DISCRETE AND CONTINUOUS
DYNAMICAL SYSTEMS SERIES B
Volume **19**, Number **6**, August **2014**

doi:10.3934/dcdsb.2014.19.1667

pp. **1667–1687**

COMPACT IMPLICIT INTEGRATION FACTOR METHODS FOR A FAMILY OF SEMILINEAR FOURTH-ORDER PARABOLIC EQUATIONS

Lili Ju

Department of Mathematics, University of South Carolina
Columbia, SC 29208, USA
and
Beijing Computational Science Research Center
Beijing, 100084, China

Xinfeng Liu

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

Wei Leng

State Key Laboratory of Scientific and Engineering Computing
Chinese Academy of Sciences, Beijing, 100190, China

(Communicated by Qing Nie)

Abstract. When developing efficient numerical methods for solving parabolic types of equations, severe temporal stability constraints on the time step are often required due to the high-order spatial derivatives and/or stiff reactions. The implicit integration factor (IIF) method, which treats spatial derivative terms explicitly and reaction terms implicitly, can provide excellent stability properties in time with nice accuracy. One major challenge for the IIF is the storage and calculation of the dense exponentials of the sparse discretization matrices resulted from the linear differential operators. The compact representation of the IIF (cIIF) can overcome this shortcoming and greatly save computational cost and storage. On the other hand, the cIIF is often hard to be directly applied to deal with problems involving cross derivatives. In this paper, by treating the discretization matrices in diagonalized forms, we develop an efficient cIIF method for solving a family of semilinear fourth-order parabolic equations, in which the bi-Laplace operator is explicitly handled and the computational cost and storage remain the same as to the classic cIIF for

2010 *Mathematics Subject Classification.* 65M06, 35K25.

Key words and phrases. Integration factor method, compact representation, implicit scheme, bi-Laplace operator, semilinear fourth-order parabolic equations.

L. Ju's research is partially supported by US National Science Foundation under grant number DMS-1215659 and US Department of Energy, Office of Science, Advanced Scientific Computing Research and Biological and Environmental Research programs through the SciDAC project "PISCEES" under grant number DE-SC0008087-ER65393. X. Liu's research is partially supported by US National Science Foundation under grant numbers DMS1019544 and DMS1308948, and University of South Carolina ASPIRE grant. W. Leng's research is partially supported by National 863 Project of China under grant number 2012AA01A309, and National Center for Mathematics and Interdisciplinary Sciences of Chinese Academy of Sciences.

second-order problems. In particular, the proposed method can deal with not only stiff nonlinear reaction terms but also various types of homogeneous or inhomogeneous boundary conditions. Numerical experiments are finally presented to demonstrate effectiveness and accuracy of the proposed method.

1. **Introduction.**

Let Ω be an open rectangular domain in $\mathbb{R}^d$ and a final time $T > 0$. In this paper, we consider a family of semilinear fourth-order parabolic equations taking the following form:

$$\begin{cases} \dfrac{\partial u}{\partial t} = -D\Delta^2 u + f(u), & \mathbf{x} \in \Omega, t \in [0, T], \\ u|_{t=0} = u_0, & \mathbf{x} \in \Omega, \end{cases} \tag{1}$$

where $D > 0$ is the diffusion coefficient. Different boundary conditions such as the Dirichlet boundary condition, periodic boundary condition or Neumann boundary condition will all be studied in this paper. One of major difficulties in numerically solving such equations is how to efficiently handle the bi-Laplace operator coupled with the stiff nonlinear reaction term $f(u)$. In general, the time step size relies heavily on stiffness of the reactions and treatment of the high-order derivatives. Integration factor (IF) or exponential differencing time (ETD) methods are among the effective approaches to deal with temporal stability constraints associated with reaction-diffusion equations [5, 14, 15]. By treating the highest order spatial derivatives exactly in time direction, the IF or ETD methods are able to achieve excellent temporal stability [6, 7, 14, 16]. To deal with additional stability constraints from the stiff reactions, an implicit integration factor (IIF) method [21] was 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 are decoupled from the diffusion term to avoid solving large nonlinear systems involving both diffusions and reactions, such as in the standard implicit method for reaction-diffusion equations. The IIF method has very nice stability, for example, its second-order scheme being linearly unconditionally stable [21].

In the IF based methods the dominant computational cost arises from storage and calculation of exponentials of matrices resulting from discretization of the linear differential operators. To overcome this difficulty in high spatial dimensions, compact representation of the discretization matrices was introduced in the context of the IIF method [20]. In the compact implicit integration factor (cIIF) method in two dimensions, 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 ones for the IIF while still preserving the stability property of the IIF. In three or higher dimensions, the cIIF is significantly more efficient in both storage and computational cost since the discretized matrix for each spatial direction has the same size as the classic IIF in one dimension. In addition, the cIIF method is robust in its implementation and integration with other spatial and temporal discretization algorithms. For example, it can handle general curvilinear coordinates as well as combining with adaptive mesh refinements in a straightforward fashion [18]. One also can apply the cIIF to stiff reactions and diffusions while using other specialized hyperbolic solvers (e.g. WENO methods [19, 12]) for convection terms to solve reaction-diffusion-convection equations efficiently [26].

On the other hand, the cIIF is often hard to be directly applied to deal with problems involving cross derivatives. In this paper, we develop a cIIF method for solving a family of semilinear fourth-order parabolic equations, in which the bi-Laplace operator is explicitly handled. In this approach we use standard central finite differences for spatial discretization coupled with compact implicit integration factor methods for time discretization. The discretized matrices arising from a compact representation of the diffusion operator need to be diagonalized just once and pre-calculated before each time step iteration. We also discuss several other key issues in the construction and implementation of the cIIF method such as fast evaluations of matrix-vector multiplications and stable and accurate incorporations of inhomogeneous boundary conditions, while how to deal with inhomogeneous boundary conditions with cIIF was not addressed at all in previous studies [20, 21]. This new approach is found to have similar stability properties and computational cost as the cIIF for second-order problems.

In the remainder of the paper, we first derive and analyze the cIIF method for the semilinear fourth-order parabolic equation (1) with various homogeneous and inhomogeneous boundary conditions (Dirichlet, Periodic and Neumann boundary conditions) in two dimensions in Section 2, then present its extension to three dimensions in Section 3. Numerical simulations given in Section 4 exhibit excellent performance of the proposed cIIF method. Finally a conclusion is drawn in Section 5.

2. Compact implicit integration factor methods in two dimensions.

In this section, we derive compact implicit integration factor (cIIF) methods for the model equation (1) in two dimensions. Suppose $\Omega = \{x_b < x < x_e, y_b < y < y_e\}$. The equation (1) can be written as

$$\frac{\partial u}{\partial t} = -D(u_{xxxx} + 2u_{xxyy} + u_{yyyy}) + f(u), \qquad (x,y) \in \Omega, t \in [0,T]. \tag{2}$$

2.1. The problem with Dirichlet boundary conditions.

We first consider the problem with a Dirichlet boundary condition

$$\begin{cases} u = g_1, & (x,y) \in \partial\Omega, t \in [0,T], \\ \Delta u = g_2, & (x,y) \in \partial\Omega, t \in [0,T]. \end{cases} \tag{3}$$

Let us discretize the spatial domain by a rectangular mesh which is uniform in each direction as follows: $(x_i, y_j) = (x_b + ih_x, y_b + jh_y)$ where $h_x = (x_e - x_b)/N_x$, $h_y = (y_e - y_b)/N_y$ and $0 \le i \le N_x$ and $0 \le j \le N_y$. We will use the central finite difference discretization scheme with second-order accuracy for the three spatial derivative terms: u_{xxxx}, u_{yyyy} and u_{xxyy}.

Set $u_{i,j} = u(t, x_i, y_j)$ for $0 \le i \le N_x$ and $0 \le j \le N_y$. Define

$$\delta_x^2 u_{i,j} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h_x^2}, \quad \delta_y^2 u_{i,j} = \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h_y^2}.$$

Then we can write the semi-discretization of (2) by the second-order central difference scheme in the following form:

$$\frac{du_{i,j}}{dt} = -D(\delta_x^2\delta_x^2 u_{i,j} + 2\delta_x^2\delta_y^2 u_{i,j} + \delta_y^2\delta_y^2 u_{i,j}) + \mathcal{F}(u_{i,j}), \tag{4}$$

for $0 < i < N_x$ and $0 < j < N_y$.

Denote the set of unknowns as

$$\mathbf{U} = \begin{pmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,N_y-1} \\ u_{2,1} & u_{2,2} & \cdots & u_{2,N_y-1} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N_x-1,1} & u_{N_x-1,2} & \cdots & u_{N_x-1,N_y-1} \end{pmatrix}_{(N_x-1)\times(N_y-1)},$$

and set

$$\mathbf{U}_{x1} = \frac{\sqrt{D}}{h_x^2} \begin{pmatrix} g_1(t,x_0,y_1) & g_1(t,x_0,y_2) & \cdots & g_1(t,x_0,y_{N_y-1}) \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \\ g_1(t,x_{N_x},y_1) & g_1(t,x_{N_x},y_2) & \cdots & g_1(t,x_{N_x},y_{N_y-1}) \end{pmatrix}_{(N_x-1)\times(N_y-1)},$$

$$\mathbf{U}_{y1} = \frac{\sqrt{D}}{h_y^2} \begin{pmatrix} g_1(t,x_1,y_0) & 0 & \cdots & 0 & g_1(t,x_1,y_{N_y}) \\ g_1(t,x_2,y_0) & 0 & \cdots & 0 & g_1(t,x_2,y_{N_y}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_1(t,x_{N_x-1},y_0) & 0 & \cdots & 0 & g_1(t,x_{N_x-1},y_{N_y}) \end{pmatrix}_{(N_x-1)\times(N_y-1)},$$

$$\mathbf{U}_{x2} = \frac{D}{h_x^2} \begin{pmatrix} g_2(t,x_0,y_1) & g_2(t,x_0,y_2) & \cdots & g_2(t,x_0,y_{N_y-1}) \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \\ g_2(t,x_{N_x},y_1) & g_2(t,x_{N_x},y_2) & \cdots & g_2(t,x_{N_x},y_{N_y-1}) \end{pmatrix}_{(N_x-1)\times(N_y-1)},$$

$$\mathbf{U}_{y2} = \frac{D}{h_y^2} \begin{pmatrix} g_2(t,x_1,y_0) & 0 & \cdots & 0 & g_2(t,x_1,y_{N_y}) \\ g_2(t,x_2,y_0) & 0 & \cdots & 0 & g_2(t,x_2,y_{N_y}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_2(t,x_{N_x-1},y_0) & 0 & \cdots & 0 & g_2(t,x_{N_x-1},y_{N_y}) \end{pmatrix}_{(N_x-1)\times(N_y-1)}.$$

Let us further define

$$\mathbf{G}_{P\times P} = \begin{pmatrix} -2 & 1 & 0 & 0 & \cdots & 0 \\ 1 & -2 & 1 & 0 & \cdots & 0 \\ & & \ddots & \ddots & \ddots & \\ 0 & \cdots & 0 & 1 & -2 & 1 \\ 0 & \cdots & 0 & 0 & 1 & -2 \end{pmatrix}_{P\times P},$$

and set $\mathbf{A} = \frac{\sqrt{D}}{h_x^2}\mathbf{G}_{(N_x-1)\times(N_x-1)}, \mathbf{B} = \frac{\sqrt{D}}{h_y^2}\mathbf{G}_{(N_y-1)\times(N_y-1)}$. Define the special operators $\textcircled{x}$ and $\textcircled{y}$ as follows:

$$\begin{aligned} (\mathbf{A}\textcircled{x}\mathbf{U})_{i,j} &= \sum_{l=1}^{N_x-1} (\mathbf{A})_{i,l} u_{l,j}, \\ (\mathbf{B}\textcircled{y}\mathbf{U})_{i,j} &= \sum_{l=1}^{N_y-1} (\mathbf{B})_{j,l} u_{i,l}. \end{aligned} \tag{5}$$

Note here that these two operators are commutative, i.e.,

$$\mathbf{B}\textcircled{y}\mathbf{A}\textcircled{x}\mathbf{U} = \mathbf{A}\textcircled{x}\mathbf{B}\textcircled{y}\mathbf{U}.$$

After incorporating with boundary conditions, the compact representation of (4) takes the following form:

$$\begin{aligned}\frac{d\mathbf{U}}{dt} \;=\; & -\mathbf{A}\textcircled{x}(\mathbf{A}\textcircled{x}\mathbf{U}+\mathbf{U}_{x1}+\mathbf{B}\textcircled{y}\mathbf{U}+\mathbf{U}_{y_1})-\mathbf{U}_{x2}\\ & -\mathbf{B}\textcircled{y}(\mathbf{A}\textcircled{x}\mathbf{U}+\mathbf{U}_{x1}+\mathbf{B}\textcircled{y}\mathbf{U}+\mathbf{U}_{y_1})-\mathbf{U}_{y2}+\mathcal{F}(\mathbf{U}),\end{aligned} \tag{6}$$

which can be simplified to

$$\frac{d\mathbf{U}}{dt}=-\mathbf{A}^2\textcircled{x}\mathbf{U}-2\mathbf{B}\textcircled{y}\mathbf{A}\textcircled{x}\mathbf{U}-\mathbf{B}^2\textcircled{y}\mathbf{U}+\mathbf{E}+\mathcal{F}(\mathbf{U}), \tag{7}$$

where $\mathcal{F}(\mathbf{U})=(f(u_{i,j}))_{(N_x-1)\times(N_y-1)}$ and

$$\mathbf{E}=-\mathbf{A}\textcircled{x}(\mathbf{U}_{x1}+\mathbf{U}_{y1})-\mathbf{U}_{x2}-\mathbf{B}\textcircled{y}(\mathbf{U}_{x1}+\mathbf{U}_{y1})-\mathbf{U}_{y2}. \tag{8}$$

Here $\mathbf{A}$ and $\mathbf{B}$ are diagonalizable and let us assume that

$$\mathbf{A}=\mathbf{P}_x\tilde{\mathbf{D}}_x\mathbf{P}_x{}^{-1},\quad \mathbf{B}=\mathbf{P}_y\tilde{\mathbf{D}}_y\mathbf{P}_y{}^{-1}, \tag{9}$$

where $\tilde{\mathbf{D}}_x$ and $\tilde{\mathbf{D}}_y$ are diagonal matrices with the eigenvalues of $\mathbf{A}$ and $\mathbf{B}$ as the diagonal elements, respectively.

Let $\mathbf{V}=\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}\mathbf{U}$, then (7) can be transformed into

$$\frac{d\mathbf{V}}{dt}=-\tilde{\mathbf{D}}_x^2\textcircled{x}\mathbf{V}-2\tilde{\mathbf{D}}_y\textcircled{y}\tilde{\mathbf{D}}_x\textcircled{x}\mathbf{V}-\tilde{\mathbf{D}}_y^2\textcircled{y}\mathbf{V}+\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}(\mathbf{E}+\mathcal{F}(\mathbf{U})), \tag{10}$$

and thus

$$\frac{d\mathbf{V}}{dt}+\tilde{\mathbf{D}}_x^2\textcircled{x}\mathbf{V}+2\tilde{\mathbf{D}}_y\textcircled{y}\tilde{\mathbf{D}}_x\textcircled{x}\mathbf{V}+\tilde{\mathbf{D}}_y^2\textcircled{y}\mathbf{V}=\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}(\mathbf{E}+\mathcal{F}(\mathbf{U})). \tag{11}$$

Suppose that the diagonal matrices are given by $\tilde{\mathbf{D}}_x=\mathrm{diag}[d_1^x,d_2^x,\cdots,d_{N_x-1}^x]$ and $\tilde{\mathbf{D}}_y=\mathrm{diag}[d_1^y,d_2^y,\cdots,d_{N_y-1}^y]$. Set $\tilde{\mathbf{H}}=(\tilde{h}_{i,j})_{(N_x-1)\times(N_y-1)}$ with $\tilde{h}_{i,j}=2d_i^x d_j^y$. Here we define an operation "(e^*)" by taking element-by-element exponentials as the following,

$$(e^*)^{\tilde{\mathbf{H}}}=(e^{\tilde{h}_{i,j}})_{(N_x-1)\times(N_y-1)}.$$

Define another operator "$\odot$" for element-by-element multiplication between two matrices with the same size by the following,

$$(\mathbf{M}\odot\mathbf{L})_{i,j}=(m_{i,j}l_{i,j}),$$

where $\mathbf{M}=(m_{i,j})$, $\mathbf{L}=(l_{i,j})$.

Then direct extension on (11) leads to

$$\begin{aligned}&\left[e^{\tilde{\mathbf{D}}_y^2 t}\textcircled{y}e^{\tilde{\mathbf{D}}_x^2 t}\textcircled{x}\left(\frac{d\mathbf{V}}{dt}+\tilde{\mathbf{D}}_x^2\textcircled{x}\mathbf{V}+2\tilde{\mathbf{D}}_y\textcircled{y}\tilde{\mathbf{D}}_x\textcircled{x}\mathbf{V}+\tilde{\mathbf{D}}_y^2\textcircled{y}\mathbf{V}\right)\right]\odot(e^*)^{\tilde{\mathbf{H}}t})=\\ &\qquad\qquad\left[e^{\tilde{\mathbf{D}}_y^2 t}\textcircled{y}e^{\tilde{\mathbf{D}}_x^2 t}\textcircled{x}\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}(\mathbf{E}+\mathcal{F}(\mathbf{U}))\right]\odot(e^*)^{\tilde{\mathbf{H}}t}),\end{aligned} \tag{12}$$

which can be simplified by

$$\frac{d(\mathbf{V}\odot(e^*)^{\mathbf{H}t})}{dt}=(\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}(\mathbf{E}+\mathcal{F}(\mathbf{U})))\odot(e^*)^{\mathbf{H}t}, \tag{13}$$

where $\mathbf{H}=(h_{i,j})_{(N_x-1)\times(N_y-1)}$ with $h_{i,j}=(d_i^x+d_j^y)^2$.

Let us discretize the time as $t_n=n\Delta t,\ n=0,1,\cdots,N_t$ with $\Delta t=T/N_t$. Then taking integration on both sides of (13) from t_n to t_{n+1} gives

$$\mathbf{V}_{n+1}\odot(e^*)^{\mathbf{H}\Delta t}-\mathbf{V}_n=\int_0^{\Delta t}(\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}(\mathbf{E}+\mathcal{F}(\mathbf{U})))\odot(e^*)^{\mathbf{H}\tau}\,d\tau. \tag{14}$$

Thus we have

$$\mathbf{V}_{n+1} = \Big(\mathbf{V}_n + \int_0^{\Delta t} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} (\mathbf{E} + \mathcal{F}(\mathbf{U}))) \odot (e^*)^{\mathbf{H}\tau}\, d\tau\Big) \odot (e^*)^{-\mathbf{H}\Delta t}. \tag{15}$$

Finally by substituting $\mathbf{V}$ by $\mathbf{U}$ in (15), the numerical scheme can be achieved for solving the model equation (1) from t^n to t^{n+1},

$$\begin{aligned} \mathbf{U}_{n+1} \;\; &= \;\; \mathbf{P}_y \textcircled{y} \mathbf{P}_x \textcircled{x} \Big((\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{U}_n) \odot (e^*)^{-\mathbf{H}\Delta t} \\ &\quad + \int_0^{\Delta t} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} (\mathbf{E} + \mathcal{F}(\mathbf{U}))) \odot (e^*)^{-\mathbf{H}(\Delta t - \tau)}\, d\tau \Big). \end{aligned} \tag{16}$$

Next we use multistep schemes to accurately evaluate the integral which appears on the right-hand-side of (16), i.e., $\int_0^{\Delta t} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} (\mathbf{E}+\mathcal{F}(\mathbf{U}))) \odot (e^*)^{-\mathbf{H}(\Delta t-\tau)}\, d\tau$.

2.1.1. *Multistep approximations of the integral from the boundary conditions.* To evaluate the integral resulted from the inhomogeneous boundary terms

$$\int_0^{\Delta t} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}(t_n + \tau)) \odot (e^*)^{-\mathbf{H}(\Delta t - \tau)}\, d\tau,$$

we need to be careful since $(e^*)^{\mathbf{H}(\Delta t - \tau)}$ contains entries which decay with highly different speeds along the time, and $\mathbf{E}(t_n + \tau)$ involves the factors of $1/h_x^4$ and $1/h_y^4$ which could quickly amplify errors arising from the time discretization, which would cause severe numerical instability. To overcome this difficulty, here we will apply an elegant approach proposed in [13], which will be described with details in the following. Let

$$\mathbf{W}(t_n + \tau) = (w_{i,j}(t_n + \tau))_{(N_x - 1)\times(N_y - 1)} = \mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}(t_n + \tau),$$

and

$$\mathbf{Q}^W = (q_{i,j}^W)_{(N_x-1)\times(N_y-1)} = \int_0^{\Delta t} \mathbf{W}(t_n + \tau) \odot (e^*)^{-\mathbf{H}(\Delta t - \tau)}\, d\tau,$$

i.e.,

$$q_{i,j}^W = \int_0^{\Delta t} e^{-h_{i,j}(\Delta t - \tau)} w_{i,j}(t_n + \tau)\, d\tau.$$

It is reasonable to assume that $\mathbf{W}(t)$ changes smoothly along the time. Thus based on the values of $\mathbf{W}(t)$ at $t_{n+1}, t_n, \cdots, t_{n+1-r_1}$, we use the Lagrange interpolation polynomial $P_{r_1}^W(\tau)$ of degree r_1 to approximate $\mathbf{W}(t_n + \tau)$ (they all can be calculated from the given inhomogeneous boundary conditions):

$$P_{r_1}^W(t_n + \tau) = \sum_{s=-1}^{r_1 - 1} \omega_{r_1,s}(\tau) \mathbf{W}(t_{n-s}), \tag{17}$$

with $\omega_{r_1,s}(\tau) = \prod_{\substack{l=-1\\ l\neq s}}^{r_1-1} \dfrac{\tau + l\Delta t}{(l-s)\Delta t}$. Then we have

$$\mathbf{W}(t_n + \tau) \approx P_{r_1}^W(\tau) + O(\Delta t^{r_1+1}). \tag{18}$$

Thus we can approximate $q_{i,j}^W$ by

$$q_{i,j}^W \approx \sum_{s=-1}^{r_1-1} w_{i,j}(t_{n-s}) \int_0^{\Delta t} e^{-h_{i,j}(\Delta t - \tau)} \omega_{r_1,s}(\tau)\, d\tau = \sum_{s=-1}^{r_1-1} w_{i,j}(t_{n-s}) \alpha_{i,j}^{(r_1,s)}, \tag{19}$$

where $\alpha_{i,j}^{(r_1,s)} = \displaystyle\int_0^{\Delta t} e^{-h_{i,j}(\Delta t-\tau)}\omega_{r_1,s}(\tau)\,d\tau$. We remark that $\alpha_{i,j}^{(r_1,s)}$ is independent of the time steps since a uniform time step size Δt is used here. The main idea here is to evaluate $\alpha_{i,j}^{(r_1,s)}$ exactly to avoid any loss of accuracy and numerical instability. For instance, the values of $\{\alpha_{i,j}^{(r_1,s)}\}$ for $r_1 = 0, 1, 2$ are listed as below.

$$r_1 = 0: \qquad \alpha_{i,j}^{(0,-1)} = \phi_0; \tag{20}$$

$$r_1 = 1: \qquad \alpha_{i,j}^{(1,-1)} = \phi_1, \quad \alpha_{i,j}^{(1,0)} = \phi_0 - \phi_1; \tag{21}$$

$$r_1 = 2: \qquad \alpha_{i,j}^{(2,-1)} = \frac{1}{2}(\phi_1+\phi_2), \alpha_{i,j}^{(2,0)} = \phi_0 - \phi_2, \alpha_{i,j}^{(2,1)} = -\frac{1}{2}(\phi_1 - \phi_2); \tag{22}$$

where

$$\begin{cases} \phi_0 = \dfrac{1}{h_{i,j}}(1-e^{-h_{i,j}\Delta t}), \phi_1 = \dfrac{1}{h_{ij}}(1-\dfrac{\phi_0}{\Delta t}), \phi_2 = \dfrac{1}{h_{ij}}(1-\dfrac{2\phi_1}{\Delta t}), & h_{i,j} \neq 0, \\ \phi_0 = \Delta t, \phi_1 = \dfrac{\Delta t}{2}, \phi_2 = \dfrac{\Delta t}{3}, & h_{i,j} = 0. \end{cases} \tag{23}$$

Define $\mathbf{S}_{r_1,s} = (\alpha_{i,j}^{(r_1,s)})_{(N_x-1)\times(N_y-1)}$, then we can have the following approximation for the integral arising from the inhomogeneous boundary conditions

$$\int_0^{\Delta t} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}(t_n+\tau)) \odot (e^*)^{-\mathbf{H}(\Delta t-\tau)}\,d\tau \approx \sum_{s=-1}^{r_1-1} \mathbf{W}_{n-s} \odot \mathbf{S}_{r_1,s}. \tag{24}$$

Such approximation is (r_1+1)-th order accurate in time.

2.1.2. *Implicit multistep approximations of the integral from the nonlinear reaction term.* For evaluation of the integral resulted from the nonlinear term

$$\int_0^{\Delta t} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{F}(\mathbf{U}(t_n+\tau))) \odot (e^*)^{-\mathbf{H}(\Delta t-\tau)}\,d\tau,$$

we may still use a similar multistep approach to obtain an implicit approximation for the nonlinear term for the purpose of stability. However, since $\mathcal{F}(\mathbf{U}(t_n+\tau))$ itself does not involve any term related to spatial mesh size and $0 \le e^{-h_{i,j}(\Delta t-\tau)} \le 1$ for $\tau < \Delta t$, thus we can directly apply the Lagrange interpolation polynomial of degree r_2 at $t_{n+1}, t_n, \cdots, t_{n+1-r_2}$ to the whole integrand $(\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{F}(\mathbf{U}(t_n+\tau))) \odot (e^*)^{\mathbf{H}(\Delta t-\tau)}$ as proposed in [20].

It is not difficult to find that

$$\begin{aligned} &\int_0^{\Delta t} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{F}(\mathbf{U}(t_n+\tau))) \odot (e^*)^{-\mathbf{H}(\Delta t-\tau)}\,d\tau \\ &\qquad\qquad \approx \Delta t \sum_{s=-1}^{r_2-1} \beta^{r_2,s} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{F}(\mathbf{U}_{n-s})) \odot (e^*)^{-(s+1)\mathbf{H}\Delta t} \\ &\qquad\qquad = \Delta t \sum_{s=0}^{r_2-1} \beta^{r_2,s} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{F}(\mathbf{U}_{n-s})) \odot (e^*)^{-(s+1)\mathbf{H}\Delta t} \\ &\qquad\qquad\quad + \Delta t \beta^{r_2,-1} \mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{F}(\mathbf{U}_{n+1}), \end{aligned} \tag{25}$$

with $\beta^{r_2,s} = \displaystyle\int_0^{\Delta t} \omega_{r_2,s}(\tau)\, d\tau$, which is (r_2+1)-th order accurate. For example, the values of $\{\beta_{i,j}^{(r_2,s)}\}$ for $r_2 = 0, 1, 2$ are presented as below.

$$
\begin{aligned}
r_2 = 0: &\qquad \beta^{(0,-1)} = 1; && (26)\\
r_2 = 1: &\qquad \beta^{(1,-1)} = \frac{1}{2}, \quad \beta_{i,j}^{(1,0)} = \frac{1}{2}; && (27)\\
r_2 = 2: &\qquad \beta^{(2,-1)} = \frac{5}{12}, \quad \beta_{i,j}^{(2,0)} = \frac{2}{3}, \quad \beta_{i,j}^{(2,1)} = -\frac{1}{12}. && (28)
\end{aligned}
$$

Note $\{\beta_{i,j}^{(r_2,s)}\}$ are constant across all points.

2.2. **The compact implicit integration factor schemes.** Now put (24) and (25) back into (16), and we obtain a compact implicit integration factor (cIIF) scheme that is second-order accurate in space and at least $(\min\{r_1, r_2\} + 1)$-th order accurate in time:

$$
\begin{aligned}
\mathbf{U}_{n+1} \;=\; & \mathbf{P}_y \textcircled{y} \mathbf{P}_x \textcircled{x} \Big((\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{U}_n) \odot (e^*)^{-\mathbf{H}\Delta t} + \sum_{s=-1}^{r_1-1} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}_{n-s}) \\
& \odot \mathbf{S}_{r_1,s} + \Delta t \sum_{s=0}^{r_2-1} \beta^{r_2,s} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{F}(\mathbf{U}_{n-s})) \odot (e^*)^{-(s+1)\mathbf{H}\Delta t} \Big) \\
& + \Delta t \beta^{r_2,-1} \mathcal{F}(\mathbf{U}_{n+1}). \qquad (29)
\end{aligned}
$$

Clearly the cIIF scheme (29) is completely explicit when there isn't the nonlinear reaction term in the model problem (1), i.e., $f(u) \equiv 0$.

It is specially worthy noting that the nonlinear solution process (that is needed when $f(u)$ is nonlinear) in the cIIF scheme (29) is point-wise and thus very efficient by using some Newton-type iterative method for scalar functions. If we apply the same approximation approach for the first integral as discussed in Sections 2.1.1 to the second integral, then we obtain a much more expensive implicit scheme since the nonlinear solution process becomes global. When there is a function $g(x, y)$ included in the right hand side of the model equation (1), it can be treated exactly the same way as that for inhomogeneous boundary conditions.

We present the detailed scheme with some typical choice of r_1 and r_2 in the following.

- $r_1 = r_2 = 0$ (first-order in time):

$$
\begin{aligned}
\mathbf{U}_{n+1} \;=\; & \mathbf{P}_y \textcircled{y} \mathbf{P}_x \textcircled{x} \Big((\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{U}_n) \odot (e^*)^{-\mathbf{H}\Delta t} \\
& + (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}_{n+1}) \odot \mathbf{S}_{0,-1} \Big) + \Delta t \mathcal{F}(\mathbf{U}_{n+1}); \qquad (30)
\end{aligned}
$$

- $r_1 = r_2 = 1$ (second-order in time):

$$
\begin{aligned}
\mathbf{U}_{n+1} \;=\; & \mathbf{P}_y \textcircled{y} \mathbf{P}_x \textcircled{x} \Big((\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} (\mathbf{U}_n + \frac{\Delta t}{2} \mathcal{F}(\mathbf{U}_n))) \odot (e^*)^{-\mathbf{H}\Delta t} \\
& + (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}_{n+1}) \odot \mathbf{S}_{1,-1} + (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}_n) \odot \mathbf{S}_{1,0} \Big) \\
& + \frac{\Delta t}{2} \mathcal{F}(\mathbf{U}_{n+1}); \qquad (31)
\end{aligned}
$$

- $r_1 = r_2 = 2$ (third-order in time):

$$\begin{aligned}\mathbf{U}_{n+1} \;=\;& \mathbf{P}_y \textcircled{y} \mathbf{P}_x \textcircled{x} \Big((\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} (\mathbf{U}_n + \frac{2\Delta t}{3}\mathcal{F}(\mathbf{U}_n))) \odot (e^*)^{-\mathbf{H}\Delta t} \\ & -(\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \frac{\Delta t}{12}\mathcal{F}(\mathbf{U}_{n-1})) \odot (e^*)^{-2\mathbf{H}\Delta t} + (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}_{n+1}) \\ & \odot \mathbf{S}_{2,-1} + (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}_{n-1}) \odot \mathbf{S}_{2,0} + (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{E}_{n-1}) \odot \mathbf{S}_{2,1} \Big) \\ & + \frac{5\Delta t}{12}\mathcal{F}(\mathbf{U}_{n+1}). \end{aligned} \tag{32}$$

Remark 1. As discussed in [13, 24], the computational complexities of $\mathbf{P}_x \textcircled{x} \mathbf{V}$, $\mathbf{P}_x^{-1} \textcircled{x} \mathbf{V}$, $\mathbf{P}_y \textcircled{y} \mathbf{V}$, $\mathbf{P}_y^{-1} \textcircled{y} \mathbf{V}$ for any $(N_x - 1) \times (N_y - 1)$ array $\mathbf{V}$ can be reduced from $O(N^3)$ to $O(N^2 \log_2(N))$ with $N = \max\{N_x, N_y\}$ through the use of fast Discrete Sine Transform (DST) if we choose

$$d_k^x = -\frac{4D}{h_x^2}\sin^2\left(\frac{k\pi}{2N_x}\right), \quad k = 1, 2, \cdots, N_x - 1,$$

$$d_k^y = -\frac{4D}{h_y^2}\sin^2\left(\frac{k\pi}{2N_y}\right), \quad k = 1, 2, \cdots, N_y - 1.$$

Thus the overall cost of the proposed cIIF scheme in two dimensions is $O(N^2 \log_2 N)$ per time step. In addition, the overall computational storages of the proposed cIIF scheme are in the order of $O(N_x^2), O(N_y^2)$ or $O(N_x N_y)$.

2.3. **Other type of boundary conditions.**

2.3.1. *Periodic boundary conditions.*

If a periodic boundary condition such as

$$\begin{cases} u(t, x_b, y) = u(t, x_e, y),\ u_x(t, x_b, y) = u_x(t, x_e, y), & y \in [y_b, y_e], \\ u(t, x, y_b) = u(t, x, y_e),\ u_y(t, x, y_b) = u_y(t, x, y_e), & x \in [x_b, x_e], \\ \Delta u(t, x_b, y) = \Delta u(t, x_e, y),\ (\Delta u)_x(t, x_b, y) = (\Delta u)_x(t, x_e, y), & y \in [y_b, y_e], \\ \Delta u(t, x, y_b) = \Delta u(t, x, y_e),\ (\Delta u)_y(t, x, y_b) = (\Delta u)_y(t, x, y_e), & x \in [x_b, x_e], \end{cases} \tag{33}$$

is imposed on the model equation (1) and here $t \in [0, T]$. For this case, the unknown solutions can be denoted by

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

and set

$$\mathbf{G}_{P \times P} = \begin{pmatrix} -2 & 1 & 0 & 0 & \cdots & 0 & 1 \\ 1 & -2 & 1 & 0 & \cdots & 0 & 0 \\ & & \ddots & \ddots & \ddots & & \\ 0 & 0 & \cdots & 0 & 1 & -2 & 1 \\ 1 & 0 & \cdots & 0 & 0 & 1 & -2 \end{pmatrix}_{P \times P},$$

and $\mathbf{A} = \frac{\sqrt{D}}{h_x^2}\mathbf{G}_{N_x \times N_x}, \mathbf{B} = \frac{\sqrt{D}}{h_y^2}\mathbf{G}_{N_y \times N_y}$. The cIIF scheme for the periodic boundary condition takes the same from as (29) except that we have $\mathbf{E} = \mathbf{0}$ here.

Remark 2. In the case of periodic boundary conditions, the computational complexities of $\mathbf{P}_x \textcircled{x} \mathbf{V}$, $\mathbf{P}_x^{-1} \textcircled{x} \mathbf{V}$, $\mathbf{P}_y \textcircled{y} \mathbf{V}$, $\mathbf{P}_y^{-1} \textcircled{y} \mathbf{V}$ can be reduced from $O(N^3)$ to $O(N^2 \log_2(N))$ through the use of fast Discrete Fourier Transform (DFT) [13, 25] if we choose

$$d_k^x = -\frac{4D}{h_x^2} \sin^2\left(\frac{(k-1)\pi}{N_x}\right), \quad k = 1, 2, \cdots, N_x,$$
$$d_k^y = -\frac{4D}{h_y^2} \sin^2\left(\frac{(k-1)\pi}{N_y}\right), \quad k = 1, 2, \cdots, N_y.$$

2.3.2. *Neumann boundary conditions.* If a Neumann boundary condition such as

$$\begin{cases} \dfrac{\partial u}{\partial x} = b_1^x, \ \dfrac{\partial u}{\partial y} = b_1^y, & (x,y) \in \partial\Omega, t \in [0,T], \\ \dfrac{\partial \Delta u}{\partial x} = b_2^x, \ \dfrac{\partial \Delta u}{\partial y} = b_2^y, & (x,y) \in \partial\Omega, t \in [0,T], \end{cases} \tag{34}$$

is imposed on the model equation (1), then we may denote the unknown solutions by

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

and set

$$\mathbf{G}_{P\times P} = \begin{pmatrix} -2 & 2 & 0 & 0 & \cdots & 0 \\ 1 & -2 & 1 & 0 & \cdots & 0 \\ & & \ddots & \ddots & \ddots & \\ 0 & \cdots & 0 & 1 & -2 & 1 \\ 0 & \cdots & 0 & 0 & 2 & -2 \end{pmatrix}_{P\times P}.$$

and $\mathbf{A} = \dfrac{\sqrt{D}}{h_x^2} \mathbf{G}_{(N_x+1)\times(N_x+1)}, \mathbf{B} = \dfrac{\sqrt{D}}{h_y^2} \mathbf{G}_{(N_y+1)\times(N_y+1)}$. We again have

$$\mathbf{E} = -\mathbf{A} \textcircled{x} (\mathbf{U}_{x1} + \mathbf{U}_{y1}) - \mathbf{U}_{x2} - \mathbf{B} \textcircled{y} (\mathbf{U}_{x1} + \mathbf{U}_{y1}) - \mathbf{U}_{y2}, \tag{35}$$

but with

$$\mathbf{U}_{x1} = \frac{2\sqrt{D}}{h_x} \begin{pmatrix} -b_1^x(t, x_0, y_0) & -b_1^x(t, x_0, y_1) & \cdots & -b_1^x(t, x_0, y_{N_y}) \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \\ b_1^x(t, x_{N_x}, y_0) & b_1^x(t, x_{N_x}, y_1) & \cdots & b_1^x(t, x_{N_x}, y_{N_y}) \end{pmatrix}_{(N_x+1)\times(N_y+1)},$$

$$\mathbf{U}_{y1} = \frac{2\sqrt{D}}{h_y} \begin{pmatrix} -b_1^y(t, x_0, y_0) & 0 & \cdots & 0 & b_1^y(t, x_0, y_{N_y}) \\ -b_1^y(t, x_1, y_0) & 0 & \cdots & 0 & b_1^y(t, x_1, y_{N_y}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -b_1^y(t, x_{N_x}, y_0) & 0 & \cdots & 0 & b_1^y(t, x_{N_x}, y_{N_y}) \end{pmatrix}_{(N_x+1)\times(N_y+1)},$$

$$\mathbf{U}_{x2} = \frac{2D}{h_x} \begin{pmatrix} -b_2^x(t, x_0, y_0) & -b_2^x(t, x_0, y_1) & \cdots & -b_2^x(t, x_0, y_{N_y}) \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \\ b_2^x(t, x_{N_x}, y_0) & b_2^x(t, x_{N_x}, y_1) & \cdots & b_2^x(t, x_{N_x}, y_{N_y}) \end{pmatrix}_{(N_x+1)\times(N_y+1)},$$

$$\mathbf{U}_{y2} = \frac{2D}{h_y}\begin{pmatrix} -b_2^y(t,x_0,y_0) & 0 & \cdots & 0 & b_2^y(t,x_0,y_{N_y}) \\ -b_2^y(t,x_1,y_0) & 0 & \cdots & 0 & b_2^y(t,x_1,y_{N_y}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -b_2^y(t,x_{N_x},y_0) & 0 & \cdots & 0 & b_2^y(t,x_{N_x},y_{N_y}) \end{pmatrix}_{(N_x+1)\times(N_y+1)}.$$

The cIIF scheme for the Neumann boundary condition again takes the same form as (29).

Remark 3. In the case of Neumann boundary conditions, the computational cost of $\mathbf{P}_x ⓧ \mathbf{V}$, $\mathbf{P}_x^{-1} ⓧ \mathbf{V}$, $\mathbf{P}_y ⓨ \mathbf{V}$, $\mathbf{P}_y^{-1} ⓨ \mathbf{V}$ again can reduced from $O(N^3)$ to $O(N^2 \log_2(N))$ through the reflective extension and the use of fast Discrete Fourier Transform (DFT) [13, 25].

2.4. **Linear stability analysis.** We test the linear stability of the proposed cIIF schemes with the following linear equation

$$u_t = -(q_1+q_2)^2 u + du, \quad \text{where} \quad d > 0, \tag{36}$$

where q_1 and q_2 represent the coefficients for the bi-Laplace term in x, y directions, respectively. As an example, we consider the second-order scheme (31) without terms from boundary conditions. In order to obtain the stability conditions, we first apply the second-order scheme to solve the linear equation (36), then substitute $u_n = e^{in\theta}$ into the scheme. This results in

$$e^{i\theta} = e^{-(q_1+q_2)^2\Delta t}\left(1 + \frac{1}{2}\lambda\right) + \frac{1}{2}\lambda e^{i\theta},$$

where $\lambda = d\Delta t$. Let us solve the real part (λ_r) and the imaginary part (λ_i) of λ, and this leads to

$$\begin{cases} \lambda_r = \dfrac{2(1-e^{-2(q_1+q_2)^2\Delta t})}{(1-e^{-(q_1+q_2)^2\Delta t})^2 + 2(1+\cos\theta)e^{-(q_1+q_2)^2\Delta t}}, \\ \lambda_i = \dfrac{4\sin\theta e^{-(q_1+q_2)^2\Delta t}}{(1-e^{-(q_1+q_2)^2\Delta t})^2 + 2(1+\cos\theta)e^{-(q_1+q_2)^2\Delta t}}. \end{cases} \tag{37}$$

Since $(q_1+q_2)^2 > 0$, we have the real part $\lambda_r > 0$ for all $0 \le \theta \le 2\pi$. Therefore the stability region lies in the whole left half of complex plane, and thus the second-order scheme (31) is A-stable.

For the third-order scheme as shown in (32), a similar analysis can be performed to obtain

$$\lambda = \frac{e^{i\theta} - e^{-(q_1+q_2)^2\Delta t}}{\frac{5}{12}e^{i\theta} + \frac{2}{3}e^{-(q_1+q_2)^2\Delta t} - \frac{1}{12}e^{-2(q_1+q_2)^2\Delta t}}. \tag{38}$$

To investigate the stability region, a recurrence relation of the third order scheme is given by

$$\left(1 - \frac{5}{12}\lambda\right)u_{n+1} - e^{-(q_1+q_2)^2\Delta t}\left(1 + \frac{2}{3}\lambda\right)u_n + \frac{e^{-2(q_1+q_2)^2\Delta t}}{12}\lambda = 0.$$

Thus the characteristic polynomial is

$$\left(1 - \frac{5}{12}\lambda\right)\omega^2 - e^{-(q_1+q_2)^2\Delta t}\left(1 + \frac{2}{3}\lambda\right)\omega + \frac{e^{-2(q_1+q_2)^2\Delta t}}{12}\lambda = 0,$$

which has roots

$$\omega = \frac{e^{-(q_1+q_2)^2\Delta t}\left(1+\frac{2}{3}\lambda \pm \sqrt{\frac{7}{12}\lambda^2+\lambda+1}\right)}{2-\frac{5}{6}\lambda}.$$

The stability region is then obtained by solving $|w| < 1$, i.e.,

$$\left|\frac{1+\frac{2}{3}\lambda \pm \sqrt{\frac{7}{12}\lambda^2+\lambda+1}}{2-\frac{5}{6}\lambda}\right| < e^{(q_1+q_2)^2\Delta t}. \tag{39}$$

Obviously the stability region is an increasing function of $(q_1 + q_2)^2\Delta t$. When $(q_1 + q_2)^2\Delta t \to \infty$, the stability region becomes the entire complex plane except one zero point ($\lambda = 12/5$) on the real axis. For the purpose of easy illustration, the stability regions of the third-order scheme for $(q_1 + q_2)^2\Delta t = 0, 0.5, 0.6, 0.8$ are plotted in Figure 1.

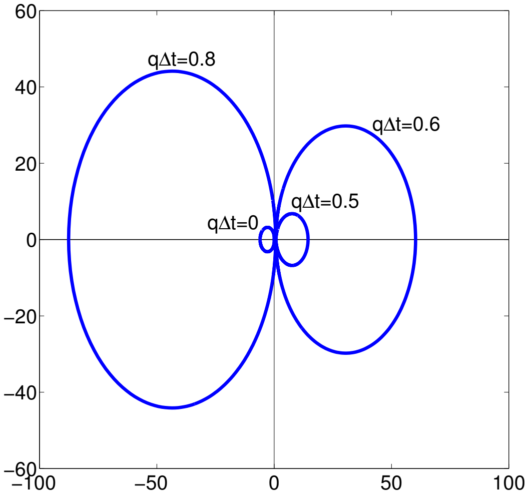

FIGURE 1. Stability regions (interior of the closed contour) for the third-order scheme (32) with $q\Delta t = 0, 0.5, 0.6, 0.8$, where $q = (q_1 + q_2)^2$.

3. Compact implicit integration factor methods in three dimensions.

In this section we derive the cIIF methods for the model problem (1) in three dimensions. Let $\Omega = \{x_b < x < x_e, y_b < y < y_e, z_b < z < z_e\}$. The equation (1) can now be written as

$$\begin{aligned}\frac{\partial u}{\partial t} &= -D(u_{xxxx} + u_{yyyy} + u_{zzzz} + 2u_{xxyy} + 2u_{yyzz} + 2u_{zzxx}) \\ &\quad + f(u), \quad (x, y, z) \in \Omega, t \in [0, T]. \end{aligned} \tag{40}$$

In this section, we only present the case with a Dirichlet boundary condition, and the derivation for other boundary conditions follows the same like the two dimensional case.

Similar to solving the two dimensional system (2), we denote h_x, h_y, h_z as the spatial mesh size, and N_x, N_y, N_z as the number of grid points in x, y, z directions, respectively. Let $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{G}$ defined the same as before in two dimensional case. We now define $\mathbf{C} = \frac{\sqrt{D}}{h_z^2}\mathbf{G}_{(N_z-1)\times(N_z-1)}$. Set $u_{i,j,k} = u(t, x_i, y_j, z_k)$ and $\Delta u_{i,j,k} = \Delta u(t, x_i, y_j, z_k)$ for $0 \le i \le N_x$, $0 \le j \le N_y$ and $0 \le k \le N_z$. Denote the unknown solutions as a three-dimensional array by $\mathbf{U} = (u_{i,j,k})_{(N_x-1)\times(N_y-1)\times(N_z-1)}$.

Define the following three-dimensional arrays with a dimension $(N_x - 1) \times (N_y - 1) \times (N_z - 1)$,

$$\mathbf{U}_{x1} = \frac{\sqrt{D}}{h_x^2}(\mu^x_{i,j,k}), \mu^x_{i,j,k} = \begin{cases} g_1(t, x_0, y_j, z_k), & i = 1, \\ 0, & 1 < i < N_x - 1, \\ g_1(t, x_{N_x}, y_j, z_k), & i = N_x - 1, \end{cases}$$

$$\mathbf{U}_{y1} = \frac{\sqrt{D}}{h_y^2}(\mu^y_{i,j,k}), \mu^y_{i,j,k} = \begin{cases} g_1(t, x_i, y_0, z_k), & j = 1, \\ 0, & 1 < j < N_y - 1, \\ g_1(t, x_i, y_{N_y}, z_k), & j = N_y - 1, \end{cases}$$

$$\mathbf{U}_{z1} = \frac{\sqrt{D}}{h_z^2}(\mu^z_{i,j,k}), \mu^z_{i,j,k} = \begin{cases} g_1(t, x_i, y_j, z_0), & k = 1, \\ 0, & 1 < k < N_z - 1, \\ g_1(t, x_i, y_j, z_{N_z}) & k = N_z - 1, \end{cases}$$

$$\mathbf{U}_{x2} = \frac{D}{h_x^2}(\nu^x_{i,j,k}), \nu^x_{i,j,k} = \begin{cases} g_2(t, x_0, y_j, z_k), & i = 1, \\ 0, & 1 < i < N_x - 1, \\ g_2(t, x_{N_x}, y_j, z_k), & i = N_x - 1, \end{cases}$$

$$\mathbf{U}_{y2} = \frac{D}{h_y^2}(\nu^y_{i,j,k}), \nu^y_{i,j,k} = \begin{cases} g_2(t, x_i, y_0, z_k), & j = 1, \\ 0, & 1 < j < N_y - 1, \\ g_2(t, x_i, y_{N_y}, z_k), & j = N_y - 1, \end{cases}$$

$$\mathbf{U}_{z2} = \frac{D}{h_z^2}(\nu^z_{i,j,k}), \nu^z_{i,j,k} = \begin{cases} g_2(t, x_i, y_j, z_0), & k = 1, \\ 0, & 1 < k < N_z - 1, \\ g_2(t, x_i, y_j, z_{N_z}) & k = N_z - 1. \end{cases}$$

With a similar analysis like two dimensional case, we can write the semi-discretization of (40) as the following compact representation,

$$\begin{aligned} \mathbf{U}_t \;=\; & -\mathbf{A}\textcircled{x}\mathbf{A}\textcircled{x}\mathbf{U} - \mathbf{B}\textcircled{y}\mathbf{B}\textcircled{y}\mathbf{U} - \mathbf{C}\textcircled{z}\mathbf{C}\textcircled{z}\mathbf{U} - 2\mathbf{A}\textcircled{x}\mathbf{B}\textcircled{y}\mathbf{U} \\ & -2\mathbf{B}\textcircled{y}\mathbf{C}\textcircled{z}\mathbf{U} - 2\mathbf{C}\textcircled{z}\mathbf{A}\textcircled{x}\mathbf{U} + \mathbf{E} + \mathcal{F}(\mathbf{U}), \end{aligned} \tag{41}$$

where the operations $\textcircled{x}$, $\textcircled{y}$ and $\textcircled{z}$ are defined as

$$\begin{aligned} (\mathbf{A}\textcircled{x}\mathbf{U})_{i,j,k} &= \sum_{l=1}^{N_x-1} (\mathbf{A})_{i,l} u_{l,j,k}, \\ (\mathbf{B}\textcircled{y}\mathbf{U})_{i,j,k} &= \sum_{l=1}^{N_y-1} (\mathbf{B})_{j,l} u_{i,l,k}, \\ (\mathbf{C}\textcircled{z}\mathbf{U})_{i,j,k} &= \sum_{l=1}^{N_z-1} (\mathbf{C})_{k,l} u_{i,j,l}, \end{aligned} \tag{42}$$

and the three-dimensional array $\mathbf{E}$ is defined correspondingly from the given inhomogeneous Dirichlet boundary conditions as

$$\begin{aligned}\mathbf{E} \quad = \quad & -\mathbf{A}\textcircled{x}(\mathbf{U}_{x1}+\mathbf{U}_{y1}+\mathbf{U}_{z1})-\mathbf{U}_{x2}-\mathbf{B}\textcircled{y}(\mathbf{U}_{x1}+\mathbf{U}_{y1}+\mathbf{U}_{z1})-\mathbf{U}_{y2}\\ & -\mathbf{C}\textcircled{z}(\mathbf{U}_{x1}+\mathbf{U}_{y1}+\mathbf{U}_{z1})-\mathbf{U}_{z2}. \end{aligned}\tag{43}$$

Once again we observe that $\mathbf{C}$ is diagonalizable with $\mathbf{C}=\mathbf{P}_z\tilde{\mathbf{D}}_z{\mathbf{P}_z}^{-1}$ where the diagonal matrix $\tilde{\mathbf{D}}_z=\mathrm{diag}[d_1^z,d_2^z,\cdots,d_{N_z-1}^z]$ with d_i^z being the eigenvalues of $\mathbf{C}$. Following the similar analysis in two dimensions, the compact implicit integration factor (cIIF) scheme in three dimensions is given as below,

$$\begin{aligned}\mathbf{U}_{n+1} \quad = \quad & \mathbf{P}_z\textcircled{z}\mathbf{P}_y\textcircled{y}\mathbf{P}_x\textcircled{x}\Big((\mathbf{P}_z^{-1}\textcircled{z}\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}\mathbf{U}_n)\odot(e^*)^{-\mathbf{H}\Delta t}\\ & +\sum_{s=-1}^{r_1-1}(\mathbf{P}_z^{-1}\textcircled{z}\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}\mathbf{E}_{n-s})\odot\mathbf{S}_{r_1,s}\\ & +\Delta t\sum_{s=0}^{r_2-1}\beta^{r_2,s}(\mathbf{P}_z^{-1}\textcircled{z}\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}\mathcal{F}(\mathbf{U}_{n-s}))\odot(e^*)^{-(s+1)\mathbf{H}\Delta t}\Big)\\ & +\Delta t\beta^{r_2,-1}\mathcal{F}(\mathbf{U}_{n+1}).\end{aligned}\tag{44}$$

where $\mathbf{H}=(h_{i,j,k})$ (with a dimension of $(N_x-1)\times(N_y-1)\times(N_z-1)$) with $h_{i,j,k}=(d_i^x+d_j^y+d_k^z)^2$ and $\mathbf{S}^W_{r_1,s}=(\alpha_{i,j,k}^{(r_1,s)})$ (with a dimension of $(N_x-1)\times(N_y-1)$ $\times(N_z-1)$) are defined as a three dimensional version of (19).

Remark 4. The overall cost of the proposed cIIF scheme in three dimensions similarly can be reduced from $O(N^4)$ to $O(N^3\log_2 N)$ where $N=\max\{N_x,N_y,N_z\}$ per time step by using FFT-based fast calculations. The overall storages of the proposed cIIF scheme in three dimensions are in the order of $O(N_x^2)$, $O(N_y^2)$, $O(N_z^2)$ or $O(N_xN_yN_z)$.

4. Numerical experiments.

In this section we will demonstrate effectiveness and accuracy of the proposed cIIF scheme through numerical examples. In all simulations the differences between the numerical solutions and the exact solutions at the final time T are measured by the L_2 and L_∞ errors.

Example 1. In this example we test the cIIF scheme in two dimensions using a simple linear equation, which takes the following form

$$\begin{cases}\dfrac{\partial u}{\partial t}=-\dfrac{3}{5\pi^4}\Delta^2u+5u, & (x,y)\in\Omega,t\in[0,T],\\ u|_{t=0}=\sin\pi(x-\frac{1}{4})\cos 2\pi(y-\frac{1}{8}), & (x,y)\in\Omega,\end{cases}\tag{45}$$

where $\Omega=[-1,1]\times[-1,1]$. The exact solution is given by

$$u(x,y,t)=e^{-10t}\sin\pi\Big(x-\frac{1}{4}\Big)\cos 2\pi\Big(y-\frac{1}{8}\Big).\tag{46}$$

The Dirichlet or Neumann boundary conditions are derived accordingly from the exact solution. Note that the exact solution (46) automatically satisfies the periodic boundary conditions. We set the final time $T=1$.

First we test the cIIF scheme with $r_1=r_2=1$ as depicted in (31) which is theoretically second-order accurate in both space and time. We run the experiments at five different spatial and temporal resolutions. Numerical results are reported Tables 1 (for Dirichlet, Neumann and Periodic boundary conditions). As expected we can clearly see the second-order convergence in both space and time for all cases.

To test the cIIF scheme with $r_1 = r_2 = 2$ which is third-order accurate in time, we fix a uniform fine spatial grid with $N_x = N_y = 2048$, and conduct a series of simulations by varying time step sizes. Numerical results are reported in the bottom parts of Tables 1, higher-order accuracy (slightly smaller than 3) in time are clearly observed.

$(N_x \times N_y) \times N_t$	L_2 Error	Order	L_∞ Error	Order	CPU (s)
Dirichlet Boundary Condition					
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$					
$(16^2) \times 16$	1.22e-01	-	1.27e-01	-	0.05
$(32^2) \times 32$	2.81e-02	2.12	2.92e-02	2.12	0.11
$(64^2) \times 64$	6.89e-03	2.03	7.14e-03	2.03	0.26
$(128^2) \times 128$	1.71e-03	2.01	1.78e-03	2.00	1.29
$(256^2) \times 256$	4.28e-04	2.00	4.43e-04	2.01	11.13
Accuracy test of the cIIF scheme with $r_1 = r_2 = 2$					
$(2048^2) \times 8$	1.38e-03	-	1.48e-03	-	94.44
$(2048^2) \times 16$	2.12e-04	2.70	2.25e-04	2.72	183.85
$(2048^2) \times 32$	2.95e-05	2.85	3.12e-05	2.85	355.14
$(2048^2) \times 64$	3.80e-06	2.96	4.05e-06	2.95	708.76
$(2048^2) \times 128$	4.03e-07	3.24	4.81e-07	3.07	1414.69
Neumann Boundary Condition					
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$					
$(16^2) \times 16$	1.39e+00	-	1.89e+00	-	0.04
$(32^2) \times 32$	3.38e-01	2.04	4.61e-01	2.04	0.06
$(64^2) \times 64$	8.41e-02	2.01	1.15e-01	2.00	0.22
$(128^2) \times 128$	2.10e-02	2.00	2.86e-02	2.01	1.09
$(256^2) \times 256$	5.25e-03	2.00	7.15e-03	2.00	13.1
Accuracy test of the cIIF scheme with $r_1 = r_2 = 2$					
$(2048^2) \times 8$	8.00e-03	-	1.15e-02	-	90.61
$(2048^2) \times 16$	1.23e-03	2.70	1.77e-03	2.70	178.74
$(2048^2) \times 32$	1.72e-04	2.84	2.48e-04	2.84	354.97
$(2048^2) \times 64$	2.31e-05	2.90	3.32e-05	2.90	708.32
$(2048^2) \times 128$	3.33e-06	2.79	4.74e-06	2.81	1424.91
Periodic Boundary Condition					
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$					
$(16^2) \times 16$	1.11e-04	-	1.09e-04	-	0.02
$(32^2) \times 32$	1.70e-05	2.71	1.70e-05	2.68	0.03
$(64^2) \times 64$	3.79e-06	2.17	3.79e-06	2.17	0.14
$(128^2) \times 128$	9.22e-07	2.04	9.22e-07	2.04	0.73
$(256^2) \times 256$	2.29e-07	2.01	2.29e-07	2.01	8.44
Accuracy test of the cIIF scheme with $r_1 = r_2 = 2$					
$(2048^2) \times 8$	4.89e-06	-	4.89e-06	-	71.94
$(2048^2) \times 16$	5.46e-07	3.16	5.46e-07	3.16	139.07
$(2048^2) \times 32$	6.50e-08	3.07	6.50e-08	3.07	277.18
$(2048^2) \times 64$	8.96e-09	2.86	8.96e-09	2.86	561.06
$(2048^2) \times 128$	1.39e-09	2.69	1.39e-09	2.69	1123.69

TABLE 1. Errors and convergence rates at the final time $T = 1$ of Example 1 (2D linear case) with inhomogeneous *Dirichlet boundary condition*, *Neumann boundary condition* and *Periodic boundary condition* respectively. The units of CPU times are in seconds (s).

Remark 5. If explicit methods such as the standard Runge-Kutta methods are applied to solve (45), extremely small time steps (proportional to $\min(h_x^4, h_y^4)$) are generally needed in order to satisfy the stability condition [9]. We have checked all the simulations in Table 1 using the second order Runge-Kutta method with the same spatial grids and time steps, none of the simulation converges due to severe stability requirement of Runge-Kutta methods. For instance, for a typical simulation in Table 1 with Dirichlet boundary condition with $N_x \times N_y = 256^2$ and $r_1 = r_2 = 1$, the second order Runge-Kutta method takes 152.34 CPU seconds in order to achieve the same accuracy as cIIF, while due to its efficiency and excellent stability condition, the proposed cIIF method only takes 11.13 seconds $(13 - 14$ times faster).

Example 2. Next we consider a similar linear problem in three dimensions,

$$\begin{cases} \dfrac{\partial u}{\partial t} = -\Delta^2 u + 12\pi^4 u, & (x,y,z) \in \Omega, t \in [0,T], \\ u|_{t=0} = \sin \pi(x - \frac{1}{4}) \cos 2\pi(y - \frac{1}{8}) \sin \pi z, & (x,y,z) \in \Omega, \end{cases} \tag{47}$$

where $\Omega = [-1,1] \times [-1,1] \times [-1,1]$ and $T = \frac{5}{12\pi^4}$. The exact solution of the above equation takes the following form,

$$u(x,y,z,t) = e^{-24\pi^4 t} \sin \pi \left(x - \frac{1}{4}\right) \cos 2\pi \left(y - \frac{1}{8}\right) \sin \pi z. \tag{48}$$

The three boundary conditions are again imposed correspondingly from the exact solution.

We first apply the second-order cIIF scheme as presented in (44) to solve the equation (47). Similar to the two dimensional case, the second-order accuracy in both time and space can be observed (see Tables 2). Due to high demanding computational cost and storage in three dimensions, we choose a uniform relatively fine spatial grid $N_x = N_y = N_z = 128$ to test the third-order (in time) cIIF scheme in time discretization. Since the order accuracy will be compromised between space (second-order) and time (third-order), a high order accuracy (bigger than 2 but less than 3) is again observed as expected.

Example 3. For this example, we consider a bi-Laplace operator with a nonlinear interaction term in two dimensions,

$$\begin{cases} \dfrac{\partial u}{\partial t} = -\dfrac{1}{40\pi^2} \Delta^2 u - \dfrac{\sin^2 u}{10}, & (x,y) \in \Omega, t \in [0,T], \\ u|_{t=0} = \sin \pi(x - \frac{1}{4}) \sin \pi(y - \frac{1}{8}), & (x,y) \in \Omega, \end{cases} \tag{49}$$

where $\Omega = [-1,1] \times [-1,1]$ and $T = 1$. Inhomogeneous Dirichlet or Neumann boundary conditions are imposed along the time evolution. For instance, the inhomogeneous Dirichlet boundary conditions are given by

$$\begin{cases} u = e^{-t} \sin \pi(x - \frac{1}{4}) \sin \pi(y - \frac{1}{8}), \\ \Delta u = -2\pi^2 e^{-t} \sin \pi(x - \frac{1}{4}) \sin \pi(y - \frac{1}{8}), \end{cases}$$

and the inhomogeneous Neumann boundary conditions are given by

$$\begin{cases} \dfrac{\partial u}{\partial x} = \pi e^{-t} \cos \pi(x - \frac{1}{4}) \sin \pi(y - \frac{1}{8}), \dfrac{\partial u}{\partial y} = \pi e^{-t} \sin \pi(x - \frac{1}{4}) \cos \pi(y - \frac{1}{8}), \\ \dfrac{\partial \Delta u}{\partial x} = -2\pi^3 e^{-t} \cos \pi(x - \frac{1}{4}) \sin \pi(y - \frac{1}{8}), \\ \qquad\qquad \dfrac{\partial \Delta u}{\partial y} = -2\pi^3 e^{-t} \sin \pi(x - \frac{1}{4}) \cos \pi(y - \frac{1}{8}), \end{cases}$$

$(N_x \times N_y \times N_z) \times N_t$	L_2 Error	Order	L_∞ Error	Order	CPU (s)
Dirichlet Boundary Condition					
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$					
$(16^3) \times 16$	6.61e-02	-	4.88e-02	-	0.48
$(32^3) \times 32$	1.52e-02	2.12	1.13e-02	2.11	4.89
$(64^3) \times 64$	3.71e-03	2.03	2.76e-03	2.03	95.00
$(128^3) \times 128$	9.22e-04	2.01	6.86e-04	2.01	3185.65
Accuracy test of the cIIF scheme with $r_1 = r_2 = 2$					
$(128^3) \times 4$	4.67e-03	-	4.37e-03	-	98.19
$(128^3) \times 8$	8.39e-04	2.48	7.15e-04	2.61	188.35
$(128^3) \times 16$	1.14e-04	2.88	1.00e-04	2.84	368.35
$(128^3) \times 32$	1.46e-05	2.96	1.32e-05	2.92	727.60
Neumann Boundary Condition					
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$					
$(16^3) \times 16$	1.42e+00	-	1.34e+00	-	0.51
$(32^3) \times 32$	3.52e-01	2.01	3.34e-01	2.00	5.95
$(64^3) \times 64$	8.77e-02	2.00	8.33e-02	2.00	140.31
$(128^3) \times 128$	2.19e-02	2.00	2.08e-02	2.00	2943.77
Accuracy test of the cIIF scheme with $r_1 = r_2 = 2$					
$(128^3) \times 4$	5.01e-02	-	5.20e-02	-	105.54
$(128^3) \times 8$	9.36e-03	2.42	9.64e-03	2.43	201.26
$(128^3) \times 16$	1.53e-03	2.61	1.56e-03	2.63	392.71
$(128^3) \times 32$	3.05e-04	2.37	2.97e-04	2.39	776.63
Periodic Boundary Condition					
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$					
$(16^3) \times 16$	1.24e-04	-	8.64e-05	-	0.20
$(32^3) \times 32$	2.05e-05	.60	1.44e-05	2.60	2.89
$(64^3) \times 64$	4.64e-06	2.14	3.28e-06	2.13	86.97
$(128^3) \times 128$	1.13e-06	2.04	8.01e-07	2.03	3916.57
Accuracy test of the cIIF scheme with $r_1 = r_2 = 2$					
$(128^3) \times 4$	7.34e-06	-	5.19e-06	-	113.15
$(128^3) \times 8$	1.16e-06	2.66	8.22e-07	2.66	221.40
$(128^3) \times 16$	1.78e-07	2.70	1.38e-07	2.57	478.65
$(128^3) \times 32$	2.96e-08	2.59	2.56e-08	2.43	917.23

TABLE 2. Errors and convergence rates at the final time $T = \frac{5}{12\pi^4}$ of Example 2 (3D linear case) with inhomogeneous *Dirichlet boundary condition*, *Neumann boundary condition* and *Periodic boundary condition* respectively. The units of CPU times are in seconds (s).

for $(x, y) \in \partial\Omega, t \in [0, T]$.

As an example, the numerical solutions of the equation (49) with Dirichlet and Neumann boundary conditions at the final time $T = 1$ obtained by using the second-order cIIF scheme on a grid of size $(N_x \times N_y) \times N_t = (256^2) \times 128$ are plotted in Figure 2. For this case, we check order of accuracy in time discretization by a numerical resolution study. Since the exact solution is not known for the problem (49), the approximate solution by using the third-order (in time) cIIF scheme on a uniform fine grid 2048×2048 along with a very small time step $\Delta t = 10^{-4}$, is considered as the "exact" solution. The L^2 and L^∞ errors are measured between numerical solutions with such "exact" solution. The results of errors and convergence rates in time discretization can be found from Table 3, which coincide with our expectation very well.

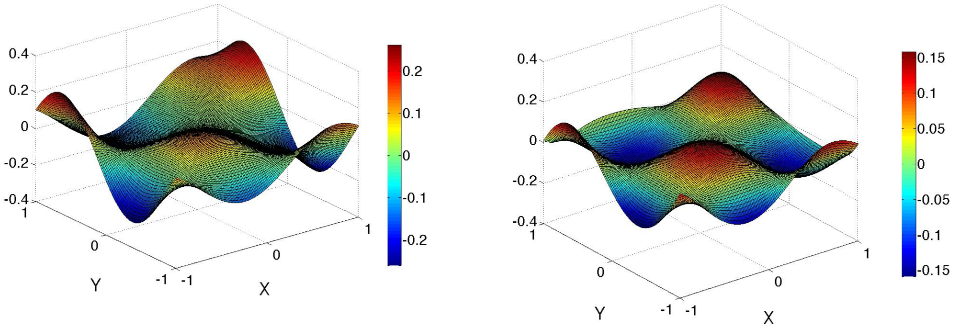

FIGURE 2. Plot of the numerical solutions u of Example 3 with *Dirichlet boundary condition* (left) and *Neumann boundary condition* (right) at the final time $T=1$ that are obtained by using the second-order cIIF scheme on a grid of size $(N_x\times N_y)\times N_t=(256^2)\times 128$.

Example 4. In this example, we consider a system of bi-Laplace equations with nonlinear interactios in two dimensions,

$$\begin{cases}\dfrac{\partial u}{\partial t}=-\dfrac{1}{10\pi^2}\Delta^2 u-\dfrac{\sin^2 u\cos v}{2}, & (x,y)\in\Omega, t\in[0,T],\\ \dfrac{\partial v}{\partial t}=-\dfrac{1}{5\pi^2}\Delta^2 v-\dfrac{\cos^2 v\sin u}{2}, & (x,y)\in\Omega, t\in[0,T],\\ u|_{t=0}=\sin\pi(x-\frac12)\cos\pi(y-\frac14), & (x,y)\in\Omega,\\ v|_{t=0}=\cos\pi(x-\frac12)\sin\pi(y-\frac14), & (x,y)\in\Omega,\end{cases}\tag{50}$$

where $\Omega=[-1,1]\times[-1,1]$ and $T=1$. Inhomogeneous Dirichlet or Neumann boundary conditions are imposed along the time evolution. For instance, the inhomogeneous Dirichlet boundary conditions are given by

$$\begin{cases}u=e^{-2t}\sin\pi(x-\frac12)\cos\pi(y-\frac14), & v=e^{-2t}\cos\pi(x-\frac12)\sin\pi(y-\frac14),\\ \Delta u=-2\pi^2e^{-2t}\sin\pi(x-\frac12)\cos\pi(y-\frac14), & \Delta v=-2\pi^2e^{-2t}\cos\pi(x-\frac12)\sin\pi(y-\frac14),\end{cases}$$

and the inhomogeneous Neumann boundary conditions are given by

$$\begin{cases}\dfrac{\partial u}{\partial x}=\pi e^{-2t}\cos\pi(x-\frac12)\cos\pi(y-\frac14), \dfrac{\partial u}{\partial y}=-\pi e^{-2t}\sin\pi(x-\frac12)\sin\pi(y-\frac14),\\ \dfrac{\partial\Delta u}{\partial x}=-2\pi^3e^{-2t}\cos\pi(x-\frac12)\cos\pi(y-\frac14), \dfrac{\partial\Delta u}{\partial y}=2\pi^3e^{-2t}\sin\pi(x-\frac12)\sin\pi(y-\frac14),\\ \dfrac{\partial v}{\partial x}=-\pi e^{-2t}\sin\pi(x-\frac12)\sin\pi(y-\frac14), \dfrac{\partial v}{\partial y}=\pi e^{-2t}\cos\pi(x-\frac12)\cos\pi(y-\frac14),\\ \dfrac{\partial\Delta v}{\partial x}=2\pi^3e^{-2t}\sin\pi(x-\frac12)\sin\pi(y-\frac14), \dfrac{\partial\Delta v}{\partial y}=-2\pi^3e^{-2t}\cos\pi(x-\frac12)\cos\pi(y-\frac14),\end{cases}$$

for $(x,y)\in\partial\Omega, t\in[0,T]$. For this example, we check the order of accuracy in time discretization with the second order cIIF scheme by a numerical resolution study. Since the exact solution is not known for the problem (50), once again the approximate solution by using the third-order (in time) cIIF scheme on a uniform fine grid 2048×2048 along with a very small time step $\Delta t=10^{-4}$, is considered as the "exact" solution. Both L^2 and L^∞ errors are measured between numerical

$(N_x \times N_y) \times N_t$	L_2 Error	Order	L_∞ Error	Order
Dirichlet Boundary Condition				
Accuracy test of the cIIF scheme with $r_1 = r_2 = 0$				
$(2048^2) \times 16$	2.19e+00	-	8.06e+00	-
$(2048^2) \times 32$	1.32e+00	0.73	4.44e+00	0.86
$(2048^2) \times 64$	6.21e-01	1.09	2.07e+00	1.10
$(2048^2) \times 128$	3.42e-01	0.94	1.05e+00	0.98
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$				
$(2048^2) \times 16$	3.82e+00	-	4.16e+00	-
$(2048^2) \times 32$	9.85e-01	1.96	1.17e+00	1.83
$(2048^2) \times 64$	2.43e-01	2.02	2.73e-01	2.10
$(2048^2) \times 128$	5.81e-02	2.06	6.90e-02	1.98
Accuracy test of the cIIF scheme with $r_1 = r_2 = 2$				
$(2048^2) \times 16$	3.71e+00	-	5.34e+00	-
$(2048^2) \times 32$	5.13e-01	2.85	7.09e-01	2.91
$(2048^2) \times 64$	7.12e-02	2.85	8.83e-02	3.01
$(2048^2) \times 128$	9.46e-03	2.91	1.13e-02	2.97
Neumann Boundary Condition				
Accuracy test of the cIIF scheme with $r_1 = r_2 = 0$				
$(2048^2) \times 16$	2.06e+00	-	6.73e+00	-
$(2048^2) \times 32$	1.05e+00	0.97	3.35e+00	1.01
$(2048^2) \times 64$	5.15e-01	1.03	1.58e+00	1.08
$(2048^2) \times 128$	2.57e-01	1.00	7.37e-01	1.10
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$				
$(2048^2) \times 16$	1.63e+00	-	1.74e+00	-
$(2048^2) \times 32$	4.14e-01	2.01	4.35e-01	2.01
$(2048^2) \times 64$	1.06e-01	2.00	1.21e-01	1.98
$(2048^2) \times 128$	2.56e-02	2.00	2.96e-02	1.98
Accuracy test of the cIIF scheme with $r_1 = r_2 = 2$				
$(2048^2) \times 16$	1.23e+00	-	2.32e+00	-
$(2048^2) \times 32$	1.73e-01	2.83	3.34e-01	2.80
$(2048^2) \times 64$	2.26e-02	2.94	4.29e-02	2.96
$(2048^2) \times 128$	3.03e-03	2.90	5.50e-03	2.96

TABLE 3. Errors and convergence rates at the final time T of Example 3 (2D semilinear case) with *Dirichlet boundary condition* and *Neumann boundary condition* respectively.

solutions with such "exact" solution. From Table 4, the second order convergence rate can be again observed as expected.

5. **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. Unlike the non-compact form, it is difficult to apply cIIF *directly* to handle diffusion terms with cross derivatives. In this paper, we have developed a cIIF method for solving a class of semilinear fourth-order parabolic equations. Because of such representation, computing exponentials of large matrices is reduced to the calculation of exponentials of matrices of significantly smaller sizes, in which the dimension of discretized matrices is the same as one dimensional case and the stability condition and computational savings and storage are similar to the original cIIF for

$(N_x \times N_y) \times N_t$	L_2 Error	Order	L_∞ Error	Order
Dirichlet Boundary Condition				
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$				
$(2048^2) \times 16$	4.52e+00	-	4.96e+00	-
$(2048^2) \times 32$	1.24e+00	1.87	1.35e+00	1.88
$(2048^2) \times 64$	3.01e-01	2.04	3.22e-01	2.02
$(2048^2) \times 128$	7.48e-02	2.01	8.24e-02	2.01
Neumann Boundary Condition				
Accuracy test of the cIIF scheme with $r_1 = r_2 = 1$				
$(2048^2) \times 16$	2.45e+00	-	2.68e+00	-
$(2048^2) \times 32$	6.11e-01	2.00	6.69e-01	2.00
$(2048^2) \times 64$	1.51e-01	2.02	1.71e-01	1.97
$(2048^2) \times 128$	3.72e-02	2.02	4.31e-02	1.99

Table 4. Errors and convergence rates at the final time $T = 1$ of Example 4 (2D system case) with *Dirichlet boundary condition* and *Neumann boundary condition* respectively.

the second-order problem. In addition, the direct and explicit incorporation of inhomogeneous boundary conditions into the cIIF is also proposed.

Although compact representation has been presented only in the context of implicit integration factor methods for semilinear fourth-order parabolic equations, such approach can easily be applied to other integration factor or exponential difference methods. Other type of equations of high-order derivatives, (e.g. Cahn-Hilliard equations of fourth-order derivatives) may also potentially be handled using the approach for better efficiency. To better deal with high spatial dimensions, one may incorporate the sparse grid [23] into the compact representation technique. The flexibility of compact representation allows either direct calculation of the exponentials of matrices or using Krylov subspace [4, 10, 22] for non-constant diffusion coefficients to compute their exponential matrix-vector multiplications for saving further in storages and 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 [8, 11, 17] or spectral methods [3, 24]. 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 to its effectiveness in implementation and good stability conditions, the method is very desirable to be incorporated with local adaptive mesh refinement [1, 2, 18], which will also be further explored in the future work.

REFERENCES

[1] M. Berger and P. Colella, Local adaptive mesh refinement for shock hydrodynamics, *Journal of Computational Physics*, **82** (1989), 64–84.

[2] M. Berger and J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, *Journal of Computational Physics*, **53** (1984), 484–512.

[3] E. O. Brigham, *The Fast Fourier Transform and its Applications*, Prentice Hall, 1988.

[4] S. Chen and Y.-T. Zhang, Krylov implicit integration factor methods for spatial discretization on high dimensional unstructured meshes: Application to discontinuous Galerkin methods, *Journal of Computational Physics*, **230** (2011), 4336–4352.

[5] S. M. Cox and P. C. Matthews, Exponential time differencing for stiff systems, *Journal of Computational Physics*, **176** (2002), 430–455.

[6] Q. Du and W. Zhu, Stability analysis and applications of the exponential time differencing schemes, *Journal of Computational Mathematics*, **22** (2004), 200–209.

[7] Q. Du and W. Zhu, Modified exponential time differencing schemes: Analysis and applications, *BIT Numerical Mathematics*, **45** (2005), 307–328.

[8] R. Eymard, T. Gallouët and R. Herbin, Finite volume methods, *Handbook of Numerical Analysis*, **7** (2000), 713–1020.

[9] B. Gustafsson, H.-O. Kreiss and J. Oliger, *Time Dependent Problems and Difference Methods*, volume 67. Wiley New York, 1995.

[10] M. Hochbruck and C. Lubich, On krylov subspace approximations to the matrix exponential operator, *SIAM Journal on Numerical Analysis*, **34** (1997), 1911–1925.

[11] A. Jameson, W. Schmidt and E. Turkel, *Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes*, The 14th AIAA Fluid and Plasma Dynamics Conference, 1981.

[12] G.-S. Jiang and C.-W. Shu, Efficient implementation of weighted ENO schemes, *Journal of Computational Physics*, **126** (1996), 202–228.

[13] L. Ju, J. Zhang, L. Zhu and Q. Du, *Fast Explicit Integration Factor Methods for Semilinear Parabolic Equations*, Journal of Scientific Computing, 2014.

[14] A.-K. Kassam and L. N. Trefethen, Fourth-order time stepping for stiff PDEs, *SIAM Journal on Scientific Computing*, **26** (2005), 1214–1233.

[15] B. Kleefeld, A. Khaliq and B. Wade, An ETD Crank-Nicolson method for reaction-diffusion systems, *Numerical Methods for Partial Differential Equations*, **28** (2012), 1309–1335.

[16] S. Krogstad, Generalized integrating factor methods for stiff PDEs, *Journal of Computational Physics*, **203** (2005), 72–88.

[17] R. LeVeque, *Numerical Methods for Conservation Laws*, Birkhauser, 1992.

[18] X. Liu and Q. Nie, Compact integration factor methods for complex domains and adaptive mesh refinement, *Journal of computational physics*, **229** (2010), 5692–5706.

[19] X.-D. Liu, S. Osher and T. Chan, Weighted essentially non-oscillatory schemes, *Journal of Computational Physics*, **115** (1994), 200–212.

[20] Q. Nie, F. Wan, Y.-T. Zhang and X. Liu, Compact integration factor methods in high spatial dimensions, *Journal of Computational Physics*, **277** (2008), 5238–5255.

[21] Q. Nie, Y.-T. Zhang and R. Zhao, Efficient semi-implicit schemes for stiff systems, *Journal of Computational Physics*, **214** (2006), 521–537.

[22] Y. Saad, Analysis of some krylov subspace approximations to the matrix exponential operator, *SIAM Journal on Numerical Analysis*, **29** (1992), 209–228.

[23] J. Shen and H. Yu, Efficient spectral sparse grid methods and applications to high-dimensional elliptic problems, *SIAM Journal on Scientific Computing*, **32** (2010), 3228–3250.

[24] C. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, volume 10. SIAM, 1992.

[25] A. Wiegmann, *Fast Poisson, Fast Helmholtz and Fast Linear Elastostatic Solvers on Rectangular Parallelepipeds*, Lawrence Berkeley National Laboratory, Paper LBNL-43565, 1999.

[26] S. Zhao, J. Ovadia, X. Liu, Y. Zhang and Q. Nie, Operator splitting implicit integration factor methods for stiff reaction–diffusion–advection systems, *Journal of Computational Physics*, **230** (2011), 5996–6009.

Received November 2013; revised January 2014.

E-mail address: ju@math.sc.edu
E-mail address: xfliu@math.sc.edu
E-mail address: wleng@lsec.cc.ac.cn