Solving the chemical master equation by a fast adaptive finite state projection based on the stochastic simulation algorithm[☆]

R.B. Sidje^{a,*}, H.D. Vo^a

^a*Department of Mathematics, The University of Alabama, Tuscaloosa, AL 35487, USA*

Abstract

The mathematical framework of the chemical master equation (CME) uses a Markov chain to model the biochemical reactions that are taking place within a biological cell. Computing the probability distribution of this Markov chain allows us to track the composition of molecules inside the cell over time, with important practical applications in a number of areas such as molecular biology or medicine. However the CME is typically difficult to solve, since the state space involved can be very large or even countably infinite. We present a novel way of using the stochastic simulation algorithm (SSA) to reduce the size of the finite state projection method. Numerical experiments that demonstrate the effectiveness of the reduction are included.

Keywords: chemical master equation, model reduction, matrix exponential, stochastic simulation algorithm, finite state projection.

1. Introduction

Models of cellular processes promise great benefits in important fields such as molecular biology or medicine. When molecules exist in large numbers, models can be formulated using concentrations. And since concentrations are continuous quantities, this way of doing so allows using reaction rate equations (RREs), which are deterministic, nonlinear ordinary differential equations. Within a cell, however, some key regulatory molecules exist only in small numbers, in which case it becomes more appropriate to formulate the models in a discrete and stochastic setting. The mathematical framework that underpins this is a continuous-time, discrete-state, Markov process, and computing its probability distribution amounts to solving the chemical master equation (CME).

While promising many insights, the CME is difficult to solve, especially for large models. Consequently, researchers often resort to simulating trajectories, using most notably Gillespie's stochastic simulation algorithm (SSA) [2] or its improved variants, e.g., [1].

We outline a novel solution technique here, namely a SSA-driven reduction of the state space that builds on the principle that the CME aims at computing a probability

[☆]Supported by NSF grant DMS-1320849

*Corresponding author (email: roger.b.sidje@ua.edu)

vector $\mathbf{p} = (p_1, \dots, p_n)^T \in [0, 1]^n$, with components that sum to one, $\sum_{i=1}^n p_i = 1$. A key consequence of this is that if numerous nonnegative real numbers sum to one, then some of these numbers are necessarily zero or small. That is, for very large problems, $n \gg 1$, the probability sum condition implies that some of the components must necessarily be zero or negligible, and precisely because the CME has such extremely high dimension, the more of them there will be. As our paper will show, dropping those negligible components allows us to reduce the size dramatically.

We build on the finite state projection (FSP) of Munsky et Kammash [9] and the Krylov-FSP [8] that was an early improvement to the basic FSP. We note that there have been other developments targeted at special problems such as those with a clear separation of time-scales [14], or those reducible to one-dimensional processes [15]. Applying the quasi steady state assumption when possible can also reduce the state space in some cases [6, 16]. Several other general studies such as [5, 12, 13] have investigated ways to speed up the calculations. But solving the CME of realistic models with many species and many reactions remains a formidable challenge to all methods. Our new SSA-based reduction drives the FSP in a fast, economical, and adaptive way.

The organization of the paper is as follows. Section 2 motivates this study. Section 3 gives some background on modeling biochemical reactions by way of the CME. Section 4 summarizes the basic FSP method and introduces our proposed SSA-driven approach. Section 5 describes the original Krylov-FSP method and how to incorporate our new reduction. Section 6 shows the performance of our method on some examples. Section 7 finally gives some concluding remarks.

2. Challenge and Motivation

What makes the CME so hard to deal with is because it is exhaustive—it enumerates *all* the possible states that the cell can ever have, resulting in a size that is extremely large and difficult to handle. In reality, however, at any given time, there could be several states that are unlikely and that could be safely ignored in the calculations. These could be past states that the cell already had and is unlikely to have again, or they could be future states that are too far in the future to be relevant at present. By ignoring those improbable states, we can reduce the size to a tractable level. While this principle is quite promising [6, 9, 12], it raises the challenging points that we discuss next.

First, states that may be improbable right now can become probable the next instant because the makeup of the cell is continuously changing as a result of the reactions that are taking place within it. Hence the subset of probable states is changing over time, but the merit of our approach hinges on the core principle that this subset remains small compared to the set of all possible states that the cell can ever have and that are encoded in the full blown CME. The key to efficiency is therefore to only track this reduced subset and deliberately ignore unlikely states, in a way that is adjusted dynamically over time.

The second point is related to the first point in the following sense: how then do we distinguish probable states from improbable ones? If we do a good job in the selection, we speed up the calculations. However, if we mistakenly exclude relevant states, we ultimately end up with unreliable answers, or even worse, erroneous or meaningless answers that we may wrongly believe to be right. This is a serious concern that is not simple to resolve because a sorting strategy may come at a significant overhead due to the fact that we deal with extremely large models over possibly thousands of time steps.

Our research is bringing fresh ideas from advanced computing and differential equations into the field of stochastic chemical kinetics. We are investigating inexact (or relaxed) techniques with other novel reduction techniques. What we present in this work is a fundamental cornerstone that is shaping our investigations.

3. The Chemical Master Equation

As mentioned earlier, when there are small numbers of some molecular species, as is the case for many key regulatory elements in biological systems, it is more appropriate to use a discrete and stochastic framework to model such systems. The underlying Markov chain is briefly described in this section, which also serves to outline the terminology and notation.

Consider a biochemical system with $N \geq 1$ different chemical species that are interacting via $M \geq 1$ chemical reactions. The state of the system is a vector $\mathbf{x} \equiv (x_1, \dots, x_N)^T$ of nonnegative integers where x_i is the current population of the i th species. Transitions between states occur when (and only when) a reaction occurs. Each reaction $1 \leq j \leq M$ entails a *stoichiometric* vector $\boldsymbol{\nu}_j$ that represents the change when the reaction occurs; if the system is in state $\mathbf{x}$ and reaction j occurs, then the system transitions to state $\mathbf{x} + \boldsymbol{\nu}_j$. Each state $\mathbf{x}$ entails M *propensities* $\alpha_1(\mathbf{x}), \dots, \alpha_M(\mathbf{x})$ that determine the relative chance of each reaction occurring if the system is in state $\mathbf{x}$. The propensities are defined by the requirement that, given $\mathbf{x}(t) = \mathbf{x}$, $\alpha_j(\mathbf{x})dt$ is the probability of reaction j occurring in the next infinitesimal time interval $[t, t + dt)$, where the dependence on time has now been made explicit.

Let the probability of being in state $\mathbf{x}$ at time t be denoted by $P(\mathbf{x}; t)$ and consider the way that this changes over time. With appropriate assumptions, it can be shown, see for example Higham [4] for a readable overview, that for each state $\mathbf{x}$, the previous description of the model implies that this probability satisfies the CME, which is the following discrete, parabolic, partial differential equation:

$$\frac{\partial P(\mathbf{x}; t)}{\partial t} = \sum_{j=1}^M \alpha_j(\mathbf{x} - \boldsymbol{\nu}_j) P(\mathbf{x} - \boldsymbol{\nu}_j; t) - P(\mathbf{x}; t) \sum_{j=1}^M \alpha_j(\mathbf{x}).$$

It may be written in an equivalent matrix-vector form by enumerating all the states. In this case, if there are n possible states, $\mathbf{x}_1, \dots, \mathbf{x}_n$, the CME takes the form of a system of linear ordinary differential equations (ODEs)

$$\begin{cases} \mathbf{p}'(t) = \mathbf{A}\mathbf{p}(t), & t \in [0, t_f] \\ \mathbf{p}(0) = \mathbf{p}_0 \end{cases}, \quad (1)$$

where the probability vector $\mathbf{p} = (p_1, \dots, p_n)^T$ is such that each component $p_i = P(\mathbf{x}_i, t) = \text{Prob}\{\mathbf{x}(t) = \mathbf{x}_i\}$, the probability of being at state $\mathbf{x}_i$ at time t , for $i = 1, \dots, n$. It is customary to equivalently identify a state $\mathbf{x}_i$ just by its index i in the enumeration. We shall use both identifications interchangeably. The vector $\mathbf{p}_0 = \mathbf{p}(0)$ is an initial probability distribution and $\mathbf{A}$ is a sparse n -by- n matrix representing the infinitesimal generator of the Markov chain that underpins the CME. For $i \neq j$, the entry a_{ij} holds the propensity for the system to transition to state i from state j , or 0 if unreachable.

Thus the a_{ij} , for $i \neq j$, are nonnegative. And to be a valid infinitesimal generator, the diagonal terms are defined as $a_{jj} := -\sum_i a_{ij}$, which means that $\mathbf{A}$ has zero column sum and so probability is conserved. The solution of (1) at time t is

$$\mathbf{p}(t) = \exp(t\mathbf{A})\mathbf{p}_0, \quad (2)$$

where $\exp(t\mathbf{A}) = \sum_{n=0}^{\infty} \frac{(t\mathbf{A})^n}{n!}$ is the usual matrix exponential represented in Taylor series (among other equivalent definitions). Since there is no particular structure common to all the models, calculations cannot rely on generic simplifications.

4. Model Reduction

4.1. FSP method

With the CME prone to have a huge state space and in theory, even a potentially countably infinite set of states, a crucial goal is how to effectively reduce the state space in a meaningful way that still captures enough of the cell dynamics. The finite state projection (FSP) algorithm of Munsky and Khammash [9] is one such model reduction method that we summarize here since we build on it. The principle is to truncate the system to a smaller subsystem that captures enough information while remaining tractable. Let $J = \{1, \dots, k\}$, then the matrix in (2) is replaced by $\mathbf{A}_J$ where

$$\mathbf{A} = \left(\begin{array}{c|c} \mathbf{A}_J & * \\ \hline * & * \end{array} \right) \in \mathbb{R}^{n \times n},$$

i.e., with $k = |J|$ being the cardinality of J , $\mathbf{A}_J$ is a $k \times k$ submatrix of the true operator $\mathbf{A}$. The states indexed by J then form the *finite state projection*. The FSP method takes

$$\mathbf{p}(t_f) \approx \exp(t_f \mathbf{A}_J) \mathbf{p}_J(0). \quad (3)$$

Note the subscript J that characterizes the truncation just described and note that the initial distribution is truncated similarly. In fact, more generally, $\mathbf{A}_J$ need not simply be a *principal* submatrix. Rather, assume that J is an arbitrary subset of $\{1, \dots, n\}$ and that for consistency $\mathbf{A}_J$ is defined with the same size as the matrix $\mathbf{A} = [a_{ij}]$ using

$$(\mathbf{A}_J)_{ij} = \begin{cases} a_{ij} & \text{if } i, j \in J \\ 0 & \text{if } i \notin J \text{ or } j \notin J \end{cases}.$$

Similarly, $\mathbf{p}_J$ is defined from $\mathbf{p} = (p_1, \dots, p_n)^T$ using

$$(\mathbf{p}_J)_i = \begin{cases} p_i & \text{if } i \in J \\ 0 & \text{otherwise} \end{cases}.$$

The FSP approximation still takes the form (3). Obviously, all computations are done in practice on the effectively truncated matrix, justifying why the FSP is a reduction method. Conceptually, such a truncation can be understood as an implicit reordering $\mathbf{P}^T \mathbf{A} \mathbf{P}$ where $\mathbf{P}$ is an appropriate permutation matrix that makes the effectively

truncated matrix a principal submatrix. Munsy and Khammash [9] have proved that $\|\mathbf{p}_J(t)\|_1$ increases as J expands, and that if J expands enough so that

$$\Gamma_J = \|\mathbf{p}_J(t)\|_1 = \mathbf{1}^T \exp(t\mathbf{A}_J)\mathbf{p}_J(0) \geq 1 - \epsilon, \quad \mathbf{1} = (1, \dots, 1)^T, \quad (4)$$

then the computed $\mathbf{p}_J(t)$ approximates the exact $\mathbf{p}(t)$ within the tolerance ϵ , i.e.,

$$\|\mathbf{p}(t) - \mathbf{p}_J(t)\|_1 \leq \epsilon.$$

Thus (4) is used as a stopping condition for the basic FSP algorithm in practice: start from $J = J_0$ that captures at least all nonzero entries of $\mathbf{p}(0)$ and expand until (4) is met.

4.2. SSA-driven reduction method

The original FSP method expands the projection simply by increasing the size of the principal submatrix. A more general expansion is by reachability of the model so that, depending on the enumeration of states, this may mean that the submatrix used is not simply the intersection of the first k rows and columns. Achieving this efficiently is far from trivial because we are dealing with a huge state space, and the overhead of a complex search quickly becomes prohibitive when it is compounded over thousands of integration steps. Also remember that since the diagonal entry satisfies $a_{jj} := -\sum_{i \neq j} a_{ij}$, all the states have to be enumerated somehow — so that we can compute the associates propensities that define the j th column of the infinitesimal generator, even if only a principal submatrix is subsequently used, unless a_{jj} is approximated by other means. However, expanding the state via reachability is of great importance because the probability mass is kept by the most relevant states.

The novel way of monitoring the reduction that we present here uses the SSA. Indeed, we can use the SSA to quickly generate a sequence

$$\left\{ \mathbf{x}(t_0 + \sum s_i) \right\}_{t_0 + \sum s_i \leq t_1} = \left\{ \mathbf{x}(t_0), \mathbf{x}(t_0 + s_0), \mathbf{x}(t_0 + s_0 + s_1), \dots, \mathbf{x}(t_1) \right\},$$

where $\mathbf{x}(t_0 + \sum s_i)$ is a state reached by the SSA at time $t_0 + \sum s_i$, in which the inner $\{s_i\}$ are the times between events in the SSA. Taking these states as the most likely reachable states within the integration interval $[t_0, t_1]$, we build a more compact and adaptive infinitesimal generator that changes across integration steps to only capture the current states. Section 5.3 gives a formal description of the method.

5. Integration of the CME

The integration involved in the CME amounts to evaluating a matrix exponential, which is a treacherous task when dealing with the very large matrices that will typically arise in realistic biological models. Furthermore, one is often interested in the evolution of the probability distribution over time, and so requires many solutions at intermediate time points as well as at the final time (the original FSP implementation did not provide for this). Addressing these issues much more effectively is integral to our research.

5.1. Galerkin method and orthogonal polynomials

The Krylov approximation is an example of a Galerkin method that converts the original large problem to a much smaller problem in a Krylov subspace, as done in the Expokit package [11] that we are building upon. The approximation to the matrix exponential takes the form

$$\exp(\tau \mathbf{A})\mathbf{v} \approx \mathbf{V}_m \exp(\tau \mathbf{H}_m)(\beta \mathbf{e}_1), \quad (5)$$

$$m \ll n, \quad \beta = \|\mathbf{v}\|_2, \quad \mathbf{e}_1 = (1, 0, \dots, 0)^T,$$

where $\mathbf{V}_m \in \mathbb{R}^{n \times m}$ and $\mathbf{H}_m \in \mathbb{R}^{m \times m}$ are the orthonormal basis and upper Hessenberg matrix resulting from the well-known Arnoldi process. The smaller $\exp(\tau \mathbf{H}_m)$ is computed using the Padé technique (we will soon see the effectiveness of the Krylov method in the numerical section where $m = 30$ while n can be in the millions). A special feature of the Krylov method is that it is *matrix-free*, i.e., $\mathbf{A}$ (or its submatrices) need not be formed explicitly because it interacts only through matrix-vector products $\mathbf{A}\mathbf{v}$, making it possible to handle very large problems while keeping the operator in functional form. These aspects should be kept in mind when reading our description.

5.2. Embedded exponentials

Unlike the original FSP, the Krylov-FSP does not repeatedly compute $\exp(t_f \mathbf{A}_{J_k})\mathbf{p}_{J_k}(0)$ with the same t_f until $\mathbf{A}_{J_k}$ is sufficiently large. Rather, it uses the embedded scheme

$$\mathbf{p}(t_f) \approx \exp(\tau_K \mathbf{A}_{J_K}) \dots \exp(\tau_0 \mathbf{A}_{J_0})\mathbf{p}(0), \quad (6)$$

where the $\{\tau_k\}$ are stepsizes with $t_f = \sum_{k=0}^K \tau_k$ and K denoting the total number of steps eventually needed. The submatrices $\mathbf{A}_{J_k}$ correspond to the intermediate projections J_k . And for (6) to be mathematically consistent, the operands should be viewed as padded with zeros as necessary. If the submatrices are the same, the scheme formally simplifies to a classical step-by-step integration scheme as implemented in Expokit, which has a built-in strategy for selecting the stepsizes $\{\tau_k\}$ as done in variable stepsize ODE solvers. The evaluation of (6) inherits this built-in strategy, with the special feature that the matrices now change between the internal steps. This embedded scheme is a key aspect that yields a significant improvement in execution time.

As for the error control, we stress that the computed approximation is tentative at each step k , until it passes a post-check. To see how this is done, first note that if the projection fails to include highly likely states, there will be a loss in the probability mass. Thus the check for acceptance uses a FSP-like criteria. More precisely it is required that

$$\mathbb{1}^T \tilde{\mathbf{p}}(t_{k+1}) \geq 1 - \epsilon \frac{t_{k+1}}{t_f}, \quad t_{k+1} = t_k + \tau_k, \quad (7)$$

where $\tilde{\mathbf{p}}(t_{k+1}) \approx \exp(\tau_k \mathbf{A}_{J_k})\tilde{\mathbf{p}}_{J_k}(t_k)$ is effectively approximated by the Krylov scheme (5) seeded with $\mathbf{A} \equiv \mathbf{A}_{J_k}$ and $\mathbf{v} \equiv \tilde{\mathbf{p}}_{J_k}(t_k)$, the approximation at the current t_k . Expokit's built-in error control first suggests a τ_k so that the approximation (5) meets a given tolerance (regardless of the FSP criteria). The Krylov-FSP inherits this control and additionally checks (7). If this fails, the current stepsize τ_k and approximation to $\tilde{\mathbf{p}}(t_{k+1})$ are rejected. Then the approximation is recomputed with a reduced stepsize, until (7)

is satisfied. A smaller τ_k aims at making J_k more likely to account for states that the system can occupy within the interval $[t_k, t_k + \tau_k]$. Importantly, not all choices of τ_k require enlarging $\mathbf{A}_{J_k}$ and incurring the cost of a bigger subsystem – recall from (6) that each step brings us closer to t_f anyway. Further refinements are done in our new SSA-driven method as we explain next.

5.3. SSA projection

The integration strategies just described in the two sections above were initially given in [8] and pursued in subsequent studies. Our new contribution is the SSA-driven reduction that we now formally present. Let $J_0, \dots, J_k$ be a sequence of sets approximating the most likely reachable states at time $t_0, \dots, t_k$, respectively. Let $\text{SSA}(\mathbf{x}_{t_k}, t_k, \tau_k)$ be the (random) path of a SSA run over $[t_k, t_k + \tau_k]$ starting at a given state $\mathbf{x}_{t_k}$,

$$\begin{aligned} \text{SSA}(\mathbf{x}_{t_k}, t_k, \tau_k) &= \left\{ \mathbf{x}_{t_k + \sum s_i} \right\}_{t_k + \sum s_i \leq t_k + \tau_k} \\ &= \left\{ \mathbf{x}_{t_k}, \mathbf{x}_{t_k + s_0}, \mathbf{x}_{t_k + s_0 + s_1}, \dots, \mathbf{x}_{t_k + \tau_k} \right\}. \end{aligned}$$

Let $\mathbb{J}$ be the set of all possible states, and let $\mathbb{J}_k = J_0 \cup \dots \cup J_k$ be the states enumerated so far. Define $i_{\mathbf{x}}$ as the index function that returns the index of the state $\mathbf{x}$ from the enumeration in $\mathbb{J}_k$ and $\mathbf{x}_i$ its inverse (the correspondence between the two equivalent identifications of a state). Thus if $\tilde{\mathbf{p}}(t_k)$ approximates the probability vector at time t_k , its $i_{\mathbf{x}}$ -th component approximates the probability that the system is in state $\mathbf{x}_i$ at time t_k . Let ℓ be a chosen width for a window of previous steps $t_{k-\ell}, \dots, t_k$. Taking

$$\mu(\mathbf{x}_i) = \max_{t_j \in \{t_{k-\ell}, \dots, t_k\}} \tilde{p}_i(t_j),$$

we approximate the next set of most likely reachable states using the multi-step and multi-stage scheme:

$$J_{k+\frac{1}{3}} = \{i \in J_k, \mu(\mathbf{x}_i) \geq \varepsilon_{\text{tol}}\} \quad (8a)$$

$$J_{k+\frac{2}{3}} = \bigcup_{i \in J_{k+\frac{1}{3}}} \left\{ \text{SSA}(\mathbf{x}_i, t_k, \tau_k) \right\} \quad (8b)$$

$$J_{k+1} = \mathcal{R}^r \left(J_{k+\frac{2}{3}} \right), \quad (8c)$$

where $\mathcal{R}^r(J)$ is the r -step expansion of J , i.e., the set of states reachable from J in at most r reactions, which is formally defined by the recurrence:

$$\mathcal{R}^0(J) = J, \quad \mathcal{R}(J) = J \cup \bigcup_{k=1}^M \left\{ i \in \mathbb{J} : \mathbf{x}_i = \mathbf{x}_j + \boldsymbol{\nu}_k, j \in J \right\}, \quad \mathcal{R}^r(J) = \mathcal{R}(\mathcal{R}^{r-1}(J)).$$

The first stage (8a) is meant to drop states that have become improbable. The second stage (8b) performs predictive SSA runs, each of which scouts a relatively cheap single path confined to a small interval of length τ_k , the stepsize for which the Krylov approximation to the matrix exponential is expected to be numerically worthwhile. The third stage (8c) is meant to widen (or smooth) the path by r -step reachability.

Another alternative is to switch the last two stages, i.e., first expand by r -step reachability and then use the SSA to widen (or randomize) the expansion:

$$J_{k+\frac{1}{3}} = \{i \in J_k, \mu(\mathbf{x}_i) \geq \varepsilon_{\text{tol}}\} \quad (9a)$$

$$J_{k+\frac{2}{3}} = \mathcal{R}^r \left(J_{k+\frac{1}{3}} \right) \quad (9b)$$

$$J_{k+1} = \bigcup_{i \in J_{k+\frac{2}{3}}} \left\{ \text{SSA}(\mathbf{x}_i, t_k, \tau_k) \right\}. \quad (9c)$$

We found (8) more effective than (9), which is why our experiments will focus on the former for the sake of remaining concise. Either way, the idea of (8) or (9) is to track states that are not too far in the future as they fan out from states in J_k according to the SSA (or a variant), while at the same time dropping past states that have become unlikely (based on a measure μ that can be tuned). Hence the next projection can dynamically stretch or shrink, with the latter happening especially if several states are dropped at the same time that the SSA runs coalesce to fewer and/or common states.

Once states in a current projection J_k are known, the associated submatrix $\mathbf{A}_{J_k}$ is constructed to advance the integration. The smaller the projection the more compact the truncation will be, since its dimension is the cardinality of J_k . Note that in practice we perform the indexing of states using a hash table and the construction of $\mathbf{A}_{J_k}$ is intertwined with the discovery of J_k . Consider a projection and its corresponding submatrix initialized to account for the current probable states, then as more states become known through transitions found either by the SSA or r -step expansions, the propensities are concomitantly appended to the submatrix using sparse data structures. We mentioned before that having the states in any order conceptually amounts to a reordering that is not done explicitly, so the order of discovery is nonessential and the dimension of the reduced problem is always effectively $|J_k|$. More interesting is the indexing function, $i_{\mathbf{x}}$, to uniquely identify a state for mapping purposes, or consistent comparisons between different algorithmic variants. Intricate data structures and technical details that could otherwise obfuscate the main gist of this paper will be reported elsewhere.

We made an important remark earlier that it is not always necessary to enlarge $\mathbf{A}_{J_k}$ and incur the extra cost of doing so. In the implementation presented here, we initialize J_0 with all the states having nonzero probabilities in $\mathbf{p}(0)$, augmented with an r_0 -step expansion from these states using a user-specified value of r_0 . Then after each integration step we drop states that have become unlikely, i.e., we carry the first stage (8a). But we expand the projection to include more states only when the FSP criteria (7) fails, i.e., we carry the other two stages (8b) and (8c) conditionally (recall also that we reduce the stepsize as explained before in section 5.2). A subtle point worth highlighting here is that although we sample a single SSA path from each relevant state $\mathbf{x}_i$ per run, since re-runs are from states in the current J_k that contains $\mathbf{x}_i$, there could be more SSA paths originating from the same $\mathbf{x}_i$ (including the path that transited at $\mathbf{x}_i$ when it was not yet in the projection). It is also instructive to view these SSA runs as a delayed sprouting of branches on paths from an earlier state where $\mathbf{x}_i$ originated, which is reminiscent of a *lazy evaluation* strategy. Ultimately if coalescence does not occur, the J_k that is used may include multiple SSA paths that transit at $\mathbf{x}_i$, together with possible branches from these paths and their r -step expansions. This explains why, in the end, enough information is gathered to make J_k a good estimate of probable states in the small interval $[t_k, t_k + \tau_k]$.

The fact that this process is monitored by the error control (7) means that potential SSA re-runs kick in only as needed and adapt to different regimes of a problem in a seamless, elegant, and efficient manner, as we shall soon see in the numerical results.

While our discussion has outlined the core principles, implementation strategies can differ by tweaking the criteria in (7); choosing μ differently; reducing ε_{tol} to keep more past states; or even setting $\varepsilon_{\text{tol}} = 0$ on small to medium-sized problems so as not to drop any past state (as in the original FSP); or repeating the SSA based on user-defined settings. Our approach is somewhat reminiscent of how the Jacobian matrix is updated or frozen in classical implicit integrators for differential equations, but with the special feature that, in our context, we will also be changing the dimension of the matrix (not just its entries).

It is clear that the variety of parameters would necessitate extensive investigation that would significantly lengthen our presentation beyond the scope of this foundational description. We have already conducted such an investigation for the original Krylov-FSP that interested readers may find in [10]. Subsequent developments targeting the new approach presented here are in progress and will be communicated elsewhere, as well as other technical details as already mentioned. We note the recent work of Hjartarson et al [5] that combined the CME with the SSA, but there are key differences with our approach. In particular, they use the SSA not so much to tune the projection J_k as we do, but rather to estimate the probabilities of states outside of it, while they update their projection itself by the sliding window technique of Wolf et al [12].

Another early combination of the original FSP with the SSA is the multiple time interval finite state projection (MTI-FSP) of Munsky and Khammash [13]. But MTI-FSP uses the SSA only to select the initial projection, and not in subsequent steps as done here. Yet another marked difference is the control of the stepsize, which is constant in MTI-FSP and not variable like here. And when a projection fails it is repeatedly expanded to suit that constant stepsize by adding states in the directions where the probability mass leaks the most, using the last approximation to identify these directions.

6. Numerical Results

We implemented the suggested scheme (6), with each step computed using a Krylov-FSP approximation (via Expokit with dimension $m = 30$ and tolerance 10^{-8}) and a reduced matrix obtained from the novel predictive SSA approach (8), using $r = 1$, $\ell = 0$ and $\varepsilon_{\text{tol}} = 10^{-16}$. We compare with the original Krylov-FSP where the reduced matrix is obtained as a principal submatrix. Both use $\epsilon = 10^{-6}$ in the FSP criteria (7). These current numerical results already demonstrate the effectiveness of our new approach.

We noted earlier that a starting $\mathbf{A}_{J_0}$ is needed and that we construct it using an initial r_0 -expansion from states with nonzero probabilities in $\mathbf{p}(0)$. In the experiments, we took $r_0 = 10, 30, 5$ for the Michaelis-Menten, gene toggle, and Goutsias examples, respectively.

6.1. Michaelis-Menten enzyme kinetics

We begin with the well known Michaelis-Menten enzyme kinetics in Table 1. Reaction rates match those in [3], and we similarly use a maximum of 100 substrates and $t_f = 70$. The resulting matrix is of order 5151, with which we obtain the ‘exact’ solution by using the vanilla Expokit (with tolerance 10^{-14}) that does not have any reduction strategy.

Table 1: Michaelis-Menten reactions and propensities. ($[X]$ is the number of copies of the species X .)

	reaction	propensity	rate constant (s^{-1})
1.	$S + E \xrightarrow{\kappa_1} ES$	$\alpha_1 = \kappa_1 [S] [E]$	$\kappa_1 = 1.0$
2.	$ES \xrightarrow{\kappa_2} E + S$	$\alpha_2 = \kappa_2 [ES]$	$\kappa_2 = 1.0$
3.	$ES \xrightarrow{\kappa_3} P + E$	$\alpha_3 = \kappa_3 [ES]$	$\kappa_3 = 0.1$

Figure 1: Michaelis-Menten results (log scale). **(A) Left y-axis:** comparison of the size of the SSA-driven projection and the principal submatrix. **Right y-axis:** approximation error (between the ‘exact’ solution and our approximation), and gap (between the original Krylov-FSP and our approximation). **(B):** Comparison of CPU time.

From the numerical results in Fig. 1 (log scale), it is striking to observe the difference in projection sizes between the old and new methods. In effect, only a few components remain of significance over time and our novel SSA-driven reduction is capable of capturing them in a way that benefits the computations. With the previous approach based on using the principal submatrix, we see an increase of the size of the reduced matrix to the point of reaching the size of the full matrix. But with our new approach, we see that the size initially grows but as more states become improbable, they are dropped and the size decreases accordingly. The decrease is so startling that one may wonder about the correctness of the method. We plot the quantities, error and gap, that confirm the accuracy of our new approach. The error measures the difference between the ‘exact’ solution and our approximation, while the gap measures the difference between the original Krylov-FSP and our approximation, both are in L^∞ -norm. The decrease of size is explained by noting from reaction 3 in Table 1 that forming a product from the enzyme-substrate complex is an irreversible reaction, so the part of the state space that is reachable shrinks with time, and it contains an absorbing state corresponding to the situation where all substrates have been converted to products. Many states therefore become unlikely and need not be kept over time, as is otherwise done by the full blown CME.

6.2. A stochastic model of gene toggle

We consider the CME of a stochastic gene toggle model (Table 2) that was solved in [7] using the principal submatrix approach. We set the initial state as $\mathbf{x}_0 = ([U], [V]) = (85, 5)$ and the final time $t_f = 100$.

The results in Fig. 2 show again that our new approach does a good job at dropping states that become unlikely, ending with one-fifth of the size of the principal submatrix without being detrimental to the accuracy, as confirmed by checking the error and gap. As before, the ‘exact’ solution was obtained by using the vanilla Expokit that does not have any reduction strategy. The matrix used accounted for 390,046 states collected through 800-step reachability expansion from the initial state. During the time interval that we are interested in, the ‘exact’ solution always summed to 1 and is therefore close to the exact probability vector (modulo the tolerance set in Expokit, which was 10^{-14}).

The gene toggle model is reputed for its bistability (reached at $t \approx 29$ in this example). To further highlight the strength of our approach, we show in Fig. 3 the evolution of our SSA-driven projection, which is a set-based approach able to dynamically track probable states even if they are in disjoint parts of the state space. Contrast with a sliding window approach [5, 12] that would need a rectangle big enough to contain all the disjoint parts.

Table 2: Reaction channels of the gene toggle example. For the parameters, we set $d_1 = d_2 = 1.0$, $s = 0.1$, $K_1 = 1.0 \times \text{scale}$, $K_2 = 1.0 \times \text{scale}$, $\beta_1 = 4.0 \times \text{scale}$, $\beta_2 = 4.0 \times \text{scale}$, $\alpha_1 = \alpha_2 = 0.2 \times \text{scale}$, $\gamma = 1.0$, $\eta = 1.0$, $\text{scale} = 100$. ($[X]$ is the number of copies of the species X.)

	reaction	propensity
1.	$U \longrightarrow \emptyset$	$\left(d_1 + \frac{s\gamma}{1+s}\right) [U]$
2.	$\emptyset \longrightarrow U$	$\eta \left(\alpha_1 + \frac{\beta_1 K_1^3}{K_1^3 + [V]^3}\right)$
3.	$V \longrightarrow \emptyset$	$d_2 [V]$
4.	$\emptyset \longrightarrow V$	$\eta \left(\alpha_2 + \frac{\beta_2 K_2^3}{K_2^3 + [U]^3}\right)$

6.3. Goutsias model of regulated transcription

This test problem is adopted from a model of transcriptional regulation proposed by Goutsias [3] and is setup as in [7]. There are 6 chemical species M , D , RNA , DNA , $DNA.D$, $DNA.2D$, that interact through 10 reaction channels (Table 3). We integrate the CME for this model to $t_f = 300s$ starting with the initial state

$$\mathbf{x}_0 = ([M], [D], [RNA], [DNA], [DNA.D], [DNA.2D]) = (2, 6, 0, 2, 0, 0).$$

Contrary to the two previous examples, the support of the probability vector in the Goutsias example does not shift, but rather spreads across a growing number of states as the system evolves. This compounds the challenge of solving this example because many states remain probable. Nevertheless, we observed a dramatic reduction of the projection size when using our approach compared to the original Krylov-FSP (Fig. 4). Doing the computations using the original approach on a node of the Dense Memory Cluster of the Alabama Supercomputer Center took over 8 hours to complete. Whereas our novel predictive SSA scheme took only about 467 seconds on a personal laptop running Ubuntu

Figure 2: Gene toggle results (log scale). **(A) Left y-axis:** comparison of the size of the SSA-driven projection and the principal submatrix. **Right y-axis:** approximation error (between the ‘exact’ solution and our approximation), and gap (between the original Krylov-FSP and our approximation). **(B):** Comparison of CPU time.

Figure 3: Gene toggle evolution. Snapshots of our SSA-driven projection and the probability distribution. The axes in each subplot represent the numbers of copies of U (y -axis) and V (x -axis). All subplots share the same color map (in log scale), revealing the probability distribution at the indicated times. This is the output of the SSA-driven projection, which dynamically tracked probable states in the shaded regions and left out improbable states in the blank regions by either dropping or never retaining these states. The equilibrium distribution was reached at $t \approx 29$ with bistability implicating disjoint parts.

Linux with 8GB of memory. Despite the widening difference between the two projections, checking the gap shows that the two computed approximations remain close.

Due to the enormous state space, we do not have the ‘exact’ solution from where to compute the true error. We instead check against the classical SSA. We sampled 5 million trajectories using a direct implementation of the SSA in FORTRAN. From these we obtained the histogram in Fig. 5, approximating the final probability distribution for states reached at $t_f = 300$. The approximation using the SSA differs from ours by 2.91×10^{-5} in the L^∞ norm.

Table 3: Reaction channels in Goutsias model of regulated transcription. The parameter $A = 6.0221415 \times 10^{23}$ is Avogadro's number, and $V = 10^{-15}L$ is the system volume chosen for this experiment. ($[X]$ is the number of copies of the species X .)

reaction	propensity	rate constant (s^{-1})
1. $RNA \rightarrow RNA + M$	$c_1[RNA]$	$c_1 = 0.043$
2. $M \rightarrow \emptyset$	$c_2[M]$	$c_2 = 0.0007$
3. $DNA.D \rightarrow RNA + DNA.D$	$c_3[DNA.D]$	$c_3 = 0.0715$
4. $RNA \rightarrow \emptyset$	$c_4[RNA]$	$c_4 = 0.0039$
5. $DNA + D \rightarrow DNA.D$	$c_5[DNA][D]$	$c_5 = \frac{0.012 \times 10^9}{AV}$
6. $DNA.D \rightarrow DNA + D$	$c_6[DNA.D]$	$c_6 = 0.4791$
7. $DNA.D + D \rightarrow DNA.2D$	$c_7[DNA.D][D]$	$c_7 = \frac{0.00012 \times 10^9}{AV}$
8. $DNA.2D \rightarrow DNA.D + D$	$c_8[DNA.2D]$	$c_8 = 0.8765 \times 10^{-11}$
9. $M + M \rightarrow D$	$\frac{c_9}{2}[M]([M] - 1)$	$c_9 = \frac{0.05 \times 10^9}{AV}$
10. $D \rightarrow M + M$	$c_{10}[D]$	$c_{10} = 0.5$

Figure 4: Goutsias results (log scale). **(A) Left y -axis:** comparison of the size of the SSA-driven projection and the principal submatrix. Many states remain likely but our new approach tracks the most relevant ones more efficiently. **Right y -axis:** gap between the original Krylov-FSP and our approximation. **(B):** Comparison of CPU time.

Figure 5: Goutsias example. Marginal probability distribution for the number of RNA molecules at $t_f = 300$, as approximated by our method (solid line) and 5 million SSA trajectories (bars).

7. Conclusion

Our research aims at solving the CME using a combination of techniques: model reduction techniques; use of a predictor SSA to quickly scout the most reachable states at any given time; efficient embedded techniques for the reduced problem; inexact/relaxed function evaluations in the integration schemes. The SSA-driven projection presented here is a central element with quite promising results.

Acknowledgement

This work was supported by NSF grant DMS-1320849. Resources and technical support from the Alabama Supercomputer Authority are also acknowledged. We thank the anonymous referees for their comments that improved the paper.

References

- [1] Y. Cao, H. Li, and L.R. Petzold. Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *J. Chem. Phys.* 121(9):4059–4067, 2004.
- [2] D.T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25):2340–2361, 1977.
- [3] J. Goutsias. Quasiequilibrium approximation of fast reaction kinetics in stochastic biochemical systems. *J. Chem. Phys.*, 122(18):184102, 2005.
- [4] D.J. Higham. Modeling and simulating chemical reactions. *SIAM Rev.*, 50(2):347–368, 2008.
- [5] A. Hjartarson, J. Ruess and J. Lygeros. Approximating the solution of the chemical master equation by combining finite state projection and stochastic simulation. In 52nd IEEE Conference on Decision and Control, Florence, Italy, pp. 751–756, 2013.
- [6] S. MacNamara, A.M. Bersani, K. Burrage, and R.B. Sidje. Stochastic chemical kinetics and the total quasi-steady-state assumption: application to the stochastic simulation algorithm and chemical master equation. *J. Chem. Phys.*, 129(9):095105, 2008.
- [7] S. MacNamara, K. Burrage, and R. B. Sidje. Multiscale modeling of chemical kinetics via the master equation. *Multiscale Model. Simul. (SIAM)*, 6(4):1146–1168, 2008.
- [8] K. Burrage, M. Hegland, S. MacNamara, and R.B. Sidje. A Krylov-based finite state projection algorithm for solving the chemical master equation arising in the discrete modelling of biological systems. In Markov 150th Anniversary Meeting, A. N. Langville and W. J. Stewart (eds), pp. 21–38, 2006. Bosc Books. ISBN 1932482342.
- [9] B. Munsky, and M. Khammash. The finite state projection algorithm for the solution of the chemical master equation. *J. Chem. Phys.*, 124:044104, 2006.
- [10] H.D. Vo, and R.B. Sidje. Implementation of variable parameters in the Krylov-based finite state projection for solving the chemical master equation. (submitted), 2015.
- [11] R.B. Sidje. Expokit: A software package for computing matrix exponentials. *ACM Transactions on Math. Softw.*, 24(1):130–156, 1998.
- [12] V. Wolf, R. Goel, M. Mateescu and T.A. Henzinger. Solving the chemical master equation using sliding windows. *BMC Systems Biology*, 4:42, 2010.
- [13] B. Munsky, and M. Khammash. A multiple time interval finite state projection algorithm for the solution to the chemical master equation. *J. Comp. Phys.*, 226(1):818–835, 2007.
- [14] S. Peles, B. Munsky, and M. Khammash. Reduction and solution of the chemical master equation using time scale separation and finite state projection. *J. Chem. Phys.*, 125(20), 204104, 2006.
- [15] S. Smith, V. Shahrezaei. General transient solution of the one-step master equation in one dimension. *Phys. Rev. E*, 91(6):062119, 2015.
- [16] J. K. Kim, K. Josić and M. R. Bennett. The validity of quasi-steady-state approximations in discrete stochastic simulations. *Biophys. J.*, 107(3):783–793, 2014.