dual-Core CPU, and the code is available on GitHub: Numerical Experiments Code.

3.1 A Simple Transcription-Translation Model

We start by considering a simple transcription-translation network containing basic patterns in the central dogma of molecular biology. The network consists of three reactions:

$$\text{mRNA} \xrightarrow{c_1} \text{mRNA} + \text{Protein}, \qquad \varnothing \xrightarrow{c_2} \text{mRNA}, \qquad \text{mRNA} \xrightarrow{c_3} \varnothing, \tag{13}$$

where an mRNA molecule translates proteins as well as being transcribed and degraded. We also assume that the protein degradation proceeds much slower than the above reactions so that in a relatively short time interval, the protein degradation can be ignored. In this example, we choose to continuously observe protein dynamics and estimate the mRNA copy number, and therefore we let $\mathbf{X}(t) \in \mathbb{Z}_{\geq 0}$ keep track of the mRNA copy number and $\mathbf{Y}(t) \in \mathbb{Z}_{\geq 0}$ keep track of the protein copy number.

3.1.1 FFSP vs. Kalman Filter and Particle Filter

We first compare the performance of our FFSP to the classical Kalman filter and the particle filter introduced in [36]. We generated the hidden and observation process trajectories using a Gillespie simulation algorithm. To construct the Kalman filter, we approximate the dynamics of the concentrations $\left(\mathbf{X}^{\Omega}(t), \mathbf{Y}^{\Omega}(t)\right) = \left(\frac{\mathbf{X}(t)}{\Omega}, \frac{\mathbf{Y}(t)}{\Omega}\right)$ (where Ω is a sampling volume) with the linear noise approximation (see [12] for a review) and establish Kalman filters on top of it. Then, we used the generated observation process trajectory as input for the first FFSP algorithm (Algorithm 1) and the particle filter; whereas, for the Kalman filter, we employed the same observation trajectory re-scaled with sampling volume Ω (see section **S3** of the Supporting Material for more details). The generated hidden process trajectory functions as a mean to test the quality of the estimates provided by the three filters. Moreover, we chose $\mathcal{X}_J = \{0, 1, \ldots, 500\}$ as the truncated state space for the FFSP, $N_{tot} = 10000$ as the particle size of the particle filter, $\Omega = 100$ as sampling volume, and $t_f = 1s$ as final simulation time, and $\mathbf{Z}_0 = [0, 0]$ as initial state. We assumed that the network follows mass-action kinetics and the propensity functions are $a_1(z_1, z_2) = c_1 z_1$, $a_2(z_1, z_2) = c_2$, $a_3(z_1, z_2) = c_3 z_1$. For the reaction rates, we set $c_1 = 100, c_2 = 10, c_3 = 5$.

Finally, the numerical results are shown in Fig. 3. In this figure, the observation process is monotonically increasing as protein degradation is absent in our model. To evaluate the performance of these three filters, we plot all their estimates together with the exact trajectory of the hidden process. As it can be seen, the FFSP and the particle filter show a good agreement in capturing the general trend of the hidden process, contrarily from the Kalman filter. Specifically, the Kalman filter is unable to follow the growth of the hidden process and even gives negative-valued estimates. This can be due to the fact that the linear noise approximation is not always a good approximation of the original model especially when the molecular copy numbers are not very large, and, furthermore, this approximate model cannot guarantee the non-negativity of the approximated solution. Another reason for the failure of the Kalman filter could be the Gaussian assumption, as it is not the most appropriate for several biological examples. All these results indicate the superior performance of our FFSP over the Kalman filter in this example.

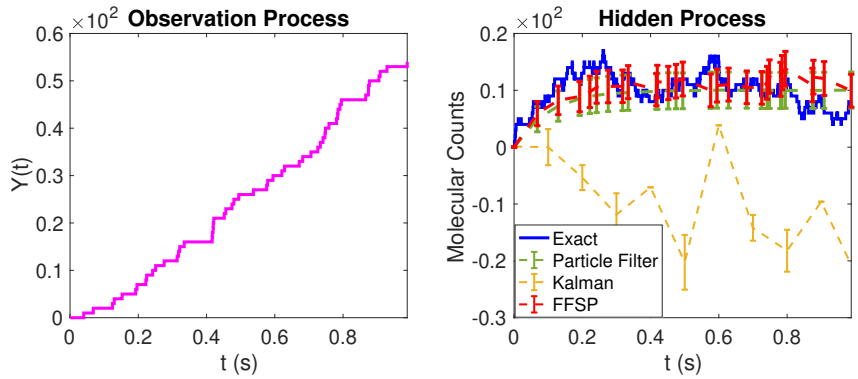

Figure 3. Performance of the FFSP, Kalman filter, and particle filter in the simple transcription-translation model. The first plot shows protein dynamics (the observation process) up to the final time. The second plot shows the exact trajectory of the mRNA dynamics (the hidden process) and the conditional mean estimates by three stochastic filters together with their standard deviations. From the second plot, the FFSP and particle filter show good agreements on the estimations; on the contrast, the Kalman filter fails to accurately estimate the hidden process, even it provides negative estimates.

3.1.2 Comparison of two FFSP algorithms

We focus now on comparing the performance of the two proposed FFSP algorithms concerning the error propagation of the approximate filtering solution. To this end, we test the chemical reaction network in Eq. (13) with the same hidden (mRNA) and observed (protein) processes. To be consistent with the error analysis hypotheses, we chose the following propensity functions: $a_1(z_1, z_2) = \frac{c_1 z_1}{c_4 + z_1}$, $a_2(z_1, z_2) = c_2$, and $a_3(z_1, z_2) = c_3 z_1$ so that the propensity of the observable reaction (the first one) is upper bounded. The propensity functions of the observable reactions play a crucial role in the error bound behaviour (see Theorem 2). Therefore, for different values of the parameters c_1 and c_4, we may obtain distinct error bounds and dynamic scenarios. In Figs. 4 to 6 we show three different behaviours of the error bounds and of the dynamics over time. In all the three settings, we again simulated the observation and hidden process trajectories with a Gillespie algorithm and fed the former as input to both FFSPs algorithms. Furthermore, we set $\mathcal{X}_J = \{0, 1, \ldots, 500\}$ as hidden species truncated state, $\mathbf{Z}_0 = [5, 0]$ as initial condition and $t_f = 2s$.

Example 1: Low protein production rate. In this first case, we set $c_1 = 25$, $c_2 = 4$, $c_3 = 1$, and $c_4 = 10$, so that the protein production rate saturates at a relatively low value 25. From the numerical result shown in Fig. 4, we can see that the observation process (the protein dynamics) still increases quite rapidly, in contrast to the mRNA copy number, which in turn fluctuates around the same value set as initial condition. Moreover, the two FFSP algorithms perform incredibly well in estimating the hidden process and show a very good agreement on all the estimates.

However, the error-bound growth tends to behave differently in the two algorithms (see Table 2).

For this choice of parameters, the propensity $a_1(z_1, z_2)$ tends to have lower values at each jump time; therefore, every time it gets pre-multiplicated with the error of the previous jump time (see Theorem 2), there is less accumulation of the error over time. Nevertheless, as it can be seen in Table 2, the error bound of the first FFSP algorithm grows immediately after the first few jump times. Such a feature can be expected, given

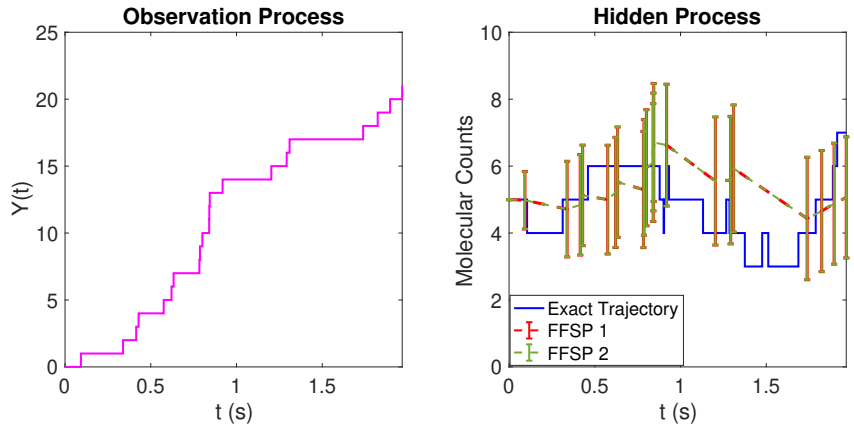

Figure 4. Comparison of the two FFSP algorithms in the simple transcription-translation model with a low protein production rate. The first plot shows the protein dynamics (the observation process) up to the final time. The second plot shows the exact trajectory of the mRNA dynamics (the hidden process) and the conditional mean estimates by the two FFSP algorithms (FFSP 1 and FFSP 2) together with their standard deviations. Specifically, the FFSP algorithms succeed in following the trend of the mRNA dyanmics and show a good agreement on their estimates, corroborating their equivalence.

	t_0	t_1	t_4	t_8	t_{14}	t_{17}	t_{21}
Jump times	0	0.09	0.43	0.78	0.92	1.3	1.9
1st FFSP's error bound	0	1×10^{-1}	2	2	2	2	2
2nd FFSP's error bound	0	$< 10^{-15}$	$< 10^{-15}$	4.9×10^{-15}	8.2×10^{-12}	6.9×10^{-9}	1.6×10^{-4}
Filter difference	0	$< 10^{-15}$	$< 10^{-15}$	4.9×10^{-15}	8.2×10^{-12}	6.9×10^{-9}	1.6×10^{-4}

Table 2. Error bounds of the two FFSP algorithms in case of the low protein production rate. The last line shows the L_1 distance between the conditional distributions provided by the two FFSP algorithms.

the poor upper bound. On the contrary, the error bound of the second FFSP algorithm is incredibly small for the first few jump times; then, we can notice a moderately rapid growth, but it stays pretty low until the last jump time with an order of magnitude of 10^{-4}).

The last row of Table 2 shows the L_1 difference in the values of the conditional distributions obtained with the two FFSP algorithms, which suggests that these two filters provide incredibly similar results. To be more specific, the difference between the two filters is almost indistinguishable until t_{14} (order of magnitude from 10^{-15} to 10^{-12}) and slowly grows towards the end of the simulation, remaining appreciably low (10^{-4}). We have to recall that Algorithm 1 and Algorithm 2 are equivalent up to a constant of normalisation, which is very different in the two algorithms. The diverse normalisation strategy can be imputable of the difference in the filters values displayed towards the end of the simulation. To summarise, the second FFSP algorithm is notably very accurate, and the estimates provided by the first FFSP algorithm are considerably close to the result obtained by second one. Therefore, we can claim that Algorithm 1 is also very accurate, even though its conservative error bound grows incredibly fast from the beginning. Moreover, we cannot choose one filter against the other. Specifically, with the triangle inequality, it is possible to show that the L_1 distance of the approximate solution provided by Algorithm 1 ($\rho_{FFSP_1}(t, \mathcal{X})$) from the exact one $\rho(t, \mathcal{X})$, namely $\mathcal{E}_{FFSP_1}$, is included in the following interval $[|\mathcal{E}_{FFSPs} - \mathcal{E}_{FFSP_2}|, \mathcal{E}_{FFSPs} + \mathcal{E}_{FFSP_2}]$. Here $\mathcal{E}_{FFSPs}$ represents the L_1 distance between the two approximate solutions obtained with Algorithm 1 ($\rho_{FFSP_1}(t, \mathcal{X})$) and Algorithm 2 ($\rho_{FFSP_2}(t, \mathcal{X})$); whereas, $\mathcal{E}_{FFSP_2}$ is the $L1$ distance between the approximate solution obtained with Algorithm 2 ($\rho_{FFSP_2}(t, \mathcal{X})$) and the exact one $\rho(t, \mathcal{X})$. Then, we can observe that, for example, during the early stages of the simulation (up to t_8), $\mathcal{E}_{FFSP_1}$ is included in an interval with a significant gap in the order of magnitude (10^{-16} to 10^{-14}), while $\mathcal{E}_{FFSP_2}$ has the order of 10^{-15}. Therefore, we may conclude that both filters are appreciably accurate, and there is no reason to discard one or the other.

Example 2: High protein production rate In this case, we set $c_1 = 40, c_2 = 4, c_3 = 1, c_4 = 20$ so that the protein production rate saturates at a relatively high value 40. In Fig. 5, we again show the observation process dynamics on the left side and the hidden dynamics, together with the filter estimates for the conditional means and standard deviations on the right. We can notice a large number of jump times in the protein dynamics, leading to rapid growth in a brief time. On the other hand, the mRNA copy number stays around the same value as the initial condition, like in the previous case. Again, the two FFSP algorithms show a good match in estimating the hidden process' conditional expectations and standard deviations, but their estimates diverge after 1.8 seconds.

In this setting, the choice of parameters may lead to higher propensity values and, therefore, to a significant accumulation of the error bound over time. This is, in fact, confirmed by the error bound dynamics shown in Table 3. We can notice that the error bound of the first FFSP algorithm overgrows already from the start of the simulation. This behaviour is to be expected again, given the conservative error estimate. In contrast, the error bound of the second FFSP algorithm tends to be very low until the 18th jump time but then grows quite fast, reaching a relatively high order of magnitude (10^{-1}). Even though the second FFSP algorithm is not as accurate as in the first case, it can still be considered accurate before 1.8 seconds. After 1.9 seconds, the second FFSP algorithm loses more than one third of the total mass and therefore can not be regarded as an accurate filter.

Despite this large gap in error bounds, the difference in the conditional distribution estimated values obtained by the two filters (last row of Table 3) again shows that the

two algorithms produce appreciably similar approximated solutions up to t_{18}, when the second filter is still accurate. Therefore, until t_{18}, we can assert that both filters produce truly accurate estimates, without preferring one filter over the other, for the same reasons as in **Example 1**. Afterwards, the second filter tends to become inaccurate and we do not have means for discussing the accuracy of the first filter, given its poor error bound. In summary, we can conclude that the values of the observable propensity functions and the number of jumps of the observation process deeply impact the accuracy of the filters.

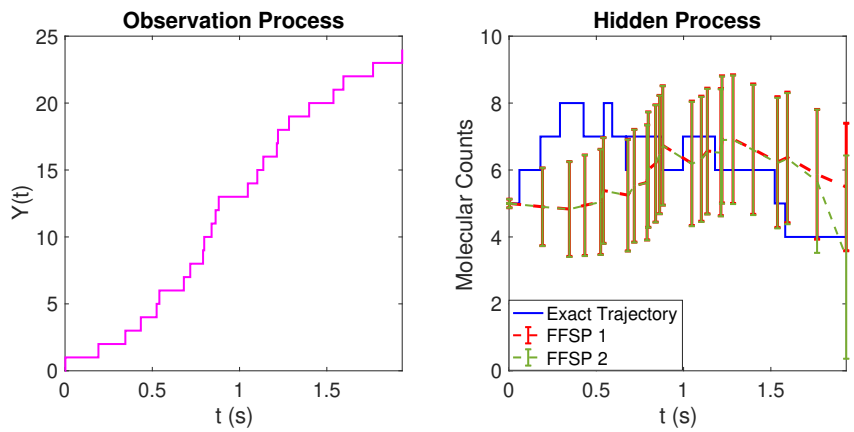

Figure 5. Comparison of the two FFSP algorithms in the simple transcription-translation model with a high protein production rate. The first plot shows protein dynamics (the observation process) up to the final time. The second plot shows the exact trajectory of the mRNA dynamics (the hidden process) and the conditional mean estimates by the two FFSP algorithm together with their standard deviations. Both the FFSP algorithms (FFSP 1 and FFSP 2) perform really well in estimating the hidden process dynamics, but their mean estimates diverge after around 1.8 seconds.

	t_0	t_1	t_2	t_5	t_{18}	t_{21}	t_{23}	t_{24}
Jump times	0	1.8×10^{-3}	0.19	0.52	1.2	1.5	1.8	1.9
1st FFSP's error bound	0	8.8×10^{-5}	0.47	2	2	2	2	2
2nd FFSP's error bound	0	$< 10^{-15}$	$< 10^{-15}$	$< 10^{-15}$	1.2×10^{-7}	1.9×10^{-4}	2.9×10^{-2}	3.7×10^{-1}
Filters difference	0	$< 10^{-15}$	$< 10^{-15}$	$< 10^{-15}$	1.2×10^{-7}	1.9×10^{-4}	2.9×10^{-2}	3.8×10^{-1}

Table 3. Error bounds of the two FFSP algorithms in case of the high protein production rate. The last line shows the L_1 distance between the conditional distributions provided by the two FFSP algorithms.

Example 3: High protein production rate but less observable jumps. Now, we investigate how different observation trajectories affect the filters' performance. To this end, we still keep $c_1 = 40, c_2 = 4, c_3 = 1, c_4 = 20$ as in the previous example but collect a simulation where the observation process jumps less frequently. In this case, we should observe less error propagation over time, and the error bounds should attain lower values at the final jump time.

As can be seen in Fig. 6, the two FFSP algorithms satisfactorily capture the behaviour of the hidden process. However, as shown in Table 4, the error bound of the first FFSP algorithm still grows very fast after the second jump time; whereas the error bound of the second FFSP algorithm behaves overall nicely and tends to grow slower

than in the previous example, attaining an order of magnitude of 10^{-4} at the final time. From these results, we can conclude that the for the same system, the error bound of the second FFSP algorithm is largely influenced by the shape of the observation trajectory, especially the number of observable jumps. On the contrast, the error bound of the first FFSP algorithm does not show such an influence.

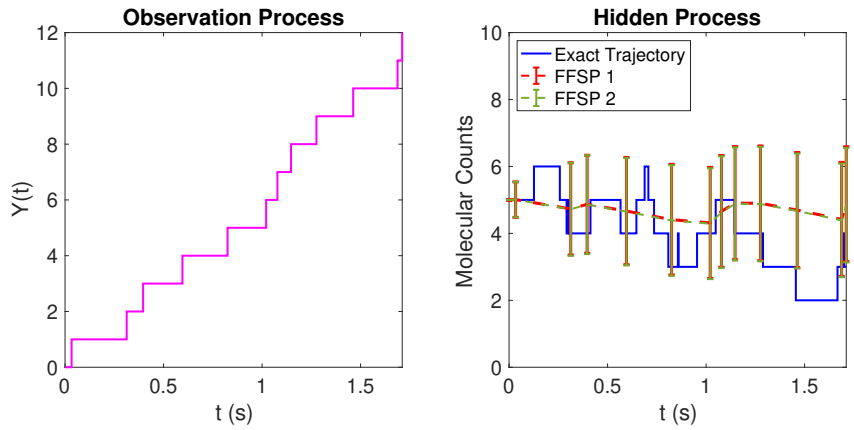

Figure 6. Comparison of the two FFSP algorithms in the simple transcription-translation model with a high protein production rate but less observable jumps. The first plot shows protein dynamics (the observation process) up to the final time. The second plot shows the exact trajectory of the mRNA dynamics (the hidden process) and the conditional mean estimates by the two stochastic filters together with their standard deviations. The two FFSP algorithms (FFSP 1 and FFSP 2) agree on all their estimates of the hidden process.

	t_0	t_1	t_2	t_3	t_5	t_8	t_{10}	$t_{11}(=t_f)$
Jump times	0	0.033	0.31	0.39	0.83	1.2	1.5	1.7
1st FFSP's error bound	0	2.4×10^{-2}	2	2	2	2	2	2
2nd FFSP's error bound	0	$< 10^{-15}$	$< 10^{-15}$	2.1×10^{-15}	1.5×10^{-12}	2.7×10^{-9}	8.6×10^{-7}	1.6×10^{-4}
Filters difference	0	$< 10^{-15}$	$< 10^{-15}$	2.1×10^{-15}	1.4×10^{-12}	2.7×10^{-9}	2.5×10^{-5}	1.6×10^{-4}

Table 4. Error bounds of the two FFSP algorithms in case of the high protein production rate but less observable jumps. The last line shows the L_1 distance between the conditional distributions provided by the two FFSP algorithms.

3.1.3 FFSP vs. Particle filter: Conditional Distribution Estimation

Now, we consider the same network (13) but revert the observed and hidden species, i.e., we let $\mathbf{X}(t)$ keep track of the protein copy number and $\mathbf{Y}(t)$ of the mRNA copy number. In this setting, the protein copy number is solely determined by the first reaction channel, and therefore the conditional distribution $\pi(t,x)$ follows a Poisson distribution with parameter $\lambda = \int_0^t c_1\mathbf{Y}(s)ds$. We again assume mass-action kinetics for the network and therefore the propensity functions remain the same as the first example. However, for reaction rates parameters, we set $c_1 = 1, c_2 = 5, c_3 = 1$.

In this case, we test the performance of the FFSP filter (Algorithm 1) against the particle filter [36] in estimating the hidden dynamics, together with the conditional distribution. We again employed the Gillespie algorithm to generate the hidden and observed trajectories, and we then used the generated observation trajectory as input for both filters. We set the hidden species state space to be $\mathcal{X} = \{0, 1, \ldots, 200\}$ for the

FFSP filter, $N_{tot} = 10000$ as particle size, $t_f = 5s$ as final time and $\mathbf{Z}_0 = [0, 5]$ as initial state. The numerical experiments are then shown in Fig. 7.

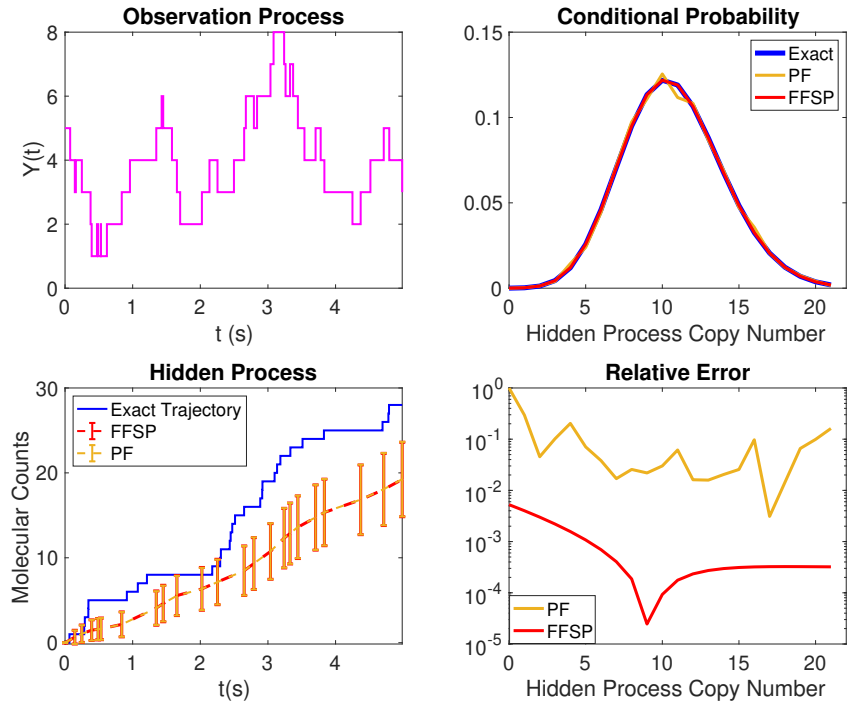

Figure 7. Comparison of the FFSP and particle filter in the simple transcription-translation model (observing the mRNA dynamics). The left column shows the observation and hidden process trajectories, together with the stochastic filters estimates and standard deviations, until the final simulation time. The top right plot, instead, displays a comparison between the exact conditional distribution and the ones estimated by the FFSP (Algorithm 1) and the particle filter at t=3s. The bottom right plot, instead, depicts the relative error of the filters from the exact conditional distribution. The result shows that the two filters have a similar performance in estimating the conditional mean and variance, but the FFSP is far more accurate in approximating the exact conditional distribution. Moreover, the particle filter takes 29.9s to be executed and the FFSP, in contrast, 11.3s, being 3 times faster.

As we can notice in the top left plot, the mRNA copy number fluctuates over time, with rapid outbursts that enhance protein translation. In this regard, we plotted the hidden process (protein) copy number in the bottom left plot as a validation of the filters estimates (conditional expectations). We can observe that the protein copy number increases rapidly over time, with the exception of some regions in which the mRNA copy numbers suddenly decreases. Overall, the two filters agree on all the estimates over time, with a general underestimation of the actual values towards at the beginning and at the end of the simulation analysis. The underestimation of the protein copy number might be due to the fact that the protein increases very rapidly at the initial time even though the mRNA copies tend to decrease. Therefore, the filters fail to follow this unexpected sharp increase at the beginning and keep underestimating the values for this whole interval.

Although the particle filter performs accurately in estimating the conditional means and standard deviations, we can discern, in the top right plot, an overall supremacy of

the FFSP in reproducing the exact conditional distribution, e.g., when estimating the peak probability. Such a behaviour can be expected, as the FFSP is a direct method, and therefore more accurate at depicting the behaviour of probability distributions. On the contrary, being particle filter a Monte-Carlo method, its performance depends on the number of particles employed in the algorithm and how well they locate at a certain state. More specifically, let $V_1(t), \dots, V_N(t)$ be N particles sampled from the conditional distribution; a particle filter use a scaled binomial distribution $\frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(V_i(t) = x)$ to approximate the conditional probability at the state x. In this case, the coefficient of error (the ratio between the standard deviation and the mean of this scaled binomial distribution) becomes $\frac{\sqrt{1-p}}{\sqrt{Np}}$ with p being the probability of finding the particle in that specific state, suggesting that the particle filter is not particularly suited in estimating rate events when p is small. This phenomenon is spotted in the bottom right plot of Fig. 7: the particle filter has very big relative error when estimating the state $x = 0$ whose conditional probability value is approximately 4.5×10^{-5}. In contrast, the relative error of the FFSP is always considerably small, even in estimating rare events.

In conclusion, though the FFSP and the particle filter have similar performance in estimating the conditional mean and variance, the FFSP has a much higher accuracy when estimating specific conditional probability, especially the rare events.

3.2 Genetic Toggle Switch

We consider now the following network, also known in biology as Toggle Switch:

$$\varnothing \xrightarrow{\frac{\alpha_1}{1+z_2^{\beta}}} S_1, \qquad S_1 \xrightarrow{1} \varnothing, \qquad \varnothing \xrightarrow{\frac{\alpha_2}{1+z_1^{\gamma}}} S_2, \qquad S_2 \xrightarrow{1} \varnothing \tag{14}$$

which is composed of two species mutually repressing each other. This elementary network shows properties like multi-stability and fluctuations that are desirable in different contexts in biology. Along these lines, a relatively new synthetic biology advance was implementing a genetic toggle switch in *Escherichia Coli* [42]. In this example, we decide to continuously observe the S_2 copy number and estimate the dynamics of S_1, and therefore we let $\mathbf{X}(t) \in \mathbb{Z}_{\geq 0}$ keep track of the S_1 copy number and $\mathbf{Y}(t) \in \mathbb{Z}_{\geq 0}$ of S_2. To be consistent with the mutual repression of the two species, we chose the subsequent propensity functions: $a_1(z_1, z_2) = \frac{\alpha_1}{1+z_2^{\beta}}$, $a_2(z_1, z_2) = z_1$, $a_3(z_1, z_2) = \frac{\alpha_2}{1+z_1^{\gamma}}$, and $a_4(z_1, z_2) = z_2$.

The following numerical examples test the two FFSP algorithms on the latent states' estimates and error bounds computation. Specifically, we chose two different parameters' scenarios so that distinct dynamics and accuracy analyses arise. In all the examples, we set $\mathbf{Z}_0 = [0, 0]$ as initial state and $\mathcal{X} = \{0, \dots, 200\}$ as hidden species truncated state space and $t_f = 5s$ as final simulation time. We again generated the observed and hidden process trajectories with a Gillespie algorithm and fed the former as input to both the filters.

Strong Repression In this case, we set $\alpha_1 = 16, \alpha_2 = 14, \beta = 1, \gamma = 1$ so that the the two species strongly repress each other's production. For this choice of parameters, switching behaviour is almost always guaranteed. Any time one of the two species is present in higher abundances, the other species' values shrink to zero almost immediately. Such a behaviour can be seen in Fig. 8. The left-sided plot shows an initial slow increase of the observed (S_2) species until time $t = 4s$, with a sharp outburst towards the end. In contrast, the right-sided plot shows the rapid increase of the hidden process from the beginning of the simulation and a rapid shrinking of its values towards the simulation's final time, as expected. Moreover, the right-sided plot, in Fig. 8, also

shows the stochastic filters' estimates and their standard deviations. Generally, we can see that both FFSP filters capture the hidden process dynamics everywhere; for the error behaviour, both filters behave nicely, providing very low error bounds, especially second FFSP algorithm whose error is always below 10^{-15}. In this case, we cannot distinguish whether it is an error of our algorithm or an error of the floating point.

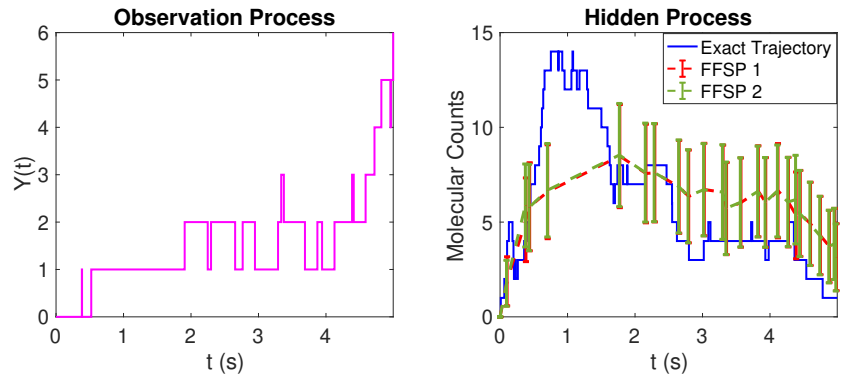

Figure 8. Performance of the two FFSP algorithms in the genetic toggle switch model (strong repression). The first plot shows the S_2 (the observation process) copy number up to the final time. The second plot shows the exact trajectory of the S_1 dynamics (the hidden process) and the conditional mean estimates by the two stochastic filters together with their standard deviations. The two FFSP algorithms (FFSP 1 and FFSP 2 in the figure's legend) agree on all the estimations of the hidden process and succeed in following the trend of the hidden process.

Mild Repression In this case, we chose $\alpha_1 = 18, \alpha_2 = 15, \beta = 1.5, \gamma = 0.01$, so that the first species repress the second one very mildly. In Fig. 9, we plot the observed and hidden process trajectories and filters' estimates, together with their standard deviations (on the left and right side of the plot, respectively). In the first part of the simulation, both filters agree on the estimations and succeed in following the trend of the hidden process. However, both filters underestimate the outbreak of the hidden species because the size of this outbreak is unexpectedly large. Even when there is no repression, every second there are 7.5 newly produced S_2 copies in average, which is only half of the size of this outbreak (see Fig. 9). In the second part of the simulation, when the hidden process values shrink, both filters can capture such a rapid decrease.

Initially, the two filters can be both very accurate: as it can be seen in Table 5, the difference in their estimates is incredibly small. Moreover, the second filter error bound tends to stay appreciably low until t_{12}. Therefore, during the beginning of the simulation, both filters are to be considered very accurate. However, after 3 seconds, the second FFSP algorithm stops functioning as it loses almost all of its mass (see Fig. 9 and Table 5). In contrast, the first filter is functioning till the end though its error bound is very large (see Fig. 9 and Table 5). To conclude, there is no strict preference among the two filters during the early stages of the simulation; the first FFSP algorithm is preferred over the final time, when it can still capture the behaviour of the hidden process despite the large error bound dynamics.

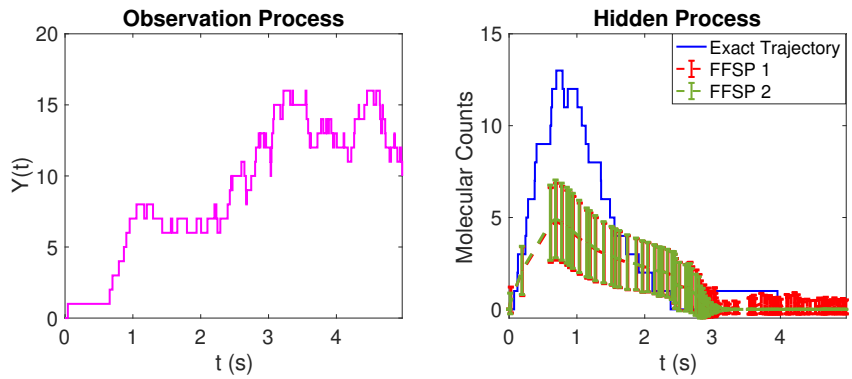

Figure 9. Performance of two FFSP algorithms in the genetic toggle switch model (mild repression). The first plot shows the S_2 (the observation process) copy number up to the final time. The second plot shows the exact trajectory of the S_1 dynamics (the hidden process) and the conditional mean estimates by the two stochastic filters together with their standard deviations. The two FFSP algorithms (FFSP 1 and FFSP 2 in the figure's legend) agree on all the estimations of the hidden process until time $t = 3s$, after which only the first FFSP filter is still able to better capture the latent states.

	t_0	t_1	t_2	t_5	t_{12}	t_{21}	t_{42}	t_{60}
Jump times	0	4.1×10^{-2}	6.6×10^{-1}	8.7×10^{-1}	1.4	2.2	2.9	3.6
1st FFSP's error bound	0	2.6×10^{-2}	9.6×10^{-1}	2	2	2	2	2
2nd FFSP's error bound	0	$< 10^{-15}$	3.6×10^{-14}	3.2×10^{-13}	1.2×10^{-9}	9×10^{-5}	7.8×10^{-1}	1
Filters difference	0	$< 10^{-15}$	3.6×10^{-14}	3.2×10^{-13}	1.2×10^{-9}	9×10^{-5}	7.8×10^{-1}	1

Table 5. Error behaviour of the two FFSP algorithms in the example of genetic toggle switch with mild inhibition

3.3 Genetic Switch: FFSP vs Particle Filter

We consider now the following network:

$$
\begin{aligned}
&G^* + \text{Protein} \xrightarrow{c_1} G && G \xrightarrow{c_2} G^* + \text{Protein} && G^* \xrightarrow{c_3} G^* + \text{mRNA} \\
&\text{mRNA} \xrightarrow{c_4} \varnothing && \text{mRNA} \xrightarrow{c_5} \text{mRNA} + \text{Protein} && \text{Protein} \xrightarrow{c_6} \varnothing
\end{aligned} \tag{15}
$$

where G and G^* are inactive and active genes, respectively activated and inactivated by the protein. Moreover, the active gene G^* transcribes mRNA copy number, which can degrade. In turn, mRNA is responsible for the protein translation, which also degrades. Here, we denote G, G^*, mRNA, and protein by S_1, S_2, S_3, and S_4, respectively. For this network, we decided to employ mass-action kinetics and, thus, the propensity functions are $a_1(z_1, z_2, z_3, z_4) = c_1 z_2 z_4$, $a_2(z_1, z_2, z_3, z_4) = c_2 z_1$, $a_3(z_1, z_2, z_3, z_4) = c_3 z_2$, $a_4(z_1, z_2, z_3, z_4) = c_4 z_3$, $a_5(z_1, z_2, z_3, z_4) = c_5 z_3$, and $a_6(z_1, z_2, z_3, z_4) = c_4 z_3$. We continuously observe protein dynamics and estimate the other remaining species; therefore, we let $\mathbf{X}(t) \in \mathbb{Z}^3_{\geq 0}$ keep track of G, G^*, and the mRNA and $\mathbf{Y}(t) \in \mathbb{Z}_{\geq 0}$ keep track of the protein.

In this numerical experiment, we compare the performance of the FFSP and the particle filter in estimating the various hidden species. To this end, we simulated the observed and hidden process trajectories with a Gillespie algorithm, and we fed the former as input to both filters. Moreover, we set $c_1 = 1$, $c_2 = 5$, $c_3 = 8$, $c_4 = 2$, $c_5 = 4$, $c_6 = 1$, $t_f = 5s$ as final simulation time, and $\mathcal{X} = \{0, \dots 100\}$ as hidden species truncated state space for the mRNA copy number.

In Section 3.3, we can spot the observation and hidden process trajectories, together with the filters' estimates and standard deviations. We can notice how the protein copy number starts with a slow increase, given that the active gene alternates periods of inactivity to activity and then experiences a sharp outburst once the gene stays mostly active. The same behaviour is overall held also by the mRNA copy number. Regarding the estimations carried out by the filters, we can see that the FFSP performs better overall than the particle filter, which tends to underestimate the actual values.

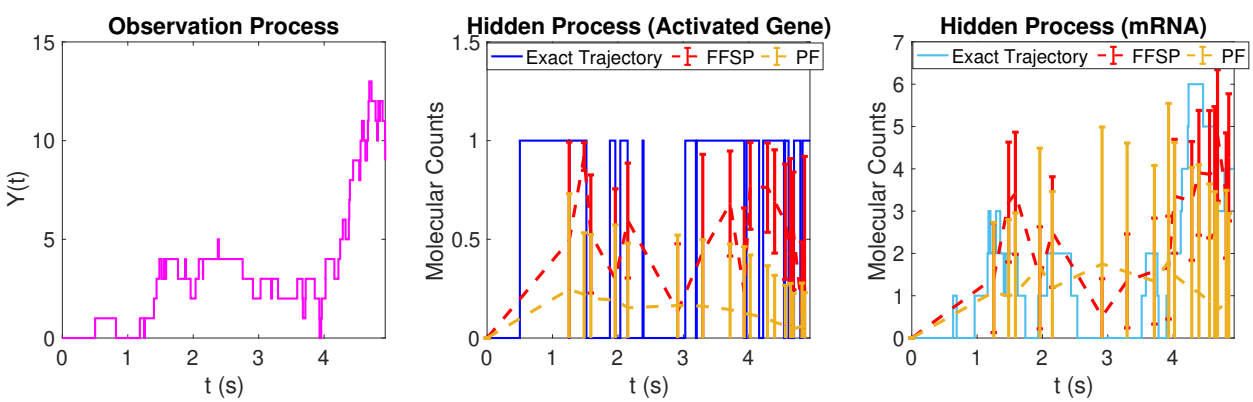

Figure 10. Performance of the FFSP and particle filter algorithms in the genetic switch model. The first plot shows the protein copy number up to the final time. The second and third plots show the exact trajectory of the activated gene and mRNA dynamics, respectively; also, we present their conditional mean estimates by the two stochastic filters together with the standard deviations. The FFSP shows a better performance in estimating both the activated gene and mRNA copy number over the particle filter, which instead underestimates the actual values overall.

Conclusion

In this paper, we developed a direct approach to solve the filtering equation for a noise-free observation process in stochastic reaction networks. Specifically, we considered a continuous-time discrete-state Markov Chain $\mathbf{Z}(t) \in \mathbb{Z}^n_{\geq 0}$ and assumed to exactly observe the dynamics of some chemical species. To postulate the filtering problem, we subsequently decomposed the network into $\mathbf{X}(t) \in \mathcal{X} \subseteq \mathbb{Z}^{n_1}_{\geq 0}$ hidden species and $\mathbf{Y}(t) \in \mathbb{Z}^{n_2}_{\geq 0}$, with $n_2 = n - n_1$, observed species. Thus, we assumed to exactly (noise-free) and time-continuously observe the trajectory of $\mathbf{Y}(t)$ and estimate the hidden species copy number $\mathbf{X}(t)$ dynamics. To be more specific, we wish to compute the conditional distribution π_t of the hidden process $\mathbf{X}(t)$ given the trajectory of the observed process $\mathbf{Y}(s)\, 0 \leq s \leq t$. In this setting, such conditional distribution is a piece-wise deterministic Markov process (PDMP): it evolves according to a non-linear dynamic equation and jumps when the observation process does so.

This conditional distribution admits an un-normalised version that is also a PDMP and evolves according to a linear evolution equation between the observation process jump times. More specifically, the evolution matrix is appreciably similar to the generator matrix of the CME and shares several valuable properties. Therefore, we solved the filtering problem by iteratively applying the Finite State Projection Method to the linear system satisfied by the un-normalised distribution between the observation process jump times. We then recovered the filtering equation's solution by normalising it before each jump time. This procedure gives rise to the Filtered Finite State Projection Method. We then developed a first accuracy analysis which results in a conservative error bound for the approximated solution. Therefore, we provided an alternative algorithm, which exploits a different normalisation technique and results in an exact error certificate as in the FSP theorem [7], at a cost of narrower applicability.

From the numerical experiments in Section 3, we can observe that the FFSP method accurately estimates the hidden dynamics of several biological networks. Moreover, our method performs better than established filters, like the Kalman filter and the particle filter, especially in estimating the full conditional probability distribution. Furthermore, our filter produces pretty accurate estimates overall if we look at the error behaviour of the proposed second algorithm.

In the future, this work could be extended in several directions. One possible direction is to mitigate the curse of dimensionality. Since our algorithm has to solve an extensive linear system between each observation process jump time, the computational cost grows exponentially with the number of species. Therefore, developing more scalable approaches for high-dimensional filtering equation in the stochastic reaction network framework is needed. A possible approach may reside in using deep learning, as it has been shown to cope quite well with high-dimensional problems in the stochastic reaction networks framework [43] and other fields [44]. Second, the algorithm could further be extended by including parameters' uncertainty, as the exact parameter values are usually unknown to scientists. Moreover, the FFSP can also be applied to solving CMEs with the Rao-Blackwell technique, which entails uncoupling the entire network into two parts and then marginalising out one by computing its conditional probability (see [45, 46] for the Rao-Blackwell technique and [33, 34] for the application of this idea together with moment closure to CMEs). The advantage of the Rao-Blackwell method is guaranteed by the Rao-Blackwell theorem (a variant of the law of total variance), suggesting this method to be more accurate than the Monte Carlo one under the same sample size. In this framework, our FFSP is ideal for the marginalisation part of the Rao-Blackwell technique when applied to biochemical reacting systems; some attempts in this direction were reported in [47].

Supporting information

S1 Supplementary Material. The proofs of the theorems. In the supplementary material, we rigorously verify the filtering equation under the assumed conditions and perform the error analysis for both FFSP algorithms.

S2 Supplementary Material. In this section of the supplementary material, we concern ourselves with the proofs of the error analysis results of Algorithm 1 and Algorithm 2.

S3 Supplementary Material. In this section, a derivation of the Kalman filter used in the numerical experiments is shown.

Acknowledgments

This work was funded by a grant from the Swiss National Science Foundation (Grant No. 182653).

References

1. Ehrenberg M. The green fluorescent protein: discovery, expression and development. Information Department, The Royal Swedish Academy of Sciences. 2008;.
2. Zhang J, Campbell RE, Ting AY, Tsien RY. Creating new fluorescent probes for cell biology. Nature reviews Molecular cell biology. 2002;3(12):906–918.
3. Stephens DJ, Allan VJ. Light microscopy techniques for live cell imaging. science. 2003;300(5616):82–86.
4. Locke JC, Elowitz MB. Using movies to analyse gene circuit dynamics in single cells. Nature Reviews Microbiology. 2009;7(5):383–392.
5. McAdams HH, Arkin A. Stochastic mechanisms in gene expression. Proceedings of the National Academy of Sciences. 1997;94(3):814–819.
6. Eldar A, Elowitz MB. Functional roles for noise in genetic circuits. Nature. 2010;467(7312):167–173.
7. Munsky B, Khammash M. The finite state projection algorithm for the solution of the chemical master equation. The Journal of chemical physics. 2006;124(4):044104.
8. Gillespie DT. Exact stochastic simulation of coupled chemical reactions. The journal of physical chemistry. 1977;81(25):2340–2361.
9. Gillespie DT. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. Journal of computational physics. 1976;22(4):403–434.
10. Gibson MA, Bruck J. Efficient exact stochastic simulation of chemical systems with many species and many channels. The journal of physical chemistry A. 2000;104(9):1876–1889.

11. Anderson DF. A modified next reaction method for simulating chemical systems with time dependent propensities and delays. The Journal of chemical physics. 2007;127(21):214107.

12. Schnoerr D, Sanguinetti G, Grima R. Approximation and inference methods for stochastic biochemical kinetics—a tutorial review. Journal of Physics A: Mathematical and Theoretical. 2017;50(9):093001.

13. Schnoerr D, Sanguinetti G, Grima R. Comparison of different moment-closure approximations for stochastic chemical kinetics. The Journal of Chemical Physics. 2015;143(18):11B610_1.

14. Grima R. A study of the accuracy of moment-closure approximations for stochastic chemical kinetics. The Journal of chemical physics. 2012;136(15):04B616.

15. Ullah M, Wolkenhauer O. Investigating the two-moment characterisation of subcellular biochemical networks. Journal of theoretical biology. 2009;260(3):340–352.

16. Bain A, Crisan D. Fundamentals of stochastic filtering. vol. 3. Springer; 2009.

17. Kalman RE. A new approach to linear filtering and prediction problems. Journal of Fluids Engineering, Transactions of the ASME. 1960;.

18. Kalman RE, Bucy RS. New Results in Linear Filtering and Prediction Theory. Journal of Basic Engineering. 1961;83(1):95–108. doi:10.1115/1.3658902.

19. Grewal MS, Andrews AP. Applications of Kalman filtering in aerospace 1960 to the present [historical perspectives]. IEEE Control Systems Magazine. 2010;30(3):69–78.

20. Stratonovich RL. Conditional Markov Processes. Theory of Probability & Its Applications. 1960;.

21. Kushner HJ. On the dynamical equations of conditional probability density functions, with applications to optimal stochastic control theory. Journal of Mathematical Analysis and Applications. 1964;.

22. Maurel MC, Michel D. Des resultats de non existence de filtre de dimension finie. Stochastics. 1984;.

23. Gordon N, Salmond DJ, Smith AFM. Novel approach to nonlinear and linear Bayesian state estimation; 1993.

24. Sun X, Jin L, Xiong M. Extended Kalman filter for estimation of parameters in nonlinear state-space models of biochemical networks. PloS one. 2008;3(11):e3758.

25. Chuang CH, Lin CL, et al. Robust estimation of stochastic gene-network systems. Journal of Biomedical Science and Engineering. 2013;6(02):213.

26. Calderazzo S, Brancaccio M, Finkenstädt B. Filtering and inference for stochastic oscillators with distributed delays. Bioinformatics. 2019;35(8):1380–1387.

27. Liu X, Niranjan M. State and parameter estimation of the heat shock response system using Kalman and particle filters. Bioinformatics. 2012;28(11):1501–1507.

28. Boys RJ, Wilkinson DJ, Kirkwood TB. Bayesian inference for a discretely observed stochastic kinetic model. Statistics and Computing. 2008;18(2):125–135.

29. Golightly A, Wilkinson DJ. Bayesian sequential inference for stochastic kinetic biochemical network models. Journal of Computational Biology. 2006;13(3):838–851.

30. Fang Z, Gupta A, Khammash M. Stochastic filtering for multiscale stochastic reaction networks based on hybrid approximations. Journal of Computational Physics. 2022;467:111441.

31. Fang Z, Gupta A, Khammash M. Stochastic filters based on hybrid approximations of multiscale stochastic reaction networks. In: 2020 59th IEEE Conference on Decision and Control (CDC). IEEE; 2020. p. 4616–4621.

32. Fang Z, Gupta A, Khammash M. Convergence of regularized particle filters for stochastic reaction networks. arXiv preprint arXiv:211007746. 2021;.

33. Zechner C, Koeppl H. Uncoupled analysis of stochastic reaction networks in fluctuating environments. PLoS computational biology. 2014;10(12):e1003942.

34. Duso L, Zechner C. Selected-node stochastic simulation algorithm. The Journal of chemical physics. 2018;148(16):164108.

35. Wiederanders HJ, Moor AL, Zechner C. Automated Generation of Conditional Moment Equations for Stochastic Reaction Networks. In: International Conference on Computational Methods in Systems Biology. Springer; 2022. p. 286–293.

36. Rathinam M, Yu M. State and parameter estimation from exact partial state observation in stochastic reaction networks. The Journal of Chemical Physics. 2021;154(3):034103.

37. Thomas PJ, Ahmadi M, Zheng H, Myers CJ. A Comparison of Weighted Stochastic Simulation Methods. bioRxiv. 2021;doi:10.1101/2021.09.21.460158.

38. Anderson DF, Kurtz TG. Stochastic analysis of biochemical systems. vol. 674. Springer; 2015.

39. Confortola F, Fuhrman M. Filtering of continuous-time Markov chains with noise-free observation and applications. Stochastics An International Journal of Probability and Stochastic Processes. 2013;85(2):216–251.

40. Gupta A, Khammash M. Unbiased estimation of parameter sensitivities for stochastic chemical reaction networks. SIAM Journal on Scientific Computing. 2013;35(6):A2598–A2620.

41. Lunz D, Batt G, Ruess J, Bonnans JF. Beyond the chemical master equation: stochastic chemical kinetics coupled with auxiliary processes. PLoS Computational Biology. 2021;17(7):e1009214.

42. Gardner TS, Cantor CR, Collins JJ. Construction of a genetic toggle switch in Escherichia coli. Nature. 2000;403(6767):339–342.

43. Gupta A, Schwab C, Khammash M. DeepCME: A deep learning framework for computing solution statistics of the chemical master equation. PLoS computational biology. 2021;17(12):e1009623.

44. Han J, Jentzen A, E W. Solving high-dimensional partial differential equations using deep learning. Proceedings of the National Academy of Sciences. 2018;115(34):8505–8510.

45. Rao CR. Information and the accuracy attainable in the estimation of statistical parameters. In: Breakthroughs in statistics. Springer; 1992. p. 235–247.

46. Blackwell D. Conditional expectation and unbiased sequential estimation. The Annals of Mathematical Statistics. 1947; p. 105–110.

47. Fang Z, Gupta A, Khammash M. A scalable approach for solving chemical master equations based on modularization and filtering. in preparation;.