Fractals, Vol. 33, No. 9 (2025) 2550085 (16 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0218348X25500859

AN EFFICIENT OPERATOR SPLITTING METHOD FOR A NORMALIZED TIME-FRACTIONAL ALLEN–CAHN EQUATION

JIAN WANG, QIN LIU and KEYONG CHEN
School of Mathematics and Statistics
Nanjing University of Information Science and Technology
Nanjing 210044, P. R. China

JUNXIANG YANG
School of Computer Science and Engineering
Faculty of Innovation Engineering, Macau University of Science and Technology, Macao 999078, P. R. China

ZIWEI HAN
School of Mathematics and Statistics
Nanjing University of Information Science and Technology
Nanjing 210044, P. R. China

SOOBIN KWAK, YUNJAE NAM, SEOKJUN HAM and JUNSEOK KIM*
Department of Mathematics, Korea University
Seoul 02841, Republic of Korea
cfdkim@korea.ac.kr

Received February 9, 2025
Accepted May 17, 2025
Published July 7, 2025

*Corresponding author.

Abstract

In this paper, we propose a normalized time-fractional Allen–Cahn (TFAC) equation, in which a time-fractional derivative replaces the conventional derivative. We apply an efficient operator splitting technique to discretize the normalized TFAC equation. Compared to the conventional AC equation, the normalized TFAC equation features a unique time scale. This unique time scale provides an intuitive perspective on the fractional time derivative, as it represents a weighted average of the temporal history of the derivative. Moreover, the total integration of the weighting function is always 1 at all times. To study the dynamic characteristics of the computational solutions of the normalized TFAC equation, we investigated the mean curvature motion with a circular initial condition under different cases. Additionally, we applied the equation to more complex shapes to observe the differences in evolution over time between the normalized TFAC equation and the traditional AC equation. The experimental results show that the parameter significantly influences the numerical solutions. By adjusting the parameter, we can control the evolution rate to achieve the desired behavior.

Keywords: Normalized Time-Fractional Allen–Cahn Equation; Operator Splitting Method; Motion by Mean Curvature.

1. INTRODUCTION

The Allen–Cahn (AC) equation, which was first introduced by Allen and Cahn in 1979 to study the dynamics of curved antiphase boundaries, is fundamentally a reaction–diffusion equation that models phase separation and interfacial motion in various scientific fields such as physics and materials science.[1,2] Yu *et al.*[3] developed an averaged L1 scheme to solve the fractal mobile/immobile AC equation with a Caputo temporal derivative of order on graded meshes. To recover pattern boundaries from original, potentially noisy images or signals, Beneš proposed an algorithm for image segmentation using the AC equation.[4] Yang *et al.*[5] proposed a modified AC equation with a space-dependent interfacial parameter. The proposed equation used a triangular mesh to effectively deal with multi-scale problems, and reduced the computational cost required to solve the AC equation. In order to find some schemes reconstructing the surface well even in the case of insufficient data, Wang *et al.*[6] introduced an innovative and efficient method for surface reconstruction from unorganized point cloud data in three-dimensional Euclidean space. Hwang *et al.*[7] proposed a fully explicit finite difference method (FDM) for solving the conservative AC equation on a cubic surface, which incorporates a two-step scheme that ensures both computational efficiency and mass conservation. Numerical experiments demonstrated that the method preserves discrete mass and captures constrained motion driven by mass-conserving mean curvature. Yao and Azaiez[8] obtained a new phase field model by coupling the energy equation with the AC equation. This phase field model can describe the solidification and melting phenomena with enhancers from a microscopic perspective. Adaptive mesh techniques are necessary in finding the solutions of the AC equation. In addition, using a new unified technique and Maple software, 54 new precise solutions to the AC equation were given. The solutions included a variety of soliton behaviors such as dark compactons, dark solitons, periodic kinks, rough waves, periodic patterns with peaked crests and troughs. Compared with previous work using the first integral method, which obtained only eight solutions, the unified approach provided more innovative results.[9]

However, the conventional AC equation did not have a memory effect. Shafiq *et al.*[10] used cubic B-spline functions to study the memory effect from the Caputo–Fabrizio time fractional diffusion equation. The Caputo–Fabrizio fractional derivative and the FDM were used to discrete space and time respectively. To ensure the errors were small in the computational process, they analyzed the stability of the algorithm and verified that the algorithm was second-order convergent both in time and space. Sohaib *et al.*[11] numerically investigated space fractional AC equation in phase separation. Similarly, the time-fractional Allen–Cahn (TFAC) equation, which applied the time-fractional derivative such as incorporating memory facts and history dependence connected with time was proposed.[12–14] Over recent decades, a variety of researches has been conducted on the TFAC equation. Olshanskii *et al.*[15] proposed an AC equation defined on a surface that varies with time, which serves as a model for phase separation and the transition from order to disorder in thin material layers. They also used a finite element method, known as trace FEM, to obtain a numerical solution for the equation. Lee and Kim[16] investigated the TFAC equation on geometric computational domains in the Caputo sense. Sohaib *et al.*[17] presented a numerical scheme to solve the TFAC equation with application in phase separation. Yuan and Zhang[18] studied long-term discrete dynamics of two-dimensional TFAC equation with Caputo fractional derivative. At the same time, an extended scalar auxiliary variable method was proposed to solve the nonlinear term in the TFAC equation and a numerical study was conducted to verify the stability of the scheme as well.[19] Similarly, the homotopy analysis method (HAM) was applied to the TFAC equation to solve its nonlinear term. After comparing results obtained using the HAM with the exact solution, the effectiveness and validity of the proposed scheme were verified.[20]

Besides, Jiang *et al.*[21] presented two highly efficient numerical schemes for the TFAC equation that preserved the maximum bound principle and energy dissipation in discrete settings. Jialili *et al.*[22] adopted the homotopy perturbation method to obtain an analytical solution, and showed that the complexity of the nonlinear problem diminished as the degree of the fractional derivative increased toward one. The (G'/G)-expansion method, which uses an auxiliary function $G(\xi)$ and its derivative $G'(\xi)/G(\xi)$ based on the TFAC equation, was used to obtain exact traveling wave solutions.[23] Furthermore, Guo *et al.*[24] proposed an efficient FDM for the TFAC equation in unbounded domains, discretized the temporal Caputo derivative using L2-1σ, and applied the Hermite–Galerkin spectral method for spatial approximation. Their results indicated that the energy of the TFAC equation decreased with the increase of α, which is the order of the Caputo derivative.[25] Yang and Zeng[26] implemented a convolution quadrature scheme with linear stabilization to tackle TFAC equation, and ensured stability by defining the time step size and discretizing space with the central difference method. Now, we present the normalized TFAC equation:

$$\frac{\partial^\beta \phi}{\partial t^\beta} = -\frac{F'(\phi)}{\epsilon^2} + \Delta\phi \quad \text{in } \Omega, \tag{1}$$

$$\mathbf{n}\cdot\nabla\phi = 0 \quad \text{on } \partial\Omega, \tag{2}$$

where $\phi = \phi(x,y,t)$ is the phase-field function, $F(\phi) = 0.25(\phi^2-1)^2$, $\Delta\phi = \phi_{xx} + \phi_{yy}$ is the diffusion term, $\mathbf{n}$ is normal to $\partial\Omega$, and

$$\frac{\partial^\beta \phi(x,t)}{\partial t^\beta} = \frac{1-\beta}{t^{1-\beta}} \int_0^t \frac{\partial \phi(x,s)}{\partial s} \frac{ds}{(t-s)^\beta}, \quad 0<\beta<1, \tag{3}$$

which satisfies

$$\frac{1-\beta}{t^{1-\beta}} \int_0^t \frac{ds}{(t-s)^\beta} = 1, \quad 0<\beta<1. \tag{4}$$

In this study, we use the simplest fourth-order double-well potential from among the class of general high-order polynomial potentials.[27] Let us define a weighting function $w_\beta^t(s)$ as follows:

$$w_\beta^t(s) = \frac{1-\beta}{t^{1-\beta}(t-s)^\beta}. \tag{5}$$

Then, from Eq. (4), we have $W_\beta(t) = \int_0^t w_\beta^t(s)\, ds = 1$. Figure 1a shows the behavior of $w_\beta^1(s)$ for fractional orders $\beta = 0.1$, 0.5, and 0.9, and it highlights the influence of β on the memory effect of the normalized time-fractional derivative. The function $w_\beta^1(s)$ measures the contribution of past states $s \in (0,1)$ at fixed time $t = 1$. As β increases, the weighting function becomes more concentrated near $s = 1$, which implies greater emphasis is placed on recent history. For $\beta = 0.1$, the function is nearly flat, which indicates that all past states contribute almost equally (long-term memory). For $\beta = 0.5$, the curve gradually increases toward $s = 1$, which reflects moderate memory localization. For $\beta = 0.9$, the function sharply rises near $s = 1$, which emphasizes short-term memory. These results demonstrate how adjusting β controls the temporal weighting, which in turn governs the evolution speed and memory characteristics of the system. Figure 1b displays $w_{0.7}^t(s)$ at various fixed times, which shows how the weighting function evolves as time progresses. This plot illustrates how the parameter t influences the distribution of memory, with the function placing varying emphasis on recent versus past states. To ensure that the weighting function maintains a unit integral, its shape changes with respect to time t.

Lazopoulos introduced Eq. (3), referred to as the L-fractional derivative, which has a significant

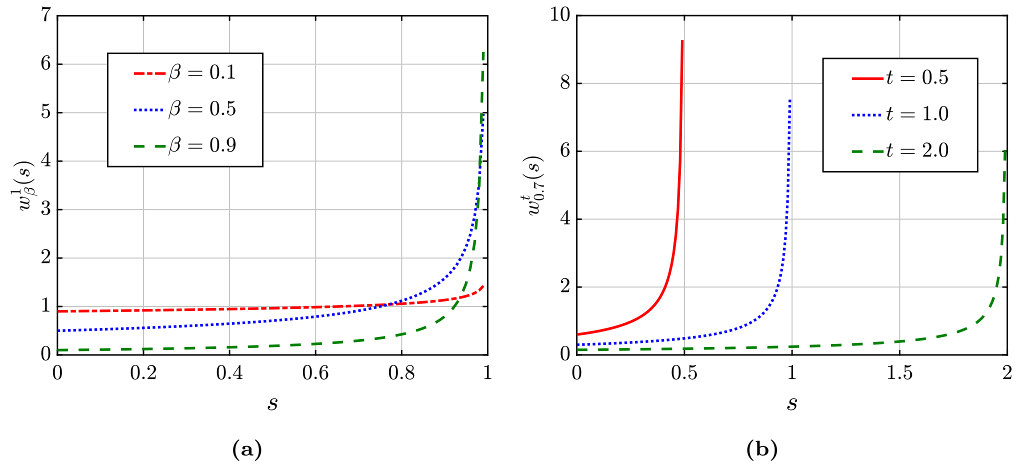


Fig. 1 **(a)** $w^1_\beta(s)$ for various values of β. **(b)** $w^t_{0.7}(s)$ with various times.

geometrical interpretation; see Refs. 28–31 for further details and additional references. The L-fractional derivative is given by

$$ {}^L_0D^\beta_t u(t) = \frac{{}^C_0D^\beta_t u(t)}{{}^C_0D^\beta_t t}, \tag{6}$$

where ${}^C_0D^\beta_t u(t)$ denotes the classical Caputo derivative, defined as follows:

$$ {}^C_0D^\beta_t u(t) = \frac{1}{\Gamma(1-\beta)}\int_0^t \frac{u'(s)}{(t-s)^\beta}\,ds. \tag{7}$$

Furthermore, for the derivation of the normalized time-fractional derivative (3) from the perspective of a weighting function with a unity integral, see Ref. 32.

The main purpose of this paper is to propose a normalized time-fractional AC equation with a unit-integral weighting function and to investigate the influence of the fractional order parameter on interface dynamics and phase evolution using a standard numerical method.

The structure of this paper is as follows. Section 2 describes computational methods for the normalized TFAC equation. Section 3 presents computational tests. Last, Sec. 4 discusses the conclusions and outlines potential directions for future research.

2. COMPUTATIONAL METHODS

Let $\Omega = (L_x,\ R_x)\times(L_y,\ R_y)$, which is discretized as follows: $\Omega_h = \{(x_i, y_j)|x_i = L_x + (i-0.5)h,\ 1 \le i \le N_x$ and $y_j = L_y + (j-0.5)h,\ 1 \le j \le N_y\}$, where $h = (R_x - L_x)/N_x$ for some positive integer N_x. Let $\phi^n_{ij} = \phi(x_i, y_j, t_n)$ and $t_n = (n-1)\Delta t$, then we have

$$\begin{aligned}
&\frac{\partial^\beta \phi(x_i,y_j,t_{n+1})}{\partial t^\beta}\\
&\quad= \frac{1-\beta}{t^{1-\beta}_{n+1}}\sum_{p=1}^{n}\int_{t_p}^{t_{p+1}} \frac{\partial\phi(x_i,y_j,s)}{\partial s}\frac{ds}{(t_{n+1}-s)^\beta}\\
&\quad\approx \sum_{p=1}^{n}\frac{1-\beta}{t^{1-\beta}_{n+1}}\int_{t_p}^{t_{p+1}}\frac{ds}{(t_{n+1}-s)^\beta}\frac{\phi^{p+1}_{ij}-\phi^p_{ij}}{\Delta t}\\
&\quad= \sum_{p=1}^{n}\frac{(n+1-p)^{1-\beta}-(n-p)^{1-\beta}}{n^{1-\beta}}\frac{\phi^{p+1}_{ij}-\phi^p_{ij}}{\Delta t}\\
&\quad= \sum_{p=1}^{n} w^n_p\frac{\phi^{p+1}_{ij}-\phi^p_{ij}}{\Delta t}\\
&\quad= w^n_n\frac{\phi^{n+1}_{ij}-\phi^n_{ij}}{\Delta t} + \sum_{p=1}^{n-1} w^n_p\frac{\phi^{p+1}_{ij}-\phi^p_{ij}}{\Delta t}\\
&\quad\approx w^n_n\frac{\partial\phi(x_i,y_j,t_{n+1})}{\partial t} + \sum_{p=1}^{n-1} w^n_p\frac{\phi^{p+1}_{ij}-\phi^p_{ij}}{\Delta t},
\end{aligned}\tag{8}$$

where $w^n_p = [(n+1-p)^{1-\beta} - (n-p)^{1-\beta}]/n^{1-\beta}$, which satisfies $\sum_{p=1}^n w^n_p = 1$. We then rewrite the normalized TFAC equation as the following approximation equation:

$$\begin{aligned}
&\frac{\partial\phi(x_i,y_j,t_{n+1})}{\partial t}\\
&\quad= -\frac{F'(\phi(x_i,y_j,t_{n+1}))}{w^n_n\epsilon^2} + \frac{1}{w^n_n}\Delta\phi(x_i,y_j,t_{n+1})\\
&\qquad -\frac{1}{w^n_n}\sum_{p=1}^{n-1} w^n_p\frac{\phi^{p+1}_{ij}-\phi^p_{ij}}{\Delta t}.
\end{aligned}\tag{9}$$

To computationally solve Eq. (9), we use the operator splitting technique to break it down into the following three steps:

Step 1.

$$\frac{\partial \phi(x_i, y_j, t_{n+1})}{\partial t} = \frac{1}{w_n^n}\phi_{xx}(x_i, y_j, t_{n+1}) + \frac{1}{2}s_{ij}^n. \tag{10}$$

Step 2.

$$\frac{\partial \phi(x_i, y_j, t_{n+1})}{\partial t} = \frac{1}{w_n^n}\phi_{yy}(x_i, y_j, t_{n+1}) + \frac{1}{2}s_{ij}^n. \tag{11}$$

Step 3.

$$\frac{\partial \phi(x_i, y_j, t_{n+1})}{\partial t} = -\frac{F'(\phi(x_i, y_j, t_{n+1}))}{w_n^n \epsilon^2}. \tag{12}$$

Here, $s_{ij}^n = -\Sigma_{p=1}^{n-1} w_p^n(\phi_{ij}^{p+1} - \phi_{ij}^p)/(w_n^n \Delta t)$. We solve the equations in Steps 1 and 2 using the FDM as follows:

Step 1.

$$\frac{\phi_{ij}^* - \phi_{ij}^n}{\Delta t} = \frac{\phi_{i-1,j}^* - 2\phi_{ij}^* + \phi_{i+1,j}^*}{w_n^n h^2} + \frac{1}{2}s_{ij}^n. \tag{13}$$

Step 2.

$$\frac{\phi_{ij}^{**} - \phi_{ij}^*}{\Delta t} = \frac{\phi_{i,j-1}^{**} - 2\phi_{ij}^{**} + \phi_{i,j+1}^{**}}{w_n^n h^2} + \frac{1}{2}s_{ij}^n. \tag{14}$$

We use a second-order discretization for the spatial variable; however, higher-order discretizations, such as a compact scheme, may also be applied.[33] The resulting discrete equations are solved using the Thomas algorithm. The solution of Step 3 is as follows:

$$\phi_{ij}^{n+1} = \frac{\phi_{ij}^{**}}{\sqrt{[1-(\phi_{ij}^{**})^2]e^{-2\Delta t/(w_n^n\epsilon^2)} + (\phi_{ij}^{**})^2}}, \tag{15}$$

which can be obtained using separation of variables.[34] These procedures are summarized in Algorithm 1.. The numerical scheme we used has second-order accuracy in space. For a detailed analysis of the numerical scheme, see Refs. 35 and 36.

3. NUMERICAL EXPERIMENTS

3.1. Thickness of the Interface Transition Layer with Various β Values

We investigate the equilibrium state of the transition layer for various β values. In the equilibrium state, the ϵ_β values for each β are interpolated based on the spatial step size. An initial condition

$$\phi(x, 0) = \begin{cases} -1 & (x \le 0.5), \\ 1 & \text{otherwise} \end{cases} \tag{16}$$

is taken in $\Omega = (0, 1)$. Parameters $h = 1/128$ and $\Delta t = 0.01h^2$ were used. Let us assume that it is in the equilibrium state if $\|\phi^{n+1} - \phi^n\|_2 < 2.5\text{e–}5$. The ϵ_β values for $\beta = 0.1,\ 0.2, \ldots, 0.9$ are as follows:

$$\epsilon_{\beta=0.1}(\xi) = 0.3761\xi - 0.0010, \tag{17}$$

$$\epsilon_{\beta=0.2}(\xi) = 0.3440\xi - 0.0009, \tag{18}$$

$$\epsilon_{\beta=0.3}(\xi) = 0.3278\xi - 0.0011, \tag{19}$$

$$\epsilon_{\beta=0.4}(\xi) = 0.3244\xi - 0.0018, \tag{20}$$

$$\epsilon_{\beta=0.5}(\xi) = 0.3213\xi - 0.0023, \tag{21}$$

$$\epsilon_{\beta=0.6}(\xi) = 0.3181\xi - 0.0026, \tag{22}$$

$$\epsilon_{\beta=0.7}(\xi) = 0.3146\xi - 0.0027, \tag{23}$$

Algorithm 1. Operator splitting scheme for the normalized TFAC equation.

```
1: Initialize $\phi_{ij}^0$ and set parameters: $\Delta t$, $h$, $\beta$, $\epsilon$
2: for $n = 1$ to $N_t$ do
3:     Compute weights $w_p^n = \frac{(n+1-p)^{1-\beta} - (n-p)^{1-\beta}}{n^{1-\beta}}$ for $p = 1, \ldots, n$
4:     Compute memory term:
           $s_{ij}^n = -\frac{1}{w_n^n \Delta t}\sum_{p=1}^{n-1} w_p^n(\phi_{ij}^{p+1} - \phi_{ij}^p)$
5:     Step 1. Solve in $x$-direction:
           $\frac{\phi_{ij}^* - \phi_{ij}^n}{\Delta t} = \frac{\phi_{i-1,j}^* - 2\phi_{ij}^* + \phi_{i+1,j}^*}{w_n^n h^2} + \frac{1}{2}s_{ij}^n$
6:     Use Thomas algorithm to solve tridiagonal system for $\phi^*$
7:     Step 2. Solve in $y$-direction:
           $\frac{\phi_{ij}^{**} - \phi_{ij}^*}{\Delta t} = \frac{\phi_{i,j-1}^{**} - 2\phi_{ij}^{**} + \phi_{i,j+1}^{**}}{w_n^n h^2} + \frac{1}{2}s_{ij}^n$
8:     Use Thomas algorithm to solve tridiagonal system for $\phi^{**}$
9:     Step 3. Solve reaction term:
           $\phi_{ij}^{n+1} = \frac{\phi_{ij}^{**}}{\sqrt{[1-(\phi_{ij}^{**})^2]e^{-2\Delta t/(w_n^n\epsilon^2)} + (\phi_{ij}^{**})^2}}$
10: end for
```

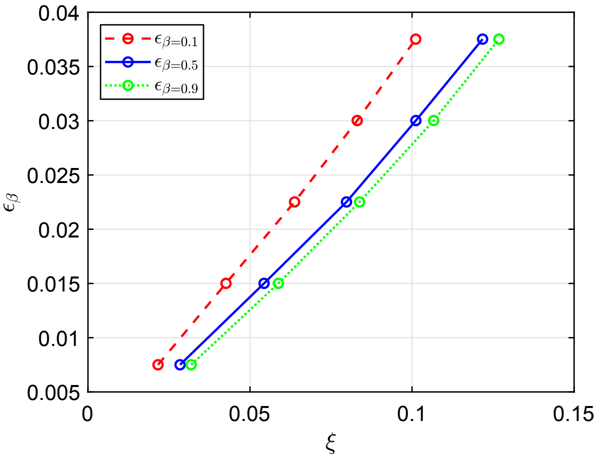


Fig. 2 Changes in ϵ over interface thickness ξ for different β values.

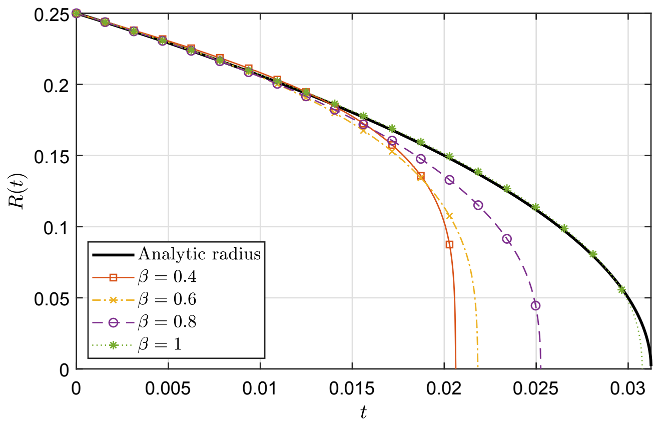


Fig. 3 Changes in radius over time for different β values.

$$\epsilon_{\beta=0.8}(\xi) = 0.3124\xi - 0.0029, \tag{24}$$

$$\epsilon_{\beta=0.9}(\xi) = 0.3154\xi - 0.0032, \tag{25}$$

where ξ is the thickness of the interface transition layer. The value of ϵ is approximated by Eqs. (17)–(25) as a function of interface thickness, depending on each β value. For $\beta = 1$, ϵ is given by $\epsilon_{\beta=1} = mh/2\sqrt{2}\text{atanh}(0.9)$, where m is a positive integer. In the following numerical experiments, the ϵ values corresponding to β are used. Figure 2 illustrates the variation of ϵ with the interface excess thickness ξ under different β values. We observe that for the same interface thickness ξ, as β increases, ϵ also increases. This indicates that, for the same interface thickness ξ, ϵ is significantly influenced by β, necessitating careful selection. In order to prevent the occurrence of mosaic phenomena, the interface thickness should be sufficiently large, approximately $4h$. If ϵ is not appropriately determined, the interface becomes pinned due to the pinning effect.[37]

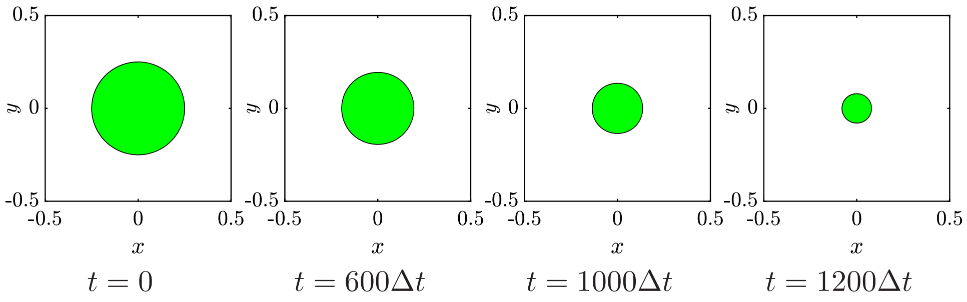
3.2. Motion by Mean Curvature

In this section, we observe the temporal evolution with the following initial condition:

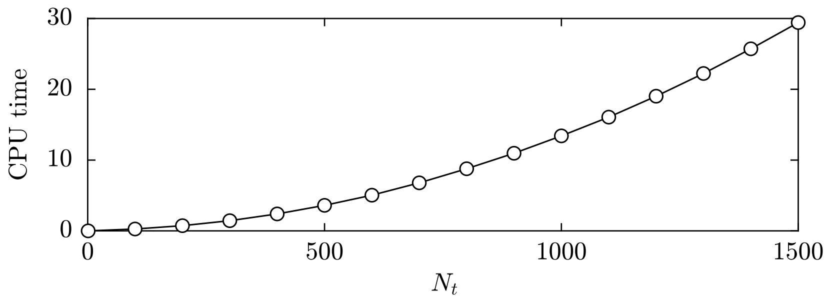
$$\phi(x, y, 0) = \tanh\left(\frac{r_0 - \sqrt{x^2 + y^2}}{\sqrt{2}\epsilon_\beta}\right),$$
$$\Omega = (-0.5, 0.5) \times (-0.5, 0.5), \tag{26}$$

where initial radius is set to $r_0 = 0.25$ for $\beta = 0.4$, 0.6, 0.8, and 1. We observe the zero-level contour of ϕ for different β values and investigate the radius $R(t)$ over time on the zero-level contour of ϕ. The analytical formula for the temporal evolution of the radius is given by $R(t) = \sqrt{(r_0)^2 - 2t}$.[38] We use the parameter values of $h = 0.01$, $\epsilon = \epsilon_\beta(10h)$, and $\Delta t = 0.2h^2$. Figure 3 shows a plot of the radius $R(t)$ of the zero-level contour of ϕ as a function of time t for different values of the parameter β. The goal of this test is to investigate how the radius evolves over time under different conditions governed by β, and it compares these numerical results with an analytic solution for the radius.

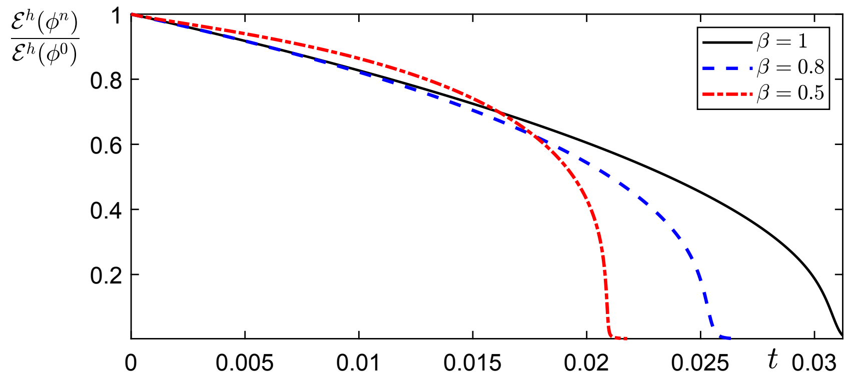
The different curves correspond to various values of β ($\beta = 0.4$, 0.6, 0.8, and 1). For each value of β, the radius $R(t)$ starts at approximately the same initial point and decreases over time, showing the shrinking of the solution domain. As β increases, the radius decreases more slowly, which indicates that a larger β slows down the shrinking process. For $\beta = 0.4$ (red squares), the radius rapidly decreases, and the curve terminates earlier than others, which shows that the solution vanishes or reaches zero much faster. For $\beta = 1$ (green circles), the radius decreases more gradually, and the curve continues further in time, meaning the system remains active for a longer period. Intermediate β values ($\beta = 0.6$ and $\beta = 0.8$) follow trends between these two extremes, with their radii decreasing faster than $\beta = 1$ but slower than $\beta = 0.4$. The black solid line represents the analytic radius $R(t)$, which is a reference solution for the case of $\beta = 1$. The numerical solutions for the various β values follow the analytic solution quite closely at first but start to deviate as time progresses. Larger β values (such as $\beta = 1$) stay closer to the analytic solution for a longer time, while smaller β values deviate earlier and show a faster decay. The analytic solution represents a more gradual decrease in radius, and the numerical solutions approximate this trend with



(a)



(b)



(c)

Fig. 4 **(a)** Snapshot results of the numerical solution at $\beta = 0.8$ for $t = 0$, $600\Delta t$, $1000\Delta t$, and $1200\Delta t$ with $\Delta t = 0.2h^2 \approx 0.00002$. **(b)** CPU time measured for the numerical experiment in **(a)**. **(c)** Temporal evolution of the normalized discrete energy with $\beta = 1,\ 0.8,\ 0.5$.

varying levels of accuracy depending on β. The fractional order parameter β controls the rate of decay of the radius. A smaller β leads to a faster collapse of the system, while a larger β results in a more gradual decay. The numerical results are fairly consistent with the analytic solution at first, but deviations appear as time progresses. The larger the β, the longer the numerical solution matches the analytic radius. This simulation result demonstrates how the radius of the system evolves over time for different β values and shows that higher β values lead to slower shrinkage, with each case compared to an analytic reference solution.

In order to provide an intuitive demonstration, numerical simulations were conducted using the parameters corresponding to $\beta = 0.8$ under the given conditions, and the experimental results are displayed in Fig. 4a (see Listing A.1 in Appendix A for code details). From Fig. 4a, it can be observed that as time increases, the shape starts from an initial circular form, with the radius continuously decreasing until it stabilizes. As the number of iterations over time increases, the weights and $\phi(x_i, y_j, t_n)$ that must be stored for calculations increase. Therefore, as the iteration increases, the computation time increases more steeply, as shown in Fig. 4b. For the energy analysis, we define the discrete total energy function as

$$\mathcal{E}^h(\phi^n) = h^2 \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} F(\phi_{ij}^n) + \frac{\epsilon^2}{2} \sum_{j=1}^{N_y} \sum_{i=1}^{N_x-1} (\phi_{i+1,j}^n - \phi_{ij}^n)^2 + \frac{\epsilon^2}{2} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y-1} (\phi_{i,j+1}^n - \phi_{ij}^n)^2$$

with the homogeneous Neumann boundary condition. Figure 4c shows the temporal evolution of the normalized discrete energy for given circular initial condition. We consider various values of β: $\beta = 1$, 0.8, and 0.5. The corresponding ϵ values are given by $\epsilon_{\beta=1} = mh/(2\sqrt{2}\tan^{-1}(0.9))$, $\epsilon_{\beta=0.8} = 0.3124mh - 0.0029$, and $\epsilon_{\beta=0.5} = 0.3213mh - 0.0023$, where $m = 8$.

Figure 5 illustrates the temporal evolution of numerical solutions for the normalized TFAC equation under different parameter values: $\beta = 0.5$ and $\beta = 1$. The corresponding values are $\epsilon_{\beta=0.5} = 0.2837mh - 0.001$ and $\epsilon_{\beta=1} = mh/(2\sqrt{2}\text{atanh}(0.9))$, where $m = 8$. In the computational domain for $\Omega = (0,1) \times (0,1)$, the grid size is $N = N_x = N_y = 256$, the spatial step size is $h = \frac{1}{N}$, and the time step size is $\Delta t = 0.245h^2$. The columns in Fig. 5 show the behavior of the system at different times: $t = 0$, $600\Delta t$, $1200\Delta t$, and $1500\Delta t$. In Fig. 5, rows a and b show the numerical solutions as a function of time for two different values of the parameter $\beta = 0.5$ and $\beta = 1$, respectively. At $t = 0$, the solutions for both $\beta = 0.5$ and $\beta = 1$ start as sharp square-like spirals. As time progresses, the spiral evolves and begins to round off and loses its sharp features. By $t = 1500\Delta t$, the spirals become very smooth. Figure 5c presents the overlaid zero-level contours for both values of β (black dotted line for $\beta = 1$ and green solid line for $\beta = 0.5$). As time progresses, the solutions diverge slightly, and the $\beta = 0.5$ solution tends to spread more than the $\beta = 1$ solution, which remains more tightly coiled even at

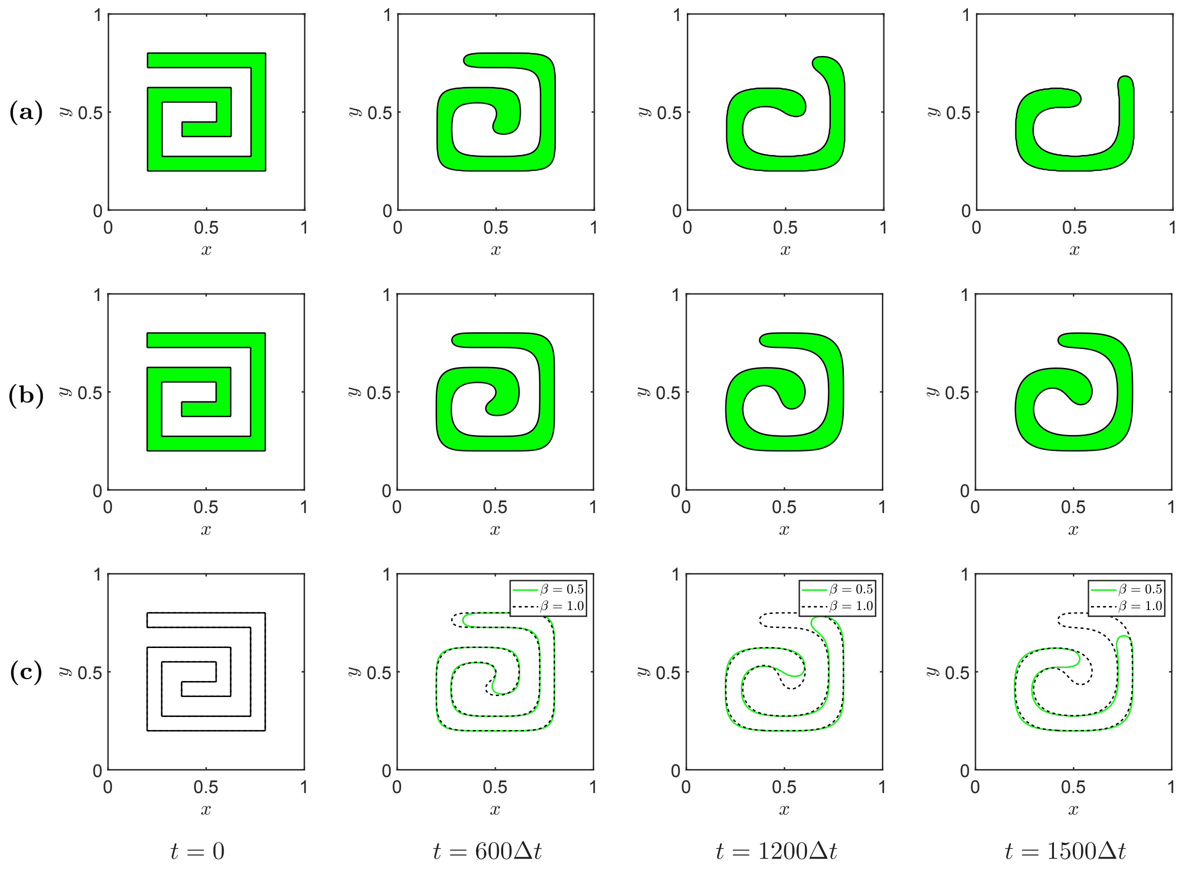


Fig. 5 Snapshot results of the numerical solutions with **(a)** $\beta = 0.5$ and **(b)** $\beta = 1$. **(c)** Overlaid zero-level contours for $\beta = 0.5$ (green solid line) and $\beta = 1$ (black dotted line). Here, the solutions are shown at $t = 0$, $600\Delta t$, $1200\Delta t$, and $1500\Delta t$ with $\Delta t = 0.245h^2 \approx 0.000003$.

$t = 1500\Delta t$. This indicates that the parameter β significantly influences the rate of spreading or opening of the spiral as time advances. The evolution of the spirals demonstrates how the system's dynamics change over time and how sensitive these dynamics are to the parameter β. The parameter $\beta = 0.5$ leads to a faster smoothing and spreading of the spiral pattern compared to $\beta = 1$. The zero-level contours help visualize how the shapes initially evolve in a similar manner, with little difference between the two cases in the early stages of evolution. However, as time progresses, the difference between the two cases becomes increasingly noticeable.

Figure 6 uses a puppy which has rounded edges to present the temporal evolution of numerical solutions under different parameter values. The initial shape and edges of the image are shown in the first column of Fig. 6. In this experiment, we chose β to be 0.3 and 1 to observe the evolution under different parameter values. The corresponding values are $\epsilon_{\beta=0.3} = 0.3278mh - 0.0011$ and $\epsilon_{\beta=1} = mh/(2\sqrt{2}\text{atanh}(0.9))$, where $m = 10$. The computational domain is $\Omega = (0,1)\times(0,1)$, with a spatial grid number of $N = N_x = N_y = 150$, a spatial step size of $h = \frac{1}{N}$, and a time step size of $\Delta t = 0.45h^2$. The columns in Fig. 6 show the behavior of the system at different times: $t = 0$, $50\Delta t$, $100\Delta t$, and $200\Delta t$. In Fig. 6, rows a and b show the numerical solutions as a function of time for two different values of the parameter $\beta = 0.3$ and $\beta = 1$, respectively. At $t = 0$, the numerical solutions for both $\beta = 0.3$ and $\beta = 1$ start as an entire puppy shape. As time progresses, the puppy shape evolves and begins to round off and loses its sharp features, especially in the tail and legs. By $t = 200\Delta t$, the tails of the puppy disappear and the legs of the puppy become very smooth and short. Figure 6c presents

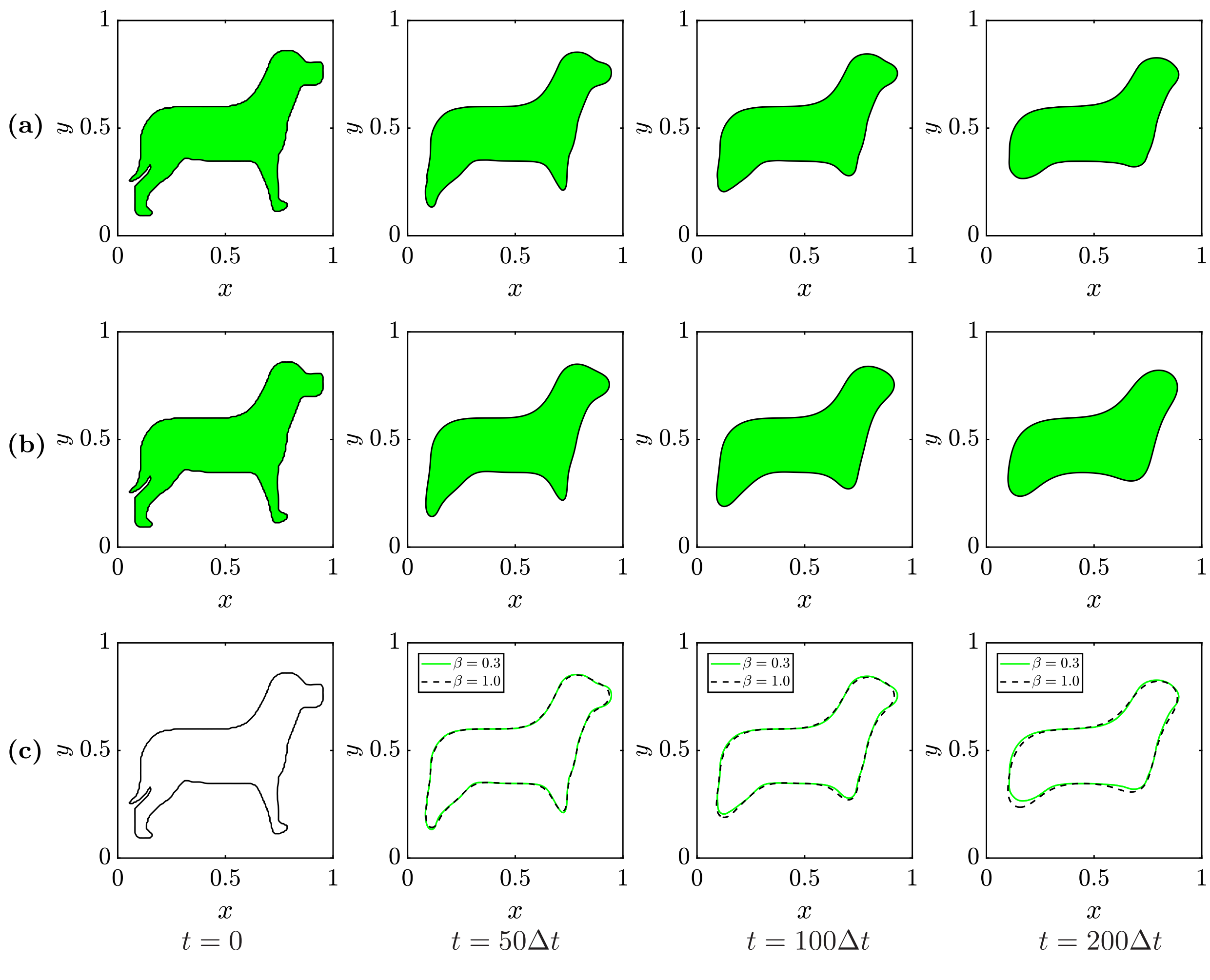


Fig. 6 Snapshot results of a puppy shape with **(a)** $\beta = 0.3$ and **(b)** $\beta = 1$. **(c)** Overlaid zero-level contours for $\beta = 0.3$ (green solid line) and $\beta = 1$ (black dotted line). Here, the solutions are shown at $t = 0$, $50\Delta t$, $100\Delta t$, and $200\Delta t$ with $\Delta t = 0.45h^2 \approx 0.00002$.

the overlaid zero-level contours for both values of β (black dotted line for $\beta = 1$ and green solid line for $\beta = 0.3$). As time progresses, the edges of the patterns in the two different cases begin to diverge gradually and show distinct changes. Compared to the case with $\beta = 1$, the contraction of the puppy shape is more pronounced when $\beta = 0.3$, and this difference becomes increasingly apparent as time advances.

In the subsequent experiment, we chose a star shape as the subject, whose initial state on $\Omega = (-1, 1) \times (-1, 1)$ is described by the following equation:

$$\phi(x, y, 0) = \tanh\left(\frac{2 + 0.5\cos(16\theta(x, y)) - \sqrt{(\pi x)^2 + (\pi y)^2}}{\sqrt{2}h}\right), \quad (27)$$

where

$$\theta(x, y) = \begin{cases} \tan^{-1}\left(\dfrac{y}{x}\right) & \text{if } x > 0, \\ \pi + \tan^{-1}\left(\dfrac{y}{x}\right) & \text{if } x \le 0. \end{cases} \quad (28)$$

In this experiment, we observe the changes in the star shape corresponding to $\beta = 0.2$ and $\beta = 1$. For these cases, $\epsilon_{\beta=0.2} = 0.3440mh - 0.0009$ and $\epsilon_{\beta=1} = mh/(2\sqrt{2}\text{atanh}(0.9))$, where $m = 20$. The number of spatial grid points is $N = N_x = N_y = 100$, the spatial step size is $h = \frac{2}{N}$, and the time step is $\Delta t = 0.3h^2$. Figure 7 represents the temporal evolution of a star shape under different parameter values: $\beta = 0.2$ and $\beta = 1$. The columns in Fig. 7 show the behavior of the system at different times: $t = 0$, $500\Delta t$, $800\Delta t$, and $1000\Delta t$. In Fig. 7, rows a and b show the star shape as a function of time for two different values of the parameter $\beta = 0.2$ and $\beta = 1$, respectively. At $t = 0$, the numerical solutions for

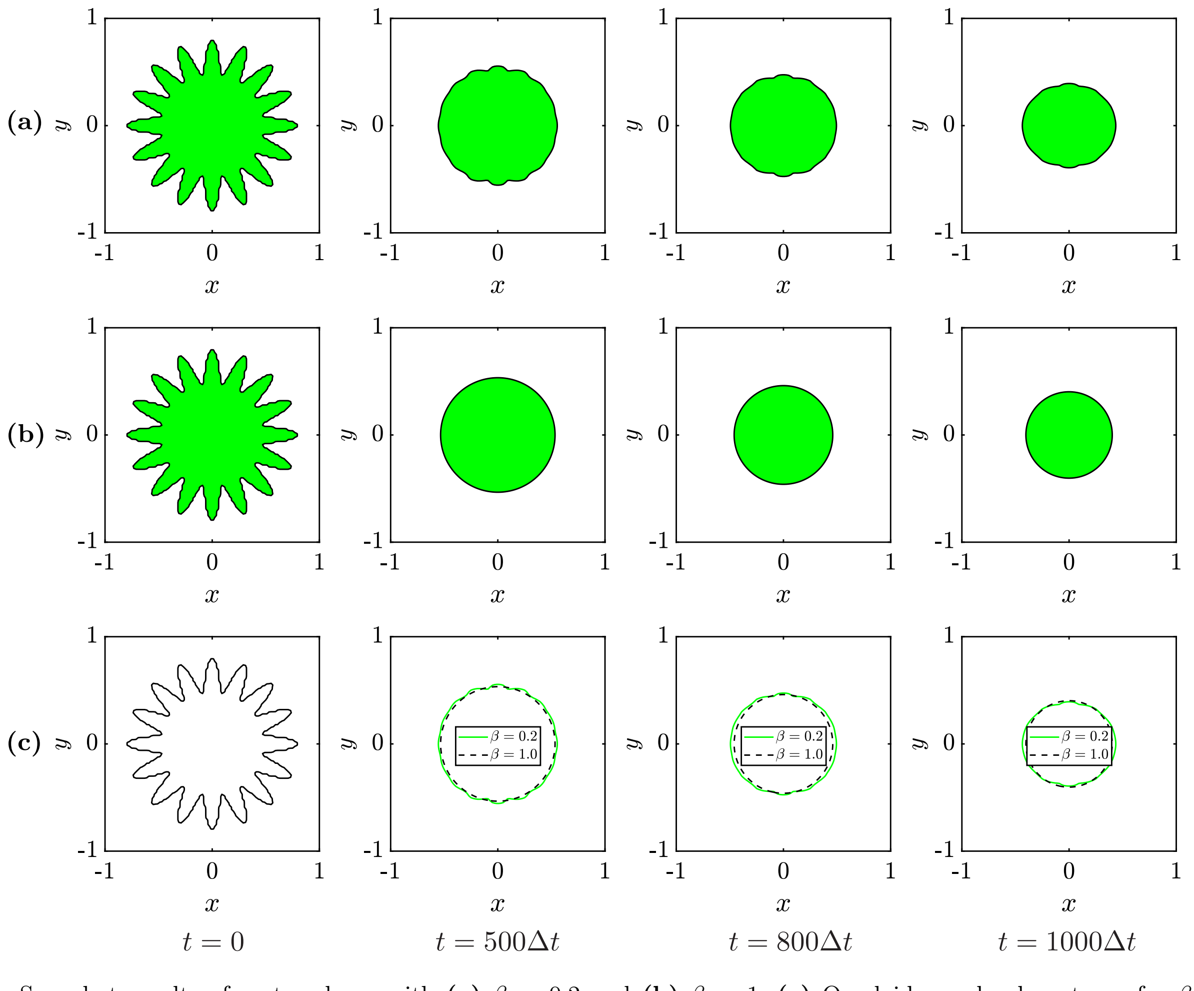


Fig. 7 Snapshot results of a star shape with **(a)** $\beta = 0.2$ and **(b)** $\beta = 1$. **(c)** Overlaid zero-level contours for $\beta = 0.2$ (green solid line) and $\beta = 1$ (black dotted line). Here, the solutions are shown at $t = 0$, $500\Delta t$, $800\Delta t$, and $1000\Delta t$ with $\Delta t = 0.3h^2 \approx 0.00012$.

both $\beta = 0.2$ and $\beta = 1$ start as a star with some narrow and tall curved corners. As time progresses, the curved corners of the star shape evolve, begin to shrink, and lose their initial height. By $t = 1000\Delta t$, the curved corners of the star shape become very smooth. Figure 7c presents the overlaid zero-level contours for both values of β (black dotted line for $\beta = 1$ and green solid line for $\beta = 0.2$). As time progresses, the solutions diverge slightly, and the $\beta = 0.2$ solution tends to spread more than the $\beta = 1$ solution, which remains more tightly coiled even at $t = 1000\Delta t$. This verifies the conclusions drawn from the experiments above.

3.3. Phase Separation

We consider an initial condition on the domain $\Omega = (0, 1) \times (0, 1)$ as

$$\phi(x, y, 0) = 0.001 \cos(5\pi(x - 1)^2) \cos(5\pi(y - 1)^2).$$

The parameters are used as $h = 1/128$ and $\Delta t = 0.1h^2$. Figure 8 presents the temporal evolution of the normalized TFAC for $\beta = 0.3$, 0.7, and 1, which shows the process of phase separation. For each case of $\beta = 0.3$, 0.7, and 1, the corresponding values of ϵ are approximately 0.0148, 0.0102, and 0.0078, respectively. As results, the dynamics of phase separation are observed in this numerical experiment. Figure 9 shows the contour plots obtained from the simulations in Fig. 8. As shown in Figs. 8 and 9, a larger value of β leads to a faster phase separation, whereas the pronounced coarsening occurs when the value of β is smaller. In the phase separation process, the zero Neumann boundary condition is used, and a perpendicular effect, where the interface meets the domain boundary at a perpendicular contact angle, is observed at the boundary.

3.4. Comparison with Caputo Derivative

We investigate the difference in the dynamics between the normalized TFAC and the Caputo

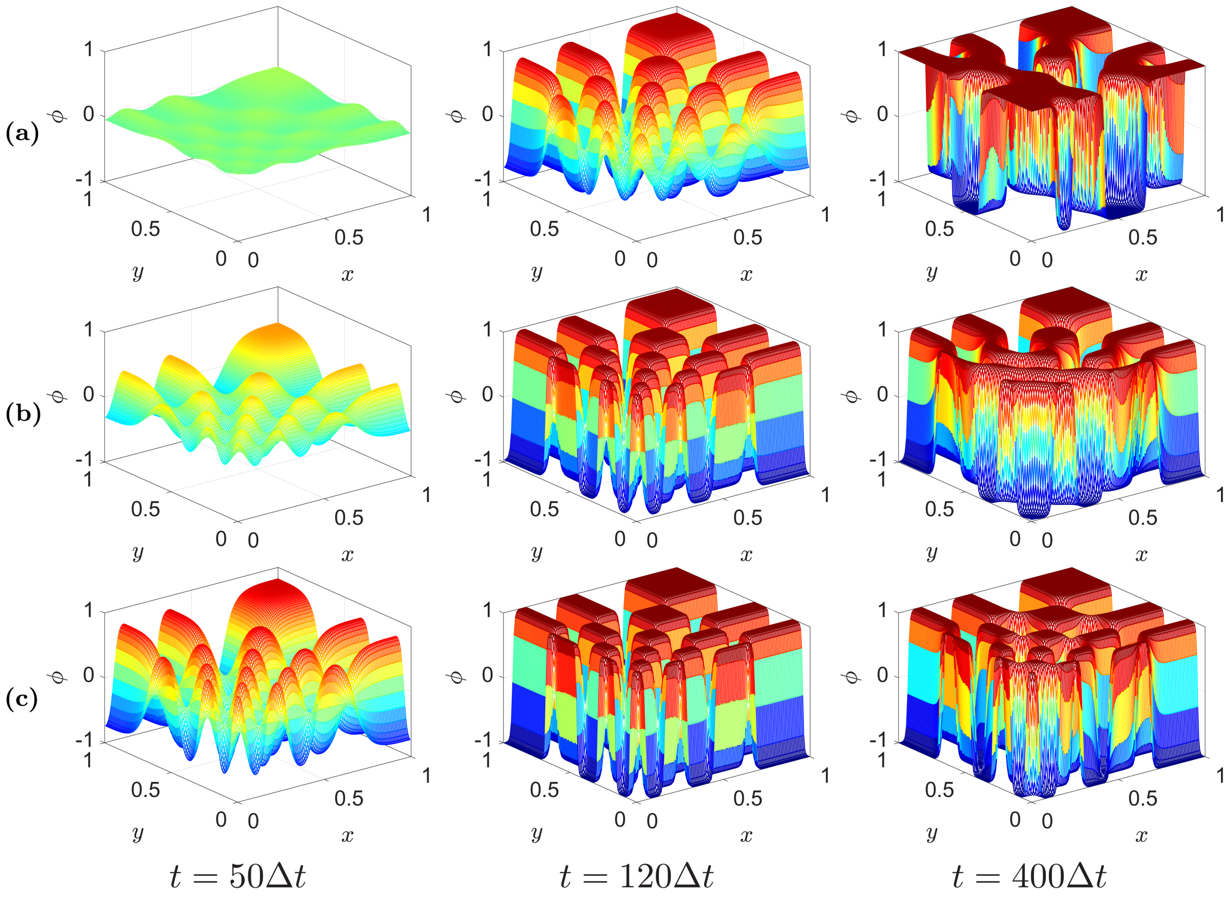


Fig. 8 The results of the temporal evolution for **(a)** $\beta = 0.3$, **(b)** $\beta = 0.7$, and **(c)** $\beta = 1$ with $\Delta t = 0.1h^2 \approx 0.000006$.

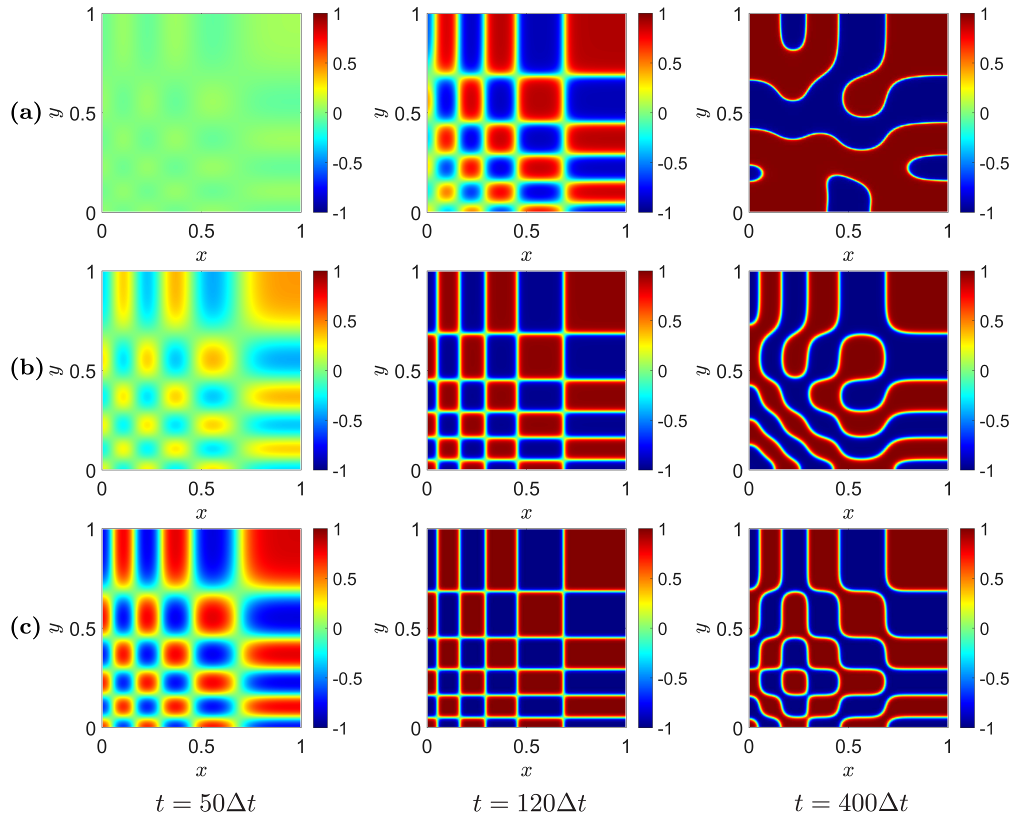


Fig. 9 The contour result is presented in Fig. 8. **(a)** $\beta = 0.3$, **(b)** $\beta = 0.7$, and **(c)** $\beta = 1$ with $\Delta t = 0.1h^2 \approx 0.000006$.

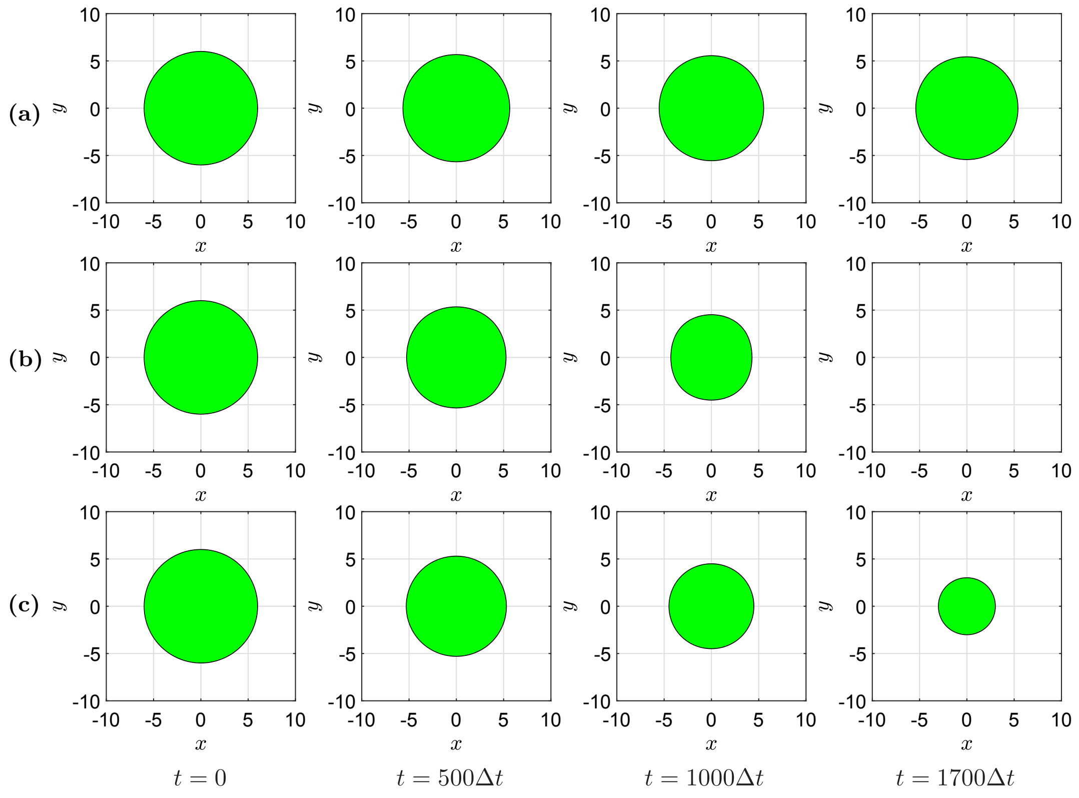


Fig. 10 The comparison of evolution results using **(a)** Caputo derivative, **(b)** the normalized time fractional derivative, and **(c)** the traditional AC with $\Delta t = 0.2h^2 \approx 0.008$.

derivative. The initial condition is set as follows:

$$\phi(x,y,0) = \tanh\left(\frac{6-\sqrt{x^2+y^2}}{\sqrt{2}\epsilon}\right), \quad (x,y)\in\Omega,$$

where the domain is given by $\Omega = (-10,10)\times(-10,10)$. The parameters are used as $h = 0.2$, $\beta = 0.4$, $\epsilon = \epsilon_\beta(10h)$, and $\Delta t = 0.2h^2$. Figure 10 presents the temporal evolution of the TFAC using the Caputo derivative and the normalized time-fractional derivative. Compared to the other two results shown in Figs. 10b and 10c, in the case of the Caputo derivative (Fig. 10a), the interface appears to be pinned during the long-term simulation. This behavior is attributed to the effect of the weighting function, which is not normalized in the Caputo derivative.

4. CONCLUSION

In this paper, we introduced a normalized TFAC equation based on the time-fractional derivative β. The normalized TFAC equation can determine the speed of the evolutionary process according to specific requirements. This equation differs from the traditional AC equation in several aspects. The normalized TFAC equation ensures that $W_\beta(t) = \int_0^t w_\beta^t(s)ds = 1$. Additionally, when β is small, the weighting function $w_\beta^t(s)$ remains relatively flat, while for larger values of β, it shows a sharp transition near time t. We conducted numerical experiments to study the dynamic characteristics of the numerical solutions of the normalized TFAC equation. We investigated the thickness of the interface transition layer for different values of $w_\beta^t(s)$. The results indicated that for the same interface thickness ξ, ϵ was significantly influenced by β. We further examined the changes in the radius of the solutions over time for different values of β, and we found that as time increases, the radius gradually decreases, with the rate of decrease accelerating. Moreover, a smaller $w_\beta^t(s)$ led to a faster collapse of the system. When $w_\beta^t(s) = 1$, the numerical solutions closely approximated the analytic solutions. Since in our normalized TFAC equation model, $\beta = 1$ corresponds to the traditional AC equation, we conducted experiments on different shapes, including spiral, puppy, and star patterns. Under the same initial conditions, compared to the traditional AC equation, the numerical solution of the corresponding TFAC equation with a smaller value of β evolves more significantly. In summary, the normalized TFAC model incorporates a fractional order parameter β that regulates the memory effect by determining the influence of past states on current dynamics. Smaller values of β represent long-term memory, while larger values prioritize recent information and results in faster morphological evolution. This behavior can be interpreted as a control of temporal diffusion or inertia, relevant in systems exhibiting memory or anomalous diffusion. Although this work emphasizes the mathematical and numerical aspects, the model provides a flexible framework for adapting β to match real-world phenomena, such as phase transitions in viscoelastic materials or varying levels of image smoothing. Future research will focus on establishing empirical connections and extending the proposed model to three-dimensional domains.

ACKNOWLEDGMENTS

Jian Wang would like to thank the Natural Science Foundation of Jiangsu Province (Grant No. BK20240689). Junseok Kim was supported by the Korea University Grant (No. K2504591). The authors sincerely appreciate the reviewers for their insightful and valuable comments, which have significantly contributed to the improvement of this paper.

ORCID

Jian Wang
https://orcid.org/0000-0003-2615-0755
Qin Liu
https://orcid.org/0009-0000-7822-6729
Keyong Chen
https://orcid.org/0009-0008-2759-5151
Junxiang Yang
https://orcid.org/0000-0003-3763-5459
Ziwei Han
https://orcid.org/0009-0004-6304-9634
Soobin Kwak
https://orcid.org/0000-0001-7693-8992
Yunjae Nam
https://orcid.org/0009-0004-6236-363X
Seokjun Ham
https://orcid.org/0000-0002-2917-312X
Junseok Kim
https://orcid.org/0000-0002-0484-9189

REFERENCES

1. S. M. Allen and J. W. Cahn, A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening, *Acta Metall.* **27** (1979) 1085–1095.
2. X. Feng and A. Prohl, Numerical analysis of the Allen–Cahn equation and approximation for mean curvature flows, *Numer. Math.* **94** (2003) 33–65.
3. F. Yu and M. Chen, Second-order error analysis for fractal mobile/immobile Allen–Cahn equation on graded meshes, *J. Sci. Comput.* **96**(2) (2023) 49.
4. M. Beneš, V. Chalupecký and K. Mikula, Geometrical image segmentation by the Allen–Cahn equation, *Appl. Numer. Math.* **51**(2–3) (2004) 187–205.
5. J. Yang, J. Wang, S. Kwak, S. Ham and J. Kim, A modified Allen–Cahn equation with a mesh size-dependent interfacial parameter on a triangular mesh, *Comput. Phys. Commun.* **304** (2024) 109301.
6. J. Wang and W. J. Jiang, Surface reconstruction algorithm using a modified Allen–Cahn equation, *Mod. Phys. Lett. B* **36**(26n27) (2022) 2250147.
7. Y. Hwang, Jyoti, S. Kwak, H. Kim and J. Kim, An explicit numerical method for the conservative Allen–Cahn equation on a cubic surface, *AIMS Math.* **9**(12) (2024) 34447–34465.
8. H. Yao and M. Azaiez, A monolithic model of solid–liquid phase change problem, *Comput. Methods Appl. Math.* **421** (2024) 116794.
9. N. A. Mohammed, M. N. Alam, I. Talib, D. L. C. Ching, T. A. Assiri, M. Hassan and I. Khan, Exact solutions of nonlinear fractional Cahn–Allen equation arises in different nonlinear physical phenomenon using unified technique, *Fractals* **33**(3) (2025) 2440061.
10. M. Shafiq, F. A. Abdullah, M. Abbas, A. Sm Alzaidi and M. B. Riaz, Memory effect analysis using piecewise cubic B-spline of time fractional diffusion equation, *Fractals* **30**(8) (2022) 2240270.
11. M. Sohaib, K. M. Furati and A. Shah, Space fractional Allen–Cahn equation and its applications in phase separation: A numerical study, *Commun. Nonlinear Sci. Numer. Simul.* **137** (2024) 108173.
12. Q. Du, Y. Jiang and Z. Zhi, Time-fractional Allen–Cahn equations: Analysis and numerical methods, *J. Sci. Comput.* **85**(2) (2020) 42.
13. H. Liu, A. Chen, H. Wang and J. Zhao, Time-fractional Allen–Cahn and Cahn–Hilliard phase-field models and their numerical investigation, *Comput. Math. Appl.* **76**(8) (2018) 1876–1892.
14. S. Habib, A. Batool, A. Islam, M. Nadeem, K. A. Gepreel and J. H. He, Study of nonlinear Hirota–Satsuma coupled KdV and coupled mKdV system with time fractional derivative, *Fractals* **29**(5) (2021) 2150108.
15. M. Olshanskii, X. Xu and V. Yushutin, A finite element method for Allen–Cahn equation on deforming surface, *Comput. Math. Appl.* **90** (2021) 148–158.
16. D. Lee and H. Kim, The time-fractional Allen–Cahn equation on geometric computational domains, *Commun. Nonlinear Sci. Numer. Simul.* **141** (2025) 108455.
17. M. Sohaib, A. Shah, K. M. Furati and H. Khaliq, A numerical scheme for time-fractional Allen–Cahn equation with application in phase separation, *Int. J. Comput. Math.* **102** (2025) 449–464.
18. W. Yuan and C. Zhang, Long-term dynamics of a stabilized time-space discretization scheme for 2D time-fractional Allen–Cahn equation with double well potential, *J. Comput. Appl. Math.* **448** (2024) 115952.
19. D. Hou, H. Zhu and C. Xu, Highly efficient schemes for time-fractional Allen–Cahn equation using extended SAV approach, *Numer. Algorithms* **88** (2021) 1077–1108.
20. A. Esen, N. M. Yagmurlu and O. Tasbozan, Approximate analytical solution to time-fractional damped Burger and Cahn–Allen equations, *Appl. Math. Inform. Sci.* **7**(5) (2013) 1951–1956.
21. H. Jiang, D. Hu, H. Huang and H. Liu, Linearly implicit schemes preserve the maximum bound principle and energy dissipation for the time-fractional Allen–Cahn equation, *J. Sci. Comput.* **101**(2) (2024) 1–27.
22. P. Jalili, B. Jalili, A. Shateri and D. D. Ganji, A novel fractional analytical technique for the time-space fractional equations appearing in oil pollution, *Int. J. Eng.-Iran* **35**(12) (2022) 2386–2394.
23. F. Batool and G. Akram, New solitary wave solutions of the time-fractional Cahn–Allen equation via the improved ($G'G\ GG$)-expansion method, *Eur. Phys. J. Plus* **133** (2018) 1–11.
24. S. Guo, L. Mei, Z. Zhang, J. Chen, Y. He and Y. Li, Finite difference/Hermite–Galerkin spectral method for multi-dimensional time-fractional nonlinear reaction–diffusion equation in unbounded domains, *Appl. Math. Model.* **70** (2019) 246–263.
25. F. A. Godínez, J. J. Rosales and M. F. Esparza-Posadas, Newton's law of cooling with Caputo derivative: Consistent dimensionality to compare with experiments, *Fractals* 30(9) (2022) 2250187.
26. Z. Yang and F. Zeng, A linearly stabilized convolution quadrature method for the time-fractional Allen–Cahn equation, *Appl. Math. Lett.* **144** (2023) 108698.
27. J. Choi, S. Ham, S. Kwak, Y. Hwang and J. Kim, Stability analysis of an explicit numerical scheme

for the Allen–Cahn equation with high-order polynomial potentials, *AIMS Math.* **9**(7) (2024) 19332–19344.
28. K. A. Lazopoulos and A. K. Lazopoulos, Fractional differential geometry of curves and surfaces, *Prog. Fract. Differ. Appl.* **2**(3) (2016) 169–186.
29. A. K. Lazopoulos and D. Karaoulanis, On L-fractional derivatives and L-fractional homogeneous equations, *Int. J. Pure Appl. Math.* **21**(2) (2016) 249–268.
30. A. K. Lazopoulos, Numerical solutions of differential equations with fractional L-derivative, *Continuum Mech. Thermodyn.* **30** (2018) 667–674.
31. M. Jornet, Theory on new fractional operators using normalization and probability tools, *Fractal Fract.* **8**(11) (2024) 665.
32. C. Lee, Y. Nam, M. Bang, S. Ham and J. Kim, Numerical investigation of the dynamics for a normalized time-fractional diffusion equation, *AIMS Math.* **9**(10) (2024) 26671–26687.
33. C. Lee, S. Ham, Y. Hwang, S. Kwak and J. Kim, An explicit fourth-order accurate compact method for the Allen–Cahn equation, *AIMS Math.* **9**(1) (2024) 735–762.
34. Y. Hwang, J. Yang, G. Lee, S. Ham, S. Kang, S. Kwak and J. Kim, Fast and efficient numerical method for solving the Allen–Cahn equation on the cubic surface, *Math. Comput. Simul.* **215** (2024) 338–356.
35. G. H. Gao, H. W. Sun and Z. Z. Sun, Stability and convergence of finite difference schemes for a class of time-fractional sub-diffusion equations based on certain superconvergence, *J. Comput. Phys.* **280** (2015) 510–528.
36. P. Zhuang and F. Liu, Implicit difference approximation for the time fractional diffusion equation, *J. Appl. Math. Comput.* **22** (2006) 87–99.
37. Z. F. Huang, Scale-coupling and interface-pinning effects in the phase-field-crystal model, *Phys. Rev. E Stat. Nonlinear Soft Matter Phys.* **87**(1) (2013) 012401.
38. J. Yang, D. Lee, S. Kwak, S. Ham and J. Kim, The Allen–Cahn model with a time-dependent parameter for motion by mean curvature up to the singularity, *Chaos Solitons Fractals* **182** (2024) 114803.

APPENDIX A

Listing A.1 is a MATLAB code for the normalized time-fractional Allen–Cahn equation. The code is used to produce the results shown in Fig. 4a.

Listing A.1 MATLAB program.

```
clear; % Clear all variables from workspace
% Domain boundaries and grid size
Lx = -0.5; Rx = 0.5; Ly = -0.5; Ry = 0.5; Nx = 100; Ny = Nx;
h = (Rx-Lx)/Nx; % Spatial step size
x = linspace(Lx+0.5*h,Rx-0.5*h,Nx); % x-coordinate grid points
y = linspace(Ly+0.5*h,Ry-0.5*h,Ny); % y-coordinate grid points
[xx,yy] = meshgrid(x,y); % 2D grid coordinates
m=10; % Parameter m for epsilon computation
ep=0.3124*(m*h)-0.0029; % Interfacial thickness parameter
beta = 0.8; % Fractional order
r = 0.25; % Radius of initial circular region
dt = 0.2*h^2; % Time step size
T = 0.5*r^2; % Total simulation time
Nt = round(T/dt); % Total number of time steps
t = linspace(0,T,Nt+1); % Time vector

u0 = zeros(Nx,Ny,Nt+1); % Initialize solution array
% Set initial condition as a hyperbolic tangent function
% for circular interface
u0(:,:,1) = tanh((r-sqrt(xx.^2+yy.^2))/(sqrt(2)*ep));
u = u0; % Initialize current solution
% Temporary storage for intermediate solutions
ou1 = zeros(Nx,Ny); ou2 = zeros(Nx,Ny);
% Coefficient vectors for Thomas algorithm
ax = zeros(1,Nx)-1/h^2; cx = ax; ay = zeros(1,Ny)-1/h^2; cy = ay;
% Plot initial condition
figure(1); clf; hold on;
contourf(x,y,u0(:,:,1)',[0 0],'FaceColor','g','LineWidth',1);
axis image; axis([Lx Rx Ly Ry])
xlabel("$x$",'Interpreter','latex'); ylabel("$y$",'Interpreter','latex');
```

```
for n = 1:Nt  % Time-stepping loop
    F = zeros(Nx,Ny);  % Initialize memory term accumulator
    if beta==1
        w(n) = 1;  % Weight for beta=1
    else
        deno = n^(1-beta);
        p = 1:n;
        % Compute weights for fractional derivative
        w = ((n+1-p).^(1-beta)-(n-p).^(1-beta))/deno;
        if n > 1
            for p = 1:n-1
                % Accumulate memory term using past solution differences
                F = F + w(p)*(u(:,:,p+1)-u(:,:,p))/dt;
            end
        end
    end
    % Prepare diagonals for x-direction Thomas solve
    bx = zeros(1,Nx) + (w(n)/dt + 2.0/h^2);
    bx(1) = w(n)/dt + 1.0/h^2;  % Adjust boundary
    bx(Nx) = w(n)/dt + 1.0/h^2;
    % Prepare diagonals for y-direction Thomas solve
    by = zeros(1,Ny) + (w(n)/dt + 2.0/h^2);
    by(1) = w(n)/dt + 1.0/h^2;  % Adjust boundary
    by(Ny) = w(n)/dt + 1.0/h^2;
    % Solve linear system along x-direction for each y-slice
    for xj = 1:Ny
    dx=w(n)*u(:,xj,n)'/dt-F(:,xj)'/2; % Right-hand side for x-direction
    ou1(:,xj) = thomas(ax,bx,cx,dx); % Solve tridiagonal system
    end
    % Solve linear system along y-direction for each x-slice
    for yi = 1:Nx
    dy=w(n)*ou1(yi,:)/dt-F(yi,:)/2; % Right-hand side for y-direction
    ou2(yi,:) = thomas(ay,by,cy,dy); % Solve tridiagonal system
    end
    % Nonlinear update formula
    E = exp(-2*dt/(ep^2*w(n)));
    u(:,:,n+1) = ou2 ./ sqrt(E + ou2.^2 * (1-E));
    % Plot solution every 100 steps
    if mod(n, 100) == 0
        figure(2); clf;
        contourf(xx,yy,u(:,:,n+1)',[0,0],'FaceColor','g','LineWidth',1);
        axis image; axis([Lx Rx Ly Ry])
        xlabel("$x$",'Interpreter','latex');
        ylabel("$y$",'Interpreter','latex'); pause(0.1);
    end
end
% Thomas algorithm to solve tridiagonal linear system
function x = thomas(alpha,beta,gamma,f)
    n = length(f);
    for i = 2:n
        mult = alpha(i)/beta(i-1);
        beta(i) = beta(i) - mult*gamma(i-1);
        f(i) = f(i) - mult*f(i-1);
    end
    x(n) = f(n)/beta(n);
    for i = n-1:-1:1
        x(i) = (f(i) - gamma(i)*x(i+1)) / beta(i);
    end
end
```