Applied Mathematics and Computation 293 (2017) 334–344

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Implementation of variable parameters in the Krylov-based finite state projection for solving the chemical master equation

H.D. Vo, R.B. Sidje*

Department of Mathematics, The University of Alabama, Tuscaloosa, AL 35487, USA

ARTICLE INFO

Keywords:
Matrix exponential
Adaptive Krylov method
Chemical master equation
Finite state projection

ABSTRACT

The finite state projection (FSP) algorithm is a reduction method for solving the chemical master equation (CME). The Krylov-FSP improved on the original FSP by using an embedded scheme where the action of the matrix exponential is evaluated by the Krylov subspace method of Expokit for greater efficiency. There are parameters that impact the method, such as the stepsize that must be controlled to ensure the accuracy of the computed matrix exponentials, or to ensure the accuracy of the FSP. Other parameters include the dimension of the Krylov basis, or even the extent of reachability when expanding the FSP. In this work, we incorporate adaptive strategies to automatically vary these parameters. Numerical experiments comparing the resulting variants are reported, showing how certain choices perform better than others.

1. Introduction

The ability to model the composition of molecules that regulate the signaling and metabolism within biological cells has significant ramifications in cell biology. When molecules exist in large numbers, as in the classical mixing tank problem for example, it is customary to quantify substances using deterministic, continuous concentrations. Within a cell, however, certain key regulatory molecules only exist in small numbers, and so it becomes more appropriate to use stochastic approaches that treat these molecules as discrete quantities.

Assume the cell has N molecular species $S_1, \ldots, S_N$, that interact through M biochemical reactions $R_1 \ldots, R_M$, and assume for modeling purposes that it has a fixed volume and constant temperature. The dynamical state of the system is a time-dependent vector $\boldsymbol{x} = (x_1, \ldots, x_N)^T$ where the components are nonnegative integers counting the different species of molecules in the cell. Thus, if the system is in state $\boldsymbol{x}$ and reaction k occurs, the system transitions to state $\boldsymbol{x} + \boldsymbol{\nu}_k$, where the state change vector $\boldsymbol{\nu}_k$ is called the *stoichiometric* vector corresponding to the kth reaction. In other words, the counter of a species increases or decreases when a chemical reaction occurs (depending on whether the reaction produces or consumes that species). Not any $\boldsymbol{x} + \boldsymbol{\nu}_k$ is a possible state since the resulting integer vector must remain nonnegative. Each state induces a set of M *propensities* $\alpha_1(\boldsymbol{x}), \ldots, \alpha_M(\boldsymbol{x})$ that determine the relative chance of each reaction occurring if the system is in state $\boldsymbol{x}$. Starting from an initial state, the cell will incrementally transition to other states based on the chain of reactions that happen over time. There are trajectorial approaches such as the stochastic simulation algorithm (SSA) and variants [2–4,14] that repeatedly simulate trajectories of the system in a Monte Carlo manner, but these are not considered here. Rather, we consider the problem of solving the chemical master equation (CME), which allows us to calculate the probability of finding the cell in a particular state at a given time.

* Corresponding author.
E-mail addresses: hvo@crimson.ua.edu (H.D. Vo), roger.b.sidje@ua.edu (R.B. Sidje).

http://dx.doi.org/10.1016/j.amc.2016.08.013
0096-3003/

Since the cell cannot contain an infinite number of molecules, the state space of interest is confined to a finite set. If we enumerate all these possible states as $\{\boldsymbol{x}_i\}_{i=1}^{n}$ with each $\boldsymbol{x}_i = (x_{1i}, \ldots, x_{Ni})^T$, the CME takes the form of a system of linear ordinary differential equations (ODEs)

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

where the probability vector $\boldsymbol{p} = (p_1, \ldots, p_n)^T$ is such that each component $p_i = \text{Prob}\{\boldsymbol{x}(t) = \boldsymbol{x}_i\}$, i.e., the probability of being at state $\boldsymbol{x}_i$ at time t, for $i = 1, \ldots, n$. It is common to equivalently identify a state $\boldsymbol{x}_i$ just by its index i in the enumeration. The vector $\boldsymbol{p}_0 = \boldsymbol{p}(0)$ is an initial probability distribution and $\boldsymbol{A}$ is a sparse n-by-n matrix representing the transition rate matrix of the Markov chain that underpins the CME.

Solving (1) is a computationally intensive task especially because n quickly grows large, a situation referred to as the curse of dimensionality. Addressing this issue was the catalyst of the finite state projection (FSP) algorithm of Munsky and Khammash [9] that truncated the state space to a more tractable size. Then, the subsequent Krylov-based FSP [1] provided an early improvement over the basic FSP by using the Krylov subspace method in Sidje's Expokit [12] to compute the matrix exponential, as well as its built-in step-by-step integration scheme.

Our present study looks back at that earlier work and explores the tuning of some of the parameters that are involved in the Krylov-FSP. Consider for instance the stepsize, which the algorithm controls in different ways to achieve two distinct goals: to get an accurate exponential from Expokit and to satisfy the FSP error bound. The first control is done by ODE techniques while the second control is done by halving. Whether this halving strategy can be improved has not been thoroughly investigated so far. Another parameter is the dimension of the Krylov basis, which is kept fixed but can be made adaptive as done by Niesen and Wright [11]. Yet another parameter is the extent of the reachability when the FSP projection is expanded. It is the variability and tuning of these parameters that we explore here.

The organization of the paper is as follows. Section 2 is a background on the FSP algorithm. Section 3 reviews the original Krylov-based FSP. Section 4 summarizes how to achieve a variable stepsize and variable dimension in the Krylov subspace method for the exponential. Section 5 discusses how similar techniques can be used to vary the stepsize and the reachability in the FSP, as an alternative to the simple stepsize halving technique of the original Krylov-FSP. Section 6 reports numerical comparisons between the variants on popular test problems from the CME literature.

2. The finite state projection method

2.1. Background

The principle of the FSP algorithm of Munsky and Khammash [9] is to truncate the system to a smaller subsystem that captures enough information while remaining tractable. Let $J = \{1, \ldots, k\}$, then the matrix in (1) is replaced by $\boldsymbol{A}_J$ where

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

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

$$\boldsymbol{p}(t_f) \approx \exp(t_f \boldsymbol{A}_J)\, \boldsymbol{p}_J(0). \tag{2}$$

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

$$(\boldsymbol{A}_J)_{ij} = \begin{cases} a_{ij} & \text{if } i, j \in J \\ 0 & \text{if } i \notin J \text{ or } j \notin J \end{cases}. \tag{3}$$

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

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

The FSP approximation still takes the form (2). Obviously, all computations are done in practice on the effectively truncated matrix, justifying why the FSP is a reduction method. Munsky and Khammash [9] proved that $\|\boldsymbol{p}_J(t)\|_1$ increases as J expands, and that if J expands enough so that

$$\|\boldsymbol{p}_J(t)\|_1 = \|\exp(t\boldsymbol{A}_J)\, \boldsymbol{p}_J(0)\|_1 \geq 1 - \varepsilon_{FSP}, \tag{4}$$

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

$$\|\boldsymbol{p}(t) - \boldsymbol{p}_J(t)\|_1 \leq \varepsilon_{FSP}.$$