Journal of Scientific Computing
62 (2), 431-455, 2015

Fast Explicit Integration Factor Methods for Semilinear Parabolic Equations

Lili Ju · Jian Zhang · Liyong Zhu · Qiang Du

Received: date / Accepted: date

Abstract In this paper, an explicit numerical method and its fast implementation are proposed and discussed for the solution of a wide class of semilinear parabolic equations including the Allen-Cahn equation as a special case. The method combines efficient decompositions of compact spatial difference operators on a regular mesh with stable and accurate exponential time integrators. It can deal with stiff nonlinearity and both homogeneous and inhomogeneous boundary conditions of different types based on multistep approximations and analytic evaluations of time integrals. Numerical experiments demonstrate effectiveness of the new method for both linear and nonlinear model problems.

Keywords Integration factor method · explicit scheme · multistep approximation · discrete fast transforms · diffusion-reaction equation · Allen-Cahn equation

Mathematics Subject Classification (2000) 65M06 · 65M22 · 65Y20

1 Introduction

Let Ω be an open rectangular domain in $\mathbb{R}^d$ and $T > 0$. We consider the numerical solution of semi-linear parabolic equations of the following form:

$$\frac{\partial u}{\partial t} = D\Delta u - f(u), \qquad \mathbf{x} \in \Omega, t \in [t_0, t_0 + T], \tag{1}$$

where D is the diffusion coefficient. Equations of the above form are of broad interests as they can represent mathematical models in various applications. For example, they can represent scalar diffusion-reaction equations in chemical reactions and population dynamics; in phase transition modeling, they

L. Ju's research is partially supported by the US National Science Foundation under grant number DMS-1215659 and the U.S. Department of Energy under grant number DE-SC0008087-ER65393. J. Zhang's research is partially supported by the Natural Science Foundation of China under grant numbers 11271350 and 91130019. L. Zhu's research is partially supported by the Natural Science Foundation of China under grant number 91130019, ISTCP of China under grant number 2010DFR00700, China Fundamental Research of Civil Aircraft under grant number MJ-F-2012-04, and the State Key Laboratory of Software Development Environment under grant number SKLSDE-2014ZX-03. Q. Du's research is partially supported by the US National Science Foundation under grant number DMS-1318586.

L. Ju
Department of Mathematics, University of South Carolina, Columbia, SC 29208, USA.
E-mail: ju@math.sc.edu

J. Zhang
Computer Network Information Center, Chinese Academy of Sciences, Beijing 100080, China.
E-mail: zhangjian@sccas.cn

L. Zhu
School of Mathematics and Systems Science, Beihang University, Beijing 100191, China
E-mail: liyongzhu@buaa.edu.cn

Q. Du
Department of Mathematics, Pennsylvania State University, University Park, PA 16802, USA.
E-mail: qdu@math.psu.edu

are often called Allen-Cahn equations [1] or Ginzburg-Landau equations with $f(u)$ being a double-well potential. To solve (1), the equation is accompanied by the initial condition $u|_{t=t_0} = u_0$ for $\mathbf{x} \in \Omega$ and various boundary conditions such as Dirichlet, periodic or Neumann boundary conditions. The equation can be viewed as the L^2 gradient flow of a free energy defined by

$$\mathcal{E}(u) = \int_{\Omega} \left\{ \frac{1}{2} |\nabla u|^2 + \frac{1}{D} F(u) \right\} d\mathbf{x}, \tag{2}$$

where $F'(u) = f(u)$ so that the energy is a decreasing function of time.

In this work, we present a numerical method for solving (1) that uses the standard central difference for spatial discretization, coupled with exponential integration factor (IF) methods and multistep approximations of stiff nonlinear terms for time discretization. Though these techniques have been well-studied respectively in the literature, we provide new contribution here to address how to integrate them seamlessly along with detailed discussions on their efficient implementation involving fast evaluation of matrix functions, combining with splitting techniques to improve stability, and a stable and accurate incorporation of inhomogeneous boundary conditions. The resulting schemes have all the essential features of explicit schemes in terms of simple implementation and fast solution, yet they are much better stable for large time steps and have higher order accuracy. We demonstrate the efficiency and accuracy of the resulting discretization scheme based on extensive numerical benchmark tests with both the standard linear diffusion equation (heat equation) and the nonlinear Allen-Cahn equation with $f(u) = \frac{D}{\epsilon^2}(u^3 - u)$ as illustrative examples.

To elaborate further, we note first that the numerical solution of equations in the form of (1) has grown into a research topic too large to be comprehensively reviewed here. While a number of spatial discretization schemes can be considered, the choice of using the standard second-order central difference approximation for the spatial Laplacian operator [9] is taken here mainly for its simplicity and compactness, especially for the special domain geometry. The compact central difference stencil provides a simple matrix structure that leads to fast matrix function evaluations such as FFT-based algorithms. As for time integration, the use of exponential integrators, including methods like exponential time difference and integration factors, has also attracted much attention [23,10,3,6,13,17–19]. A recent review and additional references on the subject can be found in [11]. Combining compact representation of spatial discretization with the integration factor approach is a natural and important idea, which was utilized for the first time in [17]. Implicit discretization schemes have been constructed there based on dense matrix-matrix multiplications and point-wise solution of decoupled nonlinear systems which can handle parabolic problems with non-stiff nonlinear terms. Our work is motivated in part by [17] and we adopt some of its notation here. To our knowledge, there is no careful study on the explicit but stable and accurate treatment of stiff nonlinearity nor direct and explicit incorporation of possibly time-dependent inhomogeneous boundary conditions in these numerical schemes. Our study here fills the gap. Moreover, we develop multistep approximations and use analytical evaluations for the time integrals resulted from the nonlinear and boundary terms. Our approach for evaluating the time integrals plays a very important role in achieving high accuracy and good stability of the proposed method which is effectively of explicit nature. For problems involving stiff nonlinearities, to avoid solving nonlinear systems while maintaining the stability in time integration, we incorporate the splitting idea into the integration factor scheme. This helps to improve stability without causing a degradation of overall accuracy due to the splitting error. The splitting technique we adopt is analogous to the splitting scheme used commonly in stabilized semi-implicit splitting methods for Allen-Cahn equations [5,6,12,22], namely, we rewrite (1) in the following form: for $\mathbf{x} \in \Omega, t \in [t_0, t_0 + T]$,

$$\frac{\partial u}{\partial t} = Lu - \hat{f}(u), \tag{3}$$

where

$$Lu = D\Delta u - \kappa u, \quad \hat{f}(u) = f(u) - \kappa u$$

with a nonnegative constant κ satisfying

$$\kappa \geq \frac{1}{2} \max \left\{ 0, \max_u f'(u) \right\} . \tag{4}$$

In addition, we describe in detail a fast implementation of the resulting discretization scheme through matrix decomposition and discrete fast transforms. Together, the method presented here provides a high order, stable and efficient computational approach to numerically solve equations of the type (1).

The rest of the paper is organized as follows: we discuss the two dimensional versions of our method in Section 2, with separate discussions for different types of boundary conditions; then in Section 3, we consider the three dimensional extension. We present some linear stability analysis of the method in Section 4 and extensive numerical results in Section 5. Finally, some conclusions are given in Section 6.

2 Fast explicit integration factor methods in two dimensions

We now derive fast explicit integration factor (fEIF) methods for the model equation (1) in two dimensions. Suppose $\Omega = \{x_b < x < x_e, y_b < y < y_e\}$. Let us discretize the spatial domain by a rectangular mesh which is uniform in each direction as follows: $(x_i, y_j) = (x_b + ih_x, y_b + jh_y)$ for $0 \le i \le N_x$ and $0 \le j \le N_y$ with $h_x = (x_e - x_b)/N_x$ and $h_y = (y_e - y_b)/N_y$. Let $u_{i,j} = u_{i,j}(t) \approx u(t, x_i, y_j)$ for $0 \le i \le N_x$ and $0 \le j \le N_y$ denote the numerical solution. We will use the second-order accurate central difference discretization scheme for the spatial derivative terms. Let us further discretize the time period as $t_n = t_0 + n\Delta t,\ n = 1, \cdots, N_t$ with $\Delta t = T/N_t$.

2.1 The problem with Dirichlet boundary conditions

We first consider the problem with a Dirichlet boundary condition

$$u = g, \qquad (x, y) \in \partial\Omega, t \in [t_0, t_0 + T]. \tag{5}$$

In this case, the set of unknowns is given as

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

Let

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

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

In addition, we define

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

and set

$$\mathbf{A} = \frac{D}{h_x^2}\mathbf{G}^D_{(N_x-1)\times(N_x-1)}, \quad \mathbf{B} = \frac{D}{h_y^2}\mathbf{G}^D_{(N_y-1)\times(N_y-1)}.$$

Define the special operators $\textcircled{x}$ and $\textcircled{y}$ as follows:

$$(\mathbf{A}\textcircled{x}\mathbf{U})_{i,j} = \sum_{l=1}^{N_x-1} (\mathbf{A})_{i,l} u_{l,j}, \qquad (\mathbf{B}\textcircled{y}\mathbf{U})_{i,j} = \sum_{l=1}^{N_y-1} (\mathbf{B})_{j,l} u_{i,l}. \tag{6}$$

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

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

Then we can write the semi-discretization of (1) in space in the following compact representation:

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

where $\mathcal{F}(\mathbf{U}) = (\hat{f}(u_{i,j}))_{(N_x-1)\times(N_y-1)}$ and $\mathcal{W} = \mathbf{U}_{x1} + \mathbf{U}_{y1}$.

2.1.1 A general time integration factor approximation

We see that $\mathbf{A}$ and $\mathbf{B}$ are obviously diagonalizable and let us assume that

$$\mathbf{A} = \mathbf{P}_x\tilde{\mathbf{D}}_x{\mathbf{P}_x}^{-1}, \quad \mathbf{B} = \mathbf{P}_y\tilde{\mathbf{D}}_y{\mathbf{P}_y}^{-1} \tag{8}$$

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

$$\tilde{\mathbf{D}}_x = \mathrm{diag}[d_1^x, d_2^x, \cdots, d_{N_x-1}^x], \quad \tilde{\mathbf{D}}_y = \mathrm{diag}[d_1^y, d_2^y, \cdots, d_{N_y-1}^y].$$

Set $\mathbf{H} = (h_{i,j})_{(N_x-1)\times(N_y-1)}$ with

$$h_{i,j} = d_i^x + d_j^y - \kappa. \tag{9}$$

Clearly we have $h_{i,j} \leq -\kappa \leq 0$.

We first define an operation"(e^*)" by t aking exponentials of an array element by element as

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

Then define another operator "$\odot$" for element by element multiplication between two arrays of same sizes as

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

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

Then, a variation of constant formula for the ODE system (7) leads to

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

We let

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

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

and

$$\mathbf{F}(t_n+\tau) = (f_{i,j}(t_n+\tau))_{(N_x-1)\times(N_y-1)} = \mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}\mathcal{F}(\mathbf{U}(t_n+\tau)),$$

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

then the above equation can be written as

$$\mathbf{U}_{n+1} = \mathbf{P}_y\textcircled{y}\mathbf{P}_x\textcircled{x}\Big((e^*)^{\mathbf{H}\mathit{\Delta} t}\odot(\mathbf{P}_y^{-1}\textcircled{y}\mathbf{P}_x^{-1}\textcircled{x}\mathbf{U}_{\mathbf{n}}) + \mathbf{Q}^W - \mathbf{Q}^F\Big). \tag{10}$$

Numerical schemes can then be developed based on the explicit and accurate evaluation of the integrals on the right-hand side of (10), that is, the terms $\mathbf{Q}^W$ and $\mathbf{Q}^F$. One key element in the design of a good quadrature is to recognize the distinct behavior of two possible approaches: one uses a polynomial approximation in time of the whole integrand, while the other approximates the terms $\mathcal{W}(t_n+\tau)$ and $\mathcal{F}(\mathbf{U}(t_n+\tau))$ (or $\mathbf{W}(t_n+\tau)$ and $\mathbf{F}(t_n+\tau)$) by a polynomial in time instead and perform the exact integration on the resulting integrands. Since $(e^*)^{\mathbf{H}(\Delta t-\tau)}$ contains entries with highly different decaying rates, $\mathcal{W}(t_n+\tau)$ involves terms on the order of $1/h_x^2$ and $1/h_y^2$, and $\mathcal{F}(\mathbf{U}(t_n+\tau))$ contains stiff nonlinear terms, it turns out that the former leads to severe numerical stability [11] issues and loss of accuracy while the latter provides a desirable alternative. On the other hand, it is reasonable to assume that the time step Δt is chosen so that $\mathbf{W}(t_n+\tau)$ and $\mathbf{F}(t_n+\tau)$ are slowly varying in τ within a few time steps, and thus are subject to good polynomial approximations in time.

2.1.2 Multistep approximations of the integrals

We adopt a multistep approach for the quadrature approximation as discussed in the above.

For evaluation of the integral $\mathbf{Q}^W$ resulted from the boundary conditions, 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 $\mathbf{W}(t_n+\tau)$ (in the spirit of Adams-Moulton, but all of which can be calculated from the given boundary condition):

$$P_{r_1}^W(t_n+\tau) = \sum_{s=-1}^{r_1-1} \omega_{r_1,s}(\tau)\mathbf{W}(t_{n-s}) \tag{11}$$

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

$$\mathbf{W}(t_n+\tau) \approx P_{r_1}^W(\tau) + O(\Delta t^{r_1+1}). \tag{12}$$

We list some of the polynomials $P_{r_1}^W(\tau)$ below:

$$\begin{aligned}
&P_0^W(\tau) = \mathbf{W}(t_{n+1});\\
&P_1^W(\tau) = \theta\mathbf{W}(t_{n+1}) + (1-\theta)\mathbf{W}(t_n);\\
&P_2^W(\tau) = \frac{1}{2}\theta(1+\theta)\mathbf{W}(t_{n+1}) + (1-\theta^2)\mathbf{W}(t_n) - \frac{1}{2}\theta(1-\theta)\mathbf{W}(t_{n-1})
\end{aligned}$$

with $\theta = \dfrac{\tau}{\Delta t}$. Thus we can approximate $q_{i,j}^W$ by

$$q_{i,j}^W \approx \sum_{s=-1}^{r_1-1} w_{i,j}(t_{n-s}) \int_0^{\Delta t} e^{h_{i,j}(\Delta t-\tau)}\omega_{r_1,s}(\tau)\, d\tau = \sum_{s=-1}^{r_1-1} w_{i,j}(t_{n-s})\alpha_{i,j}^{(r_1,s)}. \tag{13}$$

We remark that $\alpha_{i,j}^{(r_1,s)}$ is independent of the time steps $\{t_n\}$ with a uniform time step size Δt being used. The key is then to evaluate $\alpha_{i,j}^{(r_1,s)}$ exactly to avoid any loss of accuracy and numerical instability. For simplicity, we only present the values of $\{\alpha_{i,j}^{(r_1,s)}\}$ for $r_1 = 0,1,2$ below:

$$\begin{aligned}
&\alpha_{i,j}^{(0,-1)} = \phi_0;\\
&\alpha_{i,j}^{(1,-1)} = \phi_1,\ \alpha_{i,j}^{(1,0)} = \phi_0 - \phi_1;\\
&\alpha_{i,j}^{(2,-1)} = \frac{1}{2}(\phi_1+\phi_2),\ \alpha_{i,j}^{(2,0)} = \phi_0 - \phi_2,\ \alpha_{i,j}^{(2,1)} = -\frac{1}{2}(\phi_1-\phi_2)
\end{aligned}$$

where

$$\begin{cases}
\phi_0 = -\dfrac{1}{h_{i,j}}(1-e^{h_{i,j}\Delta t}),\ \phi_1 = -\dfrac{1}{h_{ij}}(1-\dfrac{\phi_0}{\Delta t}),\ \phi_2 = -\dfrac{1}{h_{ij}}(1-\dfrac{2\phi_1}{\Delta t}), & h_{i,j}\neq 0,\\
\phi_0 = \Delta t,\quad \phi_1 = \dfrac{\Delta t}{2},\quad \phi_2 = \dfrac{\Delta t}{3}, & h_{i,j} = 0.
\end{cases} \tag{14}$$

Define $\mathbf{S}^W_{r_1,s} = (\alpha^{(r_1,s)}_{i,j})_{(N_x-1)\times(N_y-1)}$, we then approximate $\mathbf{Q}^W$ as

$$\mathbf{Q}^W \approx \sum_{s=-1}^{r_1-1} \mathbf{W}_{n-s} \odot \mathbf{S}^W_{r_1,s}, \tag{15}$$

which is (r_1+1)-th order accurate.

For evaluation of the integral $\mathbf{Q}^F$ resulted from the nonlinear term, we use a similar multistep but explicit approximation, in the spirit of the Adams-Bashforth, in order to avoid nonlinear iterative solution process. We approximate $\mathbf{F}(t_n+\tau)$ using the values at $t_n, t_{n-1}, \cdots, t_{n-r_2}$. The corresponding Lagrange interpolation polynomial $P^F_{r_2}(\tau)$ of degree r_2 to approximate $\mathbf{F}(t_n+\tau)$:

$$P^F_{r_2}(t_n+\tau) = \sum_{s=0}^{r_2} \eta_{r_2,s}(\tau)\mathbf{F}(t_{n-s}), \tag{16}$$

with $\eta_{r_2,s}(\tau) = \prod\limits_{\substack{l=0\\ l\neq s}}^{r_2} \dfrac{\tau + l\Delta t}{(l-s)\Delta t}$. Then we also have

$$\mathbf{F}(t_n+\tau) \approx P^F_{r_2}(\tau) + O(\Delta t^{r_2+1}). \tag{17}$$

We list some of the polynomials $P^F_{r_2}(\tau)$ below

$$\begin{aligned}
&P^F_0(\tau) = \mathbf{F}(t_n);\\
&P^F_1(\tau) = (1+\theta)\mathbf{F}(t_n) - \theta\mathbf{F}(t_{n-1});\\
&P^F_2(\tau) = \frac{1}{2}(1+\theta)(2+\theta)\mathbf{F}(t_n) - \theta(2+\theta)\mathbf{F}(t_{n-1}) + \frac{1}{2}\theta(1+\theta)\mathbf{F}(t_{n-2}).
\end{aligned}$$

Thus we can approximate $q^F_{i,j}$ by

$$q^F_{i,j} \approx \sum_{s=0}^{r_2} f_{i,j}(t_{n-s}) \int_0^{\Delta t} e^{h_{i,j}(\Delta t-\tau)} \eta_{r_2,s}(\tau)\, d\tau = \sum_{s=0}^{r_2} f_{i,j}(t_{n-s})\beta^{(r_2,s)}_{i,j}. \tag{18}$$

We remark that $\beta^{(r_2,s)}_{i,j}$ is also independent of the time steps $\{t_n\}$ with a uniform time step size taken here. For simplicity, we only present the values of $\{\beta^{(r_2,s)}_{i,j}\}$ for $r_2 = 0, 1, 2$ below:

$$\begin{aligned}
&\beta^{(0,0)}_{i,j} = \phi_0;\\
&\beta^{(1,0)}_{i,j} = \phi_0 + \phi_1,\ \beta^{(1,1)}_{i,j} = -\phi_1;\\
&\beta^{(2,0)}_{i,j} = \frac{1}{2}(2\phi_0 + 3\phi_1 + \phi_2),\ \beta^{(2,1)}_{i,j} = -(2\phi_1+\phi_2),\ \beta^{(2,2)}_{i,j} = \frac{1}{2}(\phi_1+\phi_2);
\end{aligned}$$

where $\{\phi_i\}_{i=0}^2$ are defined in (14).

Define $\mathbf{S}^F_{r_2,s} = (\beta^{(r_2,s)}_{i,j})_{(N_x-1)\times(N_y-1)}$, then we have the approximations of $\mathbf{Q}^F$ as

$$\mathbf{Q}^F \approx \sum_{s=0}^{r_2} \mathbf{F}_{n-s} \odot \mathbf{S}^F_{r_2,s}, \tag{19}$$

which is (r_2+1)-th order accurate.

2.1.3 The explicit integration factor scheme and fast calculations

Now put (15) and (19) back to (10), we obtain a fast explicit integration factor (fEIF) scheme that is second-order accurate in space and at least $(\min\{r_1, r_2\} + 1)$-th order accurate in time:

$$\begin{aligned}\mathbf{U}_{n+1} = \mathbf{P}_y \textcircled{y} \mathbf{P}_x \textcircled{x} \Big(&(\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathbf{U_n}) \odot (e^*)^{\mathbf{H}\Delta t} \\ &+ \sum_{s=-1}^{r_1-1} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{W}_{n-s}) \odot \mathbf{S}^W_{r_1,s} - \sum_{s=0}^{r_2} (\mathbf{P}_y^{-1} \textcircled{y} \mathbf{P}_x^{-1} \textcircled{x} \mathcal{F}(\mathbf{U}_{n-s})) \odot \mathbf{S}^F_{r_2,s} \Big).\end{aligned} \tag{20}$$

Remark 1 *We specially note that*

1. *When $\hat{f}(u) \equiv 0$, the fEIF scheme (20) for solving the equation (1) is expected to be unconditionally stable when $r_1 \le 1$ since the interpolation approximation (15) is convex.*
2. *For solving the Allen-Cahn equation (the equation (3) with $f(u) = \frac{1}{\epsilon^2}(u^3 - u)$), the fEIF scheme (20) with $r_2 = 0$ is expected to be energy stable since it is analogous to the stabilized semi-implicit Euler method [24] to some extent. More detailed stability analysis for general r_2 can be carried out similarly like those given in [5, 6] for the so-called ETD (exponential time difference) Runge-Kutta schemes, see section 4 for details.*

Now let us consider fast implementation of the proposed method. Let $\mathbf{V}$ be any $(N_x - 1) \times (N_y - 1)$ array. For the operation $\mathbf{P}_x \textcircled{x} \mathbf{V}$ or $\mathbf{P}_x^{-1} \textcircled{x} \mathbf{V}$, its complexity is usually $O(N_x^2 N_y)$, but can be improved to $O(N_x N_y \log_2(N_x))$ through the use of fast Discrete Sine Transform (DST) [14] in this case. If we arrange the eigenvalues of $\mathbf{A}$ in a increasing order of its absolute values, then we have

$$d_k^x = -\frac{4D}{h_x^2} \sin^2\left(\frac{k\pi}{2N_x}\right), \quad k = 1, 2, \cdots, N_x - 1, \tag{21}$$

and

$$(\mathbf{P}_x)_{k,j} = \sin\left(\frac{\pi k j}{N_x}\right), \quad k = 1, 2, \cdots, N_x - 1, \ j = 1, 2, \cdots, N_x - 1, \tag{22}$$

can be chosen as its corresponding eigenmatrix. In this case $\mathbf{P}_x^{-1} = \dfrac{2}{N_x} \mathbf{P}_x$. It is easy to verify that $\mathbf{P}_x \textcircled{x} \mathbf{V}$ means applying a DST to each column of $\mathbf{V}$ and $\mathbf{P}_x^{-1} \textcircled{x} \mathbf{V}$ does an inverse DST, i.e.,

$$\mathbf{P}_x \textcircled{x} \mathbf{V} = \mathrm{DST}_{N_x - 1}(\mathbf{V}), \quad \mathbf{P}_x^{-1} \textcircled{x} \mathbf{V} = \mathrm{iDST}_{N_x - 1}(\mathbf{V}).$$

Similarly, let us choose

$$d_k^y = -\frac{4D}{h_y^2} \sin^2\left(\frac{k\pi}{2N_y}\right), \quad k = 1, 2, \cdots, N_y - 1, \tag{23}$$

and

$$(\mathbf{P}_y)_{k,j} = \sin\left(\frac{\pi k j}{N_y}\right), \quad k = 1, 2, \cdots, N_y - 1, \ j = 1, 2, \cdots, N_y - 1, \tag{24}$$

then $\mathbf{P}_y \textcircled{y} \mathbf{V}$ means applying a DST to each row of $\mathbf{V}$ and $\mathbf{P}_y^{-1} \textcircled{y} \mathbf{V}$ does an inverse DST, i.e.,

$$\mathbf{P}_y \textcircled{y} \mathbf{V} = \mathrm{DST}_{N_y - 1}(\mathbf{V}^T)^T, \quad \mathbf{P}_y^{-1} \textcircled{y} \mathbf{V} = \mathrm{iDST}_{N_y - 1}(\mathbf{V}^T)^T.$$

Thus the overall cost of the proposed fEIF scheme in two dimensions can be reduced from $O(N_x N_y N)$ to $O(N_x N_y \log_2 N)$ per time step where $N = \max\{N_x, N_y\}$, which is very important especially when N is quite large. The total required memory is obviously $O(N^2)$.

The proposed scheme will be next extended to other boundary boundary conditions, such as periodic and Neumann boundary conditions.

2.2 The problem with periodic boundary conditions

If a periodic boundary condition as

$$\begin{cases} u(t,x_b,y)=u(t,x_e,y),\ \dfrac{\partial u}{\partial x}(t,x_b,y)=\dfrac{\partial u}{\partial x}(t,x_e,y), & y\in[y_b,y_e],t\in[t_0,t_0+T],\\ u(t,x,y_b)=u(t,x,y_e),\ \dfrac{\partial u}{\partial y}(t,x,y_b)=\dfrac{\partial u}{\partial y}(t,x,y_e), & x\in[x_b,x_e],t\in[t_0,t_0+T], \end{cases}$$

is imposed, then we may denote the unknowns as

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

and set

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

and

$$\mathbf{A}=\frac{D}{h_x^2}\mathbf{G}^P_{N_x\times N_x},\quad \mathbf{B}=\frac{D}{h_y^2}\mathbf{G}^P_{N_y\times N_y}.$$

In addition, we have $\mathcal{W}=\mathbf{0}$. The fEIF scheme for this periodic boundary condition is then as same as (20).

In this case, the complexity of $\mathbf{P}_x\textcircled{x}\mathbf{V}$ or $\mathbf{P}_x^{-1}\textcircled{x}\mathbf{V}$ also can be improved to $O(N_xN_y\log_2(N_x))$ with $\mathbf{V}$ being any $N_x\times N_y$ array, through the use of fast Discrete Fourier Transform (DFT) [14]. If we arrange the eigenvalues of $\mathbf{A}$ as

$$d_k^x=-\frac{4D}{h_x^2}\sin^2\left(\frac{(k-1)\pi}{N_x}\right),\quad k=1,2,\cdots,N_x, \tag{25}$$

then

$$(\mathbf{P}_x)_{k,j}=\exp\left(-\mathrm{i}\frac{2\pi(k-1)(j-1)}{N_x}\right),\quad k=1,2,\cdots,N_x,\ j=1,2,\cdots,N_x, \tag{26}$$

can be chosen as its corresponding eigenmatrix. In this case $\mathbf{P}_x^{-1}=\dfrac{1}{N_x}\mathbf{P}_x^*$. Let $\mathbf{V}$ be any $N_x\times N_y$ array, it is easy to verify that $\mathbf{P}_x\textcircled{x}\mathbf{V}$ means applying a DFT to each column of $\mathbf{V}$ and $\mathbf{P}_x^{-1}\textcircled{x}\mathbf{V}$ does an inverse DFT, i.e.,

$$\mathbf{P}_x\textcircled{x}\mathbf{V}=\mathrm{DFT}_{N_x}(\mathbf{V}),\quad \mathbf{P}_x^{-1}\textcircled{x}\mathbf{V}=\mathrm{iDFT}_{N_x}(\mathbf{V}).$$

Similarly, let us choose

$$d_k^y=-\frac{4D}{h_y^2}\sin^2\left(\frac{(k-1)\pi}{N_y}\right),\quad k=1,2,\cdots,N_y, \tag{27}$$

and

$$(\mathbf{P}_y)_{k,j}=\exp\left(-\mathrm{i}\frac{2\pi(k-1)(j-1)}{N_y}\right),\quad k=1,2,\cdots,N_y,\ j=1,2,\cdots,N_y, \tag{28}$$

then $\mathbf{P}_y\textcircled{y}\mathbf{V}$ means applying a DFT to each row of $\mathbf{V}$ and $\mathbf{P}_y^{-1}\textcircled{y}\mathbf{V}$ does an inverse DFT, i.e.,

$$\mathbf{P}_y\textcircled{y}\mathbf{V}=\mathrm{DFT}_{N_y}(\mathbf{V}^T)^T,\quad \mathbf{P}_y^{-1}\textcircled{y}\mathbf{V}=\mathrm{iDFT}_{N_y}(\mathbf{V}^T)^T.$$

Thus the overall cost of the proposed fEIF scheme in two dimensions can be again reduced from $O(N_xN_yN)$ to $O(N_xN_y\log_2 N)$ per time step.

2.3 The problem with Neumann boundary conditions

If a Neumann boundary condition as

$$\begin{cases} \dfrac{\partial u}{\partial x} = b^x, & (x,y) \in \partial\Omega, t \in [t_0, t_0+T], \\ \dfrac{\partial u}{\partial y} = b^y, & (x,y) \in \partial\Omega, t \in [t_0, t_0+T], \end{cases}$$

is imposed, then we may set

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

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

and

$$\mathbf{A} = \frac{D}{h_x^2}\mathbf{G}^N_{(N_x+1)\times(N_x+1)}, \quad \mathbf{B} = \frac{D}{h_y^2}\mathbf{G}^N_{(N_y+1)\times(N_y+1)}.$$

Note that in this case $\mathbf{G}^N$ is not symmetric. We have $\mathcal{W} = \mathbf{U}_{x1} + \mathbf{U}_{y1}$ given as

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

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

The fEIF scheme for the Neumann boundary condition is again as same as (20).

However, in order to reduce the computational complexity of $\mathbf{P}_x \textcircled{x} \mathbf{V}$ or $\mathbf{P}_x^{-1} \textcircled{x} \mathbf{V}$ with $\mathbf{V}$ being any $(N_x+1)\times(N_y+1)$ array in this case from $O(N_xN_yN)$ to $O(N_xN_y\log_2 N)$ by a FFT-based fast algorithm, we have to make some special modifications. One effective and equivalent way is to use symmetric extension in each direction for unknowns $\mathbf{U}$ to obtain a periodic case [20], and we still can solve the problem *in the original domain*. Let us directly present the approach in the following. Set

$$d_k^x = -\frac{4D}{h_x^2}\sin^2\left(\frac{(k-1)\pi}{2N_x}\right), \quad k = 1,2,\cdots,N_x+1, \tag{29}$$

in the scheme (20). We remark that $\{d_k^x\}_{k=1}^{N_x+1}$ are *not* eigenvalues of $\mathbf{A}$, in stead they are the first N_x+1 eigenvalues of the periodic case with a length $2N_x$. Let $\mathbf{v} = [v_1, v_2, \cdots, v_{N_x+1}]^T$ be any vector of length N_x+1 and correspondingly define its reflection vector of length N_x-1 as $\hat{\mathbf{v}} = [v_{N_x}, v_{N_x-1}, \cdots, v_2]^T$, then we define the operator "RDFT" on $\mathbf{v}$ as to applying a DFT to $\begin{bmatrix}\mathbf{v}\\ \hat{\mathbf{v}}\end{bmatrix}$ and then taking the first N_x+1 components of the result, and "iRDFT" as to applying an inverse DFT and then taking the first N_x+1

components. Then it can be verified that $\mathbf{P}_x \textcircled{x} \mathbf{V}$ in this case means applying a RDFT to each column of $\mathbf{V}$ and $\mathbf{P}_x^{-1} \textcircled{x} \mathbf{U}$ does a inverse RDFT, i.e.,

$$\mathbf{P}_x \textcircled{x} \mathbf{V} = \mathrm{RDFT}_{N_x+1}(\mathbf{V}), \quad \mathbf{P}_x^{-1} \textcircled{x} \mathbf{V} = \mathrm{iRDFT}_{N_x+1}(\mathbf{V}).$$

Similarly, let us choose

$$d_k^y = -\frac{4D}{h_y^2} \sin^2\left(\frac{(k-1)\pi}{2N_y}\right), \quad k = 1, 2, \cdots, N_y + 1, \tag{30}$$

then $\mathbf{P}_y \textcircled{y} \mathbf{V}$ means applying a RDFT to each row of $\mathbf{V}$ and $\mathbf{P}_y^{-1} \textcircled{y} \mathbf{V}$ does an inverse RDFT, i.e.,

$$\mathbf{P}_y \textcircled{y} \mathbf{V} = \mathrm{RDFT}_{N_y+1}(\mathbf{V}^T)^T, \quad \mathbf{P}_y^{-1} \textcircled{y} \mathbf{V} = \mathrm{iRDFT}_{N_y+1}(\mathbf{V}^T)^T.$$

Thus the overall cost of the proposed fEIF scheme in two dimensions can be again reduced from $O(N_x N_y N)$ to $O(N_x N_y \log_2 N)$ per time step.

3 Fast explicit integration factor methods in three dimensions

In this section we derive fEIF methods for the model problem (1) in three dimensions. Let $\Omega = \{x_b < x < x_e, y_b < y < y_e, z_b < z < z_e\}$. We present the case with a Dirichlet boundary condition as $u = g$ on $\partial\Omega$ and discussions on other boundary condition cases simply follow.

Similar to solving the two dimensional system, we denote h_x, h_y, h_z as the spatial step size, and N_x, N_y, N_z as the number of grid points in x, y, z direction, respectively. Let $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{G}^D$ defined as before. We now define

$$\mathbf{C} = \frac{D}{h_z^2} \mathbf{G}^D_{(N_z-1)\times(N_z-1)}.$$

Set $u_{i,j,k} = u_{i,j,k}(t) \approx u(t, x_i, y_j, z_k)$ for $0 \le i \le N_x$, $0 \le j \le N_y$ and $0 \le k \le N_z$. Denote the unknowns as a three-dimensional array $\mathbf{U} = (u_{i,j,k})_{(N_x-1)\times(N_y-1)\times(N_z-1)}$.

With similar analysis, we can write the semi-discretization of (1) as the following compact representation:

$$\mathbf{U}_t = \mathbf{A} \textcircled{x} \mathbf{U} + \mathbf{B} \textcircled{y} \mathbf{U} + \mathbf{C} \textcircled{z} \mathbf{U} - \kappa \mathbf{U} + \mathcal{W} - \mathcal{F}(\mathbf{U}) \tag{31}$$

where the special operations are defined (or re-defined) as

$$\begin{aligned} (\mathbf{A} \textcircled{x} \mathbf{U})_{i,j,k} &= \sum_{l=1}^{N_x-1} (\mathbf{A})_{i,l} u_{l,j,k}, \\ (\mathbf{B} \textcircled{y} \mathbf{U})_{i,j,k} &= \sum_{l=1}^{N_y-1} (\mathbf{B})_{j,l} u_{i,l,k}, \\ (\mathbf{C} \textcircled{z} \mathbf{U})_{i,j,k} &= \sum_{l=1}^{N_z-1} (\mathbf{C})_{k,l} u_{i,j,l}, \end{aligned} \tag{32}$$

and the three-dimensional array $\mathcal{W}$ is defined correspondingly from the given Dirichlet boundary conditions as $\mathcal{W} = \mathbf{U}_{x1} + \mathbf{U}_{y1} + \mathbf{U}_{z1}$ with

$$\mathbf{U}_{x1} = \frac{D}{h_x^2} (\gamma^x_{i,j,k})_{(N_x-1)\times(N_y-1)\times(N_z-1)}, \quad \gamma^x_{i,j,k} = \begin{cases} g(t, x_0, y_j, z_k), & i = 1, \\ 0, & 1 < i < N_x - 1, \\ g(t, x_{N_x}, y_j, z_k), & i = N_x - 1, \end{cases}$$

$$\mathbf{U}_{y1} = \frac{D}{h_y^2} (\gamma^y_{i,j,k})_{(N_x-1)\times(N_y-1)\times(N_z-1)}, \quad \gamma^y_{i,j,k} = \begin{cases} g(t, x_i, y_0, z_k), & j = 1, \\ 0, & 1 < j < N_y - 1, \\ g(t, x_i, y_{N_y}, z_k), & j = N_y - 1, \end{cases}$$

$$\mathbf{U}_{z1} = \frac{D}{h_z^2}(\gamma^z_{i,j,k})_{(N_x-1)\times(N_y-1)\times(N_z-1)}, \quad \gamma^z_{i,j,k} = \begin{cases} g(t, x_i, y_j, z_0), & k = 1, \\ 0, & 1 < k < N_z - 1, \\ g(t, x_i, y_j, z_{N_z}) & k = N_z - 1. \end{cases}$$

Once again it is obvious that $\mathbf{C}$ is also diagonalizable as $\mathbf{C} = \mathbf{P}_z\tilde{\mathbf{D}}_z{\mathbf{P}_z}^{-1}$ where the diagonal matrix $\tilde{\mathbf{D}}_z = \text{diag}[d^z_1, d^z_2, \cdots, d^z_{N_z-1}]$ with d^z_i being the eigenvalues of $\mathbf{C}$. Following the similar analysis in two dimensions, a fast explicit integration factor scheme that is second-order accurate in space and and at least $(\min\{r_1, r_2\} + 1)$-th order accurate in time is given as below:

$$\begin{aligned}\mathbf{U}_{n+1} = \mathbf{P}_z ⓩ \mathbf{P}_y ⓨ \mathbf{P}_x ⓧ \Big(&(\mathbf{P}_z^{-1} ⓩ \mathbf{P}_y^{-1} ⓨ \mathbf{P}_x^{-1} ⓧ \mathbf{U_n}) \odot (e^*)^{\mathbf{H}\Delta t} \\ &+ \sum_{s=-1}^{r_1-1} (\mathbf{P}_z^{-1} ⓩ \mathbf{P}_y^{-1} ⓨ \mathbf{P}_x^{-1} ⓧ \mathcal{W}_{n-s}) \odot \mathbf{S}^W_{r_1,s} \\ &- \sum_{s=0}^{r_2} (\mathbf{P}_z^{-1} ⓩ \mathbf{P}_y^{-1} ⓨ \mathbf{P}_x^{-1} ⓧ \mathcal{F}(\mathbf{U}_{n-s})) \odot \mathbf{S}^F_{r_2,s} \Big),\end{aligned} \tag{33}$$

where $\mathbf{H} = (h_{i,j,k})_{(N_x-1)\times(N_y-1)\times(N_z-1)}$ with

$$h_{i,j,k} = d^x_i + d^y_j + d^z_k - \kappa, \tag{34}$$

$\mathbf{S}^W_{r_1,s} = (\alpha^{(r_1,s)}_{i,j,k})_{(N_x-1)\times(N_y-1)\times(N_z-1)}$ and $\mathbf{S}^F_{r_2,s} = (\beta^{(r_2,s)}_{i,j,k})_{(N_x-1)\times(N_y-1)\times(N_z-1)}$ are defined as three dimensional versions of (13) and (18), the operator "$\odot$" is also defined similarly as before, element by element multiplication between two thee-dimensional arrays. It is again easy to see that the overall cost of the fEIF scheme in three dimensions also can be reduced from $O(N_xN_yN_zN)$ to $O(N_xN_yN_z \log_2 N)$ where $N = \max\{N_x, N_y, N_z\}$ per time step by using FFT-based fast calculations.

4 Linear stability analysis

For the proposed explicit integration factor methods, we test their stability on the following linear equation

$$u_t = \mathbf{L}u + \lambda u, \tag{35}$$

with $\mathbf{L}u = -qu$ and a homogeneous boundary condition. We follow [5,6] and consider the cases where λ is complex-valued and q is a positive real number that corresponds to a Fourier mode of the self-adjoint and elliptic operator $\mathbf{L}$. We employ the boundary locus method to determine the stability regions of the proposed numerical schemes.

Let us apply the first-order (in time) fEIF scheme ($r_2 = 0$) to the equation (35), which leads to

$$u_{n+1} = e^{-q\Delta t}u_n + \psi_0(\lambda + \kappa)\Delta t u_n \tag{36}$$

where κ is the splitting parameter and $\psi_0 = (1 - e^{-(q+\kappa)\Delta t})/((q + \kappa)\Delta t)$. The characteristic equation of (36) then can be written as

$$\xi = e^{-(q+\kappa)\Delta t} + \psi_0(\lambda + \kappa)\Delta t. \tag{37}$$

where $\xi = e^{\mathrm{i}\theta}$ with $0 \le \theta < 2\pi$.

In the proposed fEIF schemes we may rewrite the splitting parameter κ in adaptive way as

$$\kappa = \alpha \max\{0, -Re(\lambda)\} \tag{38}$$

where $\alpha \ge 0$ is a constant to control the scale of the splitting. Clearly $\alpha = 0$ means that no splitting is applied. In order to satisfy the stability constraint (4) in our fEIF scheme, we need require $\alpha \ge 1/2$ for solving the model linear equation.

In the case of $Re(\lambda) \ge 0$, it is easy to see that $\kappa = \alpha \cdot 0 = 0$ from (38) for our scheme. Solving the equation (37) gives us the explicit formulas for $Re(\lambda\Delta t)$ and $Im(\lambda\Delta t)$ as

$$\begin{cases} Re(\lambda\Delta t) = \dfrac{q\Delta t(\cos(\theta) - e^{-q\Delta t})}{1 - e^{-q\Delta t}}, \\ Im(\lambda\Delta t) = \dfrac{q\Delta t \sin(\theta)}{1 - e^{-q\Delta t}}, \end{cases} \tag{39}$$

which defines the boundary curve of the stability region in the right half plane ($Re(\lambda) \geq 0$). In the case of $Re(\lambda) < 0$ we have the splitting parameter $\kappa = -\alpha Re(\lambda)$ according to (38). We then obtain the boundary locus curve equation of the stability region in the left half plane ($Re(\lambda) < 0$) from (37) as

$$\Big(e^{\alpha Re(\lambda\Delta t)} + \frac{e^{q\Delta t} - e^{\alpha Re(\lambda\Delta t)}}{q\Delta t - \alpha Re(\lambda\Delta t)}(1 - \alpha Re(\lambda\Delta t))\Big)^2 + \Big(\frac{e^{q\Delta t} - e^{\alpha Re(\lambda\Delta t)}}{q\Delta t - \alpha Re(\lambda\Delta t)} Im(\lambda\Delta t)\Big)^2 - e^{2q\Delta t} = 0, \quad (40)$$

where $Re(\lambda\Delta t)$ and $Im(\lambda\Delta t)$ are implicitly defined. Different values of $q\Delta t$ will give us a family of boundary curves for the corresponding stability regions.

Figure 1-left illustrates the stability regions (determined by (39) and (40)) of the first-order fEIF method with $\kappa = \max\{0, -Re(\lambda)\}$ for different values of $q\Delta t = 1.0, 2.0, 4.0$. It is easy to find that the stability region gets larger when the value of $q\Delta t$ gets bigger. Figure 1-right shows the stability regions of the first-order fEIF scheme with different values of $\kappa = \alpha \max\{0, -Re(\lambda)\}$ with $\alpha = 0, 0.5, 1.0, 2.0$, for a fixed $q\Delta t = 1.0$. We observe that the stability region grows larger monotonically along with the increasing of κ in the first-order fEIF scheme. We would like to specially note that larger κ may lead to a more stable but possibly less accurate scheme. Stability analysis for the proposed higher-order fEIF schemes is much more complicated and will be rigorously investigated in our future work.

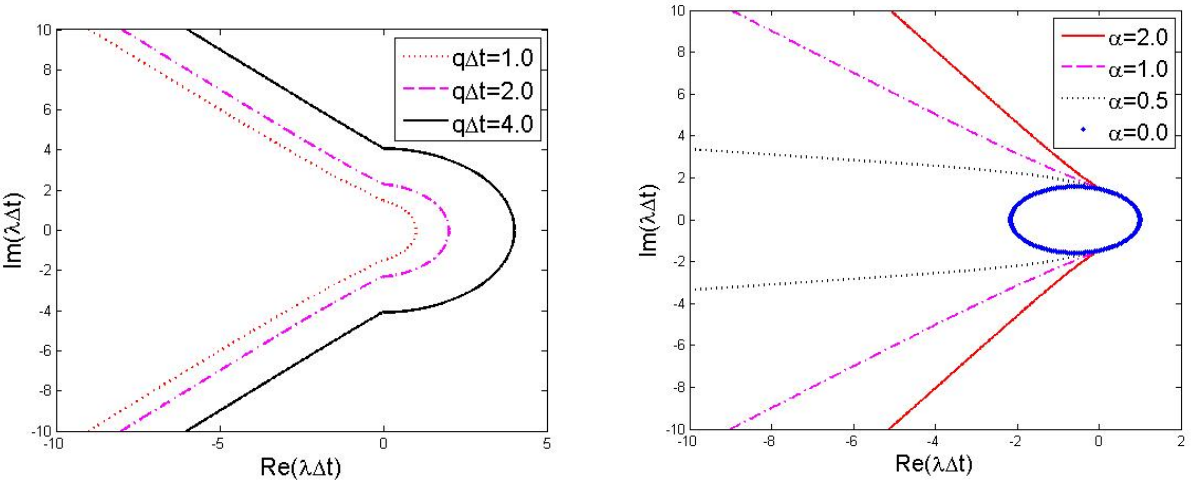

Fig. 1 Stability regions of the first-order fEIF schemes. Left: $\kappa = \max\{0, -Re(\lambda)\}$ with $q\Delta t = 1.0, 2.0, 4.0$ respectively; right: $q\Delta t = 1.0$ and $\kappa = \alpha \max\{0, -Re(\lambda)\}$ with $\alpha = 0, 0.5, 1.0, 2.0$ respectively.

5 Numerical experiments

In this section, we verify the accuracy and robustness of the proposed fEIF method through various examples. All experiments are run on a MacPro Book with 2.4GHz Intel Core i7 CPU and 8GB memory.

5.1 Linear diffusion problems

We consider linear diffusion problems first in which $f(u) \equiv 0$ and $\kappa = 0$.

Exmp 1 *Let $\Omega = [-1, 1]^2$ and $T = \frac{2}{\pi^2}$, we consider*

$$\begin{cases} \dfrac{\partial u}{\partial t} = \Delta u, & (x, y) \in \Omega, t \in [0, T], \\ u(0, x, y) = \sin(\pi(x - \frac{1}{4}))\sin(2\pi(y - \frac{1}{8})), & (x, y) \in \Omega. \end{cases} \quad (41)$$

Let us set the exact solution to be

$$u(t,x,y) = e^{-5\pi^2 t}\sin(\pi(x-\tfrac{1}{4}))\sin(2\pi(y-\tfrac{1}{8})), \tag{42}$$

which satisfies the equation and initial condition. The boundary conditions are determined accordingly from the exact solution. We test cases of Dirichlet and Neumann boundary conditions. Clearly we have non-zero boundary conditions, $u|_{\partial\Omega} \neq 0$ *and* $\dfrac{\partial u}{\partial \mathbf{n}}|_{\partial\Omega} \neq 0$.

Numerical results at the final time T of Example 1 with *Dirichlet boundary condition* and *Neumann boundary condition* produced by the proposed fEIF scheme are reported in Table 1. As expected we clearly observe the second-order accuracy of space discretization, and first ($r_1 = 0$), second ($r_1 = 1$) and third-order accuracy ($r_1 = 2$) of time discretization.

$(N_x \times N_y)$	Dirichlet Boundary Condition				Neumann Boundary Condition			
$\times N_t$	L_2 Error	CR	L_∞ Error	CR	L_2 Error	CR	L_∞ Error	CR
Accuracy test of space discretization, $r_1 = 2$								
$(16^2) \times 2048$	4.0532e-04	-	4.2578e-04	-	2.2816e-03	-	2.7967e-03	-
$(32^2) \times 2048$	9.8397e-05	2.04	1.0334e-04	2.04	5.7043e-04	2.00	6.9758e-04	2.00
$(64^2) \times 2048$	2.4425e-05	2.01	2.5636e-05	2.01	1.4261e-04	2.00	1.7431e-04	2.00
$(128^2) \times 2048$	6.0956e-06	2.00	6.3968e-06	2.00	3.5653e-05	2.00	4.3572e-05	2.00
$(256^2) \times 2048$	1.5232e-06	2.00	1.5984e-06	2.00	8.9135e-06	2.00	1.0892e-05	2.00
$(512^2) \times 2048$	3.8074e-07	2.00	3.9954e-07	2.00	2.2285e-06	2.00	2.7233e-06	2.00
Accuracy test of time discretization, $r_1 = 0$								
$(2048^2) \times 8$	3.6874e-03	-	3.8196e-03	-	2.6928e-02	-	3.3739e-02	-
$(2048^2) \times 16$	2.0877e-03	0.82	2.1589e-03	0.82	1.5225e-02	0.82	1.9110e-02	0.82
$(2048^2) \times 32$	1.1094e-03	0.91	1.1460e-03	0.91	8.0824e-03	0.92	1.0158e-02	0.91
$(2048^2) \times 64$	5.7169e-04	0.96	5.9012e-04	0.96	4.1620e-03	0.96	5.2351e-03	0.96
$(2048^2) \times 128$	2.9015e-04	0.98	2.9941e-04	0.98	2.1115e-03	0.98	2.6571e-03	0.98
$(2048^2) \times 256$	1.4617e-04	0.99	1.5080e-04	0.99	1.0634e-03	0.99	1.3384e-03	0.99
Accuracy test of time discretization, $r_1 = 1$								
$(2048^2) \times 8$	9.6990e-04	-	9.9828e-04	-	7.0387e-03	-	8.8883e-03	-
$(2048^2) \times 16$	2.4462e-04	1.99	2.5219e-04	1.98	1.7789e-03	1.98	2.2412e-03	1.99
$(2048^2) \times 32$	6.1279e-05	2.00	6.3203e-05	2.00	4.4608e-04	2.00	5.6168e-04	2.00
$(2048^2) \times 64$	1.5310e-05	2.00	1.5794e-05	2.01	1.1170e-04	2.00	1.4062e-04	2.00
$(2048^2) \times 128$	3.8105e-06	2.01	3.9323e-06	2.01	2.8035e-05	1.99	3.5287e-05	1.99
Accuracy test of time discretization, $r_1 = 2$								
$(2048^2) \times 8$	2.9868e-04	-	3.4534e-04	-	2.3042e-03	-	2.5927e-03	-
$(2048^2) \times 16$	4.5800e-05	2.71	5.1750e-05	2.74	3.5331e-04	2.71	4.0225e-04	2.69
$(2048^2) \times 32$	6.3458e-06	2.85	7.1113e-06	2.86	4.9196e-05	2.84	5.6248e-05	2.84
$(2048^2) \times 64$	8.1778e-07	2.96	9.1609e-07	2.96	6.6090e-06	2.90	7.5791e-06	2.89

Table 1 Errors and convergence rates at the final time T of Example 1 (2D linear diffusion) with *Dirichlet boundary condition* and *Neumann boundary condition* respectively using the fEIF scheme.

Exmp 2 *Let* $\Omega = [-1,1]^3$ *and* $T = 1$*, we consider the 3D pure diffusion problem*

$$\begin{cases} \dfrac{\partial u}{\partial t} = \dfrac{5}{3\pi^2}\Delta u, & (x,y,z) \in \Omega, t \in [0,T], \\ u(0,x,y,z) = \sin(\pi(x-\tfrac{1}{4}))\sin(2\pi(y-\tfrac{1}{8}))\sin(\pi z), & (x,y,z) \in \Omega, \end{cases} \tag{43}$$

and set the exact solution to be

$$u(t,x,y,z) = e^{-10t}\sin(\pi(x-\tfrac{1}{4}))\sin(2\pi(y-\tfrac{1}{8}))\sin(\pi z), \tag{44}$$

which satisfies the equation (43). The boundary conditions are still determined accordingly from the exact solution. We again test cases of Dirichlet and Neumann boundary conditions which are clearly non-zero.

Numerical results at the final time T of Example 2 with *Dirichlet boundary condition* and *Neumann boundary condition* produced by the proposed fEIF scheme are reported in Table 2. Again as expected we clearly observe the second-order accuracy of space discretization, and first ($r_1 = 0$), second ($r_1 = 1$) and third-order accuracy ($r_1 = 2$) of time discretization.

$(N_x \times N_y \times N_z)$ $\times N_t$	Dirichlet Boundary Condition				Neumann Boundary Condition			
	L_2 Error	CR	L_∞ Error	CR	L_2 Error	CR	L_∞ Error	CR
Accuracy test of space discretization, $r_1 = 2$								
$(16^3) \times 256$	1.5801e-04	-	1.3111e-04	-	1.9077e-03	-	1.6638e-03	-
$(32^3) \times 256$	3.8185e-05	2.05	3.1534e-05	2.06	4.8168e-04	1.99	4.1994e-04	1.99
$(64^3) \times 256$	9.4649e-06	2.01	7.8139e-06	2.01	1.2078e-04	2.00	1.0529e-04	2.00
$(128^3) \times 256$	2.3596e-06	2.00	1.9477e-06	2.00	3.0284e-05	2.00	2.6399e-05	2.00
Accuracy test of time discretization, $r_1 = 0$								
$(128^3) \times 4$	1.9166e-03	-	1.3850e-03	-	3.9867e-02	-	3.6911e-02	-
$(128^3) \times 8$	1.2389e-03	0.63	8.9557e-04	0.63	2.5615e-02	0.64	2.3740e-02	0.64
$(128^3) \times 16$	7.0846e-04	0.81	5.1266e-04	0.80	1.4491e-02	0.82	1.3453e-02	0.82
$(128^3) \times 32$	3.7998e-04	0.90	2.7521e-04	0.90	7.6872e-03	0.91	7.1468e-03	0.91
$(128^3) \times 64$	1.9764e-04	0.94	1.4330e-04	0.94	3.9466e-03	0.96	3.6730e-03	0.96
Accuracy test of time discretization, $r_1 = 1$								
$(128^3) \times 4$	1.3651e-03	-	9.9484e-04	-	2.6180e-02	-	2.463752e-02	-
$(128^3) \times 8$	3.3563e-04	2.02	2.4325e-04	2.03	6.7790e-03	1.95	6.3248e-03	1.96
$(128^3) \times 16$	8.1972e-05	2.03	5.9190e-05	2.04	1.7321e-03	1.97	1.6107e-03	1.97
$(128^3) \times 32$	1.8761e-05	2.12	1.3443e-05	2.14	4.5654e-04	1.92	4.2240e-04	1.93
Accuracy test of time discretization, $r_1 = 2$								
$(128^3) \times 4$	4.4571e-04	-	3.6430e-04	-	1.1102e-02	-	9.0294e-03	-
$(128^3) \times 8$	5.7597e-05	2.95	5.4321e-05	2.74	2.0703e-03	2.42	1.7377e-03	2.38
$(128^3) \times 16$	7.4723e-06	2.95	7.0839e-06	2.93	3.4295e-04	2.60	2.9102e-04	2.58

Table 2 Errors and convergence rates at the final time T of Example 2 (3D linear diffusion) with *Dirichlet boundary condition* and *Neumann boundary condition* respectively using the fEIF scheme.

5.2 Allen-Cahn equations

We now consider the Allen-Cahn equation (with a normalized diffusion coefficient) which is nonlinear and describes the process of phase separation:

$$\frac{\partial u}{\partial t} = \Delta u - \frac{1}{\epsilon^2}(u^3 - u), \qquad \mathbf{x} \in \Omega, t \in [t_0, t_0 + T], \tag{45}$$

where $\epsilon > 0$ often gives the length of the transition regions between the domains and u takes values between -1 and 1. Either no-flux or periodic boundary conditions need to be imposed. Note here $f(u) = \frac{1}{\epsilon^2}(u^3 - u)$ and $F(u) = \frac{1}{4\epsilon^2}(u^2 - 1)^2$. The Ginzburg-Landau free energy of the Allen-Cahn equation is then identified as

$$\mathcal{E}(u) = \int_\Omega \left\{ \frac{1}{2}|\nabla u|^2 + \frac{1}{4\epsilon^2}(u^2 - 1)^2 \right\} d\mathbf{x}. \tag{46}$$

To choose a suitable κ, since $|u| \le 1$ we find

$$\frac{f'(u)}{2} = \frac{1}{2\epsilon^2}(3u^2 - 1) \le \frac{1}{\epsilon^2},$$

thus we take $\kappa = \frac{2}{\epsilon^2}$ which will ensure (4). Then we apply the proposed fEIF scheme to solve

$$\frac{\partial u}{\partial t} = \Delta u - \frac{2}{\epsilon^2}u - \frac{1}{\epsilon^2}(u^3 - 3u), \qquad \mathbf{x} \in \Omega, t \in [t_0, t_0 + T], \tag{47}$$

For the Allen-Cahn equation, we have $\mathcal{W} \equiv 0$ in the proposed discretization scheme. The Allen-Cahn equation solution has sharp interface which may cause the interpolation of $\mathcal{F}$ very inaccurate for high-order schemes $r_2 \ge 1$ when the time stepsize is not small enough. In order to overcome such an instability, we take an adaptive approach in our practical implementation: Let the $ord = r_2$ be the target order we will use and defined at the beginning; along each time step, if $\mathcal{E}(u_n) > \mathcal{E}(u_{n-1})$, we then set $ord = \max\{ord - 1, 1\}$; otherwise we set $ord = \min\{ord + 1, r_2\}$. As we found in the numerical experiments, the second-order scheme in time ($r_2 = 1$) is very stable for all time stepsizes, i.e., ord always stayed in $r_2 = 1$. But for the third-order scheme ($r_2 = 2$), ord sometimes jumped back to smaller values to stabilize the numerical algorithm when the time stepsize is large.

5.2.1 Travelling wave

It is known that the Allen-Cahn equation in the whole space has a travelling wave solution. We use this to do the benchmark test [15].

Exmp 3 *Let $\Omega = [-0.5, 1.5]^2$, we consider*

$$\begin{cases} \dfrac{\partial u}{\partial t} = \Delta u - \dfrac{1}{\epsilon^2}(u^3 - u), & (x,y) \in \Omega, t \in [0,T], \\ \dfrac{\partial u}{\partial \mathbf{n}}|_{\partial\Omega} = 0, & (x,y) \in \partial\Omega, t \in [0,T], \\ u(0,x,y) = \dfrac{1}{2}\left(1 - \tanh\left(\dfrac{x}{2\sqrt{2}\epsilon}\right)\right), & (x,y) \in \Omega. \end{cases} \tag{48}$$

The zero-Neumann boundary condition is imposed to allow for an approximate exact solution (for $\epsilon << 1$) of the form

$$u(t,x,y) = \frac{1}{2}\left(1 - \tanh\left(\frac{x - st}{2\sqrt{2}\epsilon}\right)\right) \tag{49}$$

with $s = \dfrac{3}{\sqrt{2}\epsilon}$. We set $\epsilon = 0.015$ (which is small enough) and $T = \dfrac{3}{4s}$.

Numerical results at the final time T of Example 3 produced by the proposed fEIF scheme are reported in Table 3. As expected we observe the second-order accuracy of space discretization, and first ($r_2 = 0$), second ($r_2 = 1$) and third-order accuracy ($r_2 = 2$) of time discretization when the time step size gets smaller.

$(N_x \times N_y) \times N_t$	L_2 Error	CR	L_∞ Error	CR
Accuracy test of space discretization, $r_2 = 2$				
$(32^2) \times 2048$	3.6279e-01	-	8.2217e-01	-
$(64^2) \times 2048$	1.2407e-01	1.55	3.3576e-01	1.29
$(128^2) \times 2048$	3.3084e-02	1.91	9.5968e-02	1.81
$(256^2) \times 2048$	8.3458e-03	1.99	2.4407e-02	1.98
$(512^2) \times 2048$	2.0651e-03	2.01	6.0660e-03	2.01
$(1024^2) \times 2048$	4.8697e-04	2.08	1.4268e-03	2.09
Accuracy test of time discretization, $r_2 = 0$				
$(2048^2) \times 16$	9.9669e-01	-	9.9999e-01	-
$(2048^2) \times 32$	8.5052e-01	0.23	9.9984e-01	0.00
$(2048^2) \times 64$	6.7373e-01	0.34	9.9653e-01	0.01
$(2048^2) \times 128$	4.8976e-01	0.46	9.5799e-01	0.06
$(2048^2) \times 256$	3.1820e-01	0.62	7.9050e-01	0.28
$(2048^2) \times 512$	1.8283e-01	0.80	5.1435e-01	0.62
$(2048^2) \times 1024$	9.6913e-02	0.92	2.8454e-01	0.85
Accuracy test of time discretization, $r_2 = 1$				
$(2048^2) \times 16$	9.4619e-01	-	9.9998e-01	-
$(2048^2) \times 32$	7.1141e-01	0.41	9.9830e-01	0.00
$(2048^2) \times 64$	3.7671e-01	0.92	8.7778e-01	0.19
$(2048^2) \times 128$	1.4380e-01	1.39	4.1931e-01	1.07
$(2048^2) \times 256$	4.0935e-02	1.81	1.2218e-01	1.78
$(2048^2) \times 512$	1.0593e-02	1.95	3.1572e-02	1.95
$(2048^2) \times 1024$	2.5983e-03	2.02	7.7315e-03	2.03
Accuracy test of time discretization, $r_2 = 2$				
$(2048^2) \times 16$	9.4571e-01	-	9.9998e-01	-
$(2048^2) \times 32$	6.8393e-01	0.47	9.9738e-01	0.00
$(2048^2) \times 64$	1.9006e-01	1.85	5.4521e-01	0.87
$(2048^2) \times 128$	3.6496e-02	2.38	1.0793e-01	2.34
$(2048^2) \times 256$	5.7806e-03	2.66	1.7043e-02	2.66
$(2048^2) \times 512$	8.3977e-04	2.78	2.4811e-03	2.78
$(2048^2) \times 1024$	4.5386e-05	4.21	1.3302e-04	4.22

Table 3 Errors and convergence rates at the final time T of Example 3 (2D Travelling wave) using the fEIF scheme.

Exmp 4 *Let $\Omega = [-0.5, 1.5] \times [-0.5/8, 0.5/8] \times [-0.5/8, 0.5/8]$. We consider*

$$\begin{cases} \dfrac{\partial u}{\partial t} = \Delta u - \dfrac{1}{\epsilon^2}(u^3 - u), & (x, y, z) \in \Omega, t \in [0, T], \\ \dfrac{\partial u}{\partial \mathbf{n}}|_{\partial \Omega} = 0, & (x, y, z) \in \partial\Omega, t \in [0, T], \\ u(0, x, y, z) = \dfrac{1}{2}\left(1 - \tanh\left(\dfrac{x}{2\sqrt{2}\epsilon}\right)\right), & (x, y, z) \in \Omega. \end{cases} \tag{50}$$

The approximate exact solution of (50) is

$$u(t, x, y, z) = \frac{1}{2}\left(1 - \tanh\left(\frac{x - st}{2\sqrt{2}\epsilon}\right)\right) \tag{51}$$

with $s = \dfrac{3}{\sqrt{2}\epsilon}$. We again set $\epsilon = 0.015$ and $T = \dfrac{3}{4s}$ as in the 2D problem.

Numerical results at the final time T of Example 4 produced by the proposed fEIF scheme are reported in Table 4. The observation on the convergence rates coincides with that of the 2D case in Example 3.

$(N_x \times N_y \times N_z) \times N_t$	L_2 Error	CR	L_∞ Error	CR
Accuracy test of space discretization, $r_2 = 2$				
$(64 \times 4^2) \times 2048$	1.5514e-02	-	3.3583e-01	-
$(128 \times 8^2) \times 2048$	4.1585e-03	1.90	9.6220e-02	1.80
$(256 \times 16^2 \times 2048$	1.0402e-03	2.00	2.4331e-02	1.98
$(512 \times 32^2) \times 2048$	2.8903e-04	1.85	6.3397e-03	1.94
Accuracy test of time discretization, $r_2 = 0$				
$(1024 \times 64^2) \times 16$	1.2458e-01	-	9.9999e-01	-
$(1024 \times 64^2) \times 32$	1.0630e-01	0.23	9.9984e-01	0.00
$(1024 \times 64^2) \times 64$	8.4205e-02	0.34	9.9652e-01	0.01
$(1024 \times 64^2) \times 128$	6.1200e-02	0.46	9.5788e-01	0.06
$(1024 \times 64^2) \times 256$	3.9743e-02	0.62	7.9004e-01	0.28
$(1024 \times 64^2) \times 512$	2.2813e-02	0.80	5.1348e-01	0.62
Accuracy test of time discretization, $r_2 = 1$				
$(1024 \times 64^2) \times 16$	1.0933e-01	-	9.9992e-01	-
$(1024 \times 64^2) \times 32$	8.1137e-02	0.43	9.9555e-01	0.01
$(1024 \times 64^2) \times 64$	4.7060e-02	0.79	8.7746e-01	0.18
$(1024 \times 64^2) \times 128$	1.7930e-02	1.39	4.1828e-01	1.07
$(1024 \times 64^2) \times 256$	5.0674e-03	1.82	1.2101e-01	1.82
$(1024 \times 64^2) \times 512$	1.2762e-03	1.99	3.0439e-02	1.99
Accuracy test of time discretization, $r_2 = 2$				
$(1024 \times 64^2) \times 16$	1.0050e-01	-	9.9969e-01	-
$(1024 \times 64^2) \times 32$	6.4477e-02	0.64	9.7326e-01	0.04
$(1024 \times 64^2 \times 64$	2.3714e-02	1.44	5.4428e-01	0.84
$(1024 \times 64^2) \times 128$	4.5110e-03	2.39	1.0674e-01	2.35
$(1024 \times 64^2) \times 256$	6.7286e-04	2.75	1.5877e-02	2.75
$(1024 \times 64^2) \times 512$	5.9837e-05	3.49	1.3578e-03	3.55

Table 4 Errors and convergence rates at the final time T of Example 4 (3D Travelling wave) using the fEIF scheme.

5.2.2 Mean curvature flow

Finally let us consider the mean curvature flow problem [16, 7].

Exmp 5 *Let $\Omega = [-0.5, 0.5]^d$ where $d > 1$ denotes the dimension of the space. The mean curvature flow problem we will simulate is given in the following:*

$$\begin{cases} \dfrac{\partial u}{\partial t} = \Delta u - \dfrac{1}{\epsilon^2}(u^3 - u), & \mathbf{x} \in \Omega, t \in [0, T], \\ u(0, \mathbf{x}) = \tanh\left(\dfrac{R_0 - \|\mathbf{x}\|_2}{\sqrt{2}\epsilon}\right), & \mathbf{x} \in \Omega. \end{cases}$$

with $R_0 = 0.4$. This is a test example used in many earlier works, see for instance [2,5,24]. Additionally a periodic boundary condition is imposed. The problem describes the shrinking process of a circle in 2D or a sphere in 3D. Let the radius of the circular region at time t is denoted as $R(t)$ and the area/volume by $V(t)$. As $\epsilon \to 0$, we know the theoretic limit radius $R_{lim}(t)$ satisfies [24,15]

$$\frac{dR_{lm}}{dt} = \frac{1-d}{R_{lim}},$$

thus we have

$$R_{lim}(t) = \sqrt{R_0^2 + 2(1-d)t}.$$

Correspondingly it holds that

$$\begin{cases} V_{lim}(t) = \pi(R_0^2 - 2t), & d = 2, \\ V_{lim}^{2/3}(t) = \left(\frac{4}{3}\pi\right)^{2/3}(R_0^2 - 4t), & d = 3. \end{cases}$$

We first test the circle shrinking process in the 2D space and set the final time $T = 0.075$, it is easy to calculate $R_{lim}(T) = 0.1$. We choose three different values for ϵ as $\epsilon = 0.04, 0.02$, and 0.01. Numerical results on the radius of the circle at the final time T produced by the proposed fEIF scheme for all cases are reported in Table 5. By checking the results produced by the finest grid $(2048^2) \times 2048$, we take $R_\epsilon = 0.099064, 0.099689, 0.100100$ for $\epsilon = 0.04, 0.02, 0.01$ respectively, and then use them as approximate exact solutions for each of the cases to compute convergence rates, see Table 6. We still see second-order accuracy in space for all values of ϵ. The convergence rates in time stably and monotonically catch the expected ones for $r_2 = 0$ and $r_2 = 1$ along the decreasing of the time stepsizes, but show some oscillatory behavior in the case of $r_2 = 2$. The smaller the ϵ is, more finer grids are needed to achieve optimal convergence. In addition, we see $|R_\epsilon - R_{lim}|$ are $9.3606e-04$, $3.1082e-04$, $9.9782e-05$ for $\epsilon = 0.04, 0.02, 0.01$ respectively, that implies the convergence to the limit solution is approximate order of 1.731 and 1.639 with respect to ϵ.

For test of the sphere shrinking process in the 3D space, we take $\epsilon = 0.02$ and set $T = 0.0375$. The limit radius is again $R_{lim} = 0.1$. The space-time grid we used is $(256^3) \times 256$ and we apply the fEIF scheme with $r_2 = 2$. We find that the simulated radius of the sphere at the final time T is $R = 0.104677$. Figure 2 illustrates the shrinking sphere at time $t = 0, 0.02$ and 0.0375. and Figure 3 shows evolution of the volume ($V^{2/3}$) and energy of the shrinking sphere along the time (both are straight lines as expected).

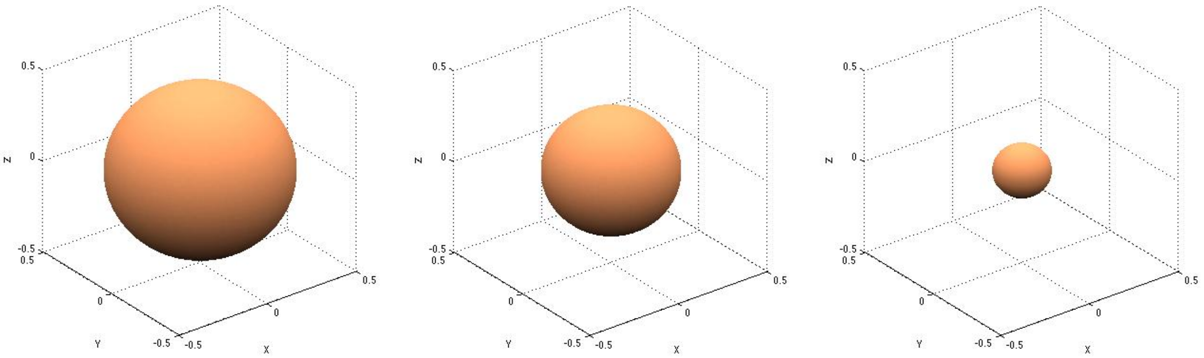

Fig. 2 Visualization of the shrinking sphere of Example 5 at time $t = 0, 0.02$ and 0.0375 respectively (from left to right) using the fEIF scheme.

6 Conclusions

In this work, a fast, accurate, stable and efficient method is discussed for the solution of semilinear parabolic equations of the type (1). The method utilizes a spatially compact difference stencil with a

$(N_x \times N_y)$ $\times N_t$	$\epsilon = 0.04$		$\epsilon = 0.02$		$\epsilon = 0.01$	
	R	$R - R_{lim}$	R	$R - R_{lim}$	R	$R - R_{lim}$
Accuracy test of space discretization, $r_2 = 2$						
$(64^2) \times 2048$	0.105418	5.4183e-03	0.121952	2.1952e-02	0.349818	2.4981e-01
$(128^2) \times 2048$	0.100681	6.8075e-04	0.105425	5.4245e-03	0.122017	2.2017e-02
$(256^2) \times 2048$	0.099466	-5.3439e-04	0.101123	1.1231e-03	0.105674	5.6744e-03
$(512^2) \times 2048$	0.099160	-8.4029e-04	0.100032	3.1716e-05	0.101441	1.4411e-03
$(1024^2) \times 2048$	0.099083	-9.1690e-04	0.099758	-2.4225e-04	0.100369	3.6883e-04
$(2048^2) \times 2048$	**0.099064**	-9.3606e-04	**0.099689**	-3.1082e-04	**0.100100**	9.9782e-05
Accuracy test of time discretization, $r_2 = 0$						
$(2048^2) \times 32$	0.348105	2.4810e-01	0.387151	2.8715e-01	0.396832	2.9683e-01
$(2048^2) \times 64$	0.303278	2.0327e-01	0.373134	2.7313e-01	0.393456	2.9345e-01
$(2048^2) \times 128$	0.249828	1.4982e-01	0.345356	2.4535e-01	0.386617	2.8661e-01
$(2048^2) \times 256$	0.200687	1.0068e-01	0.300333	2.0033e-01	0.372602	2.7260e-01
$(2048^2) \times 512$	0.162693	6.2692e-02	0.246677	1.4667e-01	0.344822	2.4482e-01
$(2048^2) \times 1024$	0.136506	3.6505e-02	0.197526	9.7525e-02	0.299778	1.9977e-01
Accuracy test of time discretization, $r_2 = 1$						
$(2048^2) \times 32$	0.187232	8.7231e-02	0.288545	1.8854e-01	0.358284	2.5828e-01
$(2048^2) \times 64$	0.128971	2.8970e-02	0.201057	1.0105e-01	0.300058	2.0005e-01
$(2048^2) \times 128$	0.107023	7.0234e-03	0.134711	3.4710e-02	0.218356	1.1835e-01
$(2048^2) \times 256$	0.100992	9.9181e-04	0.108076	8.0761e-03	0.145775	4.5775e-02
$(2048^2) \times 512$	0.099521	-4.7947e-04	0.101474	1.4739e-03	0.111247	1.1246e-02
$(2048^2) \times 1024$	0.099167	-8.3349e-04	0.100062	6.1653e-05	0.102223	2.2232e-03
Accuracy test of time discretization, $r_2 = 2$						
$(2048^2) \times 32$	0.134679	3.4678e-02	0.238688	1.3868e-01	0.346582	2.4658e-01
$(2048^2) \times 64$	0.103507	3.5071e-03	0.110308	1.0308e-02	0.220244	1.2024e-01
$(2048^2) \times 128$	0.101071	1.0714e-03	0.090471	-9.5291e-03	0.057173	-4.2826e-02
$(2048^2) \times 256$	0.099690	-3.0982e-04	0.100019	1.9341e-05	0.067282	-3.2717e-02
$(2048^2) \times 512$	0.099229	-7.7094e-04	0.099899	1.0075e-04	0.097120	-2.8804e-03
$(2048^2) \times 1024$	0.099099	-9.0148e-04	0.099782	–2.1845e-04	0.100270	2.6508e-04

Table 5 Numerical results on the radius of the shrinking circle at the final time T of Example 5 using the fEIF scheme.

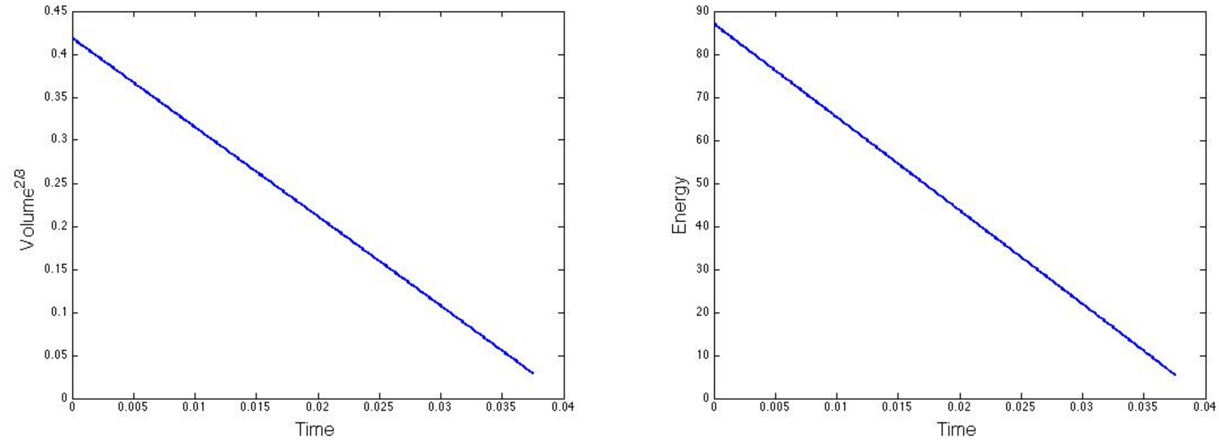

Fig. 3 Evolution of the volume ($V^{2/3}$) and energy of the shrinking sphere of Example 5 along the time using the fEIF scheme.

high order multistep integration factor in time scheme. The usual splitting technique for treating stiff nonlinearities is seamlessly incorporated into the resulting scheme to improve the stability without any degradation in the accuracy due to splitting error. While hardly discussed with details in the literature, there is much to be debated about how to deal with inhomogeneous boundary data. We present an approach that can incorporate various different types of inhomogenuous boundary data into the same discretization scheme effectively. Moreover, implementations based on fast matrix evaluations and fast transforms are considered for various boundary conditions. The use of adaptive variable order time integration schemes is also explored. We note that although most of these techniques individually have often been dealt with in numerical solution of time dependent PDEs, it is the combination of all of them

$(N_x \times N_y)$	$\epsilon = 0.04$		$\epsilon = 0.02$		$\epsilon = 0.01$	
$\times N_t$	$\lvert R - R_\epsilon \rvert$	CR	$\lvert R - R_\epsilon \rvert$	CR	$\lvert R - R_\epsilon \rvert$	CR
Accuracy test of space discretization, $r_2 = 2$						
$(64^2) \times 2048$	6.3540e-03	-	2.2263e-02	-	2.4972e-01	-
$(128^2) \times 2048$	1.6170e-03	1.97	5.7360e-03	1.96	2.1917e-02	3.51
$(256^2) \times 2048$	4.0200e-04	2.01	1.4340e-03	2.00	5.5740e-03	1.96
$(512^2) \times 2048$	9.6000e-05	2.07	3.4300e-04	2.06	1.3410e-03	2.06
$(1024^2) \times 2048$	1.9000e-05	2.34	6.9000e-05	2.31	2.6900e-04	2.32
Accuracy test of time discretization, $r_2 = 0$						
$(2048^2) \times 32$	2.4904e-01	-	2.8746e-01	-	2.9673e-01	-
$(2048^2) \times 64$	2.0421e-01	0.29	2.7345e-01	0.07	2.9336e-01	0.02
$(2048^2) \times 128$	1.5076e-01	0.44	2.4567e-01	0.15	2.8652e-01	0.03
$(2048^2) \times 256$	1.0162e-01	0.57	2.0064e-01	0.29	2.7250e-01	0.07
$(2048^2) \times 512$	6.3629e-02	0.68	1.4699e-01	0.45	2.4472e-01	0.16
$(2048^2) \times 1024$	3.7442e-02	0.77	9.7837e-02	0.59	1.9968e-01	0.29
Accuracy test of time discretization, $r_2 = 1$						
$(2048^2) \times 32$	8.8168e-02	-	1.8886e-01	-	2.5818e-01	-
$(2048^2) \times 64$	2.9907e-02	1.56	1.0137e-01	0.90	1.9996e-01	0.37
$(2048^2) \times 128$	7.9590e-03	1.91	3.5022e-02	1.53	1.1826e-01	0.76
$(2048^2) \times 256$	1.9280e-03	2.05	8.3870e-03	2.06	4.5675e-02	1.38
$(2048^2) \times 512$	4.5700e-04	2.08	1.7850e-03	2.23	1.1147e-02	2.03
$(2048^2) \times 1024$	1.0300e-04	2.14	3.7300e-04	2.26	2.1230e-03	2.39
Accuracy test of time discretization, $r_2 = 2$						
$(2048^2) \times 32$	3.5615e-02	-	1.3900e-01		2.4648e-01	-
$(2048^2) \times 64$	4.4430e-03	3.00	1.0619e-02	3.71	1.2014e-01	1.04
$(2048^2) \times 128$	2.0070e-03	1.15	9.2180e-03	0.20	4.2927e-02	1.49
$(2048^2) \times 256$	6.2600e-04	1.68	3.3000e-04	4.80	3.2818e-02	0.39
$(2048^2) \times 512$	1.6500e-04	1.92	2.1000e-04	0.65	2.9800e-03	3.46
$(2048^2) \times 1024$	3.2000e-05	2.37	9.3000e-05	1.18	1.7000e-04	4.13

Table 6 Errors and convergence rates on the radius of the shrinking circle at the final time T of Example 5 using the fEIF scheme.

together that makes the numerical method given here particularly attractive, as demonstrated through numerical tests.

We will consider extension of the proposed fEIF method to other interesting types of partial differential equations such as those involving high order spatial differential operators. Note that the fEIF scheme can be naturally generalized and applied to solving the fourth-order Cahn-Hilliard equation. Development of adaptive algorithms in time (and/or space) and use of suitable higher order accurate spatial discretizations for the fEIF schemes are also important research problems. In addition, given that solutions of nonlinear Allen-Cahn or Ginzburg-Landau type equations may exhibit very complex behavior, thus requiring a large number of unknowns to be used, efficient parallel implementation of the schemes is also very interesting works to be further explored.

References

1. S. Allen and J. W. Cahn, A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening, *Acta Metall.*, **27**, 1979, pp. 1084-1095.
2. L.-Q. Chen and J. Shen, Applications of semi-implicit Fourier-spectral method to phase field equations, *Comput. Phys. Comm.*, **108**, 1998, pp. 147-158.
3. S. Cox and P. Matthews, Exponential time differencing for stiff systems, *J. Comput. Phys.*, **176**, 2002, pp. 430-455.
4. Q. Du, C. Liu and X. Wang, A phase field approach in the numerical study of the elastic bending energy for vesicle membranes, *J. Comput. Phys.*, **198**, 2004, pp. 450-468.
5. Q. Du, and W.-X. Zhu, Stability analysis and applications of the exponential time differencing schemes and their contour integration modifications, *J. Comput. Math.*, **22**, 2004, pp. 200-209.
6. Q. Du, and W.-X. Zhu, Analysis and applications of the exponential time differencing schemes, *BIT Numer. Math..*, **45**, 2005, pp. 307-328.
7. L.C. Evans and J. Spruck, Motion of level sets by mean curvature. I, *J. Diff. Geom.*, **33**, 1991, pp. 635-681.
8. X. Feng and A. Prohl, Numerical analysis of the Allen-Cahn equation and approximation for mean curvature flows, *Numer. Math.*, **94**, 2003, pp. 33-65.
9. B. Gustafsson, H.-O. Kreiss, and J. Oliger, *Time Dependent Problems and Difference Methods*, Wiley-Interscience, New York, 1996.
10. M. Hochbruck, C. Lubich, and H. Selhofer, Exponential integrators for large systems of differential equations, *SIAM J. Sci. Comput.*, **19**, 1998, pp. 1552-1574.

11. M. Hochbruck and A. Ostermann, Exponential integrators, *Acta Numerica*, **19**, 2010, pp. 209-286.
12. D. Kessler, R. H. Nochetto, and A. Schmidt, A posteriori error control for the Allen-Cahn problem: Circumventing Gronwall's inequality, *Math. Model. Numer. Anal.*, **38**, 2004, pp. 129-142.
13. S. Krogstad, Generalized integrating factor methods for stiff PDEs, *J. Comput. Phys.*, **203**, 2005, pp. 72-88.
14. C. V. Loan, *Computational Frameworks for the Fast Fourier Transform*, SIAM, 1992.
15. Y. Li, H.-G. Lee, D. Jeong, and J. Kim, An unconditionally stable hybrid numerical method for solving the Allen-Cahn equation, *Comput. Math. Appl.*, **60**, 2010, pp. 1591-1606.
16. P. de Mottoni and M. Schatzman, Geometrical evolution of developed interfaces, *Trans. Amer. Math. Soc.*, **347**, 1989, pp. 1533-1589.
17. Q. Nie, F. Wan, Y.-T. Zhang and X.-F. Liu, Compact integration factor methods in high spatial dimensions, *J. Comput. Phys.*, **227**, 2008, pp. 5238-5255.
18. Q. Nie, Y.-T. Zhang and R. Zhao, Efficient semi-implicit schemes for stiff systems, *J. Comput. Phys.*, **214**, 2006, pp. 521-537.
19. D. Wang, L. Zhang, and Q. Nie, Array-representation integration factor method for high-dimensional systems, *J. Comput. Phys.*, **258**, 2014, pp. 585-600.
20. A. Wiegmann, Fast Poisson, Fast Helmholtz and fast linear elastrostatic solvers on rectangular parallelepipeds, Lawrence Berkeley National Laboratory, Paper #LBNL-43565, 1999.
21. C. Xu and T. Tang, Stability analysis of large time-stepping methods for epitaxial growth models, *SIAM J. Numer. Anal.*, **44**, 2006, pp. 1759-1779.
22. X. Yang, J. Feng, C. Liu and J. Shen, Numerical simulations of jet pinching-off and drop formation using an energetic variational phase-field method, *J. Comput. Phys.*, **218**, 2007, pp. 417-428.
23. L.-B. Zhang, Un schéma de semi-discrétisation en temps pour des systèmes différentiels discretises en espace par la méthode de Fourier résolution numérique des équations de Navier-Stokes stationnaires par la méthode multigrille, Ph.D. Dissertation, Universit Paris-Sud XI, 1987.
24. J. Zhang and Q. Du, Numerical studies of discrete approximations to the Allen-Cahn equation in the sharp interface limit, *SIAM J. Sci. Comput.*, **31**, 2009, pp. 3042-3063.