Journal of Computational Physics 258 (2014) 585–600

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Array-representation integration factor method for high-dimensional systems ☆

Dongyong Wang [a], Lei Zhang [b], Qing Nie [a,*]

[a] Department of Mathematics, University of California, Irvine, CA 92697, USA
[b] Beijing International Center for Mathematical Research, Peking University, Beijing 100871, China

ARTICLE INFO

Article history:
Received 15 April 2013
Received in revised form 30 October 2013
Accepted 4 November 2013
Available online 11 November 2013

Keywords:
Reaction–diffusion equations
Implicit method
Splitting method
Fokker–Planck equations
Chemical master equation

ABSTRACT

High order spatial derivatives and stiff reactions often introduce severe temporal stability constraints on the time step in numerical methods. Implicit integration method (IIF) method, which treats diffusion exactly and reaction implicitly, provides excellent stability properties with good efficiency by decoupling the treatment of reactions and diffusions. One major challenge for IIF is storage and calculation of the potential dense exponential matrices of the sparse discretization matrices resulted from the linear differential operators. Motivated by a compact representation for IIF (cIIF) for Laplacian operators in two and three dimensions, we introduce an array-representation technique for efficient handling of exponential matrices from a general linear differential operator that may include cross-derivatives and non-constant diffusion coefficients. In this approach, exponentials are only needed for matrices of small size that depend only on the order of derivatives and number of discretization points, independent of the size of spatial dimensions. This method is particularly advantageous for high-dimensional systems, and it can be easily incorporated with IIF to preserve the excellent stability of IIF. Implementation and direct simulations of the array-representation compact IIF (AcIIF) on systems, such as Fokker–Planck equations in three and four dimensions and chemical master equations, in addition to reaction–diffusion equations, show efficiency, accuracy, and robustness of the new method. Such array-presentation based on methods may have broad applications for simulating other complex systems involving high-dimensional data.

1. Introduction

Consider a general reaction–diffusion equation of the form

$$\partial_t u(\mathbf{x},t) = \sum_{i,j=1}^{n} \partial_{x_i}\left(D_{ij}(u,\mathbf{x})\partial_{x_j} u(\mathbf{x},t)\right) + F(u,\mathbf{x}), \tag{1}$$

where n is the spatial dimension and $\mathbf{x} = \{x_1,\ldots,x_n\}$. A non-linear term $F(u,\mathbf{x})$ is often interpreted as a reaction term. Coefficients D_{ij} can be either constants or functions of u and $\mathbf{x}$. The function $u(\mathbf{x},t)$ usually represents concentrations of physical or biological species with reactions among them, for which one usually has $n=2$ or $n=3$. However, $u(\mathbf{x},t)$ can

* Corresponding author.
E-mail address: qnie@uci.edu (Q. Nie).

0021-9991/$ – see front matter
http://dx.doi.org/10.1016/j.jcp.2013.11.002

also be considered as a probability density functions for stochastic dynamics described by Fokker–Planck equations [1], for which the dimension n represents the number of biochemical species.

Spatial discretization for differential equations of higher spatial dimensions (even for $n = 3$) often requires large, sometimes prohibitive, data storage and management as well as expensive CPU time at a fixed time point. In addition, temporal discretization, which strongly depends on the stiffness of reactions and treatment of the high order derivatives (e.g. the diffusion term), may lead to severe stability conditions that require very small time steps, resulting in excessive computational cost.

Integration factor (IF) or exponential time differencing (ETD) methods are effective approaches to deal with temporal stability constraints associated with high order derivatives [2–4]. By treating linear operators of the highest order derivative exactly, IF or ETD methods are able to achieve excellent temporal stability [2,5,6]. To deal with additional stability constraints from stiff reactions, a class of semi-implicit integration factor (IIF) methods [7] were developed for implicit treatment of the stiff reactions. In the IIF approach, the diffusion term is solved exactly like the IF method while the nonlinear equations resulted from the implicit treatment of reactions is decoupled from the diffusion term to avoid solving large nonlinear systems involving both diffusions and reactions, such as in a standard implicit method for reaction–diffusion equations. IIF methods have a great stability property with its second order scheme being linearly unconditionally stable.

In IF or ETD type of methods, the dominant computational cost arises from the storage and calculation of exponentials of matrices resulting from discretization of the linear differential operators in the PDEs. To deal with this difficulty, compact representation of the discretization matrices was introduced in the context of IIF method [8]. In compact implicit integration factor method (cIIF), the discretized solutions are represented in a matrix form rather than a vector while the discretized diffusion operator are represented in matrices of much smaller size than the standard matrices for IIF while preserving the stability property of the IIF. For two or three dimensions, cIIF is significantly more efficient in both storage and CPU cost. In addition, cIIF method is robust in its implementation and integration with other spatial and temporal algorithms. It can handle general curvilinear coordinates as well as combine with adaptive mesh refinements in a straightforward fashion [9]. One can also apply cIIF to stiff reactions and diffusions while using other specialized hyperbolic solvers (e.g. WENO methods [10,11]) for convection terms to solve reaction–diffusion–convection equations efficiently [12]).

One alternative approach for IF (or ETD) methods to avoid storage of the exponentials of large matrices is to use Krylov subspace method to compute the multiplication between the vector and the exponentials of matrices without explicitly forming the matrices [13,14]. The advantage of applying Krylov subspace method is that it can handle complicated diffusion operators, e.g. diffusion coefficients are spatial functions or elliptic operators contains cross-derivatives, while cIIF in previous studies [8] can only handle systems of constant diffusion coefficients and Laplacian operators restricted to two and three dimensions. In contrast to cIIF, in which exponentials of matrices are pre-calculated only once and stored for repeated usages at each time step of the temporal updating, the Krylov subspace method needs to be carried out at each time step, leading to a significant increase in CPU time.

In this paper, we introduce an array representation for the linear differential operators that may contain non-constant diffusion coefficients as well as cross-derivatives in two, three or higher dimensions. This array-representation approach is based on the idea of compact Implicit Integration Factor (cIIF), that is, when discretizing the terms with partial derivatives, regard the unknown solution as a vector with index connected to corresponding variables, while keeping other indexes fixed with unrelated variables. This new approach yields several discretization matrices of a small size that depend only on the number of derivatives in the continuous operators and the number of spatial discretization points in the direction of each derivative, in contrast to IF (or ETD) that requires exponentials of matrices whose size depend on the number of dimensions. In particular, the array representation can be incorporated into IIF to maintain the nice stability property of IIF as well as the implicit local treatment of the reactions decoupled from the diffusions. Like IIF, the second order array-representation (compact) implicit integration method (AcIIF) is A-stable. An operator splitting technique is incorporated into AcIIF for certain differential operators, resulting in non-commutable operations between discretization matrices. The AcIIF method is an extension of cIIF method that is able to deal with cross-derivatives and non-constant diffusion coefficients in addition to other applications.

To study the accuracy and efficiency of AcIIF, we implement AcIIF methods and compare it with several other existing methods for both two and three-dimensional reaction–diffusion equations. In addition, we apply AcIIF to solve Fokker–Planck equations in three and four dimensions. To demonstrate other applications of the array representation, we also use this approach to directly solve chemical master equations. In CMEs, the structure of the rate matrix for a reaction containing k species of molecules is very similar to the discretization matrix for a k-th order partial differential equation with cross-derivatives. The overall direct simulations show the excellent properties of AcIIF and its distinct advantage in high spatial dimensions.

2. Array-representation (compact) Implicit Integration Factor Method (AcIIF)

2.1. Array representation for reaction–diffusion systems in three dimensions without cross-derivatives

To illustrate the array-representation approach, we first consider three-dimensional reaction–diffusion equations without cross-derivatives and with constant diffusion coefficients and periodic boundary conditions:

$$u_t(x, y, z, t) = D\Delta u(x, y, z, t) + f\big(u(x, y, z, t)\big), \tag{2}$$

where $(x, y, z) \in \Omega = \{0 < x, y, z < l\}$. Let N_x, N_y, N_z be the number of spatial grid points in each spatial direction and h_x, h_y, h_z be the grid size, respectively. Denote U_{k_1,k_2,k_3} as the approximated solution of u at the grid point $(k_1 h_x, k_2 h_y, k_3 h_z)$. The approximation of $D\partial^2/\partial x^2$ using the second order central difference discretization can be written in terms of multi-dimensional arrays, $U = (U_{k_1,k_2,k_3})$, through a linear mapping $\mathcal{L}_x$,

$$(\mathcal{L}_x U)_{k_1,k_2,k_3} := \frac{D}{h_x^2}(U_{k_1+1,k_2,k_3} - 2U_{k_1,k_2,k_3} + U_{k_1-1,k_2,k_3}) \tag{3}$$

where $1 \leqslant k_1 \leqslant N_x$, $1 \leqslant k_2 \leqslant N_y$, and $1 \leqslant k_3 \leqslant N_z$. Similarly, using $\mathcal{L}_y$ and $\mathcal{L}_z$ to represent the approximations $D\partial^2/\partial y^2$ and $D\partial^2/\partial z^2$, respectively, Eq. (2) is approximated by

$$\frac{dU}{dt} = \mathcal{L}_x U + \mathcal{L}_y U + \mathcal{L}_z U + f(U). \tag{4}$$

Multiplying the integration factor, $e^{(\mathcal{L}_x+\mathcal{L}_y+\mathcal{L}_z)t}$, to both sides and integrating from t_n to t_{n+1}, two adjacent discretized temporal points, we derive a class of semi-implicit integration factor methods (IIF) after approximating the integral [7]. For example, the second order IIF takes the form

$$U^{n+1} - \frac{\Delta t}{2} f\left(U^{n+1}\right) = e^{(\mathcal{L}_x+\mathcal{L}_y+\mathcal{L}_z)\Delta t}\left(U^n + \frac{\Delta t}{2} f\left(U^n\right)\right) \tag{5}$$

where $U^n \approx U$ at time point t_n.

In a typical representation of the linear differential operator, the matrix $(\mathcal{L}_x + \mathcal{L}_y + \mathcal{L}_z)\Delta t$ has a size of $N_x N_y N_z \times N_x N_y N_z$. Although the matrix itself is sparse, its exponential is usually not, leading to prohibitive storage and computing cost for any fine spatial meshes. Next, we decompose this matrix into small matrices based on an array representation.

If one defines a vector by fixing the last two indices, k_2, k_3, of the three-dimensional array U,

$$U(:, k_2, k_3) = (U_{1,k_2,k_3}, U_{2,k_2,k_3}, \ldots, U_{N_x,k_2,k_3})^T. \tag{6}$$

Then the three-dimensional array U, can be treated as the collection of all such one-dimensional vector on a two-dimensional array, with all k_2, k_3 going through from 1 to N_y and from 1 to N_z, respectively. We present this collection using symbol $\bigotimes$, with the super index indicates that this collection is along x_i axis, then we have:

$$U = \bigotimes_{\substack{1\leqslant k_2\leqslant N_y \\ 1\leqslant k_3\leqslant N_z}} U(:, k_2, k_3). \tag{7}$$

Next, we define a $N_x \times N_x$ matrix $M_x = D/h_x^2 W_{N_x\times N_x}$, where

$$W_{N\times N} = \begin{pmatrix} -2 & 1 & 0 & \cdots & 1 \\ 1 & -2 & 1 & \cdots & 0 \\ 0 & 1 & -2 & 1 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 0 & \cdots & 1 & -2 \end{pmatrix}_{N\times N}. \tag{8}$$

Then, $M_x U(:, k_2, k_3)$ represents the vector and matrix multiplication for any fixed pair of k_2, k_3. Using this approach, the linear mapping $\mathcal{L}_x$ in the array representation becomes,

$$\mathcal{L}_x U = \bigotimes_{\substack{1\leqslant k_2\leqslant N_y \\ 1\leqslant k_3\leqslant N_z}} M_x U(:, k_2, k_3). \tag{9}$$

Consequently, the exponential of $\mathcal{L}_x$ in the array representation takes the following form:

$$e^{\mathcal{L}_x} U = \bigotimes_{\substack{1\leqslant k_2\leqslant N_y \\ 1\leqslant k_3\leqslant N_z}} e^{M_x} U(:, k_2, k_3), \tag{10}$$

as induced from the relation,

$$(\mathcal{L}_x)^m U = \bigotimes_{\substack{1\leqslant k_2\leqslant N_y \\ 1\leqslant k_3\leqslant N_z}} (M_x)^m U(:, k_2, k_3), \quad \forall m \in \mathbb{N}^+. \tag{11}$$

Applying the definition of linear mapping exponential yields Eq. (10).

Clearly, $\mathcal{L}_y$ and $\mathcal{L}_z$ have similar array representations,

$$\mathcal{L}_y U = \bigotimes_{\substack{1\leqslant k_1\leqslant N_x\\ 1\leqslant k_3\leqslant N_z}} M_y U(k_1,:,k_3), \qquad \mathcal{L}_z U = \bigotimes_{\substack{1\leqslant k_1\leqslant N_x\\ 1\leqslant k_2\leqslant N_y}} M_z U(k_1,k_2,:), \tag{12}$$

where $M_y = D/h_y^2 W_{N_y\times N_y}$ and $M_z = D/h_z^2 W_{N_z\times N_z}$.

Using the array representations, one can easily show that the three linear mappings $\mathcal{L}_x, \mathcal{L}_y$ and $\mathcal{L}_z$ commute with each other, i.e.,

$$\mathcal{L}_a\mathcal{L}_b U = \mathcal{L}_b\mathcal{L}_a U, \quad \text{for } a,b\in\{x,y,z\}. \tag{13}$$

This commuting property results in

$$e^{\mathcal{L}_x+\mathcal{L}_y+\mathcal{L}_z}U = e^{\mathcal{L}_x}e^{\mathcal{L}_y}e^{\mathcal{L}_z}U. \tag{14}$$

Direct application of Eq. (14) to Eq. (5) results in the following second order array-representation Implicit Integration Factor (AcIIF) method:

Algorithm 1 *(Second order AcIIF (AcIIF2)).*

$$U^{n+1} - \frac{\Delta t}{2} f\big(U^{n+1}\big) = \bigotimes_{\substack{1\leqslant k_2\leqslant N_y\\ 1\leqslant k_3\leqslant N_z}} e^{M_x\Delta t}\Bigg(\bigotimes_{\substack{1\leqslant k_1\leqslant N_x\\ 1\leqslant k_3\leqslant N_z}} e^{M_y\Delta t}\Bigg(\bigotimes_{\substack{1\leqslant k_1\leqslant N_x\\ 1\leqslant k_2\leqslant N_y}} e^{M_z\Delta t} V(k_1,k_2,:)\Bigg)(k_1,:,k_3)\Bigg)(:,k_2,k_3), \tag{15}$$

where $V = U^n + \Delta t/2 f(U^n)$.

Previously, a compact IIF (cIIF) was derived in a different fashion in two spatial dimensional systems by treating unknowns as a matrix, then the action of $\mathcal{L}_x$ is like a left product to the matrix and $\mathcal{L}_y$ is as its right product. And in three spatial dimensional cases, in addition to the left and right multiplications, a middle multiplication represents $\mathcal{L}_z$ [8]. One major advantage of both cIIF and AcIIF methods that one only needs to compute the exponentials of M_x, M_y and M_z, which are much smaller matrices (only about $N\times N$), in comparison to standard IF or ETD methods [15,16], for which the exponential $e^{(\mathcal{L}_x+\mathcal{L}_y+\mathcal{L}_z)\Delta t}$ of dimension of $N_xN_yN_z\times N_xN_yN_z$ are needed. Clearly, cIIF and AcIIF have significant savings in both CPU cost and storage, in particular, for equations in three or higher dimensions.

For the systems without cross-derivatives (e.g. Eq. (15)), the second order AcIIF (2) is equivalent to the second order cIIF method [8]. As it will be shown next, the advantage of AcIIF lies in its potential applications to reaction–diffusion systems with cross-derivatives and non-constant diffusion coefficients for which cIIF is unable to achieve.

2.2. AcIIF method for three-dimensional reaction–diffusion systems with cross-derivatives

Consider the reaction–diffusion equations with second order cross-derivatives:

$$\begin{aligned} u_t = &\left(a_1\frac{\partial^2}{\partial x^2} + 2b_1\frac{\partial^2}{\partial x\partial y} + c_1\frac{\partial^2}{\partial y^2}\right)u + \left(a_2\frac{\partial^2}{\partial x^2} + 2b_2\frac{\partial^2}{\partial x\partial z} + c_2\frac{\partial^2}{\partial z^2}\right)u\\ &+ \left(a_3\frac{\partial^2}{\partial y^2} + 2b_3\frac{\partial^2}{\partial y\partial z} + c_3\frac{\partial^2}{\partial z^2}\right)u + f(u), \end{aligned} \tag{16}$$

in a cube, $\{(x,y,z)\colon 0<x,y,z<l\}$, with periodic boundary conditions, satisfying the conditions $a_ic_i > b_i^2$, for $i=1,2,3$. Applying a standard second order central difference approximation to $a_1\frac{\partial^2}{\partial x^2} + 2b_1\frac{\partial^2}{\partial x\partial y} + c_1\frac{\partial^2}{\partial y^2}$, one obtains its approximation, denoted by $\mathcal{L}_{xy}$, as the following

$$\begin{aligned} (\mathcal{L}_{xy}U)_{k_1,k_2,k_3} = &\frac{a_1}{h_x^2}(U_{k_1+1,k_2,k_3} - 2U_{k_1,k_2,k_3} + U_{k_1-1,k_2,k_3})\\ &\times \frac{2b_1}{4h_xh_y}(U_{k_1+1,k_2+1,k_3} + U_{k_1-1,k_2-1,k_3} - U_{k_1+1,k_2-1,k_3} - U_{k_1-1,k_2+1,k_3})\\ &+ \frac{c_1}{h_y^2}(U_{k_1,k_2+1,k_3} - 2U_{k_1,k_2,k_3} + U_{k_1,k_2-1,k_3}). \end{aligned} \tag{17}$$

Using similar definitions for $\mathcal{L}_{yz}$ and $\mathcal{L}_{xz}$, the spatial approximation of Eq. (16) becomes

$$\frac{dU}{dt} = (\mathcal{L}_{xz} + \mathcal{L}_{xy} + \mathcal{L}_{yz})U + f(U). \tag{18}$$

To derive the array representation of the operator $\mathcal{L}_{xy}$, we first fix k_3 in $U(:,:,k_3)$ that represents a $N_xN_y\times N_xN_y$ matrix. Collect all these two-dimensional matrices along a vector leads to:

$$U = \bigotimes_{1\leqslant k_3\leqslant N_z} U(:,:,k_3). \tag{19}$$

Define a linear mapping, $\mathcal{A}_{xy}$, from a matrix space consisting of all $N_x \times N_y$ matrices to itself as follows:

$$\begin{aligned}(\mathcal{A}_{xy}M)_{i,j} = {} & \frac{2b_1}{4h_xh_y}(M_{i+1,j+1} + M_{i-1,j-1} - M_{i-1,j+1} - M_{i+1,j-1}) \\ & + \frac{a_1}{h_x^2}(M_{i+1,j} - 2M_{i,j} + M_{i-1,j}) + \frac{c_1}{h_y^2}(M_{i,j+1} - 2M_{i,j} + M_{i,j-1}).\end{aligned} \tag{20}$$

Then, the array representation of $\mathcal{L}_{xy}$, in terms of $\mathcal{A}_{xy}$, and its exponential become

$$\mathcal{L}_{xy}U = \bigotimes_{1\leqslant k_3\leqslant N_z} \mathcal{A}_{xy}U(:,:,k_3), \qquad e^{\mathcal{L}_{xy}}U = \bigotimes_{1\leqslant k_3\leqslant N_z} e^{\mathcal{A}_{xy}}U(:,:,k_3). \tag{21}$$

Similarly, the array representation for $\mathcal{L}_{yz}$ and $\mathcal{L}_{xz}$ may be written in terms of $\mathcal{A}_{yz}$ and $\mathcal{A}_{xz}$, respectively.

As long as $\mathcal{L}_{xy}$, $\mathcal{L}_{yz}$ and $\mathcal{L}_{xz}$ commute with each other, applying Eq. (5) using the array representation to Eq. (18) leads to the following algorithm:

Algorithm 2 *(AcIIF2 for reaction–diffusion systems with cross-derivatives).*

$$U^{n+1} - \frac{\Delta t}{2}f\left(U^{n+1}\right) = \bigotimes_{1\leqslant k_1\leqslant N_x} e^{\mathcal{A}_{yz}\Delta t}\left(\bigotimes_{1\leqslant k_2\leqslant N_y} e^{\mathcal{A}_{xz}\Delta t}\left(\bigotimes_{1\leqslant k_3\leqslant N_z} e^{\mathcal{A}_{xy}\Delta t}V(:,:,k_3)\right)(:,k_2,:)\right)(k_1,:,:), \tag{22}$$

where $V = U^n + \Delta t/2 f(U^n)$.

In this algorithm, the exponential of $\mathcal{A}_{xy}$ is a $N_xN_y \times N_xN_y$ matrix, in comparison to $\mathcal{L}_{xy}$ that is a $N_xN_yN_z \times N_xN_yN_z$ matrix. Thus applying array representation leads to significant saving.

Cross-derivatives may affect the commutable property of the discretized operators, resulting in the questions: under what conditions, $\mathcal{L}_{xy}$, $\mathcal{L}_{yz}$ and $\mathcal{L}_{xz}$ can commute with each other and what to do with the algorithms if the commuting property doesn't hold. In Section 3, we will give a sufficient condition for the commuting property. Alternatively, we next introduce a splitting technique to deal with the cases without such commuting property.

2.3. AcIIF method for reaction–diffusion systems with non-constant diffusion coefficients

When the diffusion coefficients in Eq. (16) are functions in space, minor modification is needed for the three discretization operators $\mathcal{L}_{xy}$, $\mathcal{L}_{yz}$, $\mathcal{L}_{xz}$ in array representations. Because each of the three operators depends on the other spatial dimension, we introduce a super index for $\mathcal{A}_{xy}$ to represent the operator at difference value of $z = k_3h_z$ for the array representation of $\mathcal{L}_{xy}$:

$$\mathcal{L}_{xy}U = \bigotimes_{1\leqslant k_3\leqslant N_z} \mathcal{A}_{xy}^{k_3}U(:,:,k_3). \tag{23}$$

Similarly, the array representation of $\mathcal{L}_{yz}$ and $\mathcal{L}_{xz}$ can be written in terms of $\mathcal{A}_{yz}^{k_1}$ and $\mathcal{A}_{xz}^{k_2}$.

As a result, the three operators can no longer commute with each other. Notice that

$$e^{(\mathcal{L}_{xy}+\mathcal{L}_{yz}+\mathcal{L}_{xz})\Delta t} = e^{\mathcal{L}_{xy}\Delta t}e^{\mathcal{L}_{xz}\Delta t}e^{\mathcal{L}_{yz}\Delta t} + O\left(\Delta t^2\right), \tag{24}$$

the algorithm for Eq. (22) for this general case becomes only first order in time.

The order of accuracy can be improved by using a Strang splitting scheme to approximate $e^{(\mathcal{L}_{xy}+\mathcal{L}_{yz}+\mathcal{L}_{xz})\Delta t}$. In the Strang splitting method, two linear operators $\mathcal{L}_1$ and $\mathcal{L}_2$ defined on the same linear space have the following property [17]:

$$e^{(\mathcal{L}_1+\mathcal{L}_2)\Delta t} = e^{\mathcal{L}_1\Delta t/2}e^{\mathcal{L}_2\Delta t}e^{\mathcal{L}_1\Delta t/2} + R(\Delta t), \tag{25}$$

where

$$R(\Delta t) = \left(\mathcal{L}_2\mathcal{L}_1^2d + \mathcal{L}_1^2\mathcal{L}_2 + 4\mathcal{L}_2\mathcal{L}_1\mathcal{L}_2 - 2\mathcal{L}_1\mathcal{L}_2\mathcal{L}_1 - 2\mathcal{L}_1\mathcal{L}_2^2 - 2\mathcal{L}_2^2\mathcal{L}_1\right)\Delta t^3/24. \tag{26}$$

For multiple linear operators, $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_m$, the Strang splitting method can be extended to the following by induction:

$$e^{\sum_{i=1}^m \mathcal{L}_i\Delta t} = \prod_{i=1}^{m} e^{\mathcal{L}_i\Delta t/2}\prod_{i=m}^{1} e^{\mathcal{L}_i\Delta t/2} + O\left(\Delta t^3\right). \tag{27}$$

Now, applying Strang splitting to Eq. (5) leads to a second order method in both time and space:

$$U^{n+1}-\frac{\Delta t}{2}F\left(U^{n+1}\right)=e^{\frac{\Delta t}{2}\mathcal{L}_{xy}}e^{\frac{\Delta t}{2}\mathcal{L}_{xz}}e^{\Delta t\mathcal{L}_{yz}}e^{\frac{\Delta t}{2}\mathcal{L}_{xz}}e^{\frac{\Delta t}{2}\mathcal{L}_{xy}}\left(U^{n}+\frac{\Delta t}{2}F\left(U^{n}\right)\right). \tag{28}$$

Consequently, the array representation for solving Eq. (16) with non-constant diffusion coefficients leads to

Algorithm 3 *(AcIIF2 for system (16)).*

$$\begin{aligned}
&V=U^{n}+\frac{\Delta t}{2}f\left(U^{n}\right),\\
&V^{*}=\bigotimes_{1\leqslant k_1\leqslant N_x}e^{\mathcal{A}_{yz}^{k_1}\Delta t}\left(\bigotimes_{1\leqslant k_2\leqslant N_y}e^{\mathcal{A}_{xz}^{k_2}\Delta t/2}\left(\bigotimes_{1\leqslant k_3\leqslant N_z}e^{\mathcal{A}_{xy}^{k_3}\Delta t/2}V(:,:,k_3)\right)(:,k_2,:)\right)(k_1,;,:),\\
&U^{n+1}-f\left(U^{n+1}\right)=\bigotimes_{1\leqslant k_3\leqslant N_z}e^{\mathcal{A}_{xy}^{k_3}\Delta t/2}\left(\bigotimes_{1\leqslant k_2\leqslant N_y}e^{\mathcal{A}_{xz}^{k_2}\Delta t/2}V^{*}(:,k_2,:)\right)(:,:,k_3).
\end{aligned} \tag{29}$$

Operator splitting leads to twice as many exponential-matrix and vector multiplication compared to the non-splitting case in Algorithm 2. Therefore, it is important to use appropriate order of splitting if a subset of operators can commute with each other to improve computational efficiency. For instance, for three operators $\mathcal{L}_i$, $i=1,2,3$, where $\mathcal{L}_1$ and $\mathcal{L}_2$ can commute, however, $\mathcal{L}_3$ cannot, one may have two different kinds of splittings:

$$e^{(\mathcal{L}_1+\mathcal{L}_2+\mathcal{L}_3)\Delta t}=e^{\mathcal{L}_3\Delta t/2}e^{\mathcal{L}_2\Delta t}e^{\mathcal{L}_1\Delta t}e^{\mathcal{L}_3\Delta t/2}+O\left(\Delta^3 t\right) \tag{30}$$

and

$$e^{(\mathcal{L}_1+\mathcal{L}_2+\mathcal{L}_3)\Delta t}=e^{\mathcal{L}_1\Delta t/2}e^{\mathcal{L}_2\Delta t/2}e^{\mathcal{L}_3\Delta t}e^{\mathcal{L}_2\Delta t/2}e^{\mathcal{L}_1\Delta t/2}+O\left(\Delta^3 t\right). \tag{31}$$

Clearly, the splitting in Eq. (30) computes one fewer exponential matrix and vector multiplication than the splitting in Eq. (31).

2.4. AcIIF method for high-dimensional reaction–diffusion systems

We next extend AcIIF to the reaction–diffusion equation in d spatial dimensions with $d\geqslant 3$:

$$u_t=\sum_{1\leqslant i<j\leqslant d}L_{x_ix_j}u+f(u),\quad 0<x_i<1, \tag{32}$$

where

$$L_{x_ix_j}:=a_{ij}\frac{\partial^2}{\partial x_i^2}+2b_{ij}\frac{\partial^2}{\partial x_i\partial x_j}+c_{ij}\frac{\partial^2}{\partial x_i^2},\quad 1\leqslant i<j\leqslant d, \tag{33}$$

and we assume that diffusion coefficients, a_{ij}, b_{ij} and c_{ij} are spatial functions that satisfy the elliptical conditions:

$$a_{ij}>0,\qquad c_{ij}>0,\qquad a_{ij}c_{ij}>b_{ij}^2. \tag{34}$$

We also assume that the boundary conditions for the system are periodic.

Similar to the three-dimensional case, in each direction x_i, there are N_{x_i} grid points with the grid size of h_{x_i}. We use a $N_{x_1}\times N_{x_2}\times\cdots\times N_{x_d}$ d-dimensional array $U=(U_{k_1,k_2,\ldots,k_d})$, $1\leqslant k_i\leqslant N_{x_i}$, $i=1,2,\ldots,d$, to represent the solution, and $\mathcal{L}_{x_ix_j}$ to represent the discretized operator of $L_{x_ix_j}$, $1\leqslant i<j\leqslant d$. Next, we denote $U|_{x_i,x_j}^{(k_r),r\neq i,j}$ as the matrix derived from U by fixing the dimensional index $k_r, r\neq i,j$. Thus, the array representation of $\mathcal{L}_{x_ix_j}$ becomes

$$\mathcal{L}_{x_ix_j}U=\bigotimes_{\substack{1\leqslant k_r\leqslant N_{x_r}\\ r\neq i,j}}\mathcal{A}_{x_ix_j}^{(k_r),r\neq i,j}U|_{x_i,x_j}^{(k_r),r\neq i,j} \tag{35}$$

where $\mathcal{A}_{x_ix_j}^{(k_r),r\neq i,j}$ are linear mappings from the matrix space with all $N_{x_i}\times N_{x_j}$ matrices to itself and is similarly defined in the three-dimensional case.

If $\mathcal{L}_{x_ix_j}$ commute with each other, we are able to directly apply array representation to the IIF2 method to obtain a second order AcIIF method for solving Eq. (32):

$$U^{n+1}-\frac{\Delta t}{2}F\left(U^{n+1}\right)=\bigotimes_{\substack{1\leqslant k_r\leqslant N_{x_r}\\ r\neq 1,2}}e^{\mathcal{A}_{x_1x_2}^{(k_r),r\neq 1,2}\Delta t}\bigotimes_{\substack{1\leqslant k_r\leqslant N_{x_r}\\ r\neq 1,3}}e^{\mathcal{A}_{x_1x_3}^{(k_r),r\neq 1,3}\Delta t}\cdots\bigotimes_{\substack{1\leqslant k_r\leqslant N_{x_r}\\ r\neq d-1,d}}e^{\mathcal{A}_{x_{d-1}x_d}^{(k_r),r\neq d-1,d}\Delta t}V. \tag{36}$$

If $\mathcal{L}_{x_i x_j}$ are not commutable, we apply Strang splitting and array representation to obtain the second order AcIIF method:

$$U^{n+1} - \frac{\Delta t}{2} F\left(U^{n+1}\right) = \bigotimes_{\substack{1 \leqslant k_r \leqslant N_{x_r} \\ r \neq 1,2}} e^{\mathcal{A}^{(k_r), r\neq 1,2}_{x_1 x_2} \Delta t/2} \cdots \bigotimes_{\substack{1 \leqslant k_r \leqslant N_{x_r} \\ r \neq d-1,d}} e^{\mathcal{A}^{(k_r), r\neq d-1,d}_{x_{d-1} x_d} \Delta t/2}$$
$$\times \bigotimes_{\substack{1 \leqslant k_r \leqslant N_{x_r} \\ r \neq d-1,d}} e^{\mathcal{A}^{(k_r), r\neq d-1,d}_{x_{d-1} x_d} \Delta t/2} \cdots \bigotimes_{\substack{1 \leqslant k_r \leqslant N_{x_r} \\ r \neq 1,2}} e^{\mathcal{A}^{(k_r), r\neq 1,2}_{x_1 x_2} \Delta t/2} V, \tag{37}$$

where $V = U^n + \Delta t/2 F(U^n)$ and $1 \leqslant k_r \leqslant N_{x_r}, r = 1, 2, \ldots, d$.

2.5. A sufficient condition for operator commuting

As evident in Strang splitting, proper choice of the order of operators Eq. (29) will decrease the computational cost, which can be improved if commuting operators can be found. Now, we give a sufficient condition for commutable operators.

Proposition 1. *All linear operators $\mathcal{L}_{x_i x_j}$, $1 \leqslant i < j \leqslant d$, commute with each other if the system in Eq. (32) satisfies*:

(1) *The diffusion coefficients are constant,*
(2) *The boundary conditions are periodic along each direction.*

Proof. With a given basis, the linear operator $\mathcal{L}_{x_i x_j}$ for the central difference discretization are $N_1 N_2 \ldots N_d \times N_1 N_2 \ldots N_d$ matrices in the following form:

$$\begin{pmatrix} m_1 & m_2 & \cdots & m_{N-1} & m_N \\ m_N & m_1 & m_2 & \cdots & m_{N-1} \\ m_{N-1} & m_N & m_1 & \cdots & m_{N-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ m_2 & m_3 & \cdots & m_N & m_1 \end{pmatrix} \tag{38}$$

where $N = N_1 N_2 \ldots N_d$ and m_i, $i = 1, 2, \ldots, N$, are real. Let two matrices $A = (a_i)_{N\times N}$ and $B = (b_i)_{N\times N}$ both take the form of Eq. (38). One can show directly that

$$(AB)_{ij} = \sum_{k=1}^{N} a_{k-i+2} b_{j-k+1} = \sum_{s=1}^{N} b_{s-i+2} a_{j-s+1} = (BA)_{ij}, \quad \forall i, j, \tag{39}$$

because $a_{i\pm N} = a_i$, $b_{j\pm N} = b_j$ for $\forall i, j$ where $s = j + i - k - 1$. □

This shows that Strang splitting is unnecessary for constant diffusion coefficients in high spatial dimension and Algorithm 2 is applicable for such reaction–diffusion equations.

3. Stability analysis, higher order methods, and computational costs

Next, we study the linear stability of second order AcIIF methods, derive a third method, and discuss the computational costs of the methods.

3.1. Stability analysis

Based on linear stability analyses in [7] and [8], we claim that the second order AcIIF methods, Eq. (36) and Eq. (37), are asymptotically stable for the case of $F(U) = dU$ and $L_{x_i x_j} u = -cu$, where $d < 0$ and $c > 0$ correspond to stable reactions and elliptic operators. For such a linear case, one has

$$u^{n+1} = e^{-c\Delta t}\left(u^n + \frac{d\Delta t}{2} u^n\right) + \frac{d\Delta t}{2} u^{n+1}. \tag{40}$$

Assuming $u^n = e^{in\theta}$, we obtain

$$e^{i\theta} = e^{-c\Delta t}\left(1 + \frac{1}{2}\lambda\right) + \frac{1}{2}\lambda e^{i\theta}, \tag{41}$$

where $\lambda = d\Delta t$ has a real part λ_r and imaginary part λ_i, leading to

$$\lambda_r = \frac{2(1-e^{-2c\Delta t})}{(1-e^{-c\Delta t})^2 + 2(1+\cos\theta)e^{-c\Delta t}},$$
$$\lambda_i = \frac{4(\sin\theta)ce^{-c\Delta t}}{(1-e^{-c\Delta t})^2 + 2(1+\cos\theta)e^{-c\Delta t}}, \tag{42}$$

since $c > 0$ and $\lambda_r > 0$ for $0 \leqslant \theta \leqslant 2\pi$. Then, the second order AcIIF is A-stable since the stability region includes the complex plane for all λ with $\lambda_r < 0$.

If we apply AcIIF methods Eq. (36) and Eq. (37) to Fokker–Planck equations or chemical master equations, where in each the operator $\mathcal{L}_{x_i x_j}$ defines a Markov process,

$$\frac{dU}{dt} = \mathcal{L}_{x_i x_j} U, \tag{43}$$

then we claim that AcIIF methods Eq. (36) and Eq. (37) are still A-stable.

In order to show the A-stability, we prove that, under certain norm, $\|e^{\mathcal{L}_{x_i x_j}\Delta t}\| \leqslant 1$ holds for any $\Delta t > 0$ for such $\mathcal{L}_{x_i x_j}$. Then, each $\mathcal{L}_{x_i x_j}$ can be treated as elliptic operators and the remaining proof goes through Eq. (40) to Eq. (42). Since Eq. (43) defines a Markov process, the total probability of all states, $\sum U(\Delta t) = \sum e^{\mathcal{L}_{x_i x_j}\Delta t} U(0)$, maintains a value of 1 for any time step, when $U(0)$ is a proper probability distribution, i.e. $U(0) > 0$ for each compartment and $\sum U(0) = 1$. Using the maximum norm $\|.\|_1$ and defining the corresponding linear operator norm, we first prove that for any U with $U > 0$,

$$\left\| e^{\mathcal{L}_{x_i x_j}\Delta t} U \right\|_1 = \sum \left| e^{\mathcal{L}_{x_i x_j}\Delta t} \left(\frac{U}{\sum U} \right) \right| \sum U = \|U\|_1. \tag{44}$$

Then, for $U = U^+ - U^-$ where U^+ is with all positive compartments of U,

$$\begin{aligned} \left\| e^{\mathcal{L}_{x_i x_j}\Delta t} \right\| &= \max_{U\neq 0} \frac{\|e^{\mathcal{L}_{x_i x_j}\Delta t} U\|_1}{\|U\|_1} = \max_{U\neq 0} \frac{\|e^{\mathcal{L}_{x_i x_j}\Delta t} U^+ - e^{\mathcal{L}_{x_i x_j}\Delta t} U^-\|}{\sum |U|} \\ &\leqslant \max_{U\neq 0} \left(\frac{\|e^{\mathcal{L}_{x_i x_j}\Delta t} U^+\|_1}{\sum |U|} + \frac{\|e^{\mathcal{L}_{x_i x_j}\Delta t} U^-\|_1}{\sum |U|} \right) = 1. \end{aligned} \tag{45}$$

Next, we can replace each $\mathcal{L}_{x_i x_j}$ in the Fokker–Planck equation or the chemical master equation by a negative scalar and the proof of A-stability for Eq. (36) and Eq. (37) for these two cases is done.

3.2. High order AcIIF method

If discretization operators are commutable, higher order (in time) AcIIF methods can be derived from the IIF method in a similar manner. For example, the third order IIF scheme [7] has the form:

$$U^{n+1} = e^{\mathcal{L}\Delta t} U^n + \Delta t \left(\frac{5}{12} F\left(U^{n+1}\right) + \frac{2}{3} e^{\mathcal{L}\Delta t} F\left(U^n\right) - \frac{1}{12} e^{2\mathcal{L}\Delta t} F\left(U^{n-1}\right) \right) \tag{46}$$

where $\mathcal{L} = \sum_{i,j} \mathcal{L}_{x_i x_j}$. If all $\mathcal{L}_{x_i x_j}$, $\forall i, j$ commute with each other, we then obtain

$$e^{\mathcal{L}} U = \prod_{1\leqslant i<j\leqslant d} e^{\mathcal{L}_{x_i x_j}} U. \tag{47}$$

If all discretized operators $\mathcal{L}_{x_i x_j}$ commute with each other, applying the array representation leads to the third order AcIIF method:

Algorithm 4 *(Third order AcIIF method).*

$$\begin{aligned} U^{n+1} - \frac{5\Delta t}{12} F\left(U^{n+1}\right) = &\bigotimes_{(k_r),\, r\neq 1,2} e^{\mathcal{A}_{x_1 x_2}\Delta t} \cdots \bigotimes_{(k_r),\, r\neq d-1,d} e^{\mathcal{A}_{x_{d-1} x_d}\Delta t} V_1 \\ &- \bigotimes_{(k_r),\, r\neq 1,2} e^{2\mathcal{A}_{x_1, x_2}\Delta t} \cdots \bigotimes_{(k_r),\, r\neq d-1,d} e^{2\mathcal{A}_{x_{d-1} x_d}\Delta t} V_2 \end{aligned} \tag{48}$$

$$V_1 = U^n + \frac{2\Delta t}{3} F\left(U^n\right), \qquad V_2 = \frac{\Delta t}{12} F\left(U^{n-1}\right) \tag{49}$$

where $1 \leqslant k_r \leqslant N_r$, $r = 1, 2, \ldots, d$.

In the case that some $\mathcal{L}_{x_i x_j}$ are not commutable, the splitting techniques may be applied to this subset of operators to achieve high order accuracy. However, since the formulation becomes much more tedious and complicated, we omit them here.

3.2.1. Remarks on higher order derivatives

The array representation can also be extended for equations with operators that contain high order and cross-derivatives, such as those in the following form:

$$\frac{\partial^m}{\partial x_{i_1}\partial x_{i_2}\cdots\partial x_{i_m}}. \tag{50}$$

Similarly, the second order central difference approximation in the multi-dimensional array U representation results in the discretization linear operator $\mathcal{L}_{x_{i_1},x_{i_2},\ldots,x_{i_m}}$ of the following compact form:

$$\mathcal{L}_{x_{i_1},x_{i_2},\ldots,x_{i_m}}U = \bigotimes_{1\leqslant k_r\leqslant N_r,\, r\neq i_1,i_2,\ldots,i_m} \mathcal{A}_{x_{i_1}x_{i_2}\ldots x_{i_m}} U\big|^{(k_r),r\neq i_1,i_2,\ldots,i_m}_{x_{i_1},x_{i_2},\ldots,x_{i_m}} \tag{51}$$

where $U|^{(k_r),r\neq i_1,i_2,\ldots,i_m}_{x_{i_1},x_{i_2},\ldots,x_{i_m}}$ is a m-dimensional array by fixing the index $k_r, r\neq i_1,i_2,\ldots,i_m$ of U and $\mathcal{A}_{x_{i_1}x_{i_2}\ldots x_{i_m}}$, resulting from the central difference approximation, represents a linear mapping from m-dimensional array linear space to itself.

3.3. Computational cost

For stiff reaction–diffusion equations, the size of the time step usually dictates the overall cost of the temporal updating method. For an A-stable method such as AcIIF and IIF, the cost mainly results from the formation of the exponential-matrix and the corresponding vector-matrix multiplication during each time step. In array representation, small matrices of size in $N_i\times N_i$ for the Laplacian operator or $N_iN_j\times N_iN_j$ when the second order cross-derivatives are presented in contrast to IIF in which the exponential of a $N_1N_2\ldots N_d\times N_1N_2\ldots N_d$ matrix is required. The advantage of AcIIF becomes more prominent for three or higher-dimensional systems.

For a d-spatial dimensional case ($d\geqslant 3$) with second order cross-derivatives, the computational cost for manipulating the exponential matrices in IIF is $O((N_1N_2\ldots N_d)^2)$, or $O(N^{2d})$ for $N_i=N$, $i=1,2,\ldots,d$, while the corresponding cost for AcIIF is

$$\sum_{1\leqslant i<j\leqslant d} O\big((N_iN_j)^2\big)\frac{N_1N_2\ldots N_d}{N_iN_j}. \tag{52}$$

For the case of non-constant coefficients in diffusion, it is $O(d^2N^{d+2})$ when $N_i=N$, $i=1,2,\ldots,d$. For example, a six-dimensional system requires calculating an exponential of matrix with an approximated size of $10^8\times 10^8$ when $N=20$, in contrast to the AcIIF method that only needs exponentials of matrices of a size of 400×400.

Because the exponential-matrices are small in AcIIF, one may pre-calculate the exponential matrices once and store them during the calculations. An alternative approach, which is particularly useful for matrices with sizes exceeding the memory size, is to compute the exponential-matrix vector multiplication without explicit formation of the matrices through, for example, the Krylov subspace method [13,18,19]. In the direct simulations shown in the next section, we implement Padé approximation, which has a computational cost of $O(N^2)$ (both in storage and time) to compute a matrix exponential of $N\times N$ matrix [20], for reaction–diffusion equations with or without cross-derivatives in three dimensions, and we use Krylov subspace method for the Fokker–Planck equations in three or four dimensions and chemical master equations.

3.4. Array representation for chemical master equations

Chemical master equations (CME) is a system of first order ordinary differential equations for stochastic description of the time evolution of a network of biochemical reactions [21]. The solution of the system yields the probability density vector at discrete states of the bio-chemical network in time. The system is typically stiff and it can have many components and states, presenting difficulties for numerical methods and simulations [22,23]. As seen below, the array representation provides a convenient approach to decompose the large rate matrix into smaller matrices for efficient usage of integration factor methods that can best deal with stiffness in the system.

If a given chemical reaction system consists of d molecular species, namely $X_1, X_2, \ldots, X_d$, with maximal copy numbers of $N_1, N_2, \ldots, N_d$, respectively, then the system has $N_{tot}=N_1N_2\ldots N_d$ possible states, for which a vector $x=(x_1,x_2,\ldots,x_d)$ denotes each state. The R-th reaction takes the form

$$R_r\colon\ w_1^rX_1+w_2^rX_2+\cdots+w_d^rX_d\to v_1^rX_1+v_2^rX_2+\cdots+v_d^rX_d, \tag{53}$$

where v_i^r and w_i^r, for every i and r, are non-negative integers and the system contains R number of reactions. The reaction rate at state x is $a_r(x)=a_r(x_1,x_2,\ldots,x_d)$ and the vector

$$\alpha^r=\big(v_1^r-w_1^r, v_2^r-w_2^r,\ldots,v_d^r-w_d^r\big) \tag{54}$$

denotes the change of copy number of the molecular species after the r-th reaction occurs once.

In array representation, one can use an $N_1 \times N_2 \times \cdots \times N_d$ d-dimensional array U to denote the probability density function, and each component of U, U_x, as the probability density at state x, which can be written as

$$U_x(t) = \text{Prob}[X_1 = x_1,\ X_2 = x_2,\ \ldots,\ X_d = x_d,\ \text{at time } t]. \tag{55}$$

Define the linear mapping $\mathcal{L}_r$,

$$(\mathcal{L}_r U)_x = a_r\left(x - \alpha^r\right) U_{x-\alpha^r}(t) - a_r(x) U_x. \tag{56}$$

The CME for the probability density functions becomes

$$\dot{U}(t) = \sum_{r=1}^{R} \mathcal{L}_r U(t). \tag{57}$$

To introduce the array representation for $\mathcal{L}_r$, we let $i_1^r, i_2^r, \ldots, i_{m_r}^r$ denote the indices of non-zero entries in α^r. Using the same notation as in Eq. (51), $U|_{X_{i_1^r}, X_{i_2^r}, \ldots, X_{i_{m_r}^r}}^{(x_j),\, j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r}$ denotes a m_r-dimensional array by fixing indexes x_j, $j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r$. Then, one obtains

$$U = \bigotimes_{\substack{1 \leqslant x_j \leqslant N_j \\ j \neq i_1^r, \ldots, i_{m_r}^r}} U|_{X_{i_1^r}, X_{i_2^r}, \ldots, X_{i_{m_r}^r}}^{(x_j),\, j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r}. \tag{58}$$

Define the linear mapping $\mathcal{A}_{X_{i_1^r}, X_{i_2^r}, \ldots, X_{i_{m_r}^r}}^{(x_j),\, , j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r}$ on m_r-dimensional array V as

$$\left(\mathcal{A}_{X_{i_1^r}, X_{i_2^r}, \ldots, X_{i_{m_r}^r}}^{(x_j),\, , j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r} V\right)_{(x_i),\ i \in i_1^r, i_2^r, \ldots, i_{m_r}^r} = a_r\left(x - \alpha^r\right) V_{(x_i - \alpha_i^r),\ i \in i_1^r, i_2^r, \ldots, i_{m_r}^r} - a_r(x) V_{(x_i),\ i \in i_1^r, i_2^r, \ldots, i_{m_r}^r}. \tag{59}$$

Then the array representation of $\mathcal{L}_r$ becomes

$$\mathcal{L}_r U = \bigotimes_{\substack{1 \leqslant x_j \leqslant N_j \\ j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r}} \mathcal{A}_{X_{i_1}, X_{i_2}, \ldots, X_{i_{m_r}}}^{(x_j),\, j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r} U|_{X_{i_1}, \ldots, X_{i_{m_r}}}^{(x_j),\, j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r}. \tag{60}$$

For typical systems, each $\mathcal{L}_r$, $r = 1, 2, \ldots, R$, cannot commute with one another, thus the Strang splitting method is applied to approximate the solution, resulting in a second order integration factor method (AcIIF2) for CME Eq. (57):

$$U^{n+1} = \prod_{r=1}^{R} e^{\mathcal{L}_r \Delta t/2} \prod_{r=R}^{1} e^{\mathcal{L}_r \Delta t/2} U^n, \tag{61}$$

where U^n denotes the probability density functions at time $t_n = n\Delta t$.

The exponential of $\mathcal{L}_r$ can be written in terms of the exponentials of $\mathcal{A}_{X_{i_1}, X_{i_2}, \ldots, X_{i_{m_r}}}^{(x_j),\, j \neq i_1^r, i_2^r, \ldots, i_{m_r}^r}$. If the reaction R_r only affects copy numbers of a few species, implying m_r is small, the calculation of the latter exponential is much more efficient than computing the original one. In other words, the array representation saves storage and CPU time for the system containing many molecular species while each reaction only affects the copy number of a small portion of species.

4. Numerical simulations

To explore various applications of the AcIIF methods (Eq. (36) and Eq. (37)), we apply the second order AcIIF methods to five different systems: three-dimensional reaction–diffusion equations with constant diffusion coefficients or spatially-dependent diffusion constants; three- and four-dimensional Fokker–Planck equations; and chemical master equations arising from a biological application.

4.1. Three-dimensional reaction–diffusion equation with constant diffusion coefficients

We first apply AcIIF method Eq. (36) to the following reaction–diffusion equation:

$$u_t = (0.1u_{xx} - 0.15u_{xy} + 0.1u_{yy}) + (0.1u_{xx} + 0.2u_{xz} + 0.2u_{zz}) + (0.2u_{yy} + 0.15u_{yz} + 0.1u_{zz}) + 0.8u, \tag{62}$$

where $x, y, z \in (0, 2\pi)$ with periodic boundary conditions. With the initial condition $u(x, y, z, 0) = \sin(x + y + z)$, the equation has the exact solution $u(x, y, z, t) = e^{-0.2t} \sin(x + y + z)$.

Based on the result from Section 2.5, for this case, the corresponding linear operators $\mathcal{L}_{xy}, \mathcal{L}_{yz}$ and $\mathcal{L}_{xz}$ can commute with each other. Thus, Eq. (36) is a second order scheme in both time and space. We first compare the second order

Table 1
A comparison between the second order array-representation compact IIF method (AcIIF2 in Eq. (36)) and IIF2 method. The symbol "–" denotes insufficient memory in calculation of the exponential matrix.

N	Error in L_∞		Accuracy order		CPU time	
	AcIIF2	IIF2	AcIIF2	IIF2	AcIIF2	IIF2
8	0.0672	0.0672	–	–	0.05 s	0.34 s
16	0.0169	0.0169	1.99	1.99	0.15 s	30.5 s
32	0.0042	–	2.01	–	5.13 s	–
64	0.0011	–	1.93	–	246 s	–

Table 2
A comparison between AcIIF2 method (Eq. (37)) and IIF2 for the case of non-constant diffusion coefficients.

N	Error in L_∞		Accuracy order		CPU time	
	AcIIF2	IIF2	AcIIF2	IIF2	AcIIF2	IIF2
8	0.2744	0.2754	–	–	0.29 s	1.35 s
16	0.0675	0.0678	2.02	2.02	2.2 s	155 s
32	0.0169	–	2.00	–	133 s	–

array-representation compact IIF with the standard IIF, both in second order. Because both methods are A-stable, we choose $\Delta t = 1/N = h_x/2\pi$ where $N_x = N_y = N_z = N$ and $\Delta t = 1/N = h_x/2\pi$, and simulation results are evaluated at $t = 1$. As seen in Table 1, both methods clearly show second order accuracy with similar sizes of errors as N increases, as one may expect from the analysis of both methods. On the other hand, we observe the CPU time for both methods to achieve the same accuracy is much larger in IIF than in AcIIF, because the exponential matrices in IIF have much larger size than AcIIF. When N becomes 32, IIF fails to compute as the size for matrix exponential becomes exceedingly large, leading to a lack of sufficient memory in a Matlab implementation on typical personal computers (4 GB). Even in a cluster where computing a $32^3 \times 32^3$ matrix exponential is possible, the CPU time needed will be about 2 hours, and computation of a $64^3 \times 64^3$ matrix exponential takes more than a day. On the other hand, AcIIF runs normally with good accuracy, showing clear advantages in handling larger grid numbers for convergence of solutions.

4.2. Three-dimensional diffusion–reaction system with non-constant diffusion coefficients

To test the case with non-commutable differential operators, we consider the following reaction–diffusion equations with non-constant coefficients:

$$\begin{aligned} u_t = {} & \big(0.5u_{xx} - 0.5\sin(x+y)u_{xy} + 0.5u_{yy}\big) + (0.5u_{xx} - 1/3\cos y u_{xz} + 1/3u_{zz}) \\ & + (1+\cos x)(0.5u_{yy} - 0.5u_{yz} + 1/3u_{zz})) + f(x,y,z,u), \end{aligned} \tag{63}$$

where $x, y, z \in (0, 2\pi)$ with periodic boundary conditions. With the initial condition $u(x, y, z, 0) = \sin(x + y + z)$, the equation has the exact solution $u(x, y, z, t) = e^{-0.2t}\sin(x + y + z)$. Similar to the previous case, we choose $N_x = N_y = N_z = N$, grid size $h_x = h_y = h_z = 2\pi/N$, $\Delta t = 1/N$, and $t = 1$ as the temporal point for evaluating the method.

In this non-commutable case, we need to compute N number of array-representation operators for each of $\mathcal{L}_{xy}, \mathcal{L}_{xz}$ and $\mathcal{L}_{yz}$. For example, to compute $e^{\mathcal{L}_{yz}}$, we need the following calculations:

$$\begin{aligned} & \mathcal{A}_{yz}^{k_x} \sim \big(1 + \cos\big((k_x - 1)h_x\big)\big)\big(0.5(\cdot)_{yy} - 0.5(\cdot)_{yz} + 1/3(\cdot)_{zz}\big), \\ & e^{\mathcal{L}_{yz}} = \bigotimes_{k_x} \mathcal{A}_{yz}^{k_x} U\big|_{yz}^{k_x} \end{aligned} \tag{64}$$

for $k_x = 1, 2, \ldots, N$. In the commutable case, we only compute and save three of the exponential matrices of size $N^2 \times N^2$, in contrast to $3N$ of exponential matrices of the same size. As a result, the non-commutable case takes significantly more CPU time than the commutable case as seen in Table 1 and Table 2. However, compared to the standard IIF method, AcIIF is still significantly much faster.

The order of accuracy for both AcIIF2 and IIF2 remain second order, as seen in Table 2. However, the error for the non-commutable case is larger than the commutable case at the same spatial and temporal resolutions, which is likely due to the splitting error in time. Similar to the constant diffusion case, IIF2 fails to run due to the memory problem for relatively larger N.

4.3. Three- and four-dimensional Fokker–Planck equations

The Fokker–Planck equation (FPE) describes the time evolution of the probability density function of stochastic systems [1]. The generalized FPE usually takes the following form:

$$\frac{\partial p(x,t)}{\partial t} = -\sum_{r=1}^{R}\left\{\sum_{i}^{N} n_{ri}\frac{\partial}{\partial x_i}\left(q_r(x,t) - \frac{1}{2}\sum_{j=1}^{N} n_{rj}\frac{\partial q_r(x,t)}{\partial x_j}\right)\right\}. \tag{65}$$

Here, in the case of bio-chemical reactions, R denotes the total number of chemical reactions involved in the system, N denotes the total number of different species participating the reactions, x_j denotes the copy number of j-th reactant, and n_{ri} denotes the change of copy number of reactant i when the r-th reaction occurs once. $p(x,t)$ represents the probability density of the system at the state $x = (x_1, x_2, \ldots, x_N)$ $(x \in R^{N+})$ and time t. In addition, we define

$$q_r(x,t) = w_r(x,t)p(x,t), \tag{66}$$

where $w_r(x,t)$ is the reaction propensity function for r-th reaction at state x. For example, for the following bio-chemical reactions:

$$X \xrightarrow{k_1[X]} Y, \qquad X + Y \xrightarrow{k_2[X][Y]} Z, \qquad \emptyset \xrightarrow{k_3} Z, \tag{67}$$

we have $n_1 = (-1, 1, 0)$, $n_2 = (-1, -1, 1)$, $n_3 = (0, 0, 1)$ and

$$w_1(x,y,z,t) = k_1 x, \qquad w_2(x,y,z,t) = k_2 xy, \qquad w_3(x,y,z,t) = k_3 z. \tag{68}$$

In general, FPE is a N-dimensional convection–diffusion equations with non-constant diffusive coefficients and second order cross-derivatives. Because the system may be stiff, implicit temporal methods, such as Crank–Nicolson method [24], which requires solving nonlinear systems of large size at each time step, are often needed. While directly apply IIF method, the calculation of the huge matrix exponential is unaffordable in the high-dimensional case. AcIIF, which has the good stability like Crank–Nicolson, is a better choice in solving FPE than IIF method as it divides the entire discretization matrix into multiple small pieces by the array-representation technique.

To apply AcIIF to FPE, we first study a three-dimensional case in which there are two metabolites and one enzyme, which is also studied in [25]. The reactions are:

$$\begin{aligned}
&\emptyset \xrightarrow{k_1} A, \qquad \emptyset \xrightarrow{k_2} B,\\
&A + B \xrightarrow{k_3} \emptyset,\\
&A \xrightarrow{k_4} \emptyset, \qquad B \xrightarrow{k_5} \emptyset,\\
&\emptyset \xrightarrow{k_6} E_A, \qquad E_A \xrightarrow{k_7} \emptyset,
\end{aligned} \tag{69}$$

The corresponding propensity rates are given as

$$\begin{aligned}
&k_1 = \frac{k_A[E_A]}{1 + [A]/K_I}, \qquad k_2 = k_B, \qquad k_3 = k[A][B],\\
&k_4 = \mu[A], \qquad k_5 = \mu[B], \qquad k_6 = \frac{k_{E_A}}{1 + [A]/K_R}, \qquad k_7 = \mu[E_A]
\end{aligned} \tag{70}$$

where $k_A = 0.3\ \mathrm{s}^{-1}$, $k_B = 2\ \mathrm{s}^{-1}$, $K_I = 30$, $k = 0.001\ \mathrm{s}^{-1}$, $\mu = 0.004\ \mathrm{s}^{-1}$, $K_R = 30$ and $k_{E_A} = 1\ \mathrm{s}^{-1}$ [25].

The computational domain for this system is chosen to be $\Omega_h = [0, 100] \times [0, 100] \times [0, 45]$, which is large enough such that the probability of $[A] > 100$, $[B] > 100$, $[E_A] > 45$ is sufficiently small, implying that the domain covers nearly all the possible states of the chemical reactions. After discretizing the FPE using second order central differences, we represent the density function by a three-dimensional array $U(t)$ to represent the density function. Each component $U_{i_1,i_2,i_3}(t)$ denotes the probability density for system at time t and state $[A] = i_1$, $[B] = i_2$, $[E_A] = i_3$. There are seven reactions, thus, in FPE equation (65), corresponding to $R = 7$. For r-th reaction, the corresponding discretized operator, denoted by $\mathcal{L}_r$, becomes

$$\mathcal{L}_r = \sum_{i}^{N} n_{ri}\frac{\partial}{\partial x_i}\left(q_r(x,t) + \frac{1}{2}\sum_{j=1}^{N} n_{rj}\frac{\partial q_r(x,t)}{\partial x_j}\right). \tag{71}$$

Because $\mathcal{L}_r$ contains no cross-derivatives for $r \neq 3$, we can use the array representation presented in Section 2.1. On the other hand, $\mathcal{L}_3$ contains a cross-derivative $\partial^2/\partial[A]\partial[B]$, we use the array representation presented in Section 2.3. By direct application of AcIIF (Eq. (37) based on splitting technique), we obtain an overall second order method. In particular, some of the reactions can be grouped into one matrix to reduce the number of splittings and number of calculations of exponential matrices, such as $\mathcal{R}_1$ and $\mathcal{R}_4$, which both have $\partial^2/\partial[A]^2$ and $\partial/\partial[A]$ in Eq. (71).

To study the performance of AcIIF2, we also implement the second order Runge–Kutta (RK2) method for a comparison. The error of solution in the maximal norm is based on a simulation result from the finest "spatial" grid ($N_A = N_B = 200$, $N_{E_A} = 120$) and finest time step ($\Delta t = 5 \times 10^{-3}$). The initial condition for each simulation is a Gaussian distribution centered at point $(30, 40, 20)$ with standard derivation $\sqrt{30}$.

Table 3
A comparison between the second order AcIIF and Runge–Kutta (RK2) for the three-dimensional FPE (69) at $t = 30$.

Grids (N_A, N_B, N_{E_A})	AcIIF2/RK2			
	Δt	Error in L_∞	CPU time	RK2 unstable when
(25, 25, 15)	5/0.2	$2.6 \times 10^{-4}/2.6 \times 10^{-4}$	17.6 s/71.2 s	$\Delta t \geqslant 0.3$
(40, 40, 24)	5/0.15	$1.3 \times 10^{-4}/1.4 \times 10^{-4}$	35.3 s/182.5 s	$\Delta t \geqslant 0.2$
(50, 50, 30)	5/0.1	$8.6 \times 10^{-5}/9.1 \times 10^{-5}$	65.5 s/470.2 s	$\Delta t \geqslant 0.15$

Table 4
The CPU time for different grid numbers of the fourth dimension of the FPE.

N_{E_B}	4	8	16	32
CPU time(s)	5.6	8.3	14.5	32.6

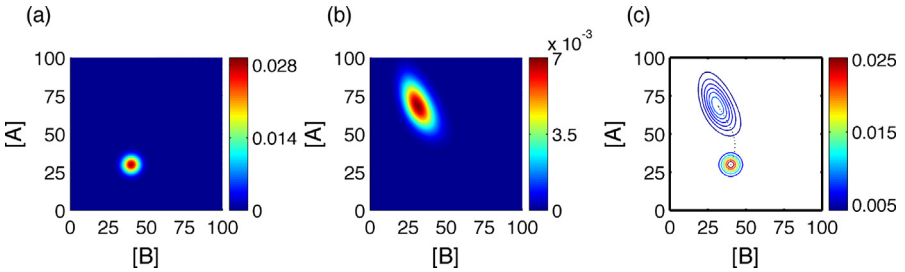

Fig. 1. Numerical solution of system (69) using AcIIF2. Temporal discretization is set by the time step $\Delta t = 1$ s, and the simulation is ran up to time $t = 50$ s. (a) Shows the initial distribution of molecular species A and B, which are Gaussian distributions centered at $(A, B) = (30, 40)$. (b) The distribution of molecular species A and B at $t = 50$ s. (c) The contour plot of initial and final distributions. The dotted black line connects the centers of the solutions of the rate equations of system (69).

First, we observe in Table 3 that a much smaller Δt is required for RK2 to converge compared to AcIIF2 due to the fact that the reactions are stiff, requiring small Δt, for non A-stable methods such as RK2. Interestingly, AcIIF2 can reach the same overall error level as RK2 using a much larger time step for the same-sized "spatial" mesh, indicating that the numerical error for solving this FPE is likely dominated by spatial discretization. Thus, a large time step is sufficient for A-stable methods, such as IIF, while small time steps are still required for RK2 due to its stability constraints. In each time step RK2 is more efficient than AcIIF2; however, AcIIF2, which requires fewer time steps for a given t, still outperforms RK2 significantly in this case. We also plot the numerical results in Fig. 1, where a grid with $N_A = N_B = 60$ and $N_{E_A} = 30$ and time step $\Delta t = 1$ s are used.

Next, we add another enzyme E_B that synthesizes metabolite B in the same way that E_A synthesizes A in the three-dimensional system (69). This extension leads to a four-dimensional FPE of four molecular species $[A]$, $[B]$, $[E_A]$ and $[E_B]$ [25],

$$
\begin{aligned}
&\emptyset \xrightarrow{\frac{k_A \cdot e_A}{1+a/K_I}} A, \qquad \emptyset \xrightarrow{\frac{k_B \cdot e_B}{1+b/K_I}} B,\\
&A + B \xrightarrow{k_2 \cdot a \cdot b} \emptyset,\\
&A \xrightarrow{\mu \cdot a} \emptyset, \qquad B \xrightarrow{\mu \cdot b} \emptyset,\\
&\emptyset \xrightarrow{\frac{k_{E_A}}{1+a/K_R}} E_A, \qquad E_A \xrightarrow{\mu \cdot e_A} \emptyset,\\
&\emptyset \xrightarrow{\frac{k_{E_B}}{1+b/K_R}} E_B, \qquad E_B \xrightarrow{\mu \cdot e_B} \emptyset,
\end{aligned}
\tag{72}
$$

where $k_A = k_B = 0.3$ s^{-1}, $k_2 = 0.001$ s^{-1}, $K_I = 60$, $\mu = 0.002$ s^{-1}, $k_{E_A} = k_{E_B} = 0.02$ s^{-1} and $K_R = 30$.

The computational domain is chosen to be $[0, 80] \times [0, 80] \times [0, 30] \times [0, 30]$ that contains nearly all possible states of the system. We choose zero Dirichlet boundary conditions with the initial condition as a Gaussian distribution centering at $(30, 40, 15, 12)$ with a standard deviation $\sqrt{40}$. There are nine reactions in the system, corresponding to nine array-representation operators. Based on commutability of the operators, we group some of them similar to the three-dimensional case to increase the overall computational efficiency. One interesting observation is that as the fourth dimension grid number N_{E_B} increases, the CPU time for increases only linearly, as seen in Table 4. For this set of simulations, we fix the other three grid numbers: $N_A = N_B = N_{E_A} = 10$, and keep doubling N_{E_B} from 4 to 32. IIF method computes the entire matrix exponential, thus its CPU time will increase by a fourth folder. While AcIIF only compute small matrix exponential, its CPU

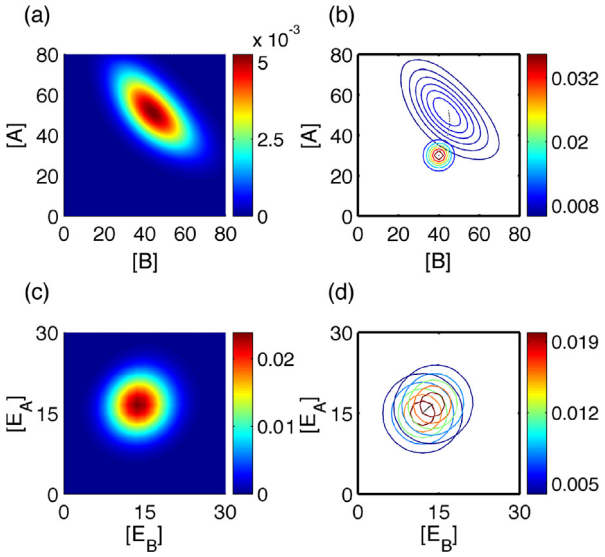

Fig. 2. Numerical solution of system (72) using AcIIF2. Temporal discretization is set by the time step $\Delta t = 1$ s, and the simulation is ran up to time $t = 35$ s. (a) The distribution of molecular species A and B at $t = 35$ s. (b) The contour plot of initial and final distributions of molecular species A and B. The dotted black line connects the centers of the solutions of the rate equations of system (72). (c) The distribution of molecular species E_A and E_B at $t = 35$ s. (d) The contour plot of initial and final distributions of molecular species E_A and E_B. The dotted black line connects the centers of the solutions of the rate equations of system (72).

time will linearly depends on N_{E_B}. Finally, we plot the numerical results for $N_A = N_B = 40$, $N_{E_A} = N_{E_B} = 20$ and time step $\Delta t = 1$ (Fig. 2).

4.4. An application to chemical master equations

The chemical master equation (CME) describes the time evolution of the probability density function. In CME, each reaction on the probability density evolution may be considered as diffusion-like operators with cross-derivatives. Thus, AcIIF can be applied to solve such equations. We consider a family of proteins X with different conformational types $X_1, X_2, \ldots, X_d$. Two conformational types X_i and X_{i+1} can conform to each other through an enzyme E. Suppose that during reactions, no new protein is created; the enzyme is abundant so that one can treat the quantity of the enzyme as a constant; and intermediate products are extremely unstable. As a result, the entire system consists of the following bio-chemical reactions:

$$\begin{aligned} &X_1 \overset{k_1^+}{\to} X_2 \overset{k_2^+}{\to} X_3 \cdots \overset{k_{d-1}^+}{\to} X_d, \\ &X_1 \overset{k_1^-}{\leftarrow} X_2 \overset{k_2^-}{\leftarrow} X_3 \cdots \overset{k_{d-1}^-}{\leftarrow} X_d, \end{aligned} \tag{73}$$

for which the total copy number of the protein X is a constant,

$$X_1 + X_2 + \cdots + X_d = N. \tag{74}$$

For simplicity, $x = (x_1, x_2, \ldots, x_d)$ denotes each state where $0 \leqslant x_i \leqslant N$, $i = 1, 2, \ldots, d$ (although some of the states cannot be reached). In particular, the reaction

$$X_i \overset{k_i^+}{\to} X_{i+1} \tag{75}$$

defines a linear mapping on probability density function in CMEs, with the following array representation:

$$\mathcal{A}^{(x_j), j\neq i, i+1}_{X_i, X_{i+1}} M_{m,n} = -k_1^+ m M_{m,n} + k_1^+ (m+1) M_{m+1,n-1} \tag{76}$$

where M is a 2-dimensional array. Other reactions can be treated in a similar way.

For a protein family with d conformational types and N total number of copies, the direct calculation of exponential of the linear mapping requires the exponentiation of a $N^d \times N^d$ matrix. However, in the array representation, only $N^2 \times N^2$ matrices' exponentials are required to be calculated. More saving in both storage and CPU time result in using the array representation when the number of species d gets larger.

To demonstrate this through direct simulations, we implement a second order array-representation integration factor method as well as the second order Runge–Kutta (a standard temporal integrator for CMEs) for the case of $N = 30$ and

Table 5
A comparison between the second order array-representation compact integration factor method (AcIIF2) and Runge–Kutta (RK2) methods for simulating CMEs.

Δt	AcIIF2		RK2	
	Error	Order of acc	Error	Order of acc
1/4	7.95×10^{-4}	–	unstable	
1/8	1.96×10^{-4}	2.02	unstable	
1/16	4.89×10^{-5}	2.00	unstable	
1/32	1.22×10^{-5}	2.00	unstable	
1/64	3.06×10^{-6}	2.00	unstable	
1/128	7.64×10^{-7}	2.00	3.74×10^{-7}	–
1/256	1.91×10^{-7}	2.00	9.30×10^{-7}	2.01

$d = 3$. The initial distribution of the molecules is set to be $P(X_1 = 30) = 1$, that is, initially all molecules take the conformational type X_1. We choose rate coefficients $k_1^+ = k_2^- = 1$, $k_1^- = 2$, $k_2^+ = 3$ and we compute the solution up to $t = 3$ using different Δt. The maximal error of the solution is estimated based on an "exact" solution computed using a very small time step by RK2.

First, the second order accuracy of AcIIF2 method is clearly observed in Table 5. As expected, RK2 requires a very small time step due to its stability constraint in contrast to AcIIF's stable and good accuracy, even at a time step as large as $\Delta t = 1/4$. As Δt decreases significantly (e.g. $\Delta t \leqslant 1/128$), RK2 becomes stable and converges as seen in Table 5. At the same size of time step, we observe AcIIF2 and RK2 has similar size of errors. Of course, using the same size of time step, RK2 takes less CPU time and storage than AIF2, with both achieving similar accuracy. However, if moderately high accuracy (e.g. 10^{-4} for this particular system) is sufficient, AIF2 shows its advantage. In particular, as the number of species increases or the rate constants become more stiff, a combination of the array representation and the integration factor method becomes even more attractive in achieving both efficiency and accuracy.

5. Discussions and conclusions

Higher order spatial derivatives and reactions of drastically different time scales demand temporal schemes of the generous stability constraint. Implicit integration factor methods, which solve exactly the linear operator of higher order spatial derivatives along with an implicit treatment of the stiff reactions, are effective approaches for such types of different equations. One unique computational challenge associated with such methods is the handling of exponentials of matrices. Here, we have introduced a new array representation for the discretization matrices of the linear differential operators. Because of such representation, computing exponentials of large matrices is reduced to the calculation of exponentials of matrices of significantly smaller sizes. The saving and advantages for array representations in both storage and CPU time escalate as the dimension of the system increases. In addition, this approach can be directly combined with the implicit integration method for an overall efficient method (termed as AcIIF) of excellent temporal stability.

Due to its advantage for high dimensions and stiff reactions, such an approach is particularly appropriate for solving reaction–diffusion equations and other diffusion-like equations, such as Fokker–Planck equations. Our direct implementation and testing of the second order AcIIF, which is linearly absolute stable, has demonstrated its advantages compared to some existing approaches. Interestingly, such array representation can also be applied to chemical master equations, ODE systems of large size that often is stiff. The computational efficiency for such applications become most evident for biochemical networks of a large number of species with each reaction in the system affecting only few species.

Although the array representation has been presented only in the context of compact implicit integration factor methods, the approach can easily be applied to other integration factor or exponential difference methods. Other type of equations of higher order derivatives, (e.g. Cahn–Hilliard equations [26] of fourth order derivatives) in addition to reaction–diffusion equations and Fokker–Planck equations may also be handled using the array representation for better efficiency. To better deal with high spatial dimensions, one can incorporate the sparse grid [27] into the array-representation technique. The flexibility of such representation allows either direct calculation of the exponentials of matrices or using Krylov subspace for computing their exponential matrix-vector multiplications for saving in storages. Overall, the array representation along with integration factor methods provides an efficient approach for solving a wide range of problems arising from biological and physical applications.

Acknowledgements

We would like to thank Jie Liang, Xinfeng Liu, Yongtao Zhang, and Hsiao-Mei Lu for many stimulating discussions. This work was supported by NSF grant DMS1161621 and NIH grants R01GM67247 and P50GM76516.

References

[1] H. Risken, The Fokker–Planck Equation: Methods of Solutions and Applications, Springer, 1996.

[2] A. Kassam Lloyd, N. Trefethen, Fourth-order time stepping for stiff PDEs, SIAM J. Sci. Comput. 26 (2005) 1214–1233.
[3] B. Kleefeld, A. Khaliq, B. Wade, An ETD Crank–Nicolson method for reaction–diffusion systems, Numer. Methods Partial Differ. Equ. 28 (2012) 1309–1335.
[4] S. Cox, P. Matthews, Exponential time differencing for stiff systems, J. Comput. Phys. 176 (2002) 430–455.
[5] Q. Du, W. Zhu, Stability analysis and application of the exponential time differencing schemes, J. Comput. Math. 22 (2004).
[6] S. Krogstad, Generalized integrating factor methods for stiff PDEs, J. Comput. Phys. 203 (2005) 72–88.
[7] Q. Nie, Y.-T. Zhang, R. Zhao, Efficient semi-implicit schemes for stiff systems, J. Comput. Phys. 214 (2006) 521–537.
[8] Q. Nie, F. Wan, Y.-T. Zhang, X. Liu, Compact integration factor methods in high spatial dimensions, J. Comput. Phys. 227 (2008) 5238–5255.
[9] X. Liu, Q. Nie, Compact integration factor methods for complex domains and adaptive mesh refinement, J. Comput. Phys. 229 (2010) 5692–5706.
[10] X.-D. Liu, S. Osher, T. Chan, Weighted essentially nonoscillatory schemes, J. Comput. Phys. 115 (1994) 200–212.
[11] G.-S. Jiang, C.-W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. 126 (1996) 202–228.
[12] S. Zhao, J. Ovadia, X. Liu, Y.-T. Zhang, Q. Nie, Operator splitting implicit integration factor methods for stiff reaction–diffusion–advection systems, J. Comput. Phys. 230 (2011) 5996–6009.
[13] S. Chen, Y.-T. Zhang, Krylov implicit integration factor methods for spatial discretization on high dimensional unstructured meshes: Application to discontinuous Galerkin method, J. Comput. Phys. 230 (2011) 4336–4352.
[14] E. Gallopoulos, Y. Saad, Efficient solution of parabolic equations by Krylov approximation methods, SIAM J. Sci. Stat. Comput. 13 (1992) 1236–1264.
[15] T. Hou, J. Lowengrub, M. Shelley, Removing the stiffness from interfacial flows with surface tension, J. Comput. Phys. 114 (1994) 312.
[16] G. Beylkin, J. Keiser, L. Vozovoi, A new class of tie discretization schemes for the solution of nonlinear PDEs, J. Comput. Phys. 147 (1998) 362–387.
[17] G. Strang, On the construction and comparison of difference schemes, SIAM J. Numer. Anal. 5 (1968) 506–517.
[18] M. Hochbruck, C. Lubich, On Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 34 (1997) 1911–1925.
[19] Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 29 (1992) 209–228.
[20] C. Moler, C. Loan, Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later, SIAM Rev. 45 (2003) 3–49.
[21] D. Gillespie, A rigorous derivation of the chemical master equation, Physica A 188 (1992) 404–425.
[22] T. Jahnke, W. Huisinga, Solving the chemical master equation for monomolecular reaction systems analytically, J. Math. Biol. 54 (2007) 1–26.
[23] V. Wolf, et al., Solving the chemical master equation using sliding windows, BMC Syst. Biol. 4 (2010).
[24] L. Ferm, P. Lotstedt, P. Sjoberg, Conservative solution of the Fokker–Planck equation for stochastic chemical equation, Technical Report, 2004.
[25] P. Sjoberg, P. Lotstedt, J. Elf, Fokker–Planck approximation of the master equation in molecular biology, Comput. Vis. Sci. 12 (2009) 37–50.
[26] J. Cahn, J.E. Hilliard, Free energy of a nonuniform system. I. Interfacial free energy, J. Chem. Phys. 28 (1958) 258–351.
[27] J. Shen, H. Yu, Efficient spectral sparse grid methods and applications to high-dimensional elliptic problems, SIAM J. Sci. Comput. 32 (2010) 3228–3250.