Journal of Computational and Applied Mathematics 348 (2019) 89–102

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

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

High order integration factor methods for systems with inhomogeneous boundary conditions

Sameed Ahmed, Xinfeng Liu *

Department of Mathematics, University of South Carolina, Columbia, SC 29208, USA

ARTICLE INFO

Article history:
Received 16 May 2017
Received in revised form 15 May 2018

Keywords:
Integration factor method
Compact representation

ABSTRACT

Due to the high order spatial derivatives and stiff reactions, severe temporal stability constraints on the time step are generally required when developing numerical methods for solving high order partial differential equations. Implicit integration method (IIF) method along with its compact form (cIIF), which treats spatial derivatives exactly and reaction terms implicitly, provides excellent stability properties with good efficiency by decoupling the treatment of reaction and spatial derivatives. 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. The compact representation for IIF (cIIF) was introduced to save the computational cost and storage for this purpose. Another challenge is finding the matrix of high order space discretization, especially near the boundaries. In this paper, we extend IIF method to high order discretization for spatial derivatives through an example of reaction diffusion equation with fourth order accuracy, while the computational cost and storage are similar to the general second order cIIF method. The method can also be efficiently applied to deal with other types of partial differential equations with both homogeneous and inhomogeneous boundary conditions. Direct numerical simulations demonstrate the efficiency and accuracy of the approach.

1. Introduction

Let Ω be an open rectangular domain in $\mathbb{R}^d$ and a final time $T > 0$. In this paper, we consider solving a system of reaction–diffusion equations:

$$\begin{cases} \dfrac{\partial u}{\partial t} = D\Delta u + f(u), & \mathbf{x} \in \Omega, \quad t \in (0,T), \\ u|_{t=0} = u_0, & \mathbf{x} \in \Omega, \end{cases} \tag{1}$$

where $D > 0$ is the diffusion coefficient. Different boundary conditions such as the homogeneous and inhomogeneous Dirichlet and Neumann boundary conditions will all be studied in this paper. Due to severe time step constraints, one of the numerical difficulties to handle such equations is to efficiently solve the diffusion term Δu coupled with the stiff nonlinear reaction term $f(u)$. In general, the time step relies heavily on the stiffness of reactions and treatment of the high order derivatives. Integration factor (IF) or exponential differencing time (ETD) methods are popular methods for temporal partial differential equations (PDEs) [1–7].

To efficiently store and compute the exponential matrices in IIF for two and three dimensional systems in Cartesian coordinates with regular meshes, a class of compact implicit integration factor (cIIF) method [8] was introduced that has the same stability properties as the original IIF [9], but with significant improvement on storage and computational savings

* Corresponding author.
E-mail address: xfliu@math.sc.edu (X. Liu).

https://doi.org/10.1016/j.cam.2018.08.036
0377-0427/

for high spatial dimensions. In order to efficiently handle the complex domains with circular or spherical symmetry, cIIF methods were generalized to curvilinear coordinates through examples of polar and spherical coordinates [10]. One can also apply cIIF to stiff reactions and diffusions while using other specialized hyperbolic solvers (e.g WENO methods [11,12]) for convection terms to solve reaction–diffusion–convection equations efficiently [13–16].

The compact form of integration factor method was often very hard to be directly applied to deal with problems involving cross derivatives. Recently in [17], the compact integration factor (cIF) method was applied to solve a family of semilinear fourth-order parabolic equations, in which the bi-Laplace operator is explicitly handled. The proposed method can also deal with not only stiff nonlinear reaction terms, but also various types of homogeneous or inhomogeneous boundary conditions, while how to deal with inhomogeneous boundary conditions with cIF was not addressed before. Meanwhile, the IF method was designed and tested primarily for reaction–diffusion equations in previous studies. More recently in [18], cIF method was extended to solve the dissipative hydrodynamic equation system for incompressible fluid mixture flows with more complex mathematical structures. The IF strategy is applied after the system is discretized in space into a large differential and algebraic equation (DAE) system, which respects the total energy dissipation. The computational cost can be dramatically reduced through the use of discrete Fourier transform (DFT) by taking advantage of the circular structure of discretized matrices. The proposed approach has exhibited great numerical stability and energy dissipation property.

One challenge for integration factor (IF) method is to find the matrix of high order space discretization, especially near the boundaries. All previous studies have mainly focused on second order discretization in space. In this paper, we generalize IF methods for efficiently handling reaction–diffusion systems with high order accuracy for various inhomogeneous boundary conditions. In this approach, we use standard fourth order central finite differences for spatial discretization coupled with compact implicit integration factor methods for time discretization. In two and three dimensional systems, the discretized matrices arising from a compact representation of the diffusion operator need to be diagonalized once and pre-calculated before each time step iteration. This new approach has similar stability properties as the general second order cIIF along with a similar computational cost. Thus, the method is particularly suitable for high order partial differential equations in high dimensional systems with high order accuracy for both homogeneous and inhomogeneous boundary conditions.

To study the accuracy and efficiency, we first derive and implement the IF method to efficiently solve reaction–diffusion systems with inhomogeneous Neumann boundary conditions. Such approach can be similarly extended to all other inhomogeneous boundary conditions. The direct numerical simulations exhibit the excellent performance of the proposed approach through extensive numerical benchmark tests with linear and nonlinear equations. The rest of the paper is organized as follows. The generalized IF method for reaction–diffusion systems with inhomogeneous Neumann boundary conditions is derived in Section 2, and numerical tests with linear and nonlinear cases are shown in Section 3. Finally a brief conclusion is drawn.

2. High Order integration factor (IF) method with inhomogeneous Boundary Conditions

One-Dimension

First we consider a one-dimensional reaction–diffusion equation with inhomogeneous Neumann boundary condition,

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} = D\dfrac{\partial^2 \mathbf{u}}{\partial x^2} + \mathbf{F}(\mathbf{u}), & x \in \Omega, \quad t \in [0, T] \\ \dfrac{\partial \mathbf{u}}{\partial x}(t, x) = g(t, x) & x \in \partial\Omega, \quad t \in [0, T] \end{cases}, \tag{2}$$

where $\Omega = [a, b]$. We first discretize the spatial domain by the mesh: $x_i = a + i \times h$, where $h = (b - a)/N$ and $0 \le i \le N$. Using the fourth order central difference discretization on the diffusion, we obtain a system of nonlinear ODEs

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = D\left(\frac{-u_{i+2} + 16u_{i+1} - 30u_i + 16u_{i-1} - u_{i-2}}{12h^2}\right) + \mathbf{F}(u_i) + \mathbf{G}. \tag{3}$$

Next we define vectors $\mathbf{U}$ and $\mathcal{G}$ and a matrix $\mathbf{A}$ by

$$\mathbf{U} = (u_0 \quad \cdots \quad u_i \quad \cdots \quad u_N)^T_{(N+1)\times 1}, \tag{4}$$

$$\mathcal{G} = \frac{D}{h^2} \times \begin{pmatrix} -\dfrac{17g(t, x_0)}{6} \\ \dfrac{11g(t, x_0)}{48} \\ 0 \\ \vdots \\ 0 \\ -\dfrac{11g(t, x_N)}{48} \\ \dfrac{17g(t, x_N)}{6} \end{pmatrix}_{(N+1)\times 1}, \tag{5}$$

and

$$\mathbf{A}=\frac{D}{12h^2}\times\begin{pmatrix} -\frac{215}{6} & 32 & 12 & -\frac{32}{3} & \frac{5}{2} & & & & \\ \frac{803}{48} & -31 & \frac{57}{4} & \frac{1}{3} & -\frac{5}{16} & & & & \\ -1 & 16 & -30 & 16 & -1 & & & & \\ & -1 & 16 & -30 & 16 & -1 & & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & & -1 & 16 & -30 & 16 & -1 \\ & & & -\frac{5}{16} & \frac{1}{3} & \frac{57}{4} & -31 & \frac{803}{48} \\ & & & \frac{5}{2} & -\frac{32}{3} & 12 & 32 & -\frac{215}{6} \end{pmatrix}_{(N+1)\times(N+1)} \tag{6}$$

In terms of these vectors and matrix, the semi-discretized form (3) becomes

$$\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t}=\mathbf{AU}+\mathcal{F}(\mathbf{U})+\mathcal{G}. \tag{7}$$

To apply the integration factor technique to the compact discretization form (7), we multiply (7) by the exponential matrix $e^{-\mathbf{A}t}$ on the left to obtain

$$\frac{\mathrm{d}(e^{-\mathbf{A}t}\mathbf{U})}{\mathrm{d}t}=e^{-\mathbf{A}t}\mathcal{F}(\mathbf{U})+e^{-\mathbf{A}t}\mathcal{G}. \tag{8}$$

Integration of (8) over one time step from t_n to $t_{n+1}\equiv t_n+\Delta t$, where Δt is the time step, leads to

$$\mathbf{U}_{n+1}=e^{\mathbf{A}\Delta t}\mathbf{U}_n+e^{\mathbf{A}\Delta t}\left(\int_0^{\Delta t}e^{-\mathbf{A}\tau}\mathcal{F}(\mathbf{U}(t_n+\tau))\mathrm{d}\tau+\int_0^{\Delta t}e^{-\mathbf{A}\tau}\mathcal{G}(t_n+\tau)\mathrm{d}\tau\right). \tag{9}$$

As discussed in [17,19], to evaluate the integral resulting from the inhomogeneous boundary terms

$$\int_0^{\Delta t}e^{-\mathbf{A}\tau}\mathcal{G}(t_n+\tau)\mathrm{d}\tau,$$

we need to be careful since $\mathcal{G}(t_n+\tau)$ contains entries which decay with highly different speeds along the time, and it involves the factors of $1/h^2$ which could quickly amplify errors arising from the time discretization, thus causing severe numerical instability. To overcome this difficulty, we will apply an elegant approach proposed in [17,19], which is described with details in the following.

To construct a scheme of rth order truncation error, we approximate the integrands in (9),

$$\mathcal{H}_1(\tau)\equiv e^{-\mathbf{A}\tau}\mathcal{F}(\mathbf{U}(t_n+\tau)),\quad \mathcal{H}_2(\tau)\equiv\mathcal{G}(t_n+\tau),$$

using a $(r-1)$th order Lagrange polynomial at a set of interpolation points $t_{n+1},t_n,\ldots,t_{n+2-r}$:

$$\mathcal{P}_1(\tau)\equiv\sum_{j=-1}^{r-2}e^{j\mathbf{A}\Delta t}\mathcal{F}(\mathbf{U}_{n-j})p_j(\tau),\quad \mathcal{P}_2(\tau)\equiv\sum_{j=-1}^{r-2}\mathcal{G}(t_n-j\Delta t)p_j(\tau),\quad 0\le\tau\le\Delta t,$$

where

$$p_j(\tau)=\prod_{k=-1,k\neq j}^{r-2}\frac{\tau+k\Delta t}{(k-j)\Delta t}. \tag{10}$$

In terms of $\mathcal{P}_1(\tau)$ and $\mathcal{P}_2(\tau)$(9) takes the form,

$$\mathbf{U}_{n+1}=e^{\mathbf{A}\Delta t}\mathbf{U}_n+e^{\mathbf{A}\Delta t}\left(\int_0^{\Delta t}\mathcal{P}_1(\tau)\mathrm{d}\tau+\int_0^{\Delta t}e^{-\mathbf{A}\tau}\mathcal{P}_2(\tau)\mathrm{d}\tau\right). \tag{11}$$

So the new rth order implicit schemes are

$$\mathbf{U}_{n+1}=e^{\mathbf{A}\Delta t}\mathbf{U}_n+\Delta t\left(\alpha_1\mathcal{F}(\mathbf{U}_{n+1})+\sum_{j=0}^{r-2}\alpha_{-j}e^{(j+1)\mathbf{A}\Delta t}\mathcal{F}(\mathbf{U}_{n-j})+e^{\mathbf{A}\Delta t}\sum_{j=-1}^{r-2}\beta_{-j}\mathcal{G}(t_n-j\Delta t)\right), \tag{12}$$

Table 1
Values of α_{-j} in (13) up to order four.

r	α_1	α_0	α_{-1}	α_{-2}
1	1	0	0	0
2	$\frac{1}{2}$	$\frac{1}{2}$	0	0
3	$\frac{5}{12}$	$\frac{2}{3}$	$\frac{-1}{12}$	0
4	$\frac{9}{24}$	$\frac{19}{24}$	$\frac{-5}{24}$	$\frac{1}{24}$

Table 2
Values of β_{-j} in (13) up to order four where ξ_k are defined in (14).

r	β_1	β_0	β_{-1}	β_{-2}
1	ξ_0	0	0	0
2	ξ_1	$-\xi_1+\xi_0$	0	0
3	$\frac{1}{2}\xi_2+\frac{1}{2}\xi_1$	$-\xi_2+\xi_0$	$\frac{1}{2}\xi_2-\frac{1}{2}\xi_1$	0
4	$\frac{1}{6}\xi_3+\frac{1}{2}\xi_2+\frac{1}{3}\xi_1$	$-\frac{1}{2}\xi_3-\xi_2+\frac{1}{2}\xi_1+\xi_0$	$\frac{1}{2}\xi_3+\frac{1}{2}\xi_2-\xi_1$	$-\frac{1}{6}\xi_3+\frac{1}{6}\xi_1$

where $\alpha_1, \alpha_0, \alpha_{-1}, \ldots, \alpha_{-r+2}$ and $\beta_1, \beta_0, \beta_{-1}, \ldots, \beta_{-r+2}$ are coefficients calculated from the integrals of the polynomials $\mathcal{P}_1(\tau)$ and $\mathcal{P}_2(\tau)$, respectively,

$$\alpha_{-j} = \frac{1}{\Delta t}\int_0^{\Delta t} \prod_{k=-1,k\neq j}^{r-2} \frac{\tau + k\Delta t}{(k-j)\Delta t} d\tau, \quad \beta_{-j} = \frac{1}{\Delta t}\int_0^{\Delta t} e^{-\mathbf{A}\tau} \prod_{k=-1,k\neq j}^{r-2} \frac{\tau + k\Delta t}{(k-j)\Delta t} d\tau, \quad -1 \le j \le r-2. \tag{13}$$

In Table 1, the values of the coefficients α_{-j} for schemes of order up to four are listed.

Define the matrices

$$\begin{aligned} \xi_0 &= \mathbf{A}^{-1}\left(\frac{1}{\Delta t}\mathbf{I} - \frac{1}{\Delta t}e^{-\mathbf{A}\Delta t}\right), \\ \xi_k &= \mathbf{A}^{-1}\left(\frac{k}{\Delta t^{k+1}}\xi_{k-1} - \frac{1}{\Delta t}e^{-\mathbf{A}\Delta t}\right), \quad k \ge 1. \end{aligned} \tag{14}$$

Then the coefficients β_{-j} for schemes of order up to four are listed in Table 2.

Remark 1. Even though the integration factor method are derived in the context of inhomogeneous Neumann boundary conditions, it can be similarly extended to inhomogeneous Dirichlet boundary conditions.

Two-Dimensions

Now we consider a two-dimensional reaction–diffusion equation with inhomogeneous Neumann boundary conditions:

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} = D(\dfrac{\partial^2 \mathbf{u}}{\partial x^2} + \dfrac{\partial^2 \mathbf{u}}{\partial y^2}) + \mathbf{F}(\mathbf{u}), & (x,y) \in \Omega, \quad t \in [0,T] \\ \dfrac{\partial \mathbf{u}}{\partial x}(t,x,y) = g_1(t,x,y) & (x,y) \in \partial\Omega, \quad t \in [0,T] \\ \dfrac{\partial \mathbf{u}}{\partial y}(t,x,y) = g_2(t,x,y) & (x,y) \in \partial\Omega, \quad t \in [0,T] \end{cases}, \tag{15}$$

where $\Omega = [a,b] \times [c,d]$. We first discretize the spatial domain by the mesh: $(x_i, y_j) = (a + i \times h_x, c + j \times h_y)$, where $h_x = (b-a)/N_x$, $h_y = (d-c)/N_y$, and $0 \le i \le N_x$ and $0 \le j \le N_j$. Using the fourth order central difference discretization on the diffusion, we obtain a system of nonlinear ODEs

$$\begin{aligned} \frac{\mathrm{d}u_{i,j}}{\mathrm{d}t} = D\Bigg(&\frac{-u_{i+2,j} + 16u_{i+1,j} - 30u_{i,j} + 16u_{i-1,j} - u_{i-2,j}}{12h_x^2} \\ &+ \frac{-u_{i,j+2} + 16u_{i,j+1} - 30u_{i,j} + 16u_{i,j-1} - u_{i,j-2}}{12h_y^2}\Bigg) + \mathbf{F}(u_{i,j}) + \mathbf{G}_1 + \mathbf{G}_2. \end{aligned} \tag{16}$$

Next we define matrices $\mathbf{U}$, $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathbf{A}$, and $\mathbf{B}$ by

$$\mathbf{U} = \begin{pmatrix} u_{0,0} & u_{0,1} & \cdots & u_{0,N_y} \\ u_{1,0} & u_{1,1} & \cdots & u_{1,N_y} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N_x,0} & u_{N_x,1} & \cdots & u_{N_x,N_y} \end{pmatrix}_{(N_x+1)\times(N_y+1)}, \tag{17}$$

$$\mathcal{G}_1=\frac{D}{h}\times\begin{pmatrix}-\frac{17g_1(t,x_0,y_0)}{6} & -\frac{17g_1(t,x_0,y_1)}{6} & \cdots & -\frac{17g_1(t,x_0,y_N)}{6}\\ \frac{11g_1(t,x_0,y_0)}{48} & \frac{11g_1(t,x_0,y_1)}{48} & \cdots & \frac{11g_1(t,x_0,y_N)}{48}\\ 0 & 0 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & 0\\ -\frac{11g_1(t,x_N,y_0)}{48} & -\frac{11g_1(t,x_N,y_1)}{48} & \cdots & -\frac{11g_1(t,x_N,y_N)}{48}\\ \frac{17g_1(t,x_N,y_0)}{6} & \frac{17g_1(t,x_N,y_1)}{6} & \cdots & \frac{17g_1(t,x_N,y_N)}{6}\end{pmatrix}_{(N_x+1)\times(N_y+1)}, \tag{18}$$

$$\mathcal{G}_2=\frac{D}{h}\times\begin{pmatrix}-\frac{17g_1(t,x_0,y_0)}{6} & \frac{11g_1(t,x_0,y_0)}{48} & 0 & \cdots & 0 & -\frac{11g_1(t,x_0,y_N)}{48} & \frac{17g_1(t,x_0,y_N)}{6}\\ -\frac{17g_1(t,x_1,y_0)}{6} & \frac{11g_1(t,x_1,y_0)}{48} & 0 & \cdots & 0 & -\frac{11g_1(t,x_1,y_N)}{48} & \frac{17g_1(t,x_1,y_N)}{6}\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots\\ -\frac{17g_1(t,x_N,y_0)}{6} & \frac{11g_1(t,x_N,y_0)}{48} & 0 & \cdots & 0 & -\frac{11g_1(t,x_N,y_N)}{48} & \frac{17g_1(t,x_N,y_0)}{6}\end{pmatrix}_{(N_x+1)\times(N_y+1)}, \tag{19}$$

$$\mathbf{A}=\frac{D}{12h_x^2}\times\begin{pmatrix}-\frac{215}{6} & 32 & 12 & -\frac{32}{3} & \frac{5}{2} & & & & & \\ \frac{803}{48} & -31 & \frac{57}{4} & \frac{1}{3} & -\frac{5}{16} & & & & & \\ -1 & 16 & -30 & 16 & -1 & & & & & \\ & -1 & 16 & -30 & 16 & -1 & & & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ & & & -1 & 16 & -30 & 16 & -1 & & \\ & & & -\frac{5}{16} & \frac{1}{3} & \frac{57}{4} & -31 & \frac{803}{48} & & \\ & & & \frac{5}{2} & -\frac{32}{3} & 12 & 32 & -\frac{215}{6} & & \end{pmatrix}_{(N_x+1)\times(N_x+1)}, \tag{20}$$

and

$$\mathbf{B}=\frac{D}{12h_y^2}\times\begin{pmatrix}-\frac{215}{6} & \frac{803}{48} & -1 & & & & & \\ 32 & -31 & 16 & -1 & & & & \\ 12 & \frac{57}{4} & -30 & 16 & \ddots & & & \\ -\frac{32}{3} & \frac{1}{3} & 16 & -30 & \ddots & -1 & -\frac{5}{16} & \frac{5}{2}\\ \frac{5}{2} & -\frac{5}{16} & -1 & 16 & \ddots & 16 & \frac{1}{3} & -\frac{32}{3}\\ & & & -1 & \ddots & -30 & \frac{57}{4} & 12\\ & & & & \ddots & 16 & -31 & 32\\ & & & & & -1 & \frac{803}{48} & -\frac{215}{6}\end{pmatrix}_{(N_y+1)\times(N_y+1)}. \tag{21}$$

In terms of these matrices, the semi-discretized form (17) becomes

$$\frac{d\mathbf{U}}{dt}=\mathbf{AU}+\mathbf{UB}+\mathcal{F}(\mathbf{U})+\mathcal{G}_1+\mathcal{G}_2. \tag{22}$$

Since **A** and **B** can be diagonalized,

$$\mathbf{A}=\mathbf{P_A D_A P_A^{-1}} \qquad \mathbf{B}=\mathbf{P_B D_B P_B^{-1}}.$$

Multiplying (22) on the left by $\mathbf{P}_\mathbf{A}^{-1}$ and on the right by $\mathbf{P}_\mathbf{B}$ yields

$$\frac{\mathrm{d}(\mathbf{P}_\mathbf{A}^{-1}\mathbf{U}\mathbf{P}_\mathbf{B})}{\mathrm{d}t} = \mathbf{D}_\mathbf{A}\mathbf{P}_\mathbf{A}^{-1}\mathbf{U}\mathbf{P}_\mathbf{B} + \mathbf{P}_\mathbf{A}^{-1}\mathbf{U}\mathbf{P}_\mathbf{B}\mathbf{D}_\mathbf{B} + \mathbf{P}_\mathbf{A}^{-1}\mathcal{F}(\mathbf{U})\mathbf{P}_\mathbf{B} + \mathbf{P}_\mathbf{A}^{-1}\mathcal{G}_1\mathbf{P}_\mathbf{B} + \mathbf{P}_\mathbf{A}^{-1}\mathcal{G}_2\mathbf{P}_\mathbf{B}. \tag{23}$$

Let

$$\mathbf{V} = \mathbf{P}_\mathbf{A}^{-1}\mathbf{U}\mathbf{P}_\mathbf{B}, \qquad \tilde{\mathcal{F}} = \mathbf{P}_\mathbf{A}^{-1}\mathcal{F}\mathbf{P}_\mathbf{B}, \qquad \tilde{\mathcal{G}}_1 = \mathbf{P}_\mathbf{A}^{-1}\mathcal{G}_1\mathbf{P}_\mathbf{B}, \qquad \tilde{\mathcal{G}}_2 = \mathbf{P}_\mathbf{A}^{-1}\mathcal{G}_2\mathbf{P}_\mathbf{B}.$$

Then (23) becomes

$$\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} = \mathbf{D}_\mathbf{A}\mathbf{V} + \mathbf{V}\mathbf{D}_\mathbf{B} + \tilde{\mathcal{F}}(\mathbf{P}_\mathbf{A}\mathbf{V}\mathbf{P}_\mathbf{B}^{-1}) + \tilde{\mathcal{G}}_1 + \tilde{\mathcal{G}}_2. \tag{24}$$

To apply the integration factor technique to the compact discretization form (24), we multiply (24) by the exponential matrix $e^{-\mathbf{D}_\mathbf{A}t}$ on the left, and $e^{-\mathbf{D}_\mathbf{B}t}$ on the right and integrate over one time step from t_n to $t_{n+1} \equiv t_n + \Delta t$, where Δt is the time step. This leads to

$$\begin{aligned}\mathbf{V}_{n+1} = e^{\mathbf{D}_\mathbf{A}\Delta t}\mathbf{V}_n e^{\mathbf{D}_\mathbf{B}\Delta t} + e^{\mathbf{D}_\mathbf{A}\Delta t}\Bigg(\int_0^{\Delta t} e^{-\mathbf{D}_\mathbf{A}\tau}\tilde{\mathcal{F}}(\mathbf{P}_\mathbf{A}\mathbf{V}(t_n+\tau)\mathbf{P}_\mathbf{B}^{-1})e^{-\mathbf{D}_\mathbf{B}\tau}\mathrm{d}\tau \\ + \int_0^{\Delta t} e^{-\mathbf{D}_\mathbf{A}\tau}\tilde{\mathcal{G}}_1(t_+\tau)e^{-\mathbf{D}_\mathbf{B}\tau}\mathrm{d}\tau + \int_0^{\Delta t} e^{-\mathbf{D}_\mathbf{A}\tau}\tilde{\mathcal{G}}_2(t_n+\tau)e^{-\mathbf{D}_\mathbf{B}\tau}\mathrm{d}\tau\Bigg)e^{\mathbf{D}_\mathbf{B}\Delta t}.\end{aligned} \tag{25}$$

Similar to the one-dimensional case, to construct a scheme of rth order truncation error, we approximate the integrands in (25) using a $(r-1)$th order Lagrange polynomial at a set of interpolation points $t_{n+1}, t_n, \ldots, t_{n+2-r}$:

$$\begin{aligned}&\mathcal{P}_1(\tau) \equiv \sum_{j=-1}^{r-2} e^{j\mathbf{D}_\mathbf{A}\Delta t}\tilde{\mathcal{F}}(\mathbf{P}_\mathbf{A}\mathbf{V}_{n-j}\mathbf{P}_\mathbf{B}^{-1})e^{j\mathbf{D}_\mathbf{D}\Delta t}p_j(\tau), \\ &\mathcal{P}_2(\tau) \equiv \sum_{j=-1}^{r-2}\tilde{\mathcal{G}}_1(t_n - j\Delta t)p_j(\tau), \quad \mathcal{P}_3(\tau) \equiv \sum_{j=-1}^{r-2}\tilde{\mathcal{G}}_2(t_n - j\Delta t)p_j(\tau), \quad 0 \le \tau \le \Delta t,\end{aligned} \tag{26}$$

where

$$p_j(\tau) = \prod_{k=-1,k\neq j}^{r-2}\frac{\tau + k\Delta t}{(k-j)\Delta t}. \tag{27}$$

In terms of $\mathcal{P}_i(\tau)$, $i = 1, 2, 3$, (25) takes the form,

$$\begin{aligned}\mathbf{V}_{n+1} = e^{\mathbf{D}_\mathbf{A}\Delta t}\mathbf{V}_n e^{\mathbf{D}_\mathbf{B}\Delta t} + e^{\mathbf{D}_\mathbf{A}\Delta t}\Bigg(\int_0^{\Delta t}\mathcal{P}_1(\tau)\mathrm{d}\tau \\ + \int_0^{\Delta t} e^{-\mathbf{D}_\mathbf{A}\tau}\mathcal{P}_2(\tau)e^{-\mathbf{D}_\mathbf{B}\tau}\mathrm{d}\tau + \int_0^{\Delta t} e^{-\mathbf{D}_\mathbf{A}\tau}\mathcal{P}_3(\tau)e^{-\mathbf{D}_\mathbf{B}\tau}\mathrm{d}\tau\Bigg)e^{\mathbf{D}_\mathbf{B}\Delta t}.\end{aligned} \tag{28}$$

Let $\mathbf{D}_\mathbf{A} = \mathrm{diag}(d_0^a, d_1^a, \ldots, d_{N_x}^a)$ and $\mathbf{D}_\mathbf{B} = \mathrm{diag}(d_0^b, d_1^b, \ldots, d_{N_y}^b)$. Note that multiplication of diagonal matrices on the left and right becomes component-wise matrix multiplication of the form

$$(\mathbf{D}_\mathbf{A}\mathcal{G}\mathbf{D}_\mathbf{B})_{i,j} = d_i^a(\mathcal{G})_{i,j}d_j^b. \tag{29}$$

So the second integration in (28) can be done component-wise. Now the new rth order implicit schemes are

$$\begin{aligned}\mathbf{V}_{n+1} = e^{\mathbf{D}_\mathbf{A}\Delta t}\mathbf{V}_n e^{\mathbf{D}_\mathbf{B}\Delta t} + \Delta t\left(\alpha_1\tilde{\mathcal{F}}(\mathbf{P}_\mathbf{A}\mathbf{V}_{n+1}\mathbf{P}_\mathbf{B}^{-1}) + \sum_{j=0}^{r-2}\alpha_{-j}e^{(j+1)\mathbf{D}_\mathbf{A}\Delta t}\tilde{\mathcal{F}}(\mathbf{P}_\mathbf{A}\mathbf{V}_{n-j}\mathbf{P}_\mathbf{B}^{-1})e^{(j+1)\mathbf{D}_\mathbf{B}\Delta t}\right) \\ + \Delta t e^{\mathbf{D}_\mathbf{A}\Delta t}\left(\sum_{j=-1}^{r-2}\beta_{-j}\circ\tilde{\mathcal{G}}_1(t_n - j\Delta t) + \sum_{j=-1}^{r-2}\beta_{-j}\circ\tilde{\mathcal{G}}_2(t_n - j\Delta t)\right)e^{\mathbf{D}_\mathbf{B}\Delta t},\end{aligned} \tag{30}$$

where "$\circ$" denotes component-wise matrix multiplication, and $\alpha_1, \alpha_0, \alpha_{-1}, \ldots, \alpha_{-r+2}$ and $\beta_1, \beta_0, \beta_{-1}, \ldots, \beta_{-r+2}$ are coefficients calculated from the integrals of the polynomials,

$$\begin{aligned}&\alpha_{-j} = \frac{1}{\Delta t}\int_0^{\Delta t}\prod_{k=-1,k\neq j}^{r-2}\frac{\tau + k\Delta t}{(k-j)\Delta t}d\tau \\ &\beta_{-j} = \frac{1}{\Delta t}\int_0^{\Delta t} e^{-(\mathbf{D}_\mathbf{A}+\mathbf{D}_\mathbf{B})\tau}\prod_{k=-1,k\neq j}^{r-2}\frac{\tau + k\Delta t}{(k-j)\Delta t}d\tau, \qquad -1 \le j \le r-2.\end{aligned} \tag{31}$$

In Table 1, the values of the coefficients α_{-j} for schemes of order up to four are listed. Define the matrices

$$(\xi_0)_{i,j} = \frac{1}{d_i^a + d_j^b}\left(\frac{1}{\Delta t} - \frac{1}{\Delta t}e^{-(d_i^a+d_j^b)\Delta t}\right)$$
$$(\xi_k)_{i,j} = \frac{1}{d_i^a + d_j^b}\left(\frac{k}{\Delta t^{k+1}}(\xi_{k-1})_{i,j} - \frac{1}{\Delta t}e^{-(d_i^a+d_j^b)\Delta t}\right) \quad k \geq 1. \tag{32}$$

Then the coefficients β_{-j} for schemes of order up to four are listed in Table 2.

From here the solution of $\mathbf{U}$ can be recovered by $\mathbf{U} = \mathbf{P_A}\mathbf{V}\mathbf{P_B}^{-1}$.

Remark 2. For the cases when A or B cannot be diagonalized, the terms from inhomogeneous boundary condition can be incorporated into the nonlinear term $\mathcal{F}$. For instance, Eq. (22) can be written as

$$\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t} = \mathbf{AU} + \mathbf{UB} + \bar{\mathcal{F}}(\mathbf{U}). \tag{33}$$

where $\bar{\mathcal{F}}(\mathbf{U}) = \mathcal{F}(\mathbf{U}) + \mathcal{G}_1 + \mathcal{G}_2$. We can follow the same ideas for compact implicit integration factor (cIIF) method as discussed in [8]. For instance, the second order cIIF2 method is given by

$$\mathbf{U}_{n+1} = e^{\mathbf{A}\Delta t}\mathbf{U}_n e^{\mathbf{B}\Delta t} + \frac{\Delta t}{2}\left(\bar{\mathcal{F}}(\mathbf{U}_n) + \bar{\mathcal{F}}(\mathbf{U}_{n+1})\right).$$

Three-Dimensions

Now we consider a three-dimensional reaction–diffusion equation with Neumann boundary conditions:

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} = D\Delta u + \mathbf{F}(\mathbf{u}), & (x,y,z) \in \Omega, \quad t \in [0,T] \\ \dfrac{\partial \mathbf{u}}{\partial x}(t,x,y,z) = g_1(t,x,y,z) & (x,y,z) \in \partial\Omega, \quad t \in [0,T] \\ \dfrac{\partial \mathbf{u}}{\partial y}(t,x,y,z) = g_2(t,x,y,z) & (x,y,z) \in \partial\Omega, \quad t \in [0,T] \\ \dfrac{\partial \mathbf{u}}{\partial z}(t,x,y,z) = g_3(t,x,y,z) & (x,y,z) \in \partial\Omega, \quad t \in [0,T] \end{cases} \tag{34}$$

where $\Omega = [a_l, a_u] \times [b_l, b_u] \times [c_l, c_u]$. Let N_x, N_y, N_z denote the number of spatial grid points in x, y, z-direction, respectively, h_x, h_y, h_z be the grid size, and $u_{i,j,k}$ represent the approximate solution at the grid point (x_i, y_j, z_k), $0 \leq i \leq N_x$, $0 \leq j \leq N_y$, and $0 \leq k \leq N_z$. A fourth order central difference discretization on the Laplacian operator yields

$$\begin{aligned} \frac{\mathrm{d}u_{i,j,k}}{\mathrm{d}t} = D\Bigg(& \frac{-u_{i+2,j,k} + 16u_{i+1,j,k} - 30u_{i,j,k} + 16u_{i-1,j,k} - u_{i-2,j,k}}{12h_x^2} \\ & + \frac{-u_{i,j+2,k} + 16u_{i,j+1,k} - 30u_{i,j,k} + 16u_{i,j-1,k} - u_{i,j-2,k}}{12h_y^2} \\ & + \frac{-u_{i,j,k+2} + 16u_{i,j,k+1} - 30u_{i,j,k} + 16u_{i,j,k-1} - u_{i,j,k-2}}{12h_z^2}\Bigg) \\ & + \mathbf{F}(u_{i,j,k}) + \mathbf{G}_{i,j,k} \end{aligned} \tag{35}$$

Define $A_x = \frac{D}{h_x^2}A_{(N_x+1)\times(N_x+1)}$, $A_y = \frac{D}{h_y^2}A_{(N_y+1)\times(N_y+1)}$, and $A_z = \frac{D}{h_z^2}A_{(N_z+1)\times(N_z+1)}$, where

$$A_{P\times P} = \begin{pmatrix} -\frac{215}{6} & 32 & 12 & -\frac{32}{3} & \frac{5}{2} & & & & & \\ \frac{803}{48} & -31 & \frac{57}{4} & \frac{1}{3} & -\frac{5}{16} & & & & & \\ -1 & 16 & -30 & 16 & -1 & & & & & \\ & -1 & 16 & -30 & 16 & -1 & & & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ & & & -1 & 16 & -30 & 16 & -1 & & \\ & & & -\frac{5}{16} & \frac{1}{3} & \frac{57}{4} & -31 & \frac{803}{48} & & \\ & & & \frac{5}{2} & -\frac{32}{3} & 12 & 32 & -\frac{215}{6} & & \end{pmatrix}_{P\times P}.$$

Then (35) has the following compact representation

$$\mathbf{U}_t = \left(\sum_{l=0}^{N_x} (A_x)_{i,l} u_{l,j,k} + \sum_{l=0}^{N_y} (A_y)_{j,l} u_{i,l,k} + \sum_{l=0}^{N_z} (A_z)_{k,l} u_{i,j,l} \right) + \mathcal{F}(\mathbf{U}) + \mathcal{G}, \tag{36}$$

where $\mathbf{U} = (u_{i,j,k})$, $\mathcal{F}(\mathbf{U}) = (\mathcal{F}(u_{i,j,k}))$, and $\mathcal{G} = (\mathcal{G}_{i,j,k})$. $\mathcal{G}$ is defined as

$$\mathcal{G}_{0,j,k} = \frac{D}{h} \times \begin{pmatrix} -\frac{17g_1(t,x_0,y_0,z_0)}{6} & \cdots & -\frac{17g_1(t,x_0,y_0,z_{N_z})}{6} \\ \vdots & \ddots & \vdots \\ -\frac{17g_1(t,x_0,y_{N_y},z_0)}{6} & \cdots & -\frac{17g_1(t,x_0,y_{N_y},z_{N_z})}{6} \end{pmatrix}_{(N_y+1)\times(N_z+1)},$$

$$\mathcal{G}_{1,j,k} = \frac{D}{h} \times \begin{pmatrix} \frac{11g_1(t,x_0,y_0,z_0)}{48} & \cdots & \frac{11g_1(t,x_0,y_0,z_{N_z})}{48} \\ \vdots & \ddots & \vdots \\ \frac{11g_1(t,x_0,y_{N_y},z_0)}{48} & \cdots & \frac{11g_1(t,x_0,y_{N_y},z_{N_z})}{48} \end{pmatrix}_{(N_y+1)\times(N_z+1)},$$

$$\mathcal{G}_{Nx-1,j,k} = \frac{D}{h} \times \begin{pmatrix} -\frac{11g_1(t,x_N,y_0,z_0)}{48} & \cdots & -\frac{11g_1(t,x_N,y_0,z_{N_z})}{48} \\ \vdots & \ddots & \vdots \\ -\frac{11g_1(t,x_N,y_{N_y},z_0)}{48} & \cdots & -\frac{11g_1(t,x_N,y_{N_y},z_{N_z})}{48} \end{pmatrix}_{(N_y+1)\times(N_z+1)},$$

$$\mathcal{G}_{N_x,j,k} = \frac{D}{h} \times \begin{pmatrix} \frac{17g_1(t,x_N,y_0,z_0)}{6} & \cdots & \frac{17g_1(t,x_N,y_0,z_{N_z})}{6} \\ \vdots & \ddots & \vdots \\ \frac{17g_1(t,x_N,y_{N_y},z_0)}{6} & \cdots & \frac{17g_1(t,x_N,y_{N_y},z_{N_z})}{6} \end{pmatrix}_{(N_y+1)\times(N_z+1)}.$$

$\mathcal{G}_{i,0,k}$, $\mathcal{G}_{i,1,k}$, $\mathcal{G}_{i,N_y-1,k}$, $\mathcal{G}_{i,N_y,k}$, and $\mathcal{G}_{i,j,0}$, $\mathcal{G}_{i,j,1}$, $\mathcal{G}_{i,j,N_z-1}$, $\mathcal{G}_{i,j,N_z}$ are similarly defined. For $i \neq 0, 1, N_x - 1, N_x, j \neq 0, 1, N_y - 1, N_y$, and $k \neq 0, 1, N_z - 1, N_z$, $\mathcal{G}_{i,j,k} = 0$.

The three summation terms in (36) are similar to the two vector-matrix multiplications in the two-dimensional case in (22). In addition to a left multiplication and a right multiplication in (22), there is a "middle" multiplication in (36).

Since A_γ can be diagonalized,

$$A_\gamma = P_\gamma D_\gamma P_\gamma^{-1}, \quad \gamma = x, y, z. \tag{37}$$

Define an operator $\mathcal{D}$ by

$$(\mathcal{D}\mathbf{U})_{i,j,k} = \sum_{f=0}^{N_z} \sum_{e=0}^{N_y} \sum_{d=0}^{N_x} (P_z^{-1})_{k,f} (P_y^{-1})_{j,e} (P_x^{-1})_{i,d} u_{d,e,f}. \tag{38}$$

Applying $\mathcal{D}$ to the first term of (36) yields

$$\begin{aligned} (\mathcal{D}\mathbf{U}_t)^{\text{1st term}} &= \sum_{f=0}^{N_z} \sum_{e=0}^{N_y} \sum_{d=0}^{N_x} (P_z^{-1})_{k,f} (P_y^{-1})_{j,e} (P_x^{-1})_{i,d} \sum_{l=0}^{N_x} (P_x D_x P_x^{-1})_{d,l} u_{l,e,f} \\ &= \sum_{f=0}^{N_z} \sum_{e=0}^{N_y} \sum_{l=0}^{N_x} (P_z^{-1})_{k,f} (P_y^{-1})_{j,e} \sum_{d=0}^{N_x} (P_x^{-1})_{i,d} (P_x D_x P_x^{-1})_{d,l} u_{l,e,f} \\ &= \sum_{f=0}^{N_z} \sum_{e=0}^{N_y} \sum_{l=0}^{N_x} (P_z^{-1})_{k,f} (P_y^{-1})_{j,e} (D_x P_x^{-1})_{i,l} u_{l,e,f} \\ &= \sum_{f=0}^{N_z} \sum_{e=0}^{N_y} \sum_{l=0}^{N_x} (P_z^{-1})_{k,f} (P_y^{-1})_{j,e} \sum_{a=0}^{N_x} (D_x)_{i,a} (P_x^{-1})_{a,l} u_{l,e,f} \end{aligned}$$

$$= \sum_{a=0}^{N_x}(D_x)_{i,a}\sum_{f=0}^{N_z}\sum_{e=0}^{N_y}\sum_{l=0}^{N_x}(P_z^{-1})_{k,f}(P_y^{-1})_{j,e}(P_x^{-1})_{a,l}u_{l,e,f}$$
$$= \sum_{a=0}^{N_x}(D_x)_{i,a}(\mathcal{D}\mathbf{U})_{a,j,k}. \tag{39}$$

Make the following substitutions,

$$\mathbf{V} = \mathcal{D}\mathbf{U}, \qquad \tilde{\mathcal{F}} = \mathcal{D}\mathcal{F}, \qquad \tilde{\mathcal{G}} = \mathcal{D}\mathcal{G}.$$

Then applying $\mathcal{D}$ to (36) yields

$$\mathbf{V}_t = \left(\sum_{l=0}^{N_x}(D_x)_{i,l}v_{l,j,k} + \sum_{l=0}^{N_y}(D_y)_{j,l}v_{i,l,k} + \sum_{l=0}^{N_z}(D_z)_{k,l}v_{i,j,l}\right) + \tilde{\mathcal{F}}(\mathcal{D}^{-1}\mathbf{V}) + \tilde{\mathcal{G}} \tag{40}$$

Define an operator $\mathcal{L}(t)$ by

$$(\mathcal{L}(t)\mathbf{U})_{i,j,k} = \sum_{n=0}^{N_z}\sum_{m=0}^{N_y}\sum_{l=0}^{N_x}(e^{-D_z t})_{k,n}(e^{-D_y t})_{j,m}(e^{-D_x t})_{i,l}u_{l,m,n}. \tag{41}$$

Taking derivatives of (41) yields

$$\frac{\mathrm{d}\mathcal{L}(t)\mathbf{U}}{\mathrm{d}t} = \mathcal{L}(t)\left(\mathbf{U}_t - \left(\sum_{l=0}^{N_x}(D_x)_{i,l}u_{l,j,k} + \sum_{l=0}^{N_y}(D_y)_{j,l}u_{i,l,k} + \sum_{l=0}^{N_z}(D_z)_{k,l}u_{i,j,l}\right)\right). \tag{42}$$

Letting $\mathcal{L}(t)$ act on both sides of (40) and using (42), we obtain

$$\frac{\mathrm{d}\mathcal{L}(t)\mathbf{V}}{\mathrm{d}t} = \mathcal{L}(t)\tilde{\mathcal{F}}(\mathcal{D}^{-1}\mathbf{V}) + \mathcal{L}(t)\tilde{\mathcal{G}}. \tag{43}$$

Integrating (43) over one time step from t_n to t_{n+1} and using a transformation $s = t_n + \tau$ for the integration, we obtain

$$\mathcal{L}(t_{n+1})\mathbf{V}_{n+1} = \mathcal{L}(t_n)\mathbf{V}_n + \mathcal{L}(t_n)\int_0^{\Delta t}\mathcal{L}(\tau)\tilde{\mathcal{F}}(\mathcal{D}^{-1}\mathbf{V}(t_n+\tau))\mathrm{d}\tau + \mathcal{L}(t_n)\int_0^{\Delta t}\mathcal{L}(\tau)\tilde{\mathcal{G}}(t_n+\tau)\mathrm{d}\tau. \tag{44}$$

Applying $\mathcal{L}(-t_{n+1})$ on both sides of (44) yields

$$\mathbf{V}_{n+1} = \mathcal{L}(-\Delta t)\mathbf{V}_n + \mathcal{L}(-\Delta t)\int_0^{\Delta t}\mathcal{L}(\tau)\tilde{\mathcal{F}}(\mathcal{D}^{-1}\mathbf{V}(t_n+\tau))\mathrm{d}\tau + \mathcal{L}(-\Delta t)\int_0^{\Delta t}\mathcal{L}(\tau)\tilde{\mathcal{G}}(t_n+\tau)(\tau)\mathrm{d}\tau. \tag{45}$$

To derive (45), we have used two identities:

$$\mathcal{L}(0)\mathbf{V} = \mathbf{V} \quad \text{and} \quad \mathcal{L}(-rt)\mathcal{L}(st)\mathbf{V} = \mathcal{L}((s-r)t)\mathbf{V} \tag{46}$$

for any two scalars r and s. Both of these can be easily proved based on the definition of $\mathcal{L}$.

Similar to one and two dimensional cases, to construct a scheme of rth order truncation error, we approximate the integrands in (45) using a $(r-1)$th order Lagrange polynomial at a set of interpolation points $t_{n+1}, t_n, \ldots, t_{n+2-r}$:

$$\mathcal{P}_1(\tau) \equiv \sum_{j=-1}^{r-2}\mathcal{L}(-j\Delta t)\tilde{\mathcal{F}}(\mathcal{D}^{-1}\mathbf{V}_{n-j})p_j(\tau), \quad \mathcal{P}_2(\tau) \equiv \sum_{j=-1}^{r-2}\tilde{\mathcal{G}}(t_n - j\Delta t)p_j(\tau), \quad 0 \le \tau \le \Delta t, \tag{47}$$

where

$$p_j(\tau) = \prod_{k=-1,k\neq j}^{r-2}\frac{\tau + k\Delta t}{(k-j)\Delta t}. \tag{48}$$

In terms of $\mathcal{P}_i(\tau)$, $i = 1,2$, (45) takes the form,

$$\mathbf{V}_{n+1} = \mathcal{L}(-\Delta t)\mathbf{V}_n + \mathcal{L}(-\Delta t)\left(\int_0^{\Delta t}\mathcal{P}_1(\tau)\mathrm{d}\tau + \int_0^{\Delta t}\mathcal{L}(\tau)\mathcal{P}_2(\tau)\mathrm{d}\tau\right). \tag{49}$$

So the new rth order implicit schemes are

$$\mathbf{V}_{n+1} = \mathcal{L}(-\Delta t)\mathbf{V}_n + \Delta t \left(\alpha_1 \tilde{\mathcal{F}}(\mathcal{D}^{-1}\mathbf{V}_{n+1}) + \sum_{j=0}^{r-2} \alpha_{-j} \mathcal{L}(-(j+1)\Delta t)\tilde{\mathcal{F}}(\mathcal{D}^{-1}\mathbf{V}_{n-j}) \right.$$
$$\left. + \ \mathcal{L}(-\Delta t) \sum_{j=-1}^{r-2} \beta_{-j} \circ \tilde{\mathcal{G}}(t_n - j\Delta t) \right), \tag{50}$$

where $\circ$ denotes component-wise multiplication of the three-dimensional matrices, and $\alpha_1, \alpha_0, \alpha_{-1}, \ldots, \alpha_{-r+2}$ and $\beta_1, \beta_0, \beta_{-1}, \ldots, \beta_{-r+2}$ are coefficients calculated from the integrals of the polynomials,

$$\alpha_{-j} = \frac{1}{\Delta t} \int_0^{\Delta t} \prod_{k=-1, k\neq j}^{r-2} \frac{\tau + k\Delta t}{(k-j)\Delta t} d\tau$$
$$\beta_{-j} = \frac{1}{\Delta t} \int_0^{\Delta t} \mathcal{L}(\tau) \prod_{k=-1, k\neq j}^{r-2} \frac{\tau + k\Delta t}{(k-j)\Delta t} d\tau, \qquad -1 \le j \le r-2. \tag{51}$$

In Table 1, the values of the coefficients α_{-j} for schemes of order up to four are listed. Define the matrices

$$(\xi_0)_{i,j,k} = \frac{1}{d_i^x + d_j^y + d_k^z} \left(\frac{1}{\Delta t} - \frac{1}{\Delta t} e^{-(d_i^x + d_j^y + d_k^z)\Delta t} \right)$$
$$(\xi_\gamma)_{i,j,k} = \frac{1}{d_i^x + d_j^y + d_k^z} \left(\frac{\gamma}{\Delta t^{\gamma+1}} (\xi_{\gamma-1})_{i,j,k} - \frac{1}{\Delta t} e^{-(d_i^x + d_j^y + d_k^z)\Delta t} \right), \quad \gamma \ge 1. \tag{52}$$

Then the coefficients β_{-j} for schemes of order up to four are listed in Table 2.

From here $\mathbf{U}$ can be recovered by $\mathbf{U} = \mathcal{D}^{-1}\mathbf{V}$, where

$$(\mathcal{D}^{-1}\mathbf{U})_{i,j,k} = \sum_{f=0}^{N_z} \sum_{e=0}^{N_y} \sum_{d=0}^{N_x} (P_z)_{k,f} (P_y)_{j,e} (P_x)_{i,d} u_{d,e,f}. \tag{53}$$

Remark 3. The scheme (50) has a form similar to the one- and two-dimensional case. The evaluation of the nonlinear term $\mathcal{F}$ at t_{n+1} is still local and decoupled from the global diffusion term such that a nonlinear system of the size $\mathcal{F}$ needs to be solved at each spatial grid point. Such approach can also be similarly extended to systems with any high spatial dimensions.

3. Numerical examples

To study the efficacy and accuracy of the fourth order compact implicit integration factor (cIIF) method, we will implement it on two- and three-dimensional systems. We test it on examples with either homogeneous or inhomogeneous boundary conditions for both linear and nonlinear systems. In the calculation, the exponential of the square matrix is computed using "expm" of MATLAB, which uses a scaling and squaring algorithm with a Pade approximation.

Because the matrix exponentials depend only on the spatial grid size, the time step, and diffusion coefficient, during the entire temporal updating, they only need to be calculated once initially for a fixed numerical resolution. The local nonlinear systems resulting from cIIF are solved iteratively using Newton's method.

In all examples, the cIIF scheme is implemented with fourth order in space and second order in time. It is implemented in MATLAB up to $T = 1$ at which the L^∞ difference between the numerical solution and the exact solution is measured. For the cases when the exact solution is not given, we take the numerical solution with relatively fine mesh as the "exact" solution. We set $h_x = h_y$ for the two-dimensional examples and $h_x = h_y = h_z$ for the three-dimensional examples. The inhomogeneous boundary condition algorithm has a higher requirement on the space to time step ratio, $\frac{h}{\Delta t}$, for stability than the homogeneous boundary condition algorithm. So we use a smaller time step for the inhomogeneous examples. The scheme is executed on a PC laptop with Intel Core 2 Solo processor with 4GB RAM. The error, spatial order, and code execution time results are in Tables 3 and 4. The fourth order accuracy can be observed for all the examples except for Example 8, and we believe that the order might be compromised since the selected time step is not sufficiently small, while the simulation takes too long for such a three-dimensional system.

Example 1. Linear problem in two-dimensions with homogeneous Neumann boundary conditions.

Table 3
Error, order, and CPU time results of the two-dimensional examples.

$N_x \times N_y \times N_t$	Error	Order	CPU Time (s)
Example 1			
$20 \times 20 \times 640$	3.28×10^{-4}	–	0.10
$40 \times 40 \times 640$	9.08×10^{-6}	5.18	0.45
$80 \times 80 \times 640$	2.04×10^{-7}	5.48	2.57
$160 \times 160 \times 640$	9.93×10^{-9}	4.36	15.43
Example 2			
$20 \times 20 \times 640$	2.09×10^{-3}	–	0.25
$40 \times 40 \times 640$	9.27×10^{-5}	4.49	0.83
$80 \times 80 \times 640$	3.00×10^{-6}	4.95	5.16
$160 \times 160 \times 640$	1.48×10^{-7}	4.34	29.68
Example 3			
$20 \times 20 \times 640$	2.02×10^{-4}	–	0.09
$40 \times 40 \times 640$	1.21×10^{-5}	4.06	0.35
$80 \times 80 \times 640$	4.72×10^{-7}	4.68	2.49
$160 \times 160 \times 640$	1.61×10^{-8}	4.87	14.99
Example 4			
$20 \times 20 \times 640$	2.13×10^{-3}	–	0.20
$40 \times 40 \times 640$	6.24×10^{-5}	5.09	1.08
$80 \times 80 \times 640$	1.52×10^{-5}	2.04	5.03
$160 \times 160 \times 640$	7.96×10^{-7}	4.26	27.92
Example 5			
$20 \times 20 \times 1280$	2.31×10^{-4}	–	1.52
$40 \times 40 \times 1280$	5.35×10^{-6}	5.43	4.74
$80 \times 80 \times 1280$	2.47×10^{-7}	4.44	23.73

Table 4
Error, order, and CPU time results of the three-dimensional examples.

$N_x \times N_y \times N_z \times N_t$	Error	Order	CPU Time (s)
Example 6			
$20 \times 20 \times 20 \times 640$	4.93×10^{-4}	–	17.33
$40 \times 40 \times 40 \times 640$	1.36×10^{-5}	5.18	96.14
$80 \times 80 \times 80 \times 640$	3.06×10^{-7}	5.48	872.26
$160 \times 160 \times 160 \times 640$	1.49×10^{-8}	4.36	9670.20
Example 7			
$20 \times 20 \times 20 \times 640$	1.99×10^{-4}	–	14.66
$40 \times 40 \times 40 \times 640$	1.20×10^{-5}	4.05	85.02
$80 \times 80 \times 80 \times 640$	4.69×10^{-7}	4.67	824.93
$160 \times 160 \times 160 \times 640$	1.60×10^{-8}	4.87	9349.11
Example 8			
$20 \times 20 \times 20 \times 1280$	2.53×10^{-4}	–	351.10
$40 \times 40 \times 40 \times 1280$	5.53×10^{-6}	5.52	3120.31
$80 \times 80 \times 80 \times 1280$	5.53×10^{-7}	3.32	38662.07

We consider a linear reaction–diffusion equation

$$\begin{cases} \dfrac{\partial u}{\partial t} = 0.2\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) + 0.1u, \qquad (x, y) \in \Omega = \{0 < x < 2\pi, \pi/2 < y < 5\pi/2\}; \\ \dfrac{\partial u}{\partial x}(0, y, t) = \dfrac{\partial u}{\partial x}(2\pi, y, t) = 0; \\ \dfrac{\partial u}{\partial y}(x, \pi/2, t) = \dfrac{\partial u}{\partial y}(x, 5\pi/2, t) = 0; \\ u(x, y, 0) = \cos x + \sin y. \end{cases} \tag{54}$$

The exact solution of the system is

$$u(x, y, t) = e^{-0.1t}(\cos x + \sin y). \tag{55}$$

Example 2. Nonlinear problem in two-dimensions with homogeneous Neumann boundary conditions.

We consider a nonlinear reaction–diffusion equation

$$\begin{cases} \dfrac{\partial u}{\partial t} = 0.2\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) + \sin u, \qquad (x, y) \in \Omega = \{0 < x < 2\pi, \pi/2 < y < 5\pi/2\}; \\ \dfrac{\partial u}{\partial x}(0, y, t) = \dfrac{\partial u}{\partial x}(2\pi, y, t) = 0; \\ \dfrac{\partial u}{\partial y}(x, \pi/2, t) = \dfrac{\partial u}{\partial y}(x, 5\pi/2, t) = 0; \\ u(x, y, 0) = \cos x + \sin y. \end{cases} \tag{56}$$

Since we do not know the exact solution, we treat the calculated solution for a very fine spatial mesh as the exact solution. The fine mesh is $1280 \times 1280 \times 640$, $(N_x \times N_y \times N_t)$. The cIIF scheme took about 12.5 h to calculate the solution on this fine mesh.

Example 3. Linear problem in two-dimensions with homogeneous Dirichlet boundary conditions.

We consider a linear reaction–diffusion equation

$$\begin{cases} \dfrac{\partial u}{\partial t} = 0.1\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) + 0.1u, \qquad (x, y) \in \Omega = \{0 < x < 2\pi, \pi/2 < y < 5\pi/2\}; \\ u(0, y, t) = u(2\pi, y, t) = 0; \\ u(x, \pi/2, t) = u(x, 5\pi/2, t) = 0; \\ u(x, y, 0) = \sin x \cos y. \end{cases} \tag{57}$$

The exact solution of the system is

$$u(x, y, t) = e^{-0.1t} \sin x \cos y. \tag{58}$$

Example 4. Nonlinear problem in two-dimensions with homogeneous Dirichlet boundary conditions.

We consider a nonlinear reaction–diffusion equation

$$\begin{cases} \dfrac{\partial u}{\partial t} = 0.1\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) + \sin u, \qquad (x, y) \in \Omega = \{0 < x < 2\pi, \pi/2 < y < 5\pi/2\}; \\ u(0, y, t) = u(2\pi, y, t) = 0; \\ u(x, \pi/2, t) = u(x, 5\pi/2, t) = 0; \\ u(x, y, 0) = \sin x \cos y. \end{cases} \tag{59}$$

Since we do not know the exact solution, we treat the calculated solution for a very fine spatial mesh as the exact solution. The fine mesh is $1280 \times 1280 \times 640$, $(N_x \times N_y \times N_t)$. The cIIF scheme took about 11.6 h to calculate the solution on this fine mesh.

Example 5. Linear problem in two-dimensions with inhomogeneous Dirichlet boundary conditions.

We consider a linear reaction–diffusion equation

$$\begin{cases} \dfrac{\partial u}{\partial t} = 0.1\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) + 0.1u, \qquad (x, y) \in \Omega = \{\pi/2 < x < 5\pi/2, 0 < y < 2\pi\}; \\ u(\pi/2, y, t) = u(5\pi/2, y, t) = e^{-0.1t} \cos y; \\ u(x, 0, t) = u(x, 2\pi, t) = e^{-0.1t} \sin x; \\ u(x, y, 0) = \sin x \cos y. \end{cases} \tag{60}$$

The exact solution of the system is

$$u(x, y, t) = e^{-0.1t} \sin x \cos y. \tag{61}$$

Example 6. Linear problem in three-dimensions with homogeneous Dirichlet boundary conditions.

We consider a linear reaction–diffusion equation

$$\begin{cases} \dfrac{\partial u}{\partial t} = 0.1\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} + \dfrac{\partial^2 u}{\partial z^2}\right) + 0.2u, \qquad (x, y, z) \in \Omega = \{0 < x < 2\pi, \pi/2 < y < 5\pi/2, \pi/2 < z < 5\pi/2\}; \\ u(0, y, z, t) = u(2\pi, y, z, t) = 0; \\ u(x, \pi/2, z, t) = u(x, 5\pi/2, z, t) = 0; \\ u(x, y, \pi/2, t) = u(x, y, 5\pi/2, t) = 0; \\ u(x, y, z, 0) = \sin x \cos y \cos z. \end{cases} \tag{62}$$

The exact solution of the system is

$$u(x,y,z,t) = e^{-0.1t}\sin x\cos y\cos z. \tag{63}$$

Example 7. Linear problem in three-dimensions with homogeneous Neumann boundary conditions.

We consider a linear reaction–diffusion equation

$$\begin{cases} \dfrac{\partial u}{\partial t} = 0.2\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} + \dfrac{\partial^2 u}{\partial z^2}\right) + 0.1u, \qquad (x,y,z) \in \Omega = \{0 < x < 2\pi, \pi/2 < y < 5\pi/2, \pi/2 < z < 5\pi/2\}; \\ \dfrac{\partial u}{\partial x}(0,y,z,t) = \dfrac{\partial u}{\partial x}(2\pi,y,z,t) = 0; \\ \dfrac{\partial u}{\partial y}(x,\pi/2,z,t) = \dfrac{\partial u}{\partial y}(x,5\pi/2,z,t) = 0; \\ \dfrac{\partial u}{\partial z}(x,y,\pi/2,t) = \dfrac{\partial u}{\partial z}(x,y,5\pi/2,t) = 0; \\ u(x,y,z,0) = \cos x + \sin y + \sin z. \end{cases} \tag{64}$$

The exact solution of the system is

$$u(x,y,t) = e^{-0.1t}(\cos x + \sin y + \sin z). \tag{65}$$

Example 8. Linear problem in three-dimensions with inhomogeneous Dirichlet boundary conditions.

We consider a linear reaction–diffusion equation

$$\begin{cases} \dfrac{\partial u}{\partial t} = 0.1\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} + \dfrac{\partial^2 u}{\partial z^2}\right) + 0.2u, \qquad (x,y,z) \in \Omega = \{\pi/2 < x < 5\pi/2, 0 < y < 2\pi, 0 < z < 2\pi\}; \\ u(\pi/2,y,z,t) = u(5\pi/2,y,z,t) = e^{-0.1t}\cos y\cos z; \\ u(x,0,z,t) = u(x,2\pi,z,t) = e^{-0.1t}\sin x\cos z; \\ u(x,y,0,t) = u(x,y,2\pi,t) = e^{-0.1t}\sin x\cos y; \\ u(x,y,z,0) = \sin x\cos y\cos z. \end{cases} \tag{66}$$

The exact solution of the system is

$$u(x,y,z,t) = e^{-0.1t}\sin x\cos y\cos z. \tag{67}$$

4. Conclusions

In high spatial dimensions, the compact representation of integration factor approach was found to be very efficient for solving systems involving high-order spatial derivatives and reactions with drastically different time scales, which in general demand temporal schemes with severe stability constraints. In general, it is difficult to develop cIIF with high order accuracy, especially for inhomogeneous boundary conditions. In this paper, we have developed a cIIF method for solving a class of stiff reaction–diffusion systems for inhomogeneous boundary conditions with fourth order accuracy in space. In this approach, the stability condition, computational savings, and storage are similar to the original cIIF with second order accuracy.

Although the high order IF method has been presented only in the context of implicit integration factor methods for reaction–diffusion equations, such approach can easily be applied to other integration factor or exponential time difference methods. Other types of equations with high-order derivatives, (e.g. Cahn–Hilliard equations with fourth-order derivatives) may also be potentially handled using this approach for better efficiency. To better deal with high spatial dimensions, one may incorporate the sparse grid [16,20,21] into the compact representation technique. The flexibility of compact representation allows either direct calculation of the exponentials of matrices, or the use of Krylov subspace [14–16,22–24] for non-constant diffusion coefficients to compute their exponential matrix–vector multiplications for further saving in storage and computational cost. In addition, the presented approach based on the finite difference framework for spatial discretization could also be extended to other discretization methods such as finite volume [25–27] or spectral methods [28,29]. Overall, the compact representation along with integration factor methods provides an efficient approach for solving a wide range of problems arising from biological and physical applications. Given its effectiveness in implementation and good stability conditions, the method is very desirable to be incorporated with local adaptive mesh refinement [10,30,31], which will also be further explored in future work.

Acknowledgment

This work was partially supported by US National Science Foundation under Grant Number DMS1308948.

References

[1] G. Beylkin, J.M. Keiser, L. Vozovoi, A new class of time discretization schemes for the solution of nonlinear PDEs, J. Comput. Phys. 147 (1998) 362–387.
[2] Q. Du, W. Zhu, Stability analysis and applications of the exponential time differencing schemes, J. Comput. Math. 22 (2004) 200.
[3] Q. Du, W. Zhu, Modified exponential time differencing schemes: analysis and applications, BIT, Numer. Math. 45 (2005) 307–328.
[4] T. Hou, J. Lowengrub, M. Shelley, Removing the stiffness from interfacial flows with surface tension, J. Comput. Phys. 114 (1994) 312.
[5] H. Jou, P. Leo, J. Lowengrub, Microstructural Evolution in Inhomogeneous Elastic Media, J. Comput. Phys. 131 (1997) 109.
[6] A.K. Kassam, L.N. Trefethen, Fourth-order time stepping for stiff PDEs, SIAM J. Sci. Comput. 26 (2005) 1214–1233.
[7] P. Leo, J. Lowengrub, Q. Nie, Microstructural Evolution in Orthotropic Elastic Media, J. Comput. Phys. 157 (2000) 44–88.
[8] Q. Nie, F. Wan, Y.T. Zhang, X. Liu, Compact integration factor methods in high spatial dimensions, J. Comput. Phys. 277 (2008) 5238–5255.
[9] Q. Nie, Y.T. Zhang, R. Zhao, Efficient semi-implicit schemes for stiff systems, J. Comput. Phys. 214 (2006) 521–537.
[10] X. Liu, Q. Nie, Compact integration factor methods for complex domains and adaptive mesh refinement, J. Comput. Phys. 229 (16) (2010) 5692–5706.
[11] X.D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, J. Comput. Phys. 115 (1) (1994) 200–212.
[12] G.S. Jiang, C.W. Shu, Efficient implementation of weighted eno schemes, J. Comput. Phys. 126 (1) (1996) 202–228.
[13] S. Zhao, J. Ovadia, X. Liu, Y. Zhang, Q. Nie, Operator splitting implicit integration factor methods for stiff reaction–diffusion-advection systems, J. Comput. Phys. 230 (15) (2011) 5996–6009.
[14] T. Jiang, Y.T. Zhang, Krylov implicit integration factor WENO methods for semilinear and fully nonlinear advection-diffusion-reaction equations, J. Comput. Phys. 253 (2013) 368–388.
[15] T. Jiang, Y.T. Zhang, Krylov single-step implicit integration factor WENO method for advection-diffusion-reaction equations, J. Comput. Phys. 311 (2016) 22–44.
[16] D. Lu, Y.T. Zhang, Krylov integration factor method on sparse grids for high spatial dimension convection–diffusion equations, J. Sci. Comput. 69 (2016) 736–763.
[17] L. Ju, X. Liu, W. Leng, Compact implicit integration factor methods for a family of semilinear fourth-order parabolic equations, Discrete Contin. Dyn. Syst. Ser. B 19 (6) (2014).
[18] Y. Gong, X. Liu, Q. Wang, Fully discretized energy stable schemes for hydrodynamic equations governing two-phase viscous fluid flows, J. Sci. Comput. http//dx.doi.org/10.1007/s10915-016-(2016)0224-7.
[19] L. Ju, J. Zhang, L. Zhu, Q. Du, Fast explicit integration factor methods for semilinear parabolic equations, J. Sci. Comput. 62 (2) (2015) 431–455.
[20] J. Shen, H. Yu, Efficient spectral sparse grid methods and applications to highdimensional elliptic problems, SIAM J. Sci. Comput. 32 (6) (2010) 3228–3250.
[21] D. Wang, W. Chen, Q. Nie, Semi-implicit integration factor methods on sparse grids for high-dimensional systems, J. Comput. Phys. 292 (2015) 43–55.
[22] S. Chen, Y.T. Zhang, Krylov implicit integration factor methods for spatial discretization on high dimensional unstructured meshes: application to discontinuous galerkin methods, J. Comput. Phys. 230 (11) (2011) 4336–4352.
[23] M. Hochbruck, C. Lubich, On krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 34 (5) (1997) 1911–1925.
[24] Y. Saad, Analysis of some krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 29 (1) (1992) 209–228.
[25] R. Eymard, T. Gallouët, R. Herbin, Finite volume methods, in: Handbook of Numerical Analysis, Vol. 7, 2000, pp. 713–1018.
[26] A. Jameson, W. Schmidt, E. Turkel, Numerical solution of the euler equations by finite volume methods using runge kutta time stepping schemes, in: 14th fluid and plasma dynamics conference, 1981, pp. 1259.
[27] R. LeVeque, Numerical Methods for Conservation Laws, Birkhauser, 1992.
[28] E.O. Brigham, The Fast Fourier Transform and Its Applications, 1988.
[29] Van. Loan, C, Computational Frameworks for the Fast Fourier Transform, SIAM, 1992.
[30] M. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1) (1989) 64–84.
[31] M. Berger, J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, J. Comput. Phys. 53 (1984) 484–512.